

НАЦІОНАЛЬНА АКАДЕМІЯ ПЕДАГОГІЧНИХ НАУК УКРАЇНИ
ІНСТИТУТ ЦИФРОВІЗАЦІЇ ОСВІТИ

Кваліфікаційна наукова
праця на правах рукопису

ПРОСКУРА СВІТЛАНА ЛЕОНІДІВНА

УДК [378.22:004-051]:37.016:004.774

ДИСЕРТАЦІЯ

**ВИКОРИСТАННЯ ВЕБОРІЄНТОВАНИХ ТЕХНОЛОГІЙ У
НАВЧАННІ ПРОГРАМУВАННЯ МАЙБУТНІХ БАКАЛАВРІВ
КОМП'ЮТЕРНИХ НАУК**

011 «Освітні, педагогічні науки»

01 «Освіта/Педагогіка»

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Всі частини тексту дисертації, під час написання яких використовувалися технології штучного інтелекту, перевірені та відредаговані автором особисто.



С. Л. ПРОСКУРА

Науковий керівник – ЛИТВИНОВА Світлана Григорівна, доктор педагогічних наук, професор, член-кореспондент НАПН України

Київ – 2026

АНОТАЦІЯ

Проскура С. Л. Використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук. Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії (PhD) за спеціальністю 011 – Освітні, педагогічні науки, освітньо-наукової програми «Інформаційно-комунікаційні технології в освіті». – Інститут цифровізації освіти Національної академії педагогічних наук України. Київ, 2026.

Зміст анотації.

Сучасний етап розвитку вищої освіти характеризується цифровою трансформацією освітнього процесу, що передбачає використання веборієнтованих технологій, хмарних сервісів, платформ дистанційного та змішаного навчання, інтерактивних засобів комунікації й цифрових освітніх ресурсів. Це зумовлює необхідність переосмислення традиційних підходів до організації навчання програмування майбутніх бакалаврів комп'ютерних наук та пошуку нових ефективних моделей формування конкурентоспроможного випускника.

Необхідність використання веборієнтованих технологій у навчанні програмування посилюється в умовах змішаного навчання, яке стало однією з провідних моделей організації освітнього процесу у закладах вищої освіти. Поєднання аудиторної та дистанційної форм навчання забезпечує гнучкість освітнього процесу, розширює можливості взаємодії між викладачем і студентами, сприяє адаптації навчання до індивідуальних потреб здобувачів освіти. Вони відкривають широкі можливості для індивідуалізації навчання, забезпечення постійного доступу до навчальних матеріалів, платформ розробки, організації колективної роботи, оцінювання результатів навчання, розвитку навичок самостійної роботи та формування цифрової компетентності майбутніх ІТ-фахівців.

Водночас аналіз сучасних наукових праць і освітньої практики свідчить про домінування переважно інструментального підходу до використання

цифрових технологій, коли вебресурси виконують функції зберігання та поширення навчальних матеріалів або забезпечення комунікації між учасниками освітнього процесу. Натомість питання комплексного використання веборієнтованих технологій як дидактичної основи навчання програмування, що передбачає інтеграцію хмарних середовищ розробки, систем контролю версій, хмарних сервісів, засобів автоматизованої перевірки програмного коду та інструментів командної взаємодії, потребує додаткового теоретичного й методичного обґрунтування

У результаті аналізу наукових праць вітчизняних та зарубіжних учених, узагальнення власного педагогічного досвіду та дослідження потреб сучасного ІТ-ринку було виявлено низку *суперечностей між*: сучасним розвитком цифрових технологій і недостатньою кількістю методичних досліджень щодо системного їх використання в закладах вищої освіти під час організації змішаної форми навчання; зростаючими вимогами ІТ-галузі до підготовки фахівців, здатних ефективно працювати у веборієнтованих середовищах розробки програмного забезпечення, та недостатньою теоретико-методичною розробленістю системного використання веборієнтованих технологій у процесі навчання програмування бакалаврів комп'ютерних наук; необхідністю об'єктивного моніторингу навчальних досягнень бакалаврів комп'ютерних наук та недостатнім обґрунтуванням методик формуального оцінювання на основі таксономії Блума та аналізу цифрового сліду.

Отже, *проблема* теоретико-методичного обґрунтування використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук, що враховує методику формуального оцінювання на основі таксономії Блума та аналізу цифрового сліду, є остаточно не розв'язаною, що потребує науково-методичного обґрунтування та розроблення відповідної методики. Відтак, дослідження «Використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук» є актуальним і затребуваним у закладах вищої освіти.

Для розв'язання окресленої проблеми було здійснено аналіз еволюції вебтехнологій – від Веб 1.0 до Веб 4.0, та встановлено, що сучасний етап розвитку характеризується посиленням інтелектуалізації вебсередовища, розширенням можливостей персоналізованої взаємодії користувача з цифровими системами та інтеграцією технологій штучного інтелекту в освітній професійній практиці.

Результатом проведеного порівняльного аналізу світового досвіду цифровізації освіти стало виокремлення трьох домінантних стратегій, кожна з яких пропонує специфічні механізми підготовки майбутніх фахівців у галузі інформаційних технологій. Вивчення атлантичної моделі, що базується на принципах децентралізації та активному впровадженні хмарних середовищ розробки, дозволило обґрунтувати доцільність переходу від локальних інструментів до хмаро орієнтованих інтегрованих середовищ розробки. Такий підхід забезпечує ідентичність навчального та професійного середовищ, що є критично важливим для формування готовності бакалаврів до роботи в сучасних ІТ-компаніях. Водночас аналіз континентальної моделі продемонстрував ефективність інтеграції спеціалізованих віртуальних лабораторій безпосередньо в системи управління навчанням. Це дозволяє реалізувати механізми автоматизованого тестування коду та забезпечити високу керованість освітнього процесу в умовах масової підготовки студентів, де об'єктивність і швидкість зворотного зв'язку відіграють вирішальну роль.

Окрему увагу в межах аналізу було приділено східноазійському вектору розвитку, який характеризується високим рівнем державної підтримки освітніх інновацій та інтеграцією систем штучного інтелекту як інтелектуальних помічників для здобувачів вищої освіти. Це підтверджує перспективність трансформації веборієнтованого навчання з формату статичного збереження ресурсів у динамічні інтелектуальні системи, що адаптуються до індивідуального темпу засвоєння алгоритмічного мислення.

Моніторинг практики підготовки майбутніх фахівців у вітчизняних закладах вищої освіти засвідчив, що 95,7% студентів самостійно опановують

технології штучного інтелекту, тоді як 75% здобувачів освіти використовують системи автоматизованої перевірки програмного коду (зокрема e-olymp) поза межами офіційно організованого освітнього процесу.

У ході дослідження було розроблено дві моделі, що відображають процес веборієнтованого навчання та особливості використання веборієнтованих технологій у підготовці майбутніх бакалаврів комп'ютерних наук як безпосередньо в освітньому процесі, так і в умовах самостійної навчальної діяльності студентів.

Відповідно до останніх рекомендацій Європейської комісії щодо розбудови цифрового освітнього простору, у дослідженні було теоретично обґрунтовано та практично розроблено підходи до оцінювання навчальних досягнень з програмування майбутніх бакалаврів комп'ютерних наук.

Особливістю пропонованого підходу є використання рамки DigComp 3.0, згідно з якою здобувач вищої освіти має досягти 7–8 (експертного) рівня володіння цифровими технологіями. Це передбачає не лише репродуктивне використання веборієнтованих інструментів, а й здатність майбутніх фахівців до самостійного створення інноваційних цифрових рішень, критичного аналізу складних даних та ефективного управління професійними проєктами у хмарних середовищах.

У процесі обґрунтування методики використання веборієнтованих технологій деталізовано процесуальний складник на прикладі дисципліни «Веборієнтовані технології. Frontend-розробка». Запропонована методика передбачає поетапне створення мінімально життєздатного продукту (MVP), що реалізується у шість взаємопов'язаних етапів – від розроблення макету інтерфейсу у Figma до розгортання готового вебзастосунку на платформі Vercel. Така організація навчальної діяльності забезпечує послідовне формування практичних навичок веброзробки та наближення освітнього процесу до реальних умов ІТ-індустрії.

У межах методики впроваджено режим програмування з використанням штучного інтелекту (ШІ), за якого технології ШІ виконують роль

інтелектуального помічника студента у процесі розроблення програмного коду, зокрема під час налагодження програм, пояснення алгоритмічних конструкцій та аналізу помилок. Це сприяє підвищенню рівня самостійності студентів і розвитку їхнього алгоритмічного мислення.

Крім того, розроблено систему оцінювання, що поєднує штрафні та заохочувальні механізми. Штрафні бали нараховуються за порушення термінів виконання завдань, тоді як заохочувальні – за активну участь у взаємному рецензуванні програмного коду, а також успішне проходження додаткових сертифікаційних курсів на платформах Coursera та FreeCodeCamp. Такий підхід сприяє підвищенню навчальної мотивації та стимулює систематичну навчальну активність здобувачів освіти.

Для доведення ефективності авторської методики було проведено педагогічний експеримент ($n = 112$) на базі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського, Міжнародного європейського університету, Запорізького національного університету, Національного університету «Запорізька політехніка», Київського столичного університету імені Бориса Грінченка. За допомогою авторського веборієнтованого застосунку інформаційної системи «E-student» встановлено, що в експериментальній групі середній бал підвищився на 15 балів – з 66,83 до 81,93. Статистичну значущість отриманих результатів підтверджено t -критерієм Велча ($t = 5,72 > 2,63$), U -критерієм Манна-Вітні ($p < 0,001$), а достовірність отриманих результатів – за критерієм кутового перетворення Фішера ($\phi^* > 2,31$).

У ході дослідження виконано всі поставлені завдання, досягнуто мети роботи. Підтверджено *гіпотезу* дослідження про те, що впровадження авторської методики навчання на основі веборієнтованих технологій сприятиме підвищенню рівня навчальних досягнень студентів із програмування.

Наукова новизна і теоретичне значення одержаних результатів. У дисертації *вперше*:

- розроблено й обґрунтовано авторську модель навчання програмування як інтегровану дидактичну систему, що базується на принципах неперервності та інтелектуальної співпраці у хмаро орієнтованому середовищі розробки;

- розроблено й обґрунтовано авторську модель і теоретично обґрунтовано методику використання веборієнтованих технологій, спрямовану на формування цифрової професійної ідентичності бакалавра;

- обґрунтовано перехід до хмаро орієнтованої парадигми навчання, у межах якої веббраузер стає єдиним освітнім середовищем для навчання програмування (зокрема через GitHub Codespaces, Replit), що усуває технічні перешкоди, пов'язані з локальним налаштуванням програмного забезпечення;

- розроблено і обґрунтовано авторську пропорцію змішаного навчання для ІТ-спеціальностей (30% традиційного, 50% дистанційного (хмаро орієнтованого) та 20% проєктного навчання), що моделює реальний життєвий цикл розробки програмного забезпечення (SDLC);

- обґрунтовано підходи до контролю, де основним об'єктом оцінювання є не фінальний файл, а історія фіксацій змін у GitHub, що дозволяє об'єктивно підтвердити академічну доброчесність та оцінити ітеративність розробки.

Доведено доцільність вживання терміна «веборієнтовані технології» як найбільш точного для ІТ-фахівців, оскільки він вказує на середовище реалізації (HTTP, DOM, браузер), на відміну від занадто широкого терміна «цифрові технології»;

Удосконалено: підходи до структурування великих обсягів навчальних відомостей за допомогою веборієнтованих інтелект-карт, що дозволяє візуалізувати складні алгоритмічні концепції; підходи до формування компетентності з програмування шляхом інтеграції ІІІ як «інтелектуального помічника», а не просто засобу генерування коду та пошуку інформації.

Встановлено, що в умовах системної цифрової трансформації (2020–2026 рр.) компетентність з програмування трансформується з вузькотехнічної навички в основну цифрову компетентність майбутнього фахівця.

Дістали подальшого розвитку класифікація веборієнтованих технологій, доповнена еволюційними характеристиками Веб 4.0 (AI-агенти, Symbiotic Web) у контексті підготовки програмістів та підходи до організації освітнього процесу в закладах вищої освіти на засадах використання ІКТ-технологій.

Практичне значення отриманих результатів дослідження полягає в тому, що розроблено:

- розроблено методику використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук в умовах змішаного навчання;
- розроблено авторську систему оцінювання, де рівні інтелектуальної діяльності за таксономією Блума (6 рівнів) вперше корелюють із рівнями цифрової компетентності DigComp 3.0 (від 1–3 базового до 7–8 експертного);
- розроблено та застосовано спеціалізований авторський веборієнтований застосунок – інформаційну систему «E-student» для автоматизованого формуального оцінювання навчальних досягнень студентів на основі рівнів таксономії Блума;
- розроблено практикум для студентів із системою завдань, побудованих на основі таксономії Блума «Веборієнтовані технології. Frontend-розробка: практикум».

Достовірність отриманих результатів. Ефективність авторської методики використання веборієнтованих технологій у навчанні майбутніх бакалаврів комп'ютерних наук підтверджено методами математичної статистики.

Основні положення, результати й висновки дисертації можуть бути використані викладачами закладів вищої освіти та вченими для подальшого обґрунтування теоретико-методологічних засад використання веборієнтованих технологій у навчанні студентів ЗВО.

Ключові слова: бакалавр комп'ютерних наук, веборієнтовані технології, програмування, DigComp 3.0, штучний інтелект, таксономія Блума, Cloud-native навчання, GitHub, програма «E-student», змішане навчання, хмаро орієнтовані середовища навчання, неперервність навчання.

ABSTRACT

Proskura S. L. The Use of Web-Oriented Technologies in Teaching Programming to future Bachelors of Computer Sciences. – Qualification scientific work in the rights of an unpublished manuscript.

Dissertation work for obtaining the scientific degree of Doctor of Philosophy (PhD) in the specialty 011 – Educational, Pedagogical Sciences, Educational-Scientific Program «Information and Communication Technologies in Education». – Institute for Digitalization of Education of the National Academy of Educational Sciences of Ukraine. – Kyiv, 2026.

Contents of the Abstract.

The modern stage of higher education development is characterized by the digital transformation of the educational process, which involves the use of web-oriented technologies, cloud services, distance and blended learning platforms, interactive communicational tools, and digital educational resources. This necessitates reconsideration of traditional approaches to organizing programming education for future Bachelors of Computer Sciences, as well as the search for new effective models for developing a competitive graduate.

The necessity of using web-oriented technologies in programming education is growing in the context of blended learning, which has become one of the leading models for organizing the educational process in higher education institutions. The combination of classroom and distance learning formats not only provides flexibility in the educational process but also expands opportunities for interaction between the teacher and students, as well as supports adapting the learning process to individual needs of learners. This opens wide opportunities for the individualization of learning, ensuring constant access to educational materials, development platforms, organizing collaborative work, assessing the learning outcomes, developing independent working skills, and forming the digital competence of future IT specialists.

At the same time, the analysis of modern scientific works and educational practice indicates the predominance of mainly an instrumental approach to the use of digital technologies, where web resources mostly perform the function of storing and

dispensing educational materials or providing communication between participants in the educational process. In contrast, the question of comprehensive use of web-oriented technologies as a didactic basis for learning programming, which involves the integration of cloud development environments, version control systems, cloud services, automated code checking tools, and tools for team interaction, requires additional theoretical and methodological substantiation.

As a result of the analysis of scientific works by domestic and foreign researchers, the generalization of personal pedagogical experience, and the study of modern IT market needs, a number of *contradictions* were identified between: the modern development of digital technologies and the insufficient number of methodological studies on the systematic use of these technologies in higher education institutions during the organization of blended learning; the growing requirements of the IT industry to training of specialists capable of effectively working in web-oriented software development environments, and the insufficient theoretical and methodological development of the systematic use of web-oriented technologies in the process of teaching programming to computer science undergraduates; the need for objective monitoring of the educational achievements of the bachelors of computer sciences and the insufficient substantiation of formative assessment methods based on Bloom's taxonomy and the analysis of digital traces.

Thus, *the problem* of theoretic-theoretical justification and modeling of the use of web-oriented technologies in teaching programming to computer science undergraduates, which takes into account the methodology of formative assessment based on Bloom's taxonomy and digital trace analysis, is not yet fully resolved, requiring scientific and methodological substantiation and the development of an appropriate methodology, while the study "The Use of Web-Oriented Technologies in Teaching Programming to future Bachelors of Computer Sciences" is relevant and in demand in higher education institutions.

To properly address the outlined problem, an analysis of web technologies development—from Web 1.0 to Web 4.0—was conducted, where it was established that the current stage of its development is characterized by intensification of web

environment intellectualization, the expansion of opportunities for personalized user interaction with digital systems, and the integration of artificial intelligence technologies into educational and professional practices.

The result of the comparative analysis of the global experience in education digitalization was the identification of three dominant strategies, each of which proposes specific mechanisms for training future specialists in the field of information technologies. The study of the Atlantic model, which is based on the principles of decentralization and the active implementation of cloud development environments, allowed substantiating the feasibility of transitioning from local tools to cloud-native IDE.

This approach ensures the identity of educational and professional environments, which is critically important for preparing the students of bachelor's degree to work in modern IT companies. At the same time, the analysis of the continental model demonstrated the effectiveness of integrating specialized virtual laboratories directly into learning management systems. This allows the implementation of automated code testing mechanisms and ensures high manageability of the educational process in conditions of mass student training, where objectivity and speed of feedback play a decisive role.

Special attention within the scope of the analysis was given to the East Asian development vector, which is characterized by a high level of state support for educational innovations and the integration of artificial intelligence systems in the role of intelligent assistants for higher education students. This confirms the prospects of transforming web-oriented learning from a format of static resource storage into dynamic intelligent systems that adapt to the individual pace of algorithmic thinking mastering.

The monitoring practice of future computer science specialists training in domestic higher education institutions has shown that 95.7% of students independently master artificial intelligence technologies, while 75% of the learners use automated code-checking systems (e-olymp, in particular) outside the officially organized educational process.

During the study, two models were developed that reflect the process of web-oriented learning and the features of using web-oriented technologies in the process of future computer science bachelors training both in the educational process and during students' independent learning activities.

In accordance with the latest recommendations of the European Commission regarding the development of the digital educational space, the study theoretically substantiated and practically developed approaches to assessing the learning achievements of future computer science bachelors. The feature of the proposed approach is the use of the DigComp 3.0 framework, according to which a higher education student should achieve level 7–8 (expert) in digital technology proficiency. This entails not only the reproductive use of web-oriented tools but also the ability of future specialists to independently create innovative digital solutions, critically analyze complex data, and effectively manage professional projects in cloud environments.

In the process of substantiating the methodology for using web-oriented technologies, the procedural component is detailed while taking the course "Frontend Development" as an example course for this purpose. The proposed methodology involves the step-by-step creation of a minimally viable product (MVP), which is implemented in six interrelated stages — from developing an interface template in Figma to deploying the finished web application on the Vercel platform. Such organization of educational activities ensures consistent formation of practical web development skills and brings the educational process closer to the real conditions of the IT industry.

Within the methodology, an AI Pair Programming mode has been implemented, in which artificial intelligence technologies act as a student's partner in the process of code developing, in particular to support debugging, explain algorithmic constructions, as well as analyze errors. This contributes to increasing students' level of independence and the development of their algorithmic thinking.

In addition, an assessment system has been developed that combines penalty and reward mechanisms. Penalty points are awarded for violating assignment

deadlines, while reward points are given for active participation in Code Review, as well as successfully completing additional certification courses on the Coursera and FreeCodeCamp platforms. This approach ensures an increase in educational motivation and stimulates systematic learning activity among students.

A pedagogical experiment ($n = 112$) was conducted at the fields of the National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute,” the International European University, Zaporizhzhia National University, the National University “Zaporizhzhia Polytechnic,” and Borys Grinchenko Kyiv Metropolitan University to prove the effectiveness of the author's methodology. Using the author's web-oriented software tool “E-student,” it was established that in the experimental group the average score increased by 15 points — from 66.83 to 81.93. The statistical significance of the obtained results was confirmed by Welch's t-test ($t = 5.72 > 2.63$), the Mann-Whitney U test ($p < 0.001$), and Fisher's angular transformation criterion ($\phi^* > 2.31$).

Within the scope of this study, all assigned tasks were completed, and the goal of the work was achieved. The research *hypothesis*, that the implementation of the author's methodology based on web-oriented technologies contributes to increasing the level of students' academic achievements in programming, was confirmed.

Scientific novelty and theoretical significance of the obtained results. In the present dissertation work *for the first time*:

- the author's model of programming education as an integrated didactic system based on the principles of continuity and intellectual collaboration in a cloud-oriented development environment has been developed and substantiated;

- the author's model as well as theoretically substantiated methodology for using web-oriented technologies aimed at forming the digital professional identity of a bachelor student, based on the principles of continuity and intellectual collaboration in a cloud-oriented development environment, have been developed and substantiated;

- the transition to the Cloud-native learning paradigm has been substantiated, where the web browser becomes the sole educational window for learning

programming (through GitHub Codespaces, Replit), eliminating technical barriers of local software setup;

- the author's proportion of blended learning for IT specialties has been developed and substantiated - 30% traditional, 50% remote (Cloud-native), and 20% project-based learning, modeling the real software development life cycle (SDLC);

- approaches to assessment have been substantiated, where the main evaluation object is not the final grade file, but the history of change records in GitHub, which allows objectively confirming academic integrity and assessing the iterative nature of development.

Has been proved: The viability of the use of the term "web-oriented technologies" has been proved as the most accurate for IT specialists, as it indicates the implementation environment (HTTP, DOM, browser), unlike the term "digital technologies", which embraces a broader meaning;

Have been improved: approaches to structuring large volumes of educational information using web-oriented mind maps, which allows visualizing complex algorithmic concepts; approaches to programming competence developing through the integration of AI as an "intelligent assistant," rather than merely a tool for code generation and information retrieval.

Has been established: that under conditions of systemic digital transformation (2020–2026), programming competence is transforming from a narrowly technical skill into a core digital competence of a specialist.

The classification of web-oriented technologies, supplemented with the evolutionary characteristics of Web 4.0 (AI agents, Symbiotic Web) in the context of programmer training, and approaches to organizing the educational process in higher education institutions based on the use of ICT technologies, *have received further development.*

The practical significance of the research results obtained lies in the fact that the following have been developed:

- the methodic for using web-oriented technologies in teaching programming to computer science bachelor students in the conditions of blended learning environment;

- the original author's assessment system, where levels of intellectual activity according to Bloom's taxonomy (6 levels) have been for the first time correlated with levels of digital competence of DigComp 3.0 (from 1–3 basic to 7–8 expert);

- the specialized original web-oriented application "E-student" has been developed by the author and applied for automated formative assessment of students' learning achievements based on the levels of Bloom's taxonomy;

- the practicum for students "Web-oriented technologies. Front-end development: Practical workshop" with a system of tasks built based on Bloom's taxonomy has been created.

Authenticity and credibility of the results obtained. The effectiveness of the author's methodology for using web-oriented technologies in the education of future computer science bachelors is confirmed by methods of mathematical statistics.

The main provisions, results, and conclusions of this dissertation work can be used by higher education instructors and researchers for further substantiation of the theoretical and methodological foundations of using web-oriented technologies in the education of university students.

Keywords: Bachelor of Computer Science, web-oriented technologies, programming, DigComp 3.0, artificial intelligence, Bloom's taxonomy, Cloud-native learning, GitHub, "E-student" program, blended learning, cloud-oriented learning environments, continuity of learning.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Розділи монографії

1. Oleksandr Burov, Evgeniy Lavrov, Svitlana Lytvynova, Olha Pinchuk, **Svitlana Proskura**, Oleksii Tkachenko, Natalia Kovalenko, Yana Chybiriak & Yana Dolgikh. Cognitive and Perceptual Reliable Performance: Comparison of Psychophysiological Limitations. *HCI International 2025 Posters. Communications in Computer and Information Science. Cham: Springer, 2025. Vol. 2523. P. 3-13.* DOI: https://doi.org/10.1007/978-3-031-94153-5_1 (Scopus).

2. **Проскура С. Л.**, Литвинова С. Г., Кронда О. П. Засоби організації дистанційного навчання в період карантину 2020 року в закладах вищої освіти України. *Екстрене дистанційне навчання в Україні* : колективна монографія / за ред. В. М. Кухаренка, В. В. Бондаренка. Харків : КП "Міська друкарня", 2020. С. 299–313.

URL: https://duan.edu.ua/images/News/UA/Departments/Management/2020/monograph_ekstr_dyst_navch.pdf

Статті у наукометричних базах Scopus

1. Lytvynova S. H., Rashevskaya N. V., **Proskura S. L.** The use of artificial intelligence in teaching students programming languages *Proceedings of the IX International Workshop on Professional Retraining and Life-Long Learning using ICT: Person-oriented Approach (3L-Person 2024), co-located with ICTERI 2024. CEUR Workshop Proceedings, 2024. Vol. 3781. P. 10–29.* URL: <https://ceur-ws.org/Vol-3781/paper01.pdf>

2. **Proskura S. L.**, Lytvynova S. H., Kronoda O. P. Students' academic achievement assessment in higher education institutions. *ICTERI 2020: Proceedings of the 16th International Conference on ICT in Education, Research and Industrial Applications. CEUR Workshop Proceedings, 2020. Vol. 2732. P. 734–746* DOI: URL: <http://ceur-ws.org/Vol-2732/20200734.pdf>

3. **Proskura S. L.**, Lytvynova S. H., Kronda O. P. Demeshkant N. Mobile learning approach as a supplementary approach in the organization of the studying process in educational institutions. *ICTERI 2020: Proceedings of the 16th International Conference on ICT in Education, Research and Industrial Applications. CEUR Workshop Proceedings*, 2020. P. 650–664. URL: <http://ceur-ws.org/Vol-2732/20200650.pdf>
4. **Proskura S. L.**, Lytvynova S. H. The approaches to Web-based education of computer science bachelors in higher education institutions. *Cloud technologies in education. CTE-2019. CEUR Workshop Proceedings*, 2020. 2643. P.609-625. .URL: <https://ceur-ws.org/Vol-2643/paper36.pdf>
5. **Proskura S. L.**, Lytvynova S. H. Organization of independent studying of future bachelors in computer science within higher education institutions of Ukraine. *ICTERI 2018 (3L-Person 2018). CEUR Workshop Proceedings*, 2018. P. 348–358. URL: http://ceur-ws.org/Vol-2104/paper_160.pdf

Статті у фахових наукових виданнях України

1. **Proskura S. L.**, Lytvynova S. H., Kronda O. P. The use of WEB-oriented technologies in the process of WEB-programming teaching for technical universities students. *Educational Dimension*, 2021. №5. URL: <https://lib.iitta.gov.ua/id/eprint/728607/1/Proskura-Lytvynova-Kronda-2021.pdf> DOI: <https://doi.org/10.31812/educdim.4724>
2. **Проскура С. Л.**, Литвинова С. Г. Формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*. 2019. № 2 (20). DOI: <https://doi.org/10.31110/2413-1571-2019-020-2-022> URL: https://fmo-journal.fizmatsspu.sumy.ua/journals/2019-v2-20/2019_2-20-Proskura_Lytvynova_FMO.pdf
3. **Проскура С. Л.** Модель формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*, 2019. № 3 (21). DOI: <https://doi.org/10.31110/2413-1571-2019-021-3-016>

URL: https://fmo-journal.fizmatsspu.sumy.ua/journals/2019-v3-21/2019_3-21-Proskura_FMO.pdf

4. **Проскура С. Л.,** Литвинова С. Г. Підготовка фахівців з інформаційних технологій у закладах вищої освіти: стан, проблеми і перспективи. *Інформаційні технології в освіті*, 2018. № 2(35). Р. 72–88. DOI: <https://doi.org/10.14308/ite000668> (дата звернення: 27.03.2026).

5. **Проскура С. Л.** Застосування інтелект-карт для підвищення якості та ефективності навчання студентів курсу програмування вищих навчальних закладів. *Актуальні питання природничо-математичної освіти*, 2017. С. 220–228. URL: https://fizmat.sspu.edu.ua/images/NAUKA/APPMO/Arhiv/APPMO_7-8_2016_1c655.pdf

Матеріали апробаційного характеру

1. **Проскура С. Л.** Педагогічні умови використання WEB-орієнтованих технологій у підготовці бакалаврів з інформаційних технологій та систем. *Цифрова трансформація науково-освітніх середовищ в умовах воєнного* : збірник матеріалів. Звітна наукова конференція Інституту цифровізації освіти НАПН України, 23 лютого 2024 р., м. Київ / упоряд.: О. П. Пінчук, Н. В. Яськова. Київ : ІЦО НАПН України, 2024. С. 130–132. URL: https://lib.iitta.gov.ua/id/eprint/740554/1/Збірник_тез_звітної_2024_v2.pdf

2. **Проскура С. Л.,** Литвинова С. Г. Моніторинг використання WEB-орієнтованих технологій бакалаврами комп'ютерних наук в процесі вивчення програмування. *Інформаційно-цифровий освітній простір України: трансформаційні процеси і перспективи розвитку* : матеріали методологічного семінару НАПН України, 4 квітня 2019 р.. Київ : НАПН України, 2019. С. 211–219. URL: <https://lib.iitta.gov.ua/id/eprint/717123/>

3. **Проскура С. Л.,** Таксономія Блума в оцінюванні результатів освітньої діяльності студентів. Звітна наукова конференція Інституту інформаційних технологій і засобів навчання НАПН України. 2020. С.83-89.

URL: <https://lib.iitta.gov.ua/id/eprint/720537/1/Збірник%20тез%20звітної%20конференції%20ІТЗН%20НАПН%20України%202020.pdf>

4. Kronda O. P., **Proskura S. L.** The level of digital technologies use in higher education institutions in the conditions of distance and blended learning. *Наукова молодь-2020* : матеріали VIII Всеукр. наук.-практ. конф., м.Київ, 2020. С. 129-131

С.URL: https://lib.iitta.gov.ua/id/eprint/722327/1/ЗБІРНИК%20НАУКОВА%20МОЛОДЬ%202020_1.pdf

5. **Проскура С. Л.** Використання Web-орієнтованих технологій в закладах вищої освіти. *Наукова молодь. 2019*: матеріали VII Всеукраїнської науково-практичної конференції. 2019. С.39-42

URL: https://lib.iitta.gov.ua/id/eprint/718530/2/Збірник%20Наукова%20молодь%202019.pdf?utm_source

6. **Проскура С. Л.,** Литвинова С. Г., Кронда О. П. Оцінювання рівня знань бакалаврів комп'ютерних наук у закладах вищої освіти. *Сучасні тенденції та фактори розвитку педагогічних та психологічних наук в Україні та країнах ЄС* : матеріали міжнародної наук.-практ. конф., Люблін. Польща, 2020. С.250-254.

URL: <http://www.baltijapublishing.lv/omp/index.php/bp/catalog/download/68/1534/3505-1>

7. **Проскура С. Л.,** Особливості організації змішаного навчання майбутніх бакалаврів комп'ютерних наук у закладах вищої освіти. *Звітна наукова конференція Інституту інформаційних технологій і засобів навчання НАПН України.* 2019. С 133 – 137.

URL: https://lib.iitta.gov.ua/id/eprint/715956/1/Збірник%20тез%20звітної%202019_9.pdf

8. **Проскура С. Л.,** Литвинова С. Г. Особливості підготовки майбутніх бакалаврів комп'ютерних наук в університетах США. *Наукова молодь-2018* : матеріали VI Всеукраїнської наук.-практ. конф. молодих учених. Інститут

інформаційних технологій і засобів навчання, м. Київ, Україна, 16 листоп. 2018.
С. 21-24 URL: https://lib.iitta.gov.ua/id/eprint/717127/?utm_source

9. **Проскура С. Л.,** Литвинова С. Г. Огляд компетентностей майбутніх бакалаврів комп'ютерних наук. *Звітна наукова конференція Інституту інформаційних технологій і засобів навчання НАПН України: збірник матеріалів наукової конференції.* 2018. С.32-34.
URL: <http://lib.iitta.gov.ua/711730/1/Збірник%20тез%20звітна%202018-output.pdf>

10. **Проскура С. Л.** Розвиток радіантного мислення студентів вищих навчальних закладів. *Концептуальні шляхи розвитку науки* : матеріали міжнародної науково-методичної конференції, 2017. С.65-66.
URL: https://lib.iitta.gov.ua/id/eprint/712031/1/proskura_rad_mysl.pdf?utm_source

11. **Проскура С. Л.** Сучасні підходи до викладання мов програмування в закладах вищої освіти. *Наукова молодь-2017* : матеріали V Всеукраїнської науково-практичної конференції. Київ, 2017. С. 313–315. URL: <http://lib.iitta.gov.ua/709994/1/Збірник конф Наукова молодь 2017.pdf>

12. **Проскура С. Л.** Інтелект-карти як засіб організації самостійної роботи студентів-програмістів. *Розвиток інтелектуальних умінь і творчих здібностей учнів та студентів у процесі навчання дисциплін природничо-математичного циклу «ІТМ» плюс – 2017»* : матеріали II Міжнародної дистанційної науково-методичної конференції, 2017. м.Суми: у 2 ч. Ч.2 /упорядн. Чашечникова О. С. С. 38-39.
URL: https://lib.iitta.gov.ua/id/eprint/712029/1/Проскура_ІнтелектКартиОрганізаціяРоботиСтуд.pdf

Зміст

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	25
ВСТУП.....	26
РОЗДІЛ 1. ТЕОРЕТИКО-МЕТОДИЧНІ ОСНОВИ НАВЧАННЯ ПРОГРАМУВАННЯ МАЙБУТНІХ БАКАЛАВРІВ КОМП'ЮТЕРНИХ НАУК	41
1.1.Аналіз поняттєво-термінологічного апарату проблеми використання веборієнтованих технологій в освітньому процесі.....	41
1.2.Особливості підготовки майбутніх бакалаврів комп'ютерних наук у зарубіжних закладах вищої освіти	55
1.2.1 Атлантична модель підготовки бакалаврів комп'ютерних наук	63
1.2.2 Континентальна модель підготовки бакалаврів комп'ютерних наук.	68
1.2.3 Східноазійська модель підготовки бакалаврів комп'ютерних наук...	72
1.3. Сучасний стан підготовки майбутніх бакалаврів комп'ютерних наук у вітчизняних закладах вищої освіти.	78
1.4. Вимоги до рівня компетентності з програмування майбутніх бакалаврів комп'ютерних наук.....	86
Висновки до розділу 1	96
Список використаних джерел до Розділу 1	97
РОЗДІЛ 2. МОДЕЛЮВАННЯ НАВЧАННЯ ПРОГРАМУВАННЯ З ВИКОРИСТАННЯМ ВЕБОРІЄНТОВАНИХ ТЕХНОЛОГІЙ.....	117
2.1 Модель використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук	117
2.2 Модель веборієнтованого навчання майбутніх бакалаврів комп'ютерних наук	145
.3 Система оцінювання навчальних досягнень майбутніх бакалаврів комп'ютерних наук з програмування.....	155
2.3.1 Оцінювання комп'ютерних практикумів за таксономією Блума.....	169
Висновки до розділу 2	172
Список використаних джерел до Розділу 2	174
РОЗДІЛ 3. МЕТОДИКА ВИКОРИСТАННЯ ВЕБОРІЄНТОВАНИХ ТЕХНОЛОГІЙ У НАВЧАННІ ПРОГРАМУВАННЯ МАЙБУТНІХ БАКАЛАВРІВ КОМП'ЮТЕРНИХ НАУК.....	191

3.1 Цільовий та змістовий складники методики використання веборієнтованих технологій.....	191
3.2 Трансформація форм і засобів навчання програмування в умовах хмаро орієнтованого середовища	209
3.3 Використання веборієнтованих технологій у процесі навчання бакалаврів комп'ютерних наук.	227
3.3.1 Особливості розроблення комп'ютерних практикумів	281
3.3.2 Самостійна робота, індивідуальний підхід	288
3.3.3 Діагностика знаннєвого складника компетентності з програмування засобами модульного контролю	300
3.4 Види контролю та підходи до оцінювання знань студентів.....	303
Висновки до розділу 3	309
Список використаних джерел до Розділу 3	310
РОЗДІЛ 4. АНАЛІЗ ТА ІНТЕРПРЕТАЦІЯ РЕЗУЛЬТАТІВ ЕКСПЕРИМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ	324
4.1. Організація та хід констатувального етапу педагогічного експерименту	326
4.2 . Аналіз результатів формувального етапу педагогічного експерименту	330
Висновки до розділу 4	341
Список використаних джерел до Розділу 4.....	342
ВИСНОВКИ	344
ДОДАТОК А.....	347
ДОДАТОК Б	354
ДОДАТОК В	361
ДОДАТОК Г	375
ДОДАТОК Д	380
ДОДАТОК Е	387
ДОДАТОК К.....	393
ДОДАТОК Л.....	394
ДОДАТОК М.....	396
ДОДАТОК Н.....	406

ДОДАТОК О.....	411
ДОДАТОК П.....	413

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ЗВО	Заклад вищої освіти
ІКТ	Інформаційно-комунікаційні технології
КД/КІ	Користувацький інтерфейс (англ. UX/UI, User Experience/ User Interface)
МОН	Міністерство освіти і науки України
ПРН	програмні результати навчання
СУН / LMS	Системи управління/керування навчанням (англ. LMS, Learning Management Systems)
ХО	Хмаро орієнтований
ХОС	Хмаро орієнтоване середовище
API	Програмний інтерфейс застосунку / інтерфейс програмування застосунків (англ. Application Programming Interface)
CI/CD	Сукупність практик та інструментів автоматизації розробки програмного забезпечення (англ. Continuous Integration / Continuous Delivery)
CS	Комп'ютерні науки (англ. Computer Science)
CMS	Системи керування контентом (англ. Content Management Systems)
ICP	Інтегроване середовище розробки (англ. IDE, Integrated Development Environment)
IS	Інформаційні системи (англ. Information Systems)
JSX	Спеціальне синтаксичне розширення для JavaScript, яке використовується в бібліотеці React (англ. JavaScript XML)
LLM	Великі мовні моделі (англ. Large Language Models)
SDLC	Повний життєвий цикл розробки програмного забезпечення (англ. Software Development Life Cycle)
МЖП	мінімально життєздатний продукт (англ. MVP, Minimum Viable Product) [

ВСТУП

Трансформація вищої освіти XXI століття характеризується розривом між високою готовністю студентів до інновацій та фрагментарністю університетського цифрового середовища, яке переважно обмежується використанням платформ Moodle і Google Classroom та не забезпечує повноцінної інтеграції сучасних інструментів штучного інтелекту в освітній процес. Ця трансформація відбувається на тлі інтенсивного розвитку інформаційно-комунікаційних технологій (ІКТ) широкого впровадження хмарних сервісів, мобільних платформ, веборієнтованих середовищ і сервісів, що суттєво змінюють підходи до взаємодії та навчання майбутніх бакалаврів комп'ютерних наук.

Аналіз світових тенденцій демонструє, що глобальний ринок праці зазнає суттєвих змін під впливом технологічного прогресу, демографічних зрушень та зростання ролі цифрових технологій [1]. За даними документу «Звіт про майбутнє робочих місць 2025» («Future of Jobs Report 2025») Всесвітнього економічного форуму, до 2030 року понад 170 мільйонів нових робочих місць буде створено внаслідок технологічних змін, при цьому очікується, що приблизно 39 % навичок, необхідних для виконання завдань на робочому місці, зміняться або стануть застарілими через розвиток цифрових технологій, зокрема ШІ, великих даних, мережевих технологій та кібербезпеки [1], [2].

У звіті наголошується, що технологічні навички, зокрема цифрова грамотність, знання інформаційно-комунікаційних технологій (ІКТ), аналітичне мислення та здатність працювати з цифровими платформами, ставатимуть дедалі більш затребуваними у професіях майбутнього, зокрема у сферах розроблення програмного забезпечення та автоматизації. Отже, трансформація ринку праці вимагає від сучасних випускників не лише ґрунтовних фахових знань, а й здатності поєднувати технічні навички з компетентністю у сфері цифрової взаємодії, комунікації та використання інтелектуальних цифрових технологій [2].

Питання цифрових навичок також перебуває в центрі уваги європейських аналітичних структур. Згідно зі звітом «Подолання розриву в навичках: прийняття цифрової трансформації» («Bridging the skills gap: embracing digital transformation») Європейського тренінгового фонду (ETF), цифрові технології суттєво змінюють зміст праці та навички, необхідні на сучасному ринку, що є критичним для формування політик освіти та професійної підготовки. Підвищення здатності адаптуватися до цифрових трансформаційних процесів та формування цифрової компетентності визначено як ключовий чинник стійкості та готовності працівників до викликів цифрової економіки [3].

У світовому контексті тенденція до інтеграції цифрових навичок у професійну діяльність має виразний педагогічний аспект. Зростання вимог до цифрової грамотності, комплексного використання веборієнтованих середовищ та онлайн-платформ для розв'язання професійних завдань актуалізує потребу у формуванні в студентів не лише компетентностей у галузі програмування як технічної дисципліни, а й здатності ефективно використовувати веборієнтовані технології для розроблення, тестування, розгортання та спільного супроводу програмних продуктів.

Таким чином, міжнародні тенденції розвитку праці та освіти підтверджують, що формування сучасної цифрової компетентності є не лише ринковою вимогою, але й педагогічною необхідністю, що потребує системного наукового обґрунтування й впровадження у зміст і технології навчання бакалаврів комп'ютерних наук.

В Україні цей процес простежується через реалізацію державної політики у сфері цифрового розвитку, зокрема через діяльність Міністерства цифрової трансформації України [4], яке координує впровадження національних цифрових сервісів і платформ. Розбудова екосистеми електронних послуг, зокрема платформи Дія та пов'язаних із нею цифрових сервісів, актуалізує потребу у висококваліфікованих фахівцях, здатних проєктувати, розробляти та підтримувати веборієнтовані програмні системи.

У цьому контексті особливого значення набуває підготовка бакалаврів комп'ютерних наук, професійна діяльність яких безпосередньо пов'язана зі створенням і супроводом вебзастосунків, хмарних сервісів, розподілених інформаційних систем. Динаміка розвитку ІТ-галузі, поширення кросплатформних рішень, використання гнучкої методології розробки та інструментів командної роботи зумовлюють необхідність модернізації змісту й технологій навчання програмування у закладах вищої освіти (ЗВО).

Водночас цифровізація освіти як соціально-педагогічний феномен не зводиться до механічного впровадження інформаційно-комунікаційних технологій (ІКТ). У Концепції розвитку цифрової економіки та суспільства України [5] підкреслюється, що цифровізація передбачає інтеграцію цифрових інструментів у всі процеси діяльності, формування кіберфізичного простору та нових моделей взаємодії суб'єктів.

У межах вітчизняного наукового дискурсу ґрунтовно досліджено теоретико-методологічні засади цифровізації освіти та використання веборієнтованих технологій як дидактичної основи підготовки фахівців. Зокрема, у працях В. Кременя [6], В. Бикова [6–9], О. Спіріна [10–14], О. Пінчук [6, 10, 15], В. Осадчого [15–16], К. Осадчої [16] та Т. Вакалюк [11, 13, 14, 15] обґрунтовано, що системна трансформація освітнього процесу залежить від розвитку ІКТ-інфраструктури, хмарних обчислень, штучного інтелекту (ШІ) та сервісно-орієнтованих архітектур. Понятійно-термінологічний апарат проблеми використання вебресурсів та концепцію «віртуального класу» як складника синтетичного середовища розвинуто С. Литвиною [17–41], тоді як Н. Сороко [42] визначила вебтехнології як методику проєктування систем, що вдосконалюються завдяки мережевій взаємодії. Питання моделювання навчальних середовищ саме для інженерів-програмістів та класифікацію їхніх професійних компетентностей висвітлено у дослідженнях В. Круглика [43, 44].

Вагомий внесок у теорію і практику змішаного навчання та впровадження «руйнівних» веборієнтованих підходів зробили В. Кухаренко [45, 46] та Ю. Триус [47]. Організаційні моделі комбінованого навчання з використанням

LMS представлені у роботах С. Семерікова [48] та А. Стрюка [48, 49], а К. Лісецький [50] розкрив потенціал веборієнтованого навчання для забезпечення контролю студента над власною освітньою траєкторією. Науковці Т. Ковалюк і Н. Кобець аналізують інтеграцію української ІТ-освіти до європейського освітнього простору [51]. Окрему увагу аспектам об'єктивізації оцінювання через математичне моделювання та таксономію Б. Блума приділили Л. Огнівчук [52], О. Пометун [53] та Н. Гупан [53].

У працях зарубіжних учених розкрито аспекти еволюції вебресурсів від статичних систем Веб 1.0, описаних Д. Гаррісом [54], до інтерактивних платформ Веб 2.0 за Т. О'Рейлі [55]; інтелектуального контенту Веб 3.0 у трактуванні Дж. Калаканіса [56]. Концептуальний зв'язок між хмарними обчисленнями та персоналізацією освітнього досвіду обґрунтували Дж. Поллок [57, 58] і К. Ватсон [59, 60]. Педагогічні стратегії дизайну онлайн-курсів програмування та чинники подолання технічних перешкод на початкових етапах навчання дослідили Н. Шенлі [61] та Н. Труонг [62]. К. Ватсон довів ефективність інтерактивних вебплатформ для зниження рівня академічних труднощів у студентів-програмістів. Значний масив досліджень присвячено автоматизації оцінювання коду. Так С. Едвардс [63] представив систему Web-SAT, а Х. Пайва [64] та А. Фігейра [64] систематизували методи статичного та динамічного аналізу програмних завдань. Сучасний етап розвитку педагогічних підходів пов'язаний із генеративним ШІ, представлений роботами С. Гуна [65] та В. Сюя [65] в аспекті розвитку алгоритмічного мислення через промпт-інжиніринг, а також П. Лагакіса [66] та С. Деметріадіса [66], які вивчали використання великих мовних моделей для аудиту коду. Ефективність колаборативних форм навчання, зокрема розподіленого парного програмування, доведено С. Цай [67]. Концепції програмного та гнучкого оцінювання як альтернативи традиційним формам контролю розвинуто у працях Л. Бартман [68], К. Квінлан [68]. Етичні аспекти та сталість ШІ-орієнтованого оцінювання у вищій освіті стали предметом вивчення М. Альфалеха [69].

Проте аналіз сучасних досліджень засвідчує, що в освітній практиці переважає інструментальний підхід до використання цифрових технологій, за якого вебресурси застосовуються переважно як засоби розміщення навчальних матеріалів або організації комунікації. Питання ж системного використання веборієнтованих технологій як дидактичної основи навчання програмування, що інтегрує середовища розробки, системи контролю версій, хмарні обчислювальні платформи, автоматизоване тестування коду та інструменти колективної взаємодії, залишається недостатньо розробленим.

Особливої актуальності ця проблема набула в умовах масового переходу до дистанційної та змішаної форми навчання у 2020 році, коли ЗВО були змушені оперативно впроваджувати цифрові платформи навчання, зокрема Moodle, Google Classroom. Попри позитивний досвід забезпечення безперервності освітнього процесу, було виявлено низку системних недоліків: фрагментарність цифрового середовища, відсутність єдиних стандартів організації навчальної діяльності, обмежені можливості інтеграції спеціалізованих інструментів програмування в освітні платформи, труднощі об'єктивного оцінювання результатів практичної діяльності студентів.

Для системного формування професійних компетентностей у бакалаврів комп'ютерних наук, зокрема компетентності з програмування, в умовах цифрової трансформації доцільно враховувати положення міжнародних рамок цифрової компетентності, зокрема на DigComp 3.0, яка визначає п'ять взаємопов'язаних сфер [70].

До основних сфер цифрової діяльності належать: здатність ефективно працювати з інформацією та даними, що передбачає їх пошук, оцінювання, опрацювання та аналітичну інтерпретацію; уміння здійснювати комунікацію та взаємодію в цифровому середовищі, організовувати спільну діяльність і використовувати цифрові платформи для колективного вирішення завдань; створення та опрацювання цифрового контенту, що охоплює розроблення програмних продуктів, інтеграцію мультимедійних ресурсів і адаптацію цифрових матеріалів для освітнього процесу; забезпечення безпеки

інформаційних та цифрових систем, що включає кібербезпеку, захист персональних даних та безпечну роботу з хмарними сервісами; а також розвиток цифрової компетентності та здатність вирішувати проблеми шляхом адаптації до нових цифрових технологій, самонавчання та застосування інноваційних рішень у професійній діяльності.

Використання підходів рамки DigComp 3.0 в освітньому процесі дозволяє не лише стандартизувати очікувані результати, а й інтегрувати сучасні веборієнтовані середовища розробки програмного забезпечення, хмарні сервіси та освітні платформи, що забезпечують формування практичних навичок розробки програмного забезпечення, колективної роботи у навчальних проєктах та вирішення прикладних завдань, відповідно до потреб сучасної ІТ-галузі та вимог цифрової трансформації суспільства.

Таким чином, виникають *суперечності між*:

- сучасним розвитком цифрових технологій і недостатньою кількістю методичних досліджень щодо системності їх використання в закладах вищої освіти під час організації змішаної форми навчання;
- зростаючими вимогами ІТ-галузі до підготовки фахівців, здатних ефективно працювати у веборієнтованих середовищах розробки програмного забезпечення, та недостатньою теоретико-методичною розробленістю системного використання веборієнтованих технологій у процесі навчання програмування бакалаврів комп'ютерних наук;
- необхідністю об'єктивного моніторингу навчальних досягнень бакалаврів комп'ютерних наук та недостатнім обґрунтуванням методик формуального оцінювання на основі таксономії Блума та аналізу цифрового сліду.

Отже, виникає *наукова проблема*: теоретико-методичне обґрунтування використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук, що враховує методику формуального оцінювання на основі таксономії Блума та аналізу цифрового сліду, є до кінця

не розв'язаною й потребує науково-методичного обґрунтування та розроблення відповідної методики.

Вибір теми дослідження «Використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук» зумовлений тим, що підготовка бакалаврів комп'ютерних наук, здатних ефективно працювати у веборієнтованих середовищах розробки програмного забезпечення, має значну актуальність та практичну значущість, але бракує достатнього науково-методичного забезпечення освітнього процесу в закладах вищої освіти.

Актуальність теми дослідження зумовлена потребою наукового обґрунтування та розроблення педагогічної моделі використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук. Системна інтеграція веборієнтованих середовищ розробки програмного забезпечення, хмарних сервісів та освітніх платформ в освітній процес є основою формування компетентності з програмування майбутніх фахівців і підвищення практичної спрямованості їхньої підготовки відповідно до вимог сучасної ІТ-галузі та цифрової трансформації суспільства.

Мета дослідження полягає у тому, щоб теоретично обґрунтувати та розробити методику використання веборієнтованих технологій у процесі навчання програмування бакалаврів комп'ютерних наук, спрямовану на розвиток компетентності з програмування.

Об'єктом дослідження є процес підготовки майбутніх бакалаврів комп'ютерних наук у закладах вищої освіти.

Предметом дослідження є використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук.

Для досягнення мети було визначено *завдання*:

1. Здійснити аналіз понятійно-термінологічного апарату та узагальнити вітчизняний і зарубіжний досвід використання веборієнтованих технологій у закладах вищої освіти.

2. Теоретично обґрунтувати та розробити педагогічну модель використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук.

3. Розробити методику використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук.

4. Розробити систему формувального оцінювання навчальних досягнень студентів та впровадити авторський веборієнтований застосунок для автоматизованого оцінювання виконання комп'ютерних практикумів на основі таксономії Блума.

5. Експериментально перевірити ефективність розробленої методики та системи оцінювання навчальних досягнень студентів закладів вищої освіти у процесі використання веборієнтованих технологій.

Зв'язок роботи з науковими програмами, планами, темами. У дисертації наведено результати досліджень, одержаних під час виконання наукових тем Інституту цифровізації освіти НАПН України «Методологія використання хмаро орієнтованих систем відкритої науки у закладах освіти» (ДР 0121U107673, 2021–2023 рр.), «Проектування і використання відкритого освітнього середовища з елементами штучного інтелекту для професійного розвитку педагогічних кадрів» (ДР № 0124U000671, 2024–2025 рр.).

Тему дисертаційної роботи було затверджено на вченій раді Інституту цифровізації освіти НАПН України 30 листопада 2017 р. (протокол № 11) та узгоджено Міжвідомчою радою з координації досліджень у галузі освіти, педагогіки і психології при НАПН України 30 січня 2018 р. (протокол № 1).

Методи дослідження. Для розв'язання поставлених завдань у дослідженні використано теоретичні, емпіричні та статистичні методи.

Зокрема, здійснено аналіз психолого-педагогічних теорій і концепцій, що стосуються предмета дослідження; узагальнено вітчизняний і зарубіжний досвід організації навчання у закладах вищої освіти на засадах використання веборієнтованих технологій; виконано проектування та моделювання процесу навчання програмування майбутніх бакалаврів комп'ютерних наук із

використанням веборієнтованих технологій в умовах дистанційного та змішаного навчання; проведено аналіз законодавчої та нормативної документації з питань розвитку вищої освіти й упровадження дистанційної та змішаної форм навчання у практику діяльності закладів вищої освіти. Також здійснено систематизацію та узагальнення теоретичних і експериментальних даних.

Для підтвердження отриманих результатів і теоретичних висновків проведено емпіричне дослідження. Для перевірки результатів дослідження використано методи науково-педагогічного експерименту, а також авторський веборієнтований застосунок для автоматизованого аналізу результатів експериментальної та контрольної груп. Статистичний аналіз даних дав змогу підтвердити достовірність отриманих результатів і основну гіпотезу дослідження.

Наукова новизна і теоретичне значення одержаних результатів. У роботі представлено низку інноваційних положень, які становлять її наукову новизну, а саме *вперше*:

- розроблено й обґрунтовано авторську модель навчання програмування як інтегровану дидактичну систему, що базується на принципах неперервності та інтелектуальної співпраці у хмаро орієнтованому середовищі розробки;

- розроблено й обґрунтовано авторську модель і теоретично обґрунтовано методику використання веборієнтованих технологій, спрямовану на формування цифрової професійної ідентичності бакалавра, що базується на принципах неперервності та інтелектуальної співпраці у хмаро орієнтованому середовищі розробки;

- обґрунтовано перехід до хмаро орієнтованої парадигми навчання, де веббраузер стає єдиним освітнім вікном навчання програмування (через GitHub Codespaces, Replit), що усуває технічні перешкоди локального налаштування програмного забезпечення;

- розроблено й обґрунтовано авторську пропорцію змішаного навчання для ІТ-спеціальностей – 30% традиційного, 50% дистанційного (хмаро орієнтованого) та 20% проєктного навчання, що моделює реальний життєвий цикл розробки програмного забезпечення (Software Development Life Cycle, SDLC);

- обґрунтовано підходи контролю, де основним об’єктом оцінювання є не фінальний файл, а історія фіксацій змін у GitHub, що дозволяє об’єктивно підтвердити академічну доброчесність та оцінити ітеративність розробки.

Доведено доцільність вживання терміна «веборієнтовані технології» як найбільш точного для ІТ-фахівців, оскільки він вказує на середовище реалізації (HTTP, DOM, браузер), на відміну від занадто широкого терміна «цифрові технології»;

Удосконалено: *підходи* до структурування великих обсягів навчальних відомостей за допомогою веборієнтованих інтелект-карт, що дозволяє візуалізувати складні алгоритмічні концепції; *підходи* до формування компетентності з програмування шляхом інтеграції ІІІ як «інтелектуального помічника», а не просто засобу генерування коду та пошуку інформації.

Встановлено, що в умовах системної цифрової трансформації (2020–2026 рр.) компетентність з програмування трансформується з вузькотехнічної навички в основну цифрову компетентність фахівця.

Дістали подальшого розвитку класифікація веборієнтованих технологій, доповнена еволюційними характеристиками Веб 4.0 (AI-агенти, Symbiotic Web) у контексті підготовки програмістів та підходи до організації освітнього процесу в закладах вищої освіти на засадах використання ІКТ-технологій.

Практичне значення отриманих результатів дослідження полягає в тому, що:

- розроблено методику використання веборієнтованих технологій у навчанні програмування бакалаврів комп’ютерних наук в умовах змішаного навчання;

- розроблено авторську систему оцінювання, де рівні інтелектуальної діяльності за таксономією Блума (6 рівнів) вперше корелюють із рівнями цифрової компетентності DigComp 3.0 (від 1–3 базового до 7–8 експертного);
- розроблено та застосовано спеціалізований авторський веборієнтований застосунок «E-student» для автоматизованого формувального оцінювання навчальних досягнень студентів на основі рівнів таксономії Блума;
- практикум для студентів із системою завдань, побудованих на основі таксономії Блума «Веборієнтовані технології. Frontend-розробка».

Дослідження здійснювалося в три етапи протягом 2018–2026 рр.

Під час I етапу (2018–2020 рр.) було окреслено ключові складники дисертаційного дослідження, а саме: визначено мету, об'єкт, предмет і завдання та обрано базу для проведення експерименту. Проведено ґрунтовний аналіз наукових джерел, сучасних педагогічних підходів і теоретичних положень, відповідно до теми дослідження. Окрему увагу приділено вивченню використання веборієнтованих технологій у вивченні програмування, організації підготовки бакалаврів комп'ютерних наук, також формуванню їх компетентностей з програмування. В результаті було обґрунтовано актуальність роботи, сформульовано гіпотезу, визначено методичний інструментарій для збору та опрацювання даних, а також розроблено етапи реалізації експерименту, та ресурсне забезпечення.

На II етапі (2021–2023 рр.) було розроблено та теоретично обґрунтовано структуру двох моделей (моделі навчання програмування бакалаврів комп'ютерних наук на засадах веборієнтованих технологій та моделі використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук); розроблено й обґрунтовано методику використання веборієнтованих технологій; розроблено й обґрунтовано авторську пропорцію змішаного навчання для ІТ-спеціальностей 30% традиційного, 50% дистанційного та 20% проєктного навчання, що моделює реальний життєвий цикл розробки ПЗ; розроблено й обґрунтовано авторську

систему оцінювання на основі таксономії Блума згідно з рівнями цифрової компетентності DigComp 3.0.

На III етапі (2024–2026 рр.) розроблено та застосовано спеціалізований авторський веборієнтований застосунок інформаційну систему «E-student» для автоматизованого формувального оцінювання навчальних досягнень студентів на основі рівнів таксономії Блума; здійснено практичну перевірку ефективності запропонованої авторської методики. Здійснено впровадження основних результатів дисертаційного дослідження в закладах вищої освіти.

Результати дослідження впроваджено в педагогічну практику викладачів ЗВО, що підтверджується довідками: Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» (№14 від 18.03.2026), Міжнародного європейського університету (№ 129-03/26 від 20.03.26), Запорізького національного університету (№ 9 від 10.03.26), Національного університету «Запорізька політехніка» (№ 11 від 11.02.26), Київського столичного університету імені Бориса Грінченка (№ 7 від 21.04.25),

Особистий внесок здобувача. У публікаціях, підготовлених у співавторстві, внесок здобувача полягає у наступному: *розроблено інструментарій моніторингу використання веборієнтованих технологій, проведено аналіз та узагальнення отриманих результатів [17]; обґрунтовано методичні засади використання веборієнтованих технологій у навчанні вебпрограмування та проаналізовано результативність їх застосування [18]; здійснено аналіз закордонного, зокрема американського досвіду підготовки бакалаврів комп'ютерних наук та визначено можливості його використання у вітчизняній системі вищої освіти [20]; проаналізовано сучасний стан підготовки фахівців з інформаційних технологій, визначено основні проблеми та перспективи розвитку системи професійної підготовки [21]; проаналізовано можливості використання засобів III в навчанні програмування та визначено напрями їх застосування під час формування компетентності з програмування студентів [22]; визначено структуру професійної компетентності, зокрема*

компетентності з програмування майбутніх бакалаврів комп'ютерних наук, та обґрунтовано педагогічні умови її формування [24]; проведено аналіз та систематизацію компетентностей майбутніх бакалаврів комп'ютерних наук відповідно до міжнародних та національних освітніх стандартів [25]; визначено та охарактеризовано підходи до веборієнтованого навчання майбутніх бакалаврів комп'ютерних наук, обґрунтовано організаційно-педагогічні особливості їх реалізації в закладах вищої освіти [32]; розроблено критерії та показники оцінювання навчальних досягнень здобувачів вищої освіти, проведено аналіз та інтерпретацію результатів педагогічного оцінювання [37]; узагальнено досвід використання засобів дистанційного навчання у ЗВО України, проведено аналіз цифрових платформ та інструментів організації освітнього процесу в умовах карантинних обмежень [38]; обґрунтовано доцільність використання мобільного навчання як додаткового засобу організації освітнього процесу та проведено аналіз його впливу на результати навчання студентів [39]; досліджено організацію самостійної роботи майбутнім бакалаврів комп'ютерних наук та визначено педагогічні умови підвищення її ефективності [40]; визначено підходи до оцінювання рівня знань бакалаврів комп'ютерних наук та проаналізовано ефективність використання сучасних засобів контролю результатів навчання [41]; здійснено аналіз психолого-педагогічних аспектів використання цифрових технологій в освітньому процесі [112]; досліджено рівень використання цифрових технологій у РДІ в умовах дистанційного та змішаного навчання, проведено аналіз результатів опитування учасників освітнього процесу [135];

Апробація результатів дослідження. Основні теоретичні й практичні результати, а також концептуальні ідеї та узагальнені висновки дисертаційного дослідження було оприлюднено у формі доповідей та тез на: міжнародних конференціях: 14-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Київ, 2018 р.); міжнародній науково-практичній конференції «Змішане навчання — інновація XXI сторіччя» (Харків, 2018 р.); 19-й міжнародній конференції «Workshop on Cloud Technologies in Education»,

СТЕ (Кривий Ріг, 2019 р.); 16-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Київ, 2020 р.); міжнародній науково-практичній конференції «Сучасні тенденції та фактори розвитку педагогічних та психологічних наук в Україні та країнах ЄС», (Люблін, Польща, 2020 р.); 17-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Херсон, 2021 р.); 19-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Львів, 2024 р.); 14-й міжнародній науково-практичній конференції з інформаційних систем та технологій «Infocom Advanced Solutions 2026» (Київ, 2026 р.); всеукраїнських конференціях: Всеукраїнській науково-практичній конференції молодих вчених «Наукова молодь» (Київ, 2017–2020 рр.); IV Всеукраїнській науково-практичній конференції молодих вчених та студентів «Інформаційні системи та технології управління (Київ, 2020 р.); XV всеукраїнській науково-практичній конференції з міжнародною участю «Неперервна освіта: актуальні дискусії» (Ужгород, 2021 р.); звітній науковій конференції Інституту цифровізації освіти НАПН України, (Київ, 2018–2023 рр.); семінарах: міжнародному семінарі «Ключові питання шкільної освіти ЄС», у межах Модуля Жана Моне «Україна – ЄС: крос-культурні порівняння в освітніх дослідженнях» (Київ, 2018 р.); всеукраїнському методологічному семінарі «Інформаційно-комунікаційні технології в освіті та наукових дослідженнях» (Київ, 2018 р.); всеукраїнському науково-методологічному семінарі «Система навчання і освіти в комп'ютерно орієнтованому середовищі» (Київ, 2018 -2026 рр.); всеукраїнському науково-практичному семінарі «Цифрова компетентність сучасного вчителя нової української школи» (Київ, 2018 р.); міжнародному семінарі «Вступ до обчислювального мислення», за участю професора Університету Дукесне Джозефа Куша (США, 2019 р.); міжнародному семінарі «Інтеграція штучного інтелекту в освіту – виклики та можливості» від провідних британських професорів (Київ, 2024 р.); у роботі оргкомітету по підготовці та проведенню XI Всеукраїнської олімпіади з інформатики та комп'ютерної техніки серед студентів закладів вищої освіти I-II рівнів акредитації (Ужгород, 2018 р.); під

час професійного стажування в компанії EPAM «IT Ukraine Association Teacher's Internship 2024 held by EPAM» (Київ, 2024 р.).

Публікації. Основні результати дослідження відображено у 24 працях, серед них 5 статей, що індексуються наукометричними базами Scopus, 5 статей у наукових фахових виданнях України категорії «Б», 12 тез доповідей у матеріалах конференцій, 1 стаття в монографії (українською мовою), 1 розділ в монографії (англійською мовою).

Структура й обсяг дисертації. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків до кожного розділу, висновків, списку використаних джерел (380 найменувань, з них 204 іноземною мовою), 12 додатків. Загальний обсяг дисертації – 419 сторінки, з них 263 сторінок основного тексту.

РОЗДІЛ 1. ТЕОРЕТИКО-МЕТОДИЧНІ ОСНОВИ НАВЧАННЯ ПРОГРАМУВАННЯ МАЙБУТНІХ БАКАЛАВРІВ КОМП'ЮТЕРНИХ НАУК

1.1. Аналіз поняттєво-термінологічного апарату проблеми використання веборієнтованих технологій в освітньому процесі

Сучасний етап розвитку вищої технічної освіти характеризується системною цифровою трансформацією, що вимагає від майбутніх бакалаврів комп'ютерних наук не лише опанування синтаксису мов програмування, а й здатності ефективно функціонувати в інтелектуалізованих цифрових екосистемах. У контексті підготовки ІТ-фахівців поняттєво-термінологічний апарат дослідження виходить за межі інструментального опису засобів і зміщується в площину формування синтетичного навчального середовища, де веборієнтовані технології виступають фундаментом професійної діяльності.

Розвиток веборієнтованих технологій зумовлює розширення можливостей та особливостей інформаційної взаємодії та комунікації в мережі Інтернет. Основою веборієнтованої технології є гіпертекстова інформаційна система типу «клієнт-сервер» [71]. Організація освітнього процесу із застосуванням веборієнтованих технологій дає змогу значно розширити можливості навчання та підвищити ефективність підготовки здобувачів освіти. Особливо важливим є їх використання у процесі вивчення дисциплін професійної підготовки майбутнім бакалаврів комп'ютерних наук, оскільки такий підхід передбачає застосування різноманітних форм подання навчального матеріалу з програмування за допомогою електронних освітніх ресурсів засобів навчального призначення [71].

Своєю чергою, інформатизація освіти стає головною умовою, що впливає на подальший успішний розвиток як інтелектуального рівня студентів, так і потенціалу нації і суспільства загалом [17]. Так, науковці В. Биков [7], О. Буров [7], констатують, що інформатизація освіти вимагає впровадження у вищу освіту інноваційних методів, засобів та форм професійної підготовки

майбутніх фахівців нової формації, де основою цього процесу є впровадження та використання сучасних інформаційно-комунікаційних технологій (ІКТ), зокрема веборієнтованих технологій [7], [17].

В освіті, теоретичне та практичне використання веборієнтованих технологій висвітлено у наукових працях С. Литвиної [17] В. Бикова [7], Н. Сороко [42], В. Круглика [43], В. Котяка [72], Н. Труонга (N. Truong) [62], В. Осадчого [15], [16], К. Осадчої [16], О. Спіріна [11], Т. Вакалюк [11] та інших вітчизняних та зарубіжних дослідників [11].

Так, науковець С. Литвинова зазначає, що веборієнтовані технології – це сукупність методів та програмно-технічних засобів, інтегрованих з метою ефективного опрацювання вебресурсів, що функціонують у вебпросторі (локальному або глобальному, зокрема в мережі Інтернет) [17], і наголошує, що застосування їх в освітньому процесі є важливим для підвищення рівня підготовки майбутніх бакалаврів ІТ-спеціальностей, зокрема бакалаврів комп'ютерних наук [18]

Н. Сороко [42], аналізуючи праці багатьох зарубіжних авторів, зазначає, що вебтехнологія – це поняття, яке використовується для позначення сукупності технологій і сервісів Інтернету. Дослідниця наголошує, що вебтехнології є методикою проєктування систем, які завдяки мережевій взаємодії удосконалюються зі збільшення кількості їх користувачів [17], [42].

У своїх наукових дослідженнях В. Круглик розглядає веборієнтовані технології як технології, у яких на рівні архітектури системи виокремлюються: частина, що забезпечує функціональність системи (backend) і частина, що забезпечує взаємодію з користувачем (frontend). Такий поділ зумовлений розподіленістю вебсистеми між серверною та клієнтською складовими [17], [43].

Завдяки використанню веборієнтованих технологій В. Круглик [43] досліджує формування веборієнтованих навчальних середовищ у професійній підготовці майбутніх інженерів–програмістів, а дослідник В. Котяк [72]

аналізує веборієнтовані системи тестування навчальних досягнень студентів [17].

У наукових дослідженнях Н. Труонг (N. Truong) [62] обґрунтовує використання веборієнтованих навчальних середовищ як ефективного засобу навчання студентів програмуванню. Автор пропонує низку заходів, спрямованих на покращення адаптації студентів до умов роботи у стандартних середовищах програмування, а саме [17], [62]:

- встановлення та налаштування середовища програмування;
- використання редактора середовища програмування;
- розуміння питань, пов'язаних з програмуванням, і застосування знань синтаксису мови програмування до написання програмного коду;
- розуміння помилок компіляції (інтерпретації);
- налагодження [62].

У своїх дослідженнях науковці В. Осадчий і К. Осадча розглядають веборієнтовані технології як важливий інструмент модернізації освітнього процесу. Автори зазначають, що використання таких технологій дає змогу створювати навчальні програми, доступ до яких здійснюється через мережу Інтернет без необхідності встановлення спеціального ПЗ на локальні пристрої користувачів. Водночас забезпечується інтерактивна взаємодія між здобувачами освіти, викладачами та навчальним контентом. Це сприяє підвищенню ефективності навчання, розширенню можливостей індивідуалізації освітнього процесу та розвитку дистанційної й змішаної форми навчання [16].

У працях учених О. Спіріна [11], Т. Вакалюк [11], обґрунтовано використання веборієнтованих технологій для навчання основ програмування майбутніх учителів інформатики. Науковці зазначають, що такий підхід є педагогічно доцільним, оскільки забезпечує організацію різних видів навчальної діяльності в цифровому освітньому середовищі. Зокрема, такі технології дають змогу проводити лекційні заняття, організовувати комп'ютерні практикуми, здійснювати тестування, виконувати модульні

контрольні роботи, а також проводити заліки й практичні частини іспитів, що сприяє підвищенню якості підготовки майбутніх фахівців [11].

У межах цього дослідження під веборієнтованими технологіями ми розуміємо такі технології, складниками яких є сукупність мов програмування, протоколів, фреймворків, сервісів та інструментів, що забезпечують створення і функціонування веборієнтованих застосунків, які переважно базуються на архітектурі «клієнт–сервер» та взаємодіють через мережу Інтернет (зокрема через веббраузер).

Специфіка використання веборієнтованих технологій у навчанні програмування полягає у забезпеченні неперервності освітнього процесу. За такого підходу інтегровані середовища розробки (ICP), системи контролю версій (Git) та комунікаційні платформи інтегруються у єдине веборієнтоване середовище, доступ до якого здійснюється через браузер. Це дає змогу усунути бар'єри, пов'язані з локальним налаштуванням ПЗ та зосередити увагу студентів безпосередньо на процесі розробки програмного коду.

Крім того, веборієнтовані технології забезпечують взаємодію між користувачем і сервером, підтримують створення динамічних вебзастосунків і сервісів, а також реалізують механізми обміну даними, відображення інформації та віддаленого доступу до навчальних ресурсів.

Основне призначення веборієнтованих технологій полягає у забезпеченні взаємодії користувача з програмним забезпеченням через веббраузер, без необхідності встановлення локальних програм на комп'ютері користувача. Такий підхід дає змогу отримувати доступ до програмних ресурсів і сервісів через мережу Інтернет, забезпечуючи їх використання незалежно від місця перебування користувача та характеристик його пристрою.

Для систематизації знань про розвиток об'єкта дослідження нами було вдосконалено класифікацію технологій, об'єднавши їх еволюційні характеристики з дидактичними можливостями в аспекті підготовки майбутніх програмістів (табл. 1.1).

Аналізуючи досвід своїх зарубіжних колег, Д. Гарріс (D. Harris) розглядає поняття Веб 1.0 як «інтернет лише для читання», що надає користувачам обмежені можливості для взаємодії та участі у наповненні Інтернету новими ресурсами [54].

Таблиця 1.1

Еволюція веборієнтованих технологій у контексті трансформації навчання програмування

Покоління Веб	Технологічний стек	Роль студента та педагогічна доцільність	Специфіка для навчання програмування
Веб 1.0 (1990–2004)	HTML, CSS, CGI, FTP	Пасивний споживач. Навчання за зразком, читання документації	Локальне написання коду; Веб як статичний довідник (напр., ранній MIT OpenCourseWare)
Веб 2.0 (2004–2015)	JS, AJAX, REST API, Git, LMS	Активний розробник. Соціальне навчання, колективна співпраця та обмін контентом	Використання GitHub для роботи в команді; поява перших Cloud IDE (CodePen, JSFiddle)
Веб 3.0 (2015–2024)	Semantic Web, AI/ML, Blockchain	Суб'єкт персоналізації. Адаптивність навчання на основі інтелектуального опрацювання даних	AI-підказки в коді; автоматизовані системи тестування алгоритмів (LeetCode, HackerRank)
Веб 4.0 (з 2024)	AI-агенти, Symbiotic Web, XR	Інтелектуальний партнер. Взаємодія людини та ШІ (AI Pair Programming)	Вседоступне середовище розробки; ШІ як когнітивний тьютор та дебагер

Перші вебзастосунки університетів, зокрема Массачусетського технологічного університету та Стенфордського університету, містили переважно статичну інформацію, а студенти могли лише переглядати її без

можливості надання зворотного зв'язку. Для створення вебсторінок використовувались такі веборієнтовані технології як HTML 3.2, CSS, FTP, простий дизайн і таблична верстка.

Веб 2.0 є важливим етапом в еволюції вебтехнологій, і спрямований на перехід від статичного представлення інформації до інтерактивних, динамічних і орієнтованих на користувача сервісів.

Термін Веб 2.0 вперше популяризував Т. О'Рейлі (Т. О'Reilly) у 2004 році, підкреслюючи зміну підходів до створення та використання вебзастосунків, за якої користувачі стають співавторами контенту, а платформи – інфраструктурою для розвитку послуг [55]. Веб 2.0 розглядається як нова парадигма розвитку програмного забезпечення та інтернет-сервісів.

Згідно з підходом Т. О'Рейлі (Т. О'Reilly) [55], Веб 2.0 слід розуміти як мережу, що функціонує як платформа та об'єднує різноманітні пристрої, забезпечуючи надання програмного забезпечення у формі постійно оновлюваного сервісу. Важливою характеристикою таких систем є їх здатність до вдосконалення зі зростанням кількості користувачів, що забезпечується через використання колективного досвіду та даних [55].

Однією з ключових відмінностей Веб 2.0 від попереднього етапу розвитку вебу (Веб 1.0) є зміна ролі користувача. Якщо у Веб 1.0 користувач виступав переважно як споживач інформації, то у Веб 2.0 він активно бере участь у процесі створення контенту та функціонування сервісів. Це проявляється у розвитку таких явищ, як блогінг, соціальні мережі, вікітехнології та платформи спільного редагування [55].

Важливим принципом Веб 2.0 є:

- *використання колективного інтелекту*, що полягає у залученні великої кількості користувачів до створення, оцінювання та структурування інформації;
- орієнтація на «довгий хвіст» (*long tail*), що передбачає залучення великої кількості малих учасників замість концентрації лише на великих гравцях;

- перехід до моделі «*програмне забезпечення як послуга*», де програмні продукти не розповсюджуються як завершені рішення, а надаються через мережу з постійним оновленням і вдосконаленням;
- *дані як стратегічний ресурс* – контроль над унікальними наборами даних, що постійно збагачуються завдяки активності користувачів, стає ключовою конкурентною перевагою компаній [55];
- використання *легких моделей програмування*, таких як REST, RSS та AJAX, що забезпечують інтеграцію різних сервісів і створення нових застосунків шляхом їх комбінування, а також можливість повторного використання вже існуючих компонентів;
- створення *багатого користувацького інтерфейсу*, що наближає вебзастосунки до рівня функціональності настільних програм і забезпечує підвищення зручності використання та розширення можливостей взаємодії користувача з системою [55];

Таким чином, Веб 2.0 можна розглядати як комплексну концепцію, що поєднує технологічні, соціальні та економічні аспекти розвитку вебсередовища. Її основними характеристиками є орієнтація на користувача, використання колективного інтелекту, домінування сервісної моделі, а також зростаюча роль даних і мережевих ефектів.

В освіті ці технології дали змогу викладачам і студентам створювати та редагувати документи в реальному часі, наприклад, брати участь у створенні одного документа кількома користувачами (Google Docs), активно користуватися електронною поштою, офісними пакетами [55].

У цей період, поряд з HTML та CSS, застосовувалися такі складові веборієнтованих технологій, як JavaScript, jQuery, AJAX; CMS (Content Management Systems), REST API, JSON, XML, відеоплатформи, чати, форуми тощо.

Веб 2.0 став поштовхом до розвитку онлайн-освіти (e-Learning), зробивши її доступнішою для широкої аудиторії. З'явилися системи управління навчанням (LMS, Learning Management Systems), що дозволяють викладачам та

студентам ефективно взаємодіяти (Moodle, Google Classroom). Тепер викладачі та студенти можуть активно спілкуватися між собою засобами Інтернету, створювати власну інформацію (тести, блоги, відео, презентації). З'явилися інструменти для інтерактивної співпраці – онлайн-тести, форуми, відеочати, що покращило якість освітнього процесу.

Освіта стала більш демократичною, інтерактивною та доступною, оскільки інформація та ресурси стали відкритими і доступними для будь-кого. Оскільки на технологічній платформі Веб 2.0 з'явилося багато одноманітних ресурсів, що, відповідно, девальвує цінність більшості з них, прийшла третя культурна версія Веб 3.0. Вона дозволила створити високоякісний контент і сервіси на базі платформи Веб 3.0, рецензувати і відбирати цікавий і корисний контент [17], [42].

На основі аналізу робіт закордонних авторів, Н. Сороко [42] зазначає, що виникнення Веб 3.0 пов'язане з появою так званих «хмарних обчислень» (англ. cloud computing), які характеризують підтримку формування і використання мережних віртуальних майданчиків, що є ситуаційною складовою логічної мережевої інфраструктури інформаційно-комунікаційної мережі з тимчасовою відкритою гнучкою архітектурою, яка за своєю будовою і часом існування відповідає персоніфікованим потребам користувача (індивідуальним і груповим ресурсам) [17], [42].

Якщо попередній етап Веб 2.0 зосереджувався на взаємодії користувача з контентом, то Веб 3.0 робить акцент на розумінні контенту машинами, інтелектуальній обробці даних і персоналізації досвіду користувача, забезпечує глибшу семантичну взаємодію, інтеграцію з фізичними пристроями (IoT) та забезпечення безпеки за допомогою блокчейну, використанні штучного інтелекту, інтелектуальну підтримку (чат-боти, віртуальні репетитори, автоматичне оцінювання).

Зазначимо, що використання таких програмних продуктів як, наприклад, Coursera, Udacity, LinkedIn Learning дає змогу адаптувати освітній процес за допомогою рекомендацій на основі ШІ. ChatGPT, Grammarly використовується

як інтелектуальні помічники у навчанні. Мобільні застосунки з машинним навчанням (ML, Machine Learning) забезпечують персоналізоване вивчення мов програмування і природничо-математичних дисциплін. Також застосовуються цифрові сертифікати з надійним захистом від підроблення (Blockchain).

Як зазначає Н. Сороко, термін Веб 3.0 був запропонований Дж. Калаканісом (J. Calacanis) [56]. Він трактує це поняття як високоякісний контент і сервіси, які створюються професіоналами на базі технологічної платформи Веб 2.0. Його пояснення виникнення Веб 3.0 полягає в тому, що оскільки Веб 2.0 – технологічна платформа, яка дозволила на її основі створювати низку сервісів, з'явилося багато одноманітних ресурсів, що, відповідно, девальвує цінність більшості з них. Тому на зміну технологічній платформі Веб 2.0 мала прийти концепція третя культурна версія Веб, яка дасть змогу рецензувати і відбирати цікавий і корисний контент [42].

Закордонні дослідники, зокрема Дж. Поллок (J. Pollock) [57], [58] та М. Ватсон (M. Watson) [59], [60] пов'язують еволюцію Веб 3.0 із концепцією хмарних обчислень як гнучкою архітектурою, що відповідає персоналізованим потребам користувача. Це корелює з нашим баченням персоналізованих освітніх траєкторій, де веборієнтовані технології забезпечують студенту-програмісту індивідуальний темп та глибину занурення в навчальний матеріал .

Головна ідея Веб 3.0 полягає в тому, що користувач виступає модератором контенту, який публікується в мережі. Веб 3.0 передбачає появу вузькоспеціалізованих ресурсів, де агрегуватимуться всі необхідні користувачеві сервіси та інструменти професійної соціальної складової, а публікація контенту модеруватиметься експертами [42].

Більшість дослідників Веб 3.0 називають семантичним (Semantic Web). Семантична павутина – це еволюційний етап розвитку мережі Інтернет, метою якого є реалізація можливості машинної обробки електронних ресурсів, що доступні у всесвітній павутині. При цьому увага зосереджується на роботі з метаданими [73].

На нашу думку, веборієнтовані технології періоду Веб 3.0 – це новітні веборієнтовані технології, які дали змогу поєднати ІІІ, семантичні технології та розподілені системи для створення інтелектуальної, персоналізованої та більш безпечної мережі.

Серед веборієнтованих технологій можна виокремити:

- *Semantic Web: RDF, OWL, SPARQL*. Мови для опису, пошуку та взаємодії з даними.
- *Штучний інтелект і машинне навчання (AI/ML)*. Алгоритми, які навчаються та покращують свою роботу з часом.
- *Big Data. Технології оброблення великих обсягів даних для прийняття рішень і формування рекомендацій*.
- *Blockchain і IPFS*. Розподілені системи для безпечного зберігання та передачі інформації.
- *JSON-LD, Knowledge Graphs*. Структуровані дані для кращої організації знань.

Приклади:

- *Google Knowledge Graph*. Система, що дозволяє пошуковику краще розуміти запити і контекст.
- *Голосові помічники: Siri, Alexa*.
- *ChatGPT*. Приклад ІІІ, що генерує тексти і відповідає на запитання.
- *Онлайн-курси з адаптивним навчанням (LinkedIn Learning, Coursera, Udemy)*.
- Використання блокчейну для сертифікації в освіті.

Веборієнтовані технології періоду Веб 4.0 – це веборієнтовані технології, які передбачають глибоку інтеграцію ІІІ, автоматизацію, Інтернет речей (IoT), віртуальну та доповнену реальність, а також створення персоналізованого, інтуїтивного та людиноцентричного цифрового досвіду. Його ще називають «Симбіотичним вебом», адже взаємодія між людиною і машиною стає максимальною гармонійною.

До характеристик веборієнтованих технології періоду Веб 4.0 можна віднести:

- *інтелектуальну взаємодію*, за якої Веб 4.0 використовує *штучний інтелект*, щоб розуміти наміри, контекст та емоції користувача. Системи можуть самостійно приймати рішення та навчатися з досвіду;
- *емоційну чутливість (емоційний інтернет)* – це розпізнавання емоцій користувача через голос, обличчя, мову тіла. Адаптація контенту відповідно до емоційного стану;
- *гармонійну взаємодію людини і машини*:- вебсервіси працюють як асистенти людини, а не лише як інструменти. Машини «відчувають», коли і як найкраще допомогти;
- *доповнену та віртуальну реальності (AR/VR)*, що забезпечує глибоке занурення в цифрове середовище (освіта, медицина, шопінг, туризм). Метавсвіт як новий рівень взаємодії;
- *гіперперсоналізацію*: Веб «знає» користувача його інтереси, звички, контекст. Кожен бачить унікальний контент, повністю адаптований до нього;
- *Інтернет речей (IoT)*: усі пристрої навколо людини взаємодіють між собою (розумний дім, авто, гаджети). Дані з'єднуються у єдину цифрову систему;
- *автономні цифрові агенти*. Віртуальні помічники здатні діяти самостійно (бронювати, відповідати, нагадувати). Агенти спілкуються не лише з людьми, а й між собою;
- *високу безпеку та етичність*. Використання блокчейну, децентралізації, шифрування для захисту даних. Прозорість у використанні даних, етичні стандарти ІІІ;
- *контекстуальність*. Веб розуміє ситуацію користувача: місце, час, потребу, оточення. Видає саме ту інформацію, яка потрібна саме зараз;
- *інтеграцію з метавсесвітами*. Цифрові простори з новими формами спілкування, роботи та розваг. Створення віртуального «я» – цифрового аватара.

У таблиці 1.2 наведено веборієнтовані технології, а саме перелік програм із ШІ, AR/VR, IoT, хмарними сервісами та блокчейном, які Веб 4.0 активно використовує, щоб створити інтелектуальне, персоналізоване й глобальне освітнє та інформаційне середовище.

Таблиця 1.2

Веборієнтовані технології періоду Веб 4.0

Програма / Сервіс	Призначення
<i>AI та машинне навчання</i>	
ChatGPT (OpenAI)	Розумний помічник, навчання, генерація тексту
Google Bard / Gemini	Відповіді на запити, генерація ідей
Grammarly AI	Перевірка та редагування текстів
Synthesia	Створення відео з віртуальними аватарами
Socratic (Google)	AI-допомога учням у навчанні
<i>Доповнена та віртуальна реальність (AR/VR)</i>	
Mozilla Hubs	Віртуальні кімнати для навчання/зустрічей
Google Expeditions	Освітні AR/VR-тури
Unity / Unreal Engine	Розробка інтерактивних 3D-сцен для навчання
Tilt Brush (VR)	Малювання у 3D-просторі
CoSpaces EDU	Створення віртуальних світів для навчання
<i>Хмарні платформи та інструменти</i>	
Google Workspace	Онлайн-освіта, спільна робота, зберігання
Microsoft 365 / Teams	Онлайн-заняття, документи, інтеграція AI
Zoom / Meet / Webex	Віртуальні заняття з AI-функціями
Canva + AI	Дизайн з підтримкою штучного інтелекту
Notion AI	Інтелектуальні замітки, організація знань
<i>Big Data / Аналітика / IoT</i>	
Power BI / Tableau	Інтерактивна аналітика даних
ThingSpeak	IoT-платформа для збору й візуалізації даних
IFTTT / Zapier	Автоматизація між IoT і веб-сервісами
Arduino Web Editor	Онлайн-програмування для IoT-проектів
Google Data Studio	Безкоштовна аналітика даних з інтеграцією в хмари

Програма / Сервіс	Призначення
<i>Блокчейн та безпека</i>	
Ethereum / MetaMask	Веб3-ідентифікація, токени знань
Blockcerts	Видача та перевірка цифрових сертифікатів
OpenCerts	Платформа верифікації дипломів
Storj / IPFS	Децентралізоване зберігання даних

Веб 4.0 радикально трансформує освітній процес кожного ЗВО, роблячи його інтерактивним, персоналізованим, розумним і емоційно чутливим. Передусім ідеться про персоналізоване навчання студентів за допомогою ІШ, який аналізує рівень знань, стиль мислення, інтереси студента і формує його індивідуальну навчальну траєкторію. Платформи підлаштовують темп, формат та складність завдань під конкретного студента.

Крім того, ІШ здатний не лише оцінювати відповіді, а й пояснювати помилки, пропонувати поліпшення. Віртуальні наставники/асистенти надають миттєві поради, пояснення, перевірку завдань. ІШ перекладає матеріали на різні мови, долаючи мовні бар'єри. Навчання доступне звідусіль: навіть віддалені регіони можуть отримати освітні ресурси світового рівня. Узагальнену таблицю веборієнтованих технологій наведено в Додатку А.

Узагальнюючи результати аналізу поняттєво-термінологічного апарату, можна констатувати, що системне використання веборієнтованих технологій в освітньому процесі забезпечує якісну трансформацію форм, методів та засобів підготовки майбутніх бакалаврів комп'ютерних наук. На відміну від суто інструментального використання цифрових засобів, веборієнтований підхід дає змогу створити синтетичне навчальне середовище, що базується на архітектурі «клієнт-сервер» та забезпечує єдність освітнього й професійного інструментарію розробника.

На основі аналізу функціонального призначення веборієнтованих технологій ми пропонуємо типологію технологій, які є критично важливими саме для підготовки бакалаврів комп'ютерних наук (табл. 1.3).

Грунтовний аналіз термінологічного апарату в контексті навчання програмування дає змогу стверджувати, що термін «веборієнтовані технології» є найбільш точним і методологічно обґрунтованим з таких причин:

Таблиця 1.3

Функціональна класифікація веборієнтованих технологій у навчанні програмування

Категорія технологій	Типові інструменти та середовища	Дидактичне призначення у розрізі DigComp 3.0
Frontend-технології	HTML5, CSS3, JS, React	Створення цифрового контенту; формування UX/UI компетентності
Backend технології	Node.js, Python, REST API	Розв'язання проблем; розуміння архітектури розподілених клієнт-серверних систем
Data	SQL, PostgreSQL, MongoDB,	Робота з даними, інформаційна грамотність, аналіз та інтеграція даних.
Cloud IDE та VCS	GitHub Codespaces, Replit, Git	Комунікація та співпраця; реалізація культури перегляду коду (Code Review) та версіонування коду
Інтелектуальні системи	OpenAI API, GitHub Copilot	Розвиток алгоритмічного мислення; ШІ як «партнер для дебагінгу» (AI Pair Programming)
DevOps	Docker, Kubernetes, CI/CD (GitHub Actions, GitLab CS), хмарні сервіси Vercel	Розгортання та підтримка ПЗ, автоматизація процесів; відповідальне використання цифрових технологій
DigComp (інтерактивний рівень)	Усі перелічені технології в комплексі	Комплексний розвиток цифрової

Веборієнтований підхід забезпечує синтез навчального та професійного інструментарію. Згідно з дослідженнями С. Литвиної [19], віртуальний клас та хмарні сервіси стають складниками синтетичного середовища.

– Для програміста саме веббраузер стає універсальним цифровим інструментом, що об'єднує комунікацію (Zoom, Teams), співпрацю (GitHub) та

розробку хмаро орієнтованих (ХО) інтегрованих середовищ розробки (ICP) , усуваючи технічний розрив між навчанням в аудиторії та вдома та забезпечуючи неперервність освітнього процесу будь-де та будь-коли.

– Веборієнтовані технології за своєю природою базуються на принципі навчання через формування мереж. Якщо цифрові засоби можуть використовуватися автономно, то веборієнтовані середовища (від Веб 2.0 до Веб 4.0) реалізують колективну співпрацю через механізми запитів на внесення змін (Pull Requests), обговорення в репозиторіях та асинхронну взаємодію у віртуальних спільнотах розробників.

– Станом на 2026 рік веборієнтовані системи еволюціонували до інтелектуального вебу. Оскільки 81,4% студентів вважають ІІІ своїм асистентом [22], використання AI-інструментів реалізується саме через вебінтерфейси. Це дозволяє впроваджувати підхід, що інтегрує ІІІ безпосередньо в процес кодування всередині веборієнтованого середовища.

Таким чином, використання терміну «веборієнтовані технології» дозволяє акцентувати увагу на процесуальному складнику навчання програмування, створенні програмного продукту безпосередньо в тому середовищі, де відбувається навчання та комунікація учасників освітнього процесу [19]. Завдяки цьому студенти не лише опановують технічні навички, а й поступово формують свою професійну компетентність, зокрема компетентність з програмування, що сприяє професійному становленню майбутнього фахівця та підготовці його до роботи на глобальному ІТ-ринку.

1.2. Особливості підготовки майбутніх бакалаврів комп'ютерних наук у зарубіжних закладах вищої освіти

Проблема підвищення ефективності навчання програмування у сучасній освіті безпосередньо пов'язана з використанням веборієнтованих технологій, онлайн-платформ та цифрових освітніх середовищ. Результати досліджень свідчать, що інтеграція вебінструментів в освітній процес сприяє формуванню

практичних навичок програмування, підвищує мотивацію студентів та розширює можливості індивідуалізації навчання.

Одним із ключових напрямів є розроблення методик онлайн-навчання програмування. У дослідженні Н. Шенлі (N. Shanley) [61] та ін. обґрунтовано педагогічні стратегії дизайну та фасилітації онлайн-курсів програмування, зокрема використання проблемно-орієнтованого навчання та симуляцій у вебсередовищах, що забезпечують активну взаємодію учасників освітнього процесу [61].

Подібні підходи розглядаються також у роботі, де систематизовано методи організації онлайн-навчання програмування та запропоновано рекомендації щодо оцінювання результатів навчання у цифрових освітніх середовищах [74].

Значна увага приділяється інтеграції вебтехнологій безпосередньо у зміст курсів комп'ютерних наук та інформаційних систем. Класичним прикладом є дослідження щодо викладання технологій веброзробки у навчальних програмах комп'ютерних наук та інформаційних систем, у якому запропоновано поетапну модель формування навичок розробки вебзастосунків через опанування HTML, JavaScript та серверних технологій. Розвиток цієї ідеї представлено у сучасних дослідженнях, де підкреслюється необхідність адаптації педагогічних технологій до швидкої еволюції вебтехнологій та інструментів програмування [75].

Важливим аспектом є використання вебплатформ та цифрових освітніх сервісів. Зокрема, онлайн-курси програмування та освітні платформи забезпечують гнучке навчання та можливість самостійного опанування програмування [74], [76]. Такі платформи часто включають інтерактивні вправи, автоматичну перевірку коду та аналітику прогресу студентів, що підвищує ефективність освітнього процесу.

Дослідження також підтверджують позитивний вплив вебінструментів на результати навчання програмування. Наприклад, експериментальні результати свідчать, що використання вебсередовищ у навчанні програмування значно

підвищує академічні результати студентів та їхню навчальну мотивацію [77], [78]. Подібні висновки підтверджують і дослідження щодо впровадження вебплатформ у курсах програмування в університетах [79].

Дослідження науковця С. Ватсона (C. Watson) [60] аналізують проблему високого рівня невдач студентів у курсах початкового програмування. Автор доводить, що впровадження інтерактивних вебплатформ, систем автоматичного оцінювання та онлайн-практикумів дозволяє знизити рівень академічних труднощів та покращити результати навчання. Особливу увагу приділено ролі практикоорієнтованих завдань у вебсередовищах програмування [60].

У статті С. Едварса (S. Edwards) [63] представлено веборієнтовану систему «Web-CAT» для автоматичного оцінювання програмних завдань студентів. Система забезпечує автоматичну перевірку коду, формування тестових сценаріїв та надання детального зворотного зв'язку. Автор доводить, що використання подібних вебінструментів підвищує ефективність навчання програмування, сприяє розвитку навичок тестування програм та оптимізує роботу викладача [63].

Подальший розвиток веборієнтованих технологій навчання програмування пов'язаний зі створенням інтерактивних цифрових середовищ, які забезпечують не лише виконання програмних завдань, а й підтримують когнітивний розвиток студентів. Одним із ранніх, але концептуально важливих досліджень у цьому напрямі, є праця таких дослідників, як В. Ю. Хван (W. Y. Hwang) [80], Ч. Ю. Ван (C.Y. Wang) [80], Г. Дж. Хван (G. J. Hwang) [80]. У цій роботі запропоновано веборієнтоване середовище навчання програмування [80].

У запропонованому середовищі реалізовано можливості онлайн-редагування коду, налагодження програм, спільного аналізу рішень та взаємооцінювання студентських робіт. Результати експериментального дослідження показали, що використання такого середовища сприяє розвитку когнітивних стратегій, розв'язування програмних задач, формуванню навичок,

рефлексії та підвищенню якості навчальних результатів студентів. Автори підкреслюють, що інтеграція інструментів співпраці та миттєвого зворотного зв'язку у вебсередовища програмування створює умови для більш глибокого розуміння алгоритмічних концепцій та підвищує ефективність навчального процесу [80].

Сучасний етап розвитку цифрової освіти характеризується активним використанням технологій ШІ у навчанні програмування. Автори С. Гун (X. Gong) [65], В. Сюй (W. Xu) [65], та А. Цяо (A. Qiao) [65] у своєму дослідженні розглядають підхід до навчання програмування з використанням генеративних моделей штучного інтелекту та запит-орієнтованих стратегій взаємодії з системами. Автори аналізують, як використання підказок під час роботи з інтелектуальними системами допомагає студентам формувати навички обчислювального мислення, зокрема декомпозиції задач, абстрагування та алгоритмічного мислення [65]. Емпіричні результати показують, що використання з підтримкою підказок у середовищах програмування підвищує рівень розуміння алгоритмічних концепцій та стимулює студентів до активнішого експериментування з програмним кодом. Таким чином, інтеграція генеративного штучного інтелекту у вебсередовища програмування відкриває нові можливості для персоналізації навчання та підтримки індивідуальних освітніх траєкторій [65].

Важливим напрямом удосконалення веборієнтованих систем навчання програмування є використання адаптивних механізмів та гейміфікації. Так, Д. Маріоно (D. Maryono) [81] та група співавторів представляють платформу NgodingSeru.com – адаптивну систему електронного навчання програмування, яка поєднує інструменти автоматичного аналізу коду, адаптивного добору навчальних завдань та елементи гейміфікації [81].

Платформа використовує механізми накопичення балів, рівнів складності та інтерактивних викликів, що стимулює навчальну мотивацію студентів. Результати експерименту свідчать про значний позитивний вплив системи на розвиток навичок розв'язування програмних задач; статистичний аналіз

показав високий ефект впливу. Автори роблять висновок, що адаптивні гейміфіковані вебсистеми можуть суттєво підвищувати ефективність формування програмувальних компетентностей [81].

Окрему увагу у сучасних дослідженнях приділено колаборативним формам навчання програмування у вебсередовищах. Зокрема, у роботі С. Цай (C. Tsai) [67] та співавторів досліджується ефективність поєднання онлайн-взаємонавчання та розподіленого парного програмування. Автори доводять, що така модель навчання сприяє активнішому залученню студентів до процесу розв'язування програмних задач, покращує комунікацію між учасниками навчального процесу та сприяє формуванню навичок командної роботи у програмуванні [67]. Експериментальні результати показали, що студенти, які працювали у форматі онлайн-парного програмування, демонстрували вищий рівень розуміння програмних концепцій та кращі результати виконання практичних завдань [67].

Суттєвим компонентом сучасних вебсередовищ навчання програмування є системи автоматичного оцінювання програмних завдань. У статті Б. Перейра (B Pereira Cipriano) [82], Н. Фашада (N. Fachada) [82] та А. Алвеша (P. Alves) [82] представлено інструмент Drop Project – платформу для автоматичного аналізу та оцінювання студентських програм. Система інтегрується з системами управління навчанням та дозволяє викладачам організовувати автоматичну перевірку програмного коду з використанням тестових наборів, статичного аналізу та системи контролю версій. Використання такого інструменту значно оптимізує процес оцінювання програмних завдань і забезпечує студентам оперативний зворотний зв'язок щодо якості їхнього програмного коду [82].

Подальший розвиток автоматизованого оцінювання програмних завдань пов'язаний із використанням моделей ШІ. Дослідники П. Лагакіс (P. Lagakis) [66], С. Деметріадіс (S. Demetriadis) [66], Г. Псатас (G. Psathas) [66] пропонують застосування великих мовних моделей для автоматичного аналізу та оцінювання програмних рішень студентів. Автори демонструють, що великі

мовні можуть ефективно аналізувати структуру програмного коду, визначати логічні помилки та надавати студентам змістовні рекомендації щодо вдосконалення програм. Використання таких технологій відкриває перспективи створення інтелектуальних систем підтримки навчання програмування, здатних автоматично адаптувати навчальні матеріали до потреб студентів [66].

Важливе теоретичне підґрунтя для розвитку методик навчання програмування формують систематичні дослідження педагогічних підходів у цій галузі. У роботі Й. Фейтоза (Y. Feitosa) [83], М. Сілва (M. Silva) [83] здійснено систематичне картування наукових досліджень, присвячених розвитку навичок розв'язування задач у курсах початкового програмування. Особливу вагу автори приділяють застосуванню тестування ПЗ та підходу Test-Driven Development як засобів підтримки процесу розв'язання задач і формування логічного мислення студентів. Результати дослідження свідчать про позитивний вплив цих підходів на навчання програмування початківців та вказують на перспективність їх подальшого використання у курсах початкового програмування [83].

Таким чином, результати аналізу наукових джерел свідчать, що веборієнтовані технології відіграють ключову роль у модернізації навчання програмування. Їх використання сприяє формуванню практичних компетентностей студентів, забезпечує інтерактивність освітнього процесу та створює умови для індивідуалізації освітньої траєкторії. Подальші дослідження мають бути спрямовані на розроблення комплексних моделей використання веборієнтованих технологій у підготовці майбутніх фахівців з комп'ютерних наук.

Аналіз зарубіжного досвіду модернізації ІТ-освіти дозволяє констатувати перехід від інструментального використання цифрових засобів до формування інтелектуалізованих освітніх екосистем, що базуються на веборієнтованих технологіях. Ключовим напрямом цієї трансформації у 2025–2026 роках стало впровадження Європейської рамки цифрової компетентності DigComp 3.0 [70]. На відміну від попередніх редакцій, DigComp 3.0 розглядає цифрову

компетентність не як набір технічних навичок, а як здатність бакалавра ефективно функціонувати в умовах активної інтеграції ІІІ у всі види професійної діяльності.

У контексті навчання програмування це зумовлює зміну парадигми: від механічного опанування синтаксису до формування здатності працювати в неперервних веборієнтованих середовищах. Принциповим є те, що використання таких інструментів, як GitHub Copilot, ChatGPT або XO ICP, більше не є факультативним – вони стають органічним складником методики викладання. Студент у закордонних ЗВО постає як суб'єкт, що поєднує технічні, знаннєві та етичні складники при розробленні програмного коду.

Важливою інновацією, що забезпечує гнучкість закордонних моделей освіти, є технологічна нейтральність та орієнтація на SaaS-моделі (Software as a Service). Це стимулює використання веборієнтованих рішень, доступних з будь-якого пристрою, що дозволяє студентам працювати з реальними репозиторіями, інтегрувати інтерфейси програмування застосунків та застосовувати практики безперервної інтеграції і безперервного постачання (БІ/БП)) без прив'язки до локальних обчислювальних ресурсів університету.

Для систематизації цього досвіду нами було розроблено матрицю відповідності між сферами DigComp 3.0 та веборієнтованими технологіями, що застосовуються у провідних університетах світу (табл. 1.4).

Таблиця 1.4

Матриця відповідності DigComp 3.0 веборієнтованим технологіям у навчанні програмування

Напрямок DigComp 3.0	Складник у навчанні програмування	Веб-орієнтовані технології	Очікувані результати навчання
Інформаційна та дата-грамотність	Пошук рішень, робота з технічною документацією та Big Data	Stack Overflow, MDN Web Docs, ChatGPT (як пошуковий асистент)	Здатність критично оцінювати та фільтрувати код; вміння знаходити оптимальні алгоритмічні рішення у мережі.

Напрямок DigComp 3.0	Складник у навчанні програмування	Веб-орієнтовані технології	Очікувані результати навчання
Комунікація та співпраця	Командна розробка (Teamwork), професійний фідбек	GitHub, GitLab, Bitbucket (Pull Requests, Code Review)	Формування культури колективної розробки; опанування систем контролю версій та етики професійної взаємодії.
Створення цифрового контенту	Проектування, кодування та життєвий цикл ПЗ	Web-IDE (Replit, CodePen), GitHub Actions, Vercel	Вміння створювати програмні продукти; розуміння процесів деплойменту та безперервної інтеграції (CI/CD).
Розв'язання проблем	Дебагінг, алгоритмічне мислення, адаптація до нових стеків	LeetCode, HackerRank, онлайн-судді (система як суддя0, AI- дебагери	Здатність до адаптивного вирішення задач; вміння тестувати код та автоматизувати пошук помилок у веб- середовищі.
Безпека	Кібергігієна коду, захист персональних даних, етика AI	Хмарні сервіси (AWS, Azure), Auth0 (керування доступом)	Усвідомлення вразливостей коду; навички захисту інтелектуальної власності та етичного використання AI- інструментів.

Закордонний досвід реалізації цих підходів варіюється залежно від історичних та управлінських особливостей освітніх систем. Дослідник І. Соколова виділила три домінуючі моделі вищої освіти: *атлантичну* (децентралізована, орієнтована на ринок), *континентальну* (централізована, з акцентом на державні стандарти) та *східноазійську* (висока дисципліна та державна підтримка інновацій) [84]. Спільним для всіх моделей є впровадження

свободи вибору дисциплін та перехід до 8-рівневої системи компетентності (від базового кодування до експертного проектування архітектури).

1.2.1 Атлантична модель підготовки бакалаврів комп'ютерних наук

Атлантична модель підготовки бакалаврів комп'ютерних наук характерна для таких країн, як США, Канада, Велика Британія, Австралія, Нова Зеландія. Найбільш відомі університети, які готують бакалаврів-програмістів наведено у таблиці 1.5.

Таблиця 1.5

Університети, які здійснюють підготовку бакалаврів комп'ютерних наук

№	Університети	Країна	Місто
1	Гарвардський університет (Harvard University)	США	Кембридж, штат Массачусетс
2	Массачусетський технологічний інститут (MIT)	США	Кембридж, штат Массачусетс
3	Стенфордський університет (Stanford University)	США	Стенфорд, Каліфорнія
4	Оксфордський університет (University of Oxford)	Велика Британія	Оксфорд
5	Кембриджський університет (University of Cambridge)	Велика Британія	Кембридж
6	Університет Торонто (University of Toronto)	Канада	Торонто
7	Університет Макгілла (McGill University)	Канада	Монреаль
8	Мельбурнський університет (University of Melbourne)	Австралія	Мельбурн
9	Австралійський національний університет (ANU)	Австралія	Канберра
10	Лондонська школа економіки та політичних наук (LSE)	Велика Британія	Лондон

Загальними рисами при складанні освітніх програм атлантичної моделі підготовки бакалаврів комп'ютерних наук є незалежність університетів від

держави; конкурентність і ринковість; тісний зв'язок з бізнес-спільнотою [20]; поєднання освіти, науки і бізнесу; орієнтація на практичність; модульна структура навчання; розвинена система грантів і стипендій; демократія, науковість та високий рівень компетентності, соціальна детермінація, гуманізація, правова пріоритетність, інформаційно-комунікаційні технології.

Ключовою особливістю цієї моделі є «освітня свобода під відповідальністю», де ЗВО самостійно визначають зміст програм, орієнтуючись на динамічні вимоги глобального технологічного ринку. Станом на 2025–2026 роки атлантична модель демонструє лідерство у впровадженні ХО парадигми навчання, яка передбачає п'ять ключових трансформацій освітнього процесу.

Перша. Відмова від локального середовища. Раніше студенту доводили встановлювати на власний комп'ютер компілятори, системи керування базами даних інтегровані середовища розробки (ICP) та налаштовувати зміни середовища. Хмаро орієнтований (ХО) підхід передбачає що веббраузер є єдиним вікном професійну діяльність (GitHub Codespaces, Gitpod, AWS Cloud). Студент отримує доступ до потужних обчислювальних ресурсів навіть за допомогою малопотужного ноутбука

Друга. Контейнеризація навчальних завдань. Парадигма передбачає, що кожен комп'ютерний практикум надається студенту у вигляді контейнера. Це гарантує, що код, який коректно функціонує у викладача, так само працюватиме і у студента (вирішення проблеми роботи на окремих машинах). Крім того, студент працює з мікросервісною архітектурою вже з першого курсу.

Третя. Система БІ/БП як дидактичний засіб, представляє собою сукупність програмних інструментів і методів автоматизованої інтеграції, тестування та розгортання програмного забезпечення, що використовується в освітньому процесі для формування професійних компетентностей майбутніх бакалаврів комп'ютерних наук. Навчання передбачає автоматизацію життєвого циклу розробки. Студент не просто здає файл з кодом, він вивантажує код на сервер. Хмарна система автоматично: перевіряє код на помилки, запускає автотестування, розгортає проєкт на сервер, де його може перевірити викладач.

Четверта. Еластичність та масштабованість ресурсів. Парадигмою передбачено динамічний доступ до знань. Якщо студенту для виконання завдання з BigData потрібно опрацювати терабайт даних, він розгортає кластер у хмарі на годину, виконує завдання і видаляє його. Це вчить бакалавра ергономії хмарних обчислень (FinOps).

П'ята. Інтелектуалізація через Web 4.0. Оскільки всі дані про процес написання коду (таймінг, помилки, виправлення) зберігаються в хмарі, парадигма передбачає залучення III-менторів. Система аналізує, на якому етапі студент «застряг» у ХО ІСР і підказує не готове рішення, а посилання на документацію або архітектурну пораду в режимі реального часу.

Таким чином, професійне середовище бакалавра комп'ютерних наук трансформується з локально-орієнтованого у хмаро орієнтоване/віртуальне, де робоче місце перестає бути прив'язаним до конкретних апаратних потужностей (ПК) і функціонує як набір файлів у хмарній інфраструктурі.

Аналіз досвіду провідних університетів (Harvard, MIT, Stanford) дозволяє виокремити веборієнтовані технології як головний інструмент формування професійної компетентності, зокрема компетентності з програмування. Найбільш репрезентативним прикладом реалізації атлантичної моделі є вступний курс із комп'ютерних наук CS50, який використовується не лише в Гарварді та Єлі, а й масово масштабується через платформу Prometheus в Україні. Технологічна трансформація цього курсу ілюструє перехід до неперервного і повсюдного навчання:

- використання ХО ІСР дозволяє забезпечити кожному студенту ідентичне середовище розробки без необхідності складного локального встановлення програмного забезпечення. Це вирішує критичну проблему адаптації новачків, описану в дослідженнях [83] (GitHub Codespaces та VS Code Cloud);
- студенти отримують миттєвий зворотний зв'язок через веборієнтовані скрипти тестування коду, що сприяє розвитку алгоритмічного мислення на ранніх етапах (автоматизоване оцінювання);

– інтеграція спеціалізованих чат-ботів на основі генеративного ШІ (Веб 4.0) забезпечує персоналізовану підтримку 24/7, допомагаючи студентам виявляти та усувати помилки у коді без надання прямих відповідей, що відповідає етичним нормам DigComp 3.0. (ШІ-асистент).

Специфіка використання веборієнтованих технологій в інших ЗВО атлантичної моделі, зокрема в діяльності Массачусетського технологічного інституту (MIT) та Стенфордського університету, підтверджує, що веборієнтовані середовища є основою для реалізації сфери комунікації та співпраці за DigComp 3.0. Використання GitHub та GitLab для проведення групової перевірки коду формує у бакалаврів культуру колективної розробки та Git-мислення.

У Каліфорнійському університеті в Берклі (UC Berkeley) пріоритет надається хмарним обчисленням через Jupyter Notebooks та Google Colab, що дозволяє формувати дата-грамотність майбутніх фахівців у межах єдиного ХО простору. Університет Карнегі Меллона (Carnegie Mellon University(США)), акцентує на опануванні повного життєвого циклу ПЗ через сервіси БІ/БП (GitHub Actions), що забезпечує автоматичне розгортання студентських проєктів у хмару.

- Узагальнюючи, зазначимо, що атлантична модель реалізується на:
- людиноцентричному підході через побудову індивідуальних траєкторій та вільний доступ до вебресурсів 24/7;
- самостійній роботі (співвідношення 1:3 до аудиторної) і базується на використанні інтелектуальних агентів та онлайн-суддів (LeetCode, HackerRank), що робить навчання інтенсивним та практикоорієнтованим;
- цифровій професійній ідентичності бакалавра, що формується в умовах співпраці з ШІ (Веб 4.0), де інструменти, наприклад, GitHub Copilot виступають інтелектуальними асистентами у написанні чистого коду.
- Враховуючи специфіку атлантичної моделі підготовки бакалаврів комп'ютерних наук (США, Велика Британія), сформуємо розширену таблицю веборієнтованих технологій як конкретних інструментів, що інтегруються в

освітній процес провідних університетів світу для досягнення цілей рамки DigComp 3.0 (табл. 1.6).

– Таким чином, атлантична модель підготовки бакалаврів комп’ютерних наук виступає еталоном технологічної нейтральності, де акцент переноситься з вивчення конкретної мови програмування на здатність ефективно функціонувати в глобальній веборієнтованій системі.

Таблиця 1.6

Веборієнтовані технології навчання програмування в межах атлантичної моделі освіти

Категорія технологій	Конкретні інструменти	Університет-взірець	Дидактична мета та результат для студента
Cloud-native IDE (Хмарні середовища) [85]	GitHub Codespaces, VS Code Cloud, Replit	Harvard University (курс CS50)	Забезпечення ідентичного середовища розробки без локального встановлення ПЗ. Студент працює в професійній хмарній екосистемі з будь-якого пристрою
Платформи для співпраці та VCS [86], [87]	GitHub, GitLab, Bitbucket	MIT – Massachusetts Institute of Technology	Розвиток культури колективної розробки, управління версіями та проведення Code Review. Опанування етики спільної розробки
Генеративний ШІ (Веб 4.0) [88]	ChatGPT, GitHub Copilot, AI Duck	Stanford University	Інтеграція ШІ як інтелектуального асистета для дебагінгу та пояснення логіки коду. Розвиток здатності критично оцінювати пропозиції AI
Автоматизовані системи перевірки [89], [90]	HackerRank, LeetCode, check50	University of Oxford	Автоматизація перевірки алгоритмів для отримання миттєвого зворотного зв'язку. Формування алгоритмічного мислення
Інтелектуальна аналітика даних [91]	Jupyter Notebooks (Cloud), Google Colab	UC Berkeley	Формування дата-грамотності через роботу з хмарними обчисленнями. Вміння аналізувати Big Data

Категорія технологій	Конкретні інструменти	Університет-взірець	Дидактична мета та результат для студента
			та документувати код в інтерактивному режимі
Система розгортання та CI/CD [92]	GitHub Actions, Vercel, CircleCI	Carnegie Mellon University	Опанування повного життєвого циклу ПЗ: від написання коду до його реалізації. Розуміння процесів безперервної інтеграції (CI/CD)

1.2.2 Континентальна модель підготовки бакалаврів комп'ютерних наук

В основу континентальної моделі підготовки бакалаврів комп'ютерних наук покладено німецьку та французьку системи, які базуються на централізації управління (тобто концентрації більшості функцій, відповідних прав і повноважень у центральних владних структурах, наявність переважно вертикальних зв'язків, підпорядкованість або субординацію суб'єктів управління); відсутності безпосередніх зв'язків «університет - бізнес».

Освітні системи «континентальної моделі» (Нідерланди, Німеччина, Швейцарія) відрізняє ієрархічність навчальних закладів, їхній тісний зв'язок із державними структурами, пряме міністерське регулювання та відсутність безпосередніх зв'язків «університет - бізнес». Континентальна модель спирається на однорівневу схему навчання у ЗВО та ранню спеціалізацію студентів [84].

Найбільш відомі університети, які готують бакалаврів комп'ютерних наук представлено в таблиці 1.7.

Континентальна модель вищої освіти, репрезентована системами Німеччини, Франції, Австрії та Швейцарії, базується на принципах державного регулювання, централізації управління та високої академічної стабільності. На відміну від ринково-орієнтованої Атлантичної моделі, тут домінує логіка

державного замовлення, де акредитація виступає державною гарантією якості та відповідності єдиним європейським стандартам [93].

Таблиця 1.7

**Університети, які представляють континентальну модель підготовки
бакалаврів комп'ютерних наук**

№	Університети	Країна	Місто
1	Технічний університет Мюнхена (Technische Universität München (TUM))	Німеччина	Мюнхен
2	Сорбонна, Університет Париж-Сакле, École Polytechnique	Франція	Париж, Ліон, Тулуза
3	Болонський університет, Сапієнца (Рим)	Італія	Болонья, Рим, Мілан
4	Університет Саламанки, Університет Комплутенсе (Мадрид)	Іспанія	Мадрид, Барселона, Саламанка
5	Ягеллонський університет, Варшавський університет	Польща	Краків, Варшава, Вроцлав
7	Віденський університет	Австрія	Відень
8	ETH Zürich, Цюрихський університет	Швейцарія	Цюрих, Лозанна
9	Карлів університет	Чехія	Прага
10	Уппсальський університет, Стокгольмський університет	Швеція	Уппсала, Стокгольм

У 2025–2026 роках континентальна модель стала ключовим майданчиком для стандартизованого впровадження рамки DigComp 3.0. Завдяки централізованому управлінню, ЗВО системно впроваджують веборієнтовані технології в освітні програми, забезпечуючи підготовку бакалавра, чия професійна компетентність стає інтелектуальним фундаментом розвитку цифрової системи.

Аналіз досвіду провідних університетів континентальної Європи (TUM, ETH Zurich, Сорбонна) дозволяє виокремити технологічну специфіку, інструментарій моделі та особливості використання веборієнтованих технологій у підготовці бакалаврів комп'ютерних наук.

- На відміну від американської децентралізації, європейські ЗВО використовують веборієнтовані плагіни, наприклад, віртуальну лабораторію програмування (Virtual Programming Lab (VPL)) для Moodle [94]. Це створює ізольоване хмарне середовище для проведення іспитів із програмування, що гарантує об'єктивність оцінювання та дотримання стандартів кібергігієни коду (середовище, інтегроване з LMS, та модуль VPL для Moodle).

- Поширено дуальну освіту як фундамент неперервного повсюдного навчання. У Німеччині (наприклад, TUM) підготовка бакалавра наук передбачає тісну інтеграцію з IT-корпораціями (Siemens, BMW, Bosch). Веборієнтовані технології забезпечують технічну можливість такої взаємодії студент використовує одні й ті самі Cloud IDE та репозиторії як під час теоретичних занять в університеті, так і під час оплачуваної практики на підприємстві.

- Континентальна модель передбачає, що більшість часу студент опановує матеріал самостійно. Це стимулює активне використання віртуальних класів, сервісів онлайн-взаємодії (Zoom, Teams) та інтелектуальних систем підтримки навчання, де ІІІ виступає в ролі асистента, допомагаючи студенту в самостійному дебагінгу та пошуку рішень у спільнотах розробників. Зауважимо, що за цією моделлю спостерігається висока частка самостійної роботи (60%).

На основі аналізу підходів *континентальної моделі* підготовки бакалаврів комп'ютерних наук, розглянемо, як конкретні інструменти інтегруються в освітній процес провідних університетів світу для досягнення цілей рамки DigComp 3.0 (табл. 1.8).

Досвід Німеччини щодо використання формувального оцінювання та інтеграції практичного навчання на підприємствах у єдине веборієнтоване середовище є критично важливим для цифрової трансформації ІТ-освіти.

Таблиця 1.8

**Веборієнтовані технології навчання програмування в межах
континентальної моделі підготовки бакалаврів комп'ютерних наук**

Складник моделі	Технологічне рішення	Університет	Дидактичний ефект (DigComp 3.0)
Офіційне оцінювання	Moodle VPL (Virtual Programming Lab)	ETH Zurich	Безпека та прозорість контролю результатів навчання
Дуальна взаємодія	Enterprise GitHub / Cloud IDE	Technical University of Munich (TUM)	Комунікація та співпраця між академічним та промисловим середовищами
Теоретична база	LMS Campus / Open Access Portals	LMU Munich	Інформаційна грамотність та доступ до навчальних матеріалів 24/7
Інтелектуальна підтримка	AI-тьютори (локальні LLM-моделі)	École Polytechnique (Франція)	Створення контенту та етичне використання ШІ-агентів

Використання веборієнтованих технологій дозволяє нівелювати розрив між теоретичною підготовкою в університеті та вимогами реального ІТ-ринку, що є ключовою проблемою вітчизняних ЗВО.

Таким чином, континентальна модель як веборієнтоване середовище стає гарантом академічної якості, забезпечуючи системність, законодавчу прозорість та відповідність бакалавра комп'ютерних наук глобальним стандартам DigComp 3.0.

1.2.3 Східноазійська модель підготовки бакалаврів комп'ютерних наук

В основу управління вищою освітою східноазійської моделі, як зазначила І. В. Соколова, покладено значну централізацію управління освітою, сильний вплив держави на всі аспекти освітнього життя, незважаючи на достатньо розвинутий приватний освітній сектор вищої освіти [93] (в Японії, наприклад, 75% університетів — приватні); поступова лібералізація централізованого управління; визнання вищої освіти пріоритетною галуззю, сприяння просуванню кращих університетів на ринку освітніх послуг; інтернаціоналізація вищої освіти [84].

Найбільш відомі університети, що представляють східноазійську модель вищої освіти і які пропонують програми бакалаврату з комп'ютерних наук, наведено в таблиці 1.9.

У межах цієї дисертації розглянемо східноазійську модель підготовки бакалаврів комп'ютерних наук на прикладі систем п'яти країн — Японії, Китаю, Південної Кореї, Сінгапуру та Тайваню. Ця модель базується на принципах значної централізації управління, сильного державного впливу та визнання ІТ-сфери пріоритетною галуззю національної безпеки й економічного розвитку.

Таблиця 1.9

Університети, що представляють східноазійську модель підготовки бакалаврів комп'ютерних наук

№	Університети	Країна	Особливості системи освіти (у межах Східноазійської моделі)
1	Токійський університет (University of Tokyo)	Японія	Найголовніший національний університет Японії; суворі стандарти вступу; орієнтація на дослідження та розвиток технологій; підтримка урядових програм інновацій.
2	Пекінський університет (Peking University)	Китай	Найпрестижніший університет Китаю; поєднання західних академічних підходів і конфуціанських цінностей; висока дисципліна; акцент на науці, інженерії, ІТ.

№	Університети	Країна	Особливості системи освіти (у межах Східноазійської моделі)
3	Цінхуа університет (Tsinghua University)	Китай	Сильна державна підтримка; тісний зв'язок із промисловими корпораціями, дослідження у сфері технологій, інженерії та штучного інтелекту;
4	Кіотський університет (Kyoto University)	Японія	Потужний науково-дослідний університет; значна роль у розвитку природничих наук і медицини; баланс традицій і сучасності.
5	Сеульський національний університет (Seoul National University)	Південна Корея	Центральний державний університет; конкурентна система відбору студентів; тісна інтеграція з промисловістю та ІТ-компаніями (Samsung, LG).
6	Корейський передовий інститут науки та технологій (KAIST)	Південна Корея	STEM-напрями; підготовка висококваліфікованих фахівців для інноваційної економіки; співпраця з урядом і бізнесом.
7	Національний університет Сінгапуру (NUS)	Сінгапур	Поєднання британської академічної традиції та азійської дисципліни; акцент на міжнародній конкурентоспроможності.
8	Технологічний університет Наньян (NTU)	Сінгапур	Інноваційна освіта; штучний інтелект; стартапи; цифрові технології; підтримка урядових програм цифровізації.
9	Гонконзький університет (University of Hong Kong, HKU)	Китай (Гонконг, SAR)	Британська система освіти з азійськими цінностями; високий рівень академічної автономії; міжнародне викладання англійською.
10	Національний університет Тайваню (NTU)	Тайвань	Сильна державна освітня політика; інноваційні дослідження в ІТ, медицині та біотехнологіях; державне фінансування наукових проєктів.

Станом на 2025–2026 роки ця модель демонструє найвищі темпи інтеграції інтелектуальних систем у навчання програмування, що зумовлено стратегічним курсом цих країн на технологічне лідерство у розробці ШІ та робототехніки.

У межах східноазійської моделі стандарти спеціальності «Комп'ютерні науки» встановлюються централізовано через профільні міністерства (наприклад, Міністерство освіти, культури, спорту, науки та технологій Японії (МEXT) або Міністерство освіти Китайської Народної Республіки (МОЕ)). Важливою особливістю є узгодження національних вимог із міжнародною угодою Seoul Accord [95], що гарантує взаємне визнання дипломів бакалаврів-програмістів та відповідність професійних компетентностей глобальним вимогам ІТ-ринку. Держава виступає не лише регулятором, а й інвестором, що забезпечує ЗВО найсучаснішою хмарною інфраструктурою для реалізації концепції Web 4.0.

Аналіз досвіду Токійського університету (University of Tokyo) дозволяє виокремити унікальну двоступеневу структуру підготовки бакалаврів:

Цикл базових дисциплін (Foundation Courses). Протягом перших двох років студенти опановують фундаментальні математичні та світоглядні концепції, що формує здатність до критичного мислення.

Спеціалізований цикл (Senior Division). Починаючи з третього курсу, студенти прикріплюються до однієї з 20 дослідницьких лабораторій, де виконують індивідуальні науково-дослідні роботи з проектування передових ІТ-систем під керівництвом провідних вчених.

Особлива увага приділяється вивченню дисциплін, що є основою сучасних веборієнтованих систем: теорія інтелектуальних систем (AI Systems Theory), обробка природної мови (NLP), теорія обчислювальної складності та дискретна оптимізація. Це забезпечує випускникам лідерські позиції у створенні складних алгоритмічних рішень для глобальних ІТ-корпорацій.

Східноазійська модель поєднує жорстку дисципліну з інноваційними методами дистанційної творчості. У навчанні програмування активно застосовуються:

- методи проблемного навчання та комп'ютерне моделювання (використання віртуальних лабораторій для відпрацювання алгоритмів у реальному часі);

– інструменти колективної онлайн-творчості (проведення дистанційних мозкових штурмів, створення інтерактивних вебсторінок та порівняльний аналіз інформації в мережі Веб);

– використання інтелектуальних агентів (AI Tutors) як персоналізованих наставників, що забезпечують безпечне та привітне навчальне середовище, де студент почувається вільно під час дебагінгу (AI Pair Programming).

Для деталізації технологічного інструментарію розроблено порівняльну таблицю (табл. 1.10).

Таблиця 1.10

**Веборієнтовані технології навчання у східноазійській моделі освіти
підготовки ІТ-фахівців**

Компонент моделі	Технологічне рішення	Університет	Дидактична мета (DigComp 3.0)
Інтелектуальні системи	AI Systems & NLP Platforms	University of Tokyo	Створення цифрового контенту на основі ШІ-моделей
Державна інноваційність	Cloud-native Research Labs	Tsinghua University	Розв’язання складних проблем через хмаро орієнтоване моделювання
ІТ-інтеграція	Enterprise Git & IoT Hubs	Seoul National University	Комунікація та співпраця з ІТ-корпораціями (Samsung, LG)
Автономна підтримка	AI Tutors & Smart Avatars	NTU Singapore	Персоналізація навчання та емоційна підтримка студента

На відміну від атлантичної моделі, де акцент робиться на індивідуальній автономії, тут веборієнтовані технології використовуються для досягнення високих нормативних результатів, що відповідають міжнародним вимогам Seoul Accord [95].

На основі аналізу підходів східноазійській моделі розглянемо, як конкретні інструменти інтегруються в освітній процес провідних університетів для досягнення цілей рамки DigComp 3.0 (табл. 1.11).

Східноазійська модель демонструє лідерство у впровадженні програмування з використанням ШІ та інтелектуальних агентів, що дає змогу автоматизувати рутинну перевірку коду й зосередити увагу студента на архітектурних рішеннях. Використання агентів на основі ШІ в навчанні програмування забезпечує безпечне та сприятливе середовище для налагодження програмного коду, у якому студент може вільно здійснювати пошук та виправлення помилок.

Таблиця 1.11

Реалізація DigComp 3.0 у східноазійській моделі на основі веборієнтованих технологій

DigComp 3.0	Веборієнтовані технології навчання	Специфіка реалізації (на прикладі Японії, Китаю, Сінгапуру)
Інформаційна та дата-грамотність	NLP-платформи, семантичні пошукові системи	Студенти Токійського університету використовують веборієнтовані системи обробки природної мови для глибокого аналізу великих масивів технічної документації та наукових даних
Комунікація та співпраця	Enterprise Git, хмарні середовища спільної творчості	Реалізується через онлайн-мозкові штурми та спільне проєктування в мережесхвильованих творчих групах
Створення цифрового контенту	Cloud-native Research Labs, інтерактивні вебсервіси	Створення складних програмних продуктів та інтерактивних сторінок у віртуальних дослідницьких лабораторіях під керівництвом вчених
Безпека	Державні хмарні інфраструктури, IoT Hubs	Використання захищених національних хмарних платформ для роботи з IoT та робототехнікою, що формує навички кібергігієни в промислових масштабах
Розв'язання проблем	AI Tutors, віртуальні	ШІ-агенти (напр. у NTU Singapore) діють як персоналізовані наставники, допомагаючи студенту

DigComp 3.0	Веборієнтовані технології навчання	Специфіка реалізації (на прикладі Японії, Китаю, Сінгапуру)
	симулятори та моделі	дебагити код та знаходити оптимальні архітектурні рішення у реальному часі

Проте, моніторинг виявив і низку недоліків, зокрема: надмірну стандартизацію та високий психологічний тиск, що може обмежувати свободу самовираження студентів. Незважаючи на це, східноазійський досвід доводить, що веборієнтоване середовище стає глобальним інструментом інновацій, де ІІІ-агенти трансформуються з допоміжних засобів у повноцінних когнітивних асистентів майбутнього програміста. Таким чином, Східноазійська модель демонструє майбутнє інтелектуального Веб 4.0, де дисципліна, державна стратегія та передові технології формують фахівця, здатного створювати інноваційні продукти світового рівня.

Представлений зарубіжний досвід підтверджує, що сучасна закордонна ІТ-освіта розглядає програмування не як статичний набір умінь, а як процес поступового професійного зростання в умовах розвитку інтелектуальних систем. Використання зазначених вебтехнологій дозволяє університетам реалізувати рівневу систему володіння компетентностями: від базового виконання вправ у вебтренажерах до експертного проєктування архітектури хмарних застосунків.

У процесі аналізу зарубіжних систем підготовки бакалаврів комп'ютерних наук з використанням веборієнтованих технологій було виявлено низку розбіжностей, які стосуються рівнів державного управління, структури навчальних планів та глибини інтеграції в освітній процес реальних потреб ІТ-сектору.

Атлантична модель (США, Велика Британія) базується на високій децентралізації та фінансовій автономії, де кожен університет самостійно визначає зміст програм відповідно до ринкових вимог. На противагу цьому, Континентальна (ЄС) та Східноазійська моделі характеризуються значною

централізацією та жорстким державним контролем стандартів, що є ближчим до української системи освіти.

У зарубіжних ЗВО (особливо атлантичної моделі) реалізовано принцип «освітньої свободи під відповідальність», що дозволяє студентам самостійно обирати спеціалізацію та формувати індивідуальну траєкторію навчання. Програми орієнтовані на раннє та комплексне занурення студента у професію, через що у них практично відсутній великий перелік дисциплін загальної спрямованості, які досі є обов'язковими у вітчизняних планах підготовки. На практиці веборієнтовані технології використовуються для подолання розриву між теорією та практикою (наприклад, через дуальну освіту в Німеччині).

Зарубіжний досвід демонструє перехід системи здобуття компетентності з програмування згідно з DigComp 3.0, де веборієнтоване середовище виступає не просто інструментом доставки контенту, а цілісною екосистемою для професійного зростання (від базових вправ у браузері до проєктування хмарної архітектури).

Таким чином, виявлені розбіжності підкреслюють необхідність модернізації вітчизняної системи підготовки бакалаврів комп'ютерних наук, шляхом вивчення та адаптації успішних закордонних практик, зокрема у сфері створення неперервного та повсюдного веборієнтованого навчального середовища

1.3.Сучасний стан підготовки майбутніх бакалаврів комп'ютерних наук у вітчизняних закладах вищої освіти.

Сучасний стан підготовки майбутніх бакалаврів комп'ютерних наук у вітчизняних закладах вищої освіти (ЗВО) України перебуває на стадії постійної модернізації та подальшого використання веборієнтованих та хмарних технологій, штучного інтелекту в освітньому процесі.

Актуальність питання підготовки спеціалістів зі спеціальності F3 «Комп'ютерні науки» підтверджується тим, що Україна взяла курс на

створення цифрової держави, а саме реалізацію низки цифрових проєктів, які, зокрема, координує Міністерство цифрової трансформації та які формують цифровий простір держави.

До таких проєктів можна віднести: «Дія Застосунок», «Електронні послуги», «е-Резиденство», «Дія City», «Цифрова освіта», «Цифровізація освіти», «Бізнес», «Захист дітей в Інтернеті», «Євроінтеграція» (<https://thedigital.gov.ua/projects>), які включають низку підпроєктів, наприклад: реєстрація місця проживання, реєстрація речових прав, е-підприємець, е-малятко, цифровий податковий номер, прописка онлайн, подання заявок до ЗВО та ін.

Розроблення значної кількості програмних застосунків потребує від держави постійного їх оновлення та підтримання в працездатному стані, що, своєю чергою, зумовлює потребу у кваліфікованих програмістах і, відповідно, у підготовці конкурентоспроможних випускників ЗВО [21].

Наразі, динаміка розвитку ІТ-сфери, зокрема веборієнтованих, кросплатформних та ХО технологій, є настільки високою, що вимагає від випускників не лише знання сучасних напрямів, методів, методологій (зокрема Agile) і технологій розробки програмного забезпечення, а й уміння працювати в команді, володіння сучасними стратегіями, актуальними технологічними стеками та інструментами колективної розробки програмного забезпечення.

Водночас, стандарти у сфері комп'ютерних наук мають визначати загальні та фахові компетентності, відповідно до яких формуються вимоги до освітніх програм підготовки фахівців. Це, у свою чергу, зумовлює необхідність постійного оновлення навчальних планів і змісту освітніх компонентів у ЗВО, а також регулярного підвищення кваліфікації науково-педагогічних працівників [21].

Підготовка фахівців галузі знань F3 «Комп'ютерні науки» відбувається на основі державних стандартів вищої освіти України, згідно із законом «Про вищу освіту» від 01.07.2014 № 1556-VII де констатується, що студенти під час навчання в ЗВО можуть пройти відповідні рівні вищої освіти, а саме: перший

(бакалаврський) рівень; другий (магістерський) рівень; третій (освітньо-науковий) рівень; науковий рівень [104].

Здобуття вищої освіти на кожному рівні передбачає успішне виконання особою відповідної освітньої (освітньо-професійної, освітньо-наукової) чи наукової програми, що є підставою для присудження відповідного ступеня вищої освіти: бакалавра, магістра, доктора філософії, доктора наук [21].

У рамках цього дисертаційного дослідження будемо розглядати перший рівень вищої освіти ступінь «Бакалавр» галузі знань F «Інформаційні системи» спеціальності F3 «Комп'ютерні науки», який відповідає сьомому рівню Національної рамки кваліфікацій і передбачає здобуття особою теоретичних знань та практичних умінь і навичок, достатніх для успішного виконання професійних обов'язків за обраною спеціальністю [21].

Якщо йдеться про визначення бакалавра, то згідно із Законом України «Про вищу освіту», бакалавр – це освітній ступінь, що здобувається на першому рівні вищої освіти та присуджується ЗВО за результатами успішного виконання здобувачем вищої освіти освітньо-професійної програми, обсяг якої становить 180–240 кредитів ЄКТС [104].

Випускники спеціальності F3 відповідно до Національного класифікатора України ДК 003:2010 «Класифікатор професій» [105] можуть працювати за напрямками діяльності, пов'язаними з розробленням математичного, інформаційного та програмного забезпечення інформаційних систем у сфері інформаційних технологій.

З метою налагодження ефективної взаємодії ІТ-освітою та ІТ-індустрією сформульовано вимоги до посад та напрямків кар'єрного зростання фахівців у галузі програмування відповідно до рівня кваліфікацій у ІТ-сфері. Випускники можуть обіймати за такими професіями згідно з Національним класифікатором професій ДК 003:2010 [105]: 2131.2 Адміністратор бази даних; 131.2 Інженер з автоматизованих систем керування виробництвом; 2131.2 Інженер з комп'ютерних систем; 2131.2 Інженер з програмного забезпечення комп'ютерів; 3121.2 Фахівець з інформаційних технологій; 3121.2 Фахівець з

розробки та тестування програмного забезпечення; 3121.2 Фахівець з розроблення комп'ютерних програм [105].

Підготовка бакалаврів комп'ютерних наук здійснюється на основі держаного Стандарту вищої освіти за спеціальністю F3 «Комп'ютерні науки», затверджений наказом № 962 від 10.07.2019р. МОН України [106].

Стандарт визначає формування компетентного, конкурентоспроможного фахівця, здатного аналізувати, розробляти та впроваджувати інформаційні системи та інші ІТ-системи; створювати як індивідуальне, так і командне програмне забезпечення, керувати проєктами, працювати в команді. При цьому студент набуває обов'язкових загальних та фахових компетентностей спеціальності F3 «Комп'ютерні науки».

Державний стандарт (нормативна база) містить єдині мінімальні вимоги, проте кожен ЗВО може вносити зміни у власні програми. Обсяг освітньої програми бакалавра комп'ютерних наук становить 240 кредитів ЄКТС на базі повної загальної середньої освіти з терміном навчання 11 років [21].

Розробка стандартів F3 «Комп'ютерні науки» ведеться з урахуванням європейської рамки ІК-компетенцій (European e-Competence Framework) та рамки компетенцій SFIA (Skills Framework for the Information Age), що загалом відповідає міжнародним тенденціям в ІТ-індустрії.

При цьому, в освітній процес ЗВО впроваджуються стандарти вищої освіти відповідно до вимог Європейської рамки кваліфікацій (EQF, European Qualifications Framework) і рекомендацій ENQA. Основна увага приділяється: програмуванню, зокрема вебпрограмуванню; алгоритмам та структурам даних; базам даних; використанню ІІІ; кібербезпеці; застосуванню веборієнтовних та хмарних технологій; DevOps, Data Science та ін.

Стандарти вищої освіти спеціальності F3 «Комп'ютерні науки» слугують нормативною основою підготовки бакалаврів комп'ютерних наук визначаючи перелік компетентностей результати навчання та загальні вимоги до освітнього процесу. На їх основі ЗВО розробляють освітньо-професійні програми, наприклад освітньо-професійні програми спеціальності F3 «Комп'ютерні

науки» у КПІ ім. Ігоря Сікорського [107], Київському столичному університеті імені Бориса Грінченка [108], Національному університеті «Запорізька політехніка» [109], Запорізькому національному університеті [110], Львівському національному університеті [111]. У свою чергу, освітньо-професійні програми є підґрунтям для формування навчальних планів які деталізуються через робочі програми навчальних дисциплін з урахуванням психолого-педагогічних аспектів використання цифрових технологій в освітньому процесі [112].

Для забезпечення якості освіти періодично проводиться зовнішня експертиза та акредитація освітніх програм Національним агентством із забезпечення якості вищої освіти (НАЗЯВО/NAQA), результатом яких є протоколи та рішення щодо відповідності програм стандартам. Єдність вимог із європейськими стандартами (ECTS, Болонський простір) підтримує мобільність студентів і визнання дипломів у світі.

Підвищення вимог роботодавців до випускників ЗВО потребує зміни підходів до викладання основних дисциплін ІТ-спеціальностей, зокрема використання ШІ [22]. У процесі розгляду сучасних підходів до викладання мов програмування передусім доцільно зосередитися на дослідженні В. Ковалюк [113], яка пропонує застосування проєктно-орієнтованого підходу як інноваційної технології навчання. Сутність такого підходу полягає у розгляді управління освітнім процесом підготовки ІТ-фахівців, зокрема викладання мов програмування, з позицій технологій управління ІТ-проєктами. При цьому освітній процес організовується за аналогією до діяльності провідних ІТ-компаній із використанням окремих елементів бізнес-моделей ІТ-проєктів у підготовці студентів ІТ -спеціальностей [113]. Це сприяє формуванню у здобувачів освіти практичних навичок командної роботи, управління проєктною діяльністю та дає змогу підвищити рівень їх адаптації до професійних вимог сучасного ІТ -ринку праці [23].

Також Т. Ковалюк виокремлює аналогію між процесами управління ІТ -проєктами та організацією підготовки фахівців у галузі інформаційних

технологій. Авторка встановлює відповідність між основними етапами реалізації проєкту та етапами освітнього процесу а саме: ініціація (початок проєкту – наказ на зарахування студентів), планування (цілі проєкту галузеві стандарти), виконання (ресурси для виконання плану лекції, комп'ютерні практикуми, самостійні роботи, курсове, дипломне проєктування), аналіз (відповідність плану і виконання проєкту модульний та сесійний контроль), управління (корегування та узгодженість дій засідання кафедр, підвищення кваліфікації викладачів), завершення (фінал проєкту захист дипломних проєктів, дисертацій та видача відповідних дипломів) [23].

Освітньо-професійні програми спеціальності F3 «Комп'ютерні науки» передбачають значний обсяг навчального матеріалу, пов'язаного з вивченням мов програмування, який бакалаври мають опанувати упродовж чотирьох років навчання. Формування програмувальних навичок розпочинається з першого курсу під час вивчення дисципліни «Алгоритмізація та програмування». Обсяг цієї дисципліни становить близько п'яти кредитів ЄКТС і вона є базовим курсом не лише для закладів вищої освіти України, а й для закордонних університетів, що підтверджується документом CS2023 Curricula Guidelines and Computer Science Accreditation [114], [21].

На основі цієї дисципліни формуються базові компетентності бакалавра комп'ютерної наук, необхідні для подальшого вивчення дисциплін циклу професійної підготовки відповідно до освітніх програм різних спеціальностей. Рівень засвоєння навчального матеріалу дисципліни «Алгоритмізація та програмування» значною мірою визначає успішність опанування таких дисциплін, як об'єктно-орієнтоване програмування, структури даних та алгоритми, вебпрограмування, системне програмування, декларативне програмування, технологія програмування, створення програмних продуктів, кросплатформенне програмування, а також дисциплін, пов'язаних із компонентним, аспектно-орієнтованим, предметно-орієнтованим програмуванням, у тому числі і математичних дисциплін [21].

У процесі вивчення курсу «Алгоритмізація та програмування», застосовуються такі мови програмування як C/C++, C# та інші. На старших курсах студенти-програмісти опановують мови програмування JavaScript, TypeScript, Java, Python та інші сучасні мови, а також поглиблюють знання щодо використання сучасних засобів програмної розробки.[21].

У процесі аналізу освітнього процесу у ЗВО було встановлено, що навчання мов програмування реалізується через комплекс різних форм і видів навчальної діяльності, зокрема лекційні заняття, комп'ютерні практикуми, самостійну роботу студентів, модульні контрольні роботи, ректорський контроль, підсумкове оцінювання у формі іспитів і заліків, виконання та захист курсових робіт і кваліфікаційних проєктів, а також проходження виробничої та переддипломної практик [21].

Погоджуємося з твердженням О. М. Кривоноса [115], який стверджує, що у ЗВО у процесі навчання програмування використовуються вербальні (лекції) і практичні (виконання комп'ютерних практикумів, проєктів) методи, що дає змогу студентам не лише отримувати нові знання і набувати практичних навичок, але й формувати ключові компетентності, у тому числі й інформаційно-комунікаційні. Викладач виступає в ролі інструктора, наголошує на завданнях роботи, скеровує і певною мірою контролює хід її виконання. А діяльність студентів переважно практична, у якій суттєву роль відіграє самостійний розумовий процес, котрий уможливорює пошук необхідних даних і алгоритмів розв'язування задач [21], [115].

У нових освітньо-професійних програмах, починаючи з другого курсу, кожен здобувач вищої освіти має можливість формувати індивідуальну освітню траєкторію навчання відповідно до власних освітніх потреб.

На нашу думку, подальші перспективи розвитку системи підготовки бакалаврів полягають у поєднанні освітнього процесу з професійною діяльністю (дуальна освіта), розширенні співпраці із міжнародними освітніми платформами, зокрема Coursera та Udey, отриманні мікросертифікацій, а

також у створенні національних кластерів ІТ-освіти за участю бізнесу, ЗВО та держави.

Важливим чинником підвищення якості підготовки фахівців є постійне професійне зростання викладачів. Зокрема, позитивний вплив на освітній процес спеціальності F3 «Комп'ютерні науки» має регулярне проходження курсів і сертифікаційних програм (EPAM, Cisco Networking Academy, Oracle Academy, Coursera, edX, Prometheus), участь у ІТ-конференціях та стажування в ІТ-компаніях, а також здійснення наукової діяльності.

Одним із ключових показників ефективності підготовки бакалаврів комп'ютерних наук є рівень їх конкурентоспроможності, зокрема працевлаштування випускників за спеціальністю F3 «Комп'ютерні науки».

Так, у компанії EPAM Systems проаналізували інформацію про близько 10 тисяч фахівців, зокрема щодо рівня та місця здобуття ними освіти. Переважна більшість аїтівців мають закінчену вищу освіту, а 4% ще продовжують навчання. Майже 7 із 10 спеціалістів мають диплом магістра, третина – бакалавра, 2% – мають рівень кандидата наук, PhD та вище [116]. Встановлено, що 53% спеціалістів компанії є випускниками провідних закладів вищої освіти, що свідчить про важливу роль університетської освіти у формуванні професійних компетентностей ІТ-фахівців. Аналіз також демонструє залежність між рівнем освіти та кар'єрним зростанням: серед спеціалістів рівня Junior ступінь магістра мають 58%, серед Senior – 69%, а серед фахівців управлінського рівня цей показник сягає 87% [116].

Таким чином, вища освіта залишається важливим чинником професійного розвитку ІТ-спеціалістів. Водночас фундаментальні знання, здобутті у ЗВО, можуть ефективно доповнюватися практичними навичками, отриманими в межах корпоративних освітніх програм ІТ-компаній. Зокрема, у межах освітніх ініціатив EPAM Campus з моменту їх започаткування, навчання пройшли понад 59 тисяч українців [116]. Співпраця з індустрією здійснюється через спільні освітні програми, лабораторії, стажування, але ці ініціативи поки що не охоплюють усі університети та регіони країни.

Отже, сучасна система підготовки майбутніх бакалаврів з комп'ютерних наук в Україні характеризується достатньо розвинутою теоретичною базою та значним попитом серед абітурієнтів, але з одночасним наявним розривом між академічною підготовкою й вимогами індустрії. Основними напрямками подальшого розвитку є: поглиблення практичного компонента, модернізація змісту програм, розвиток матеріально-технічної бази і ширше залучення ІТ-індустрії до освітнього процесу.

Приєднання України до Болонського процесу примусило державу переглянути підходи до вищої освіти. Спроби впровадити Болонську освітню систему, у вітчизняну вищу освіту виявились досить невдалими, адже вони зачепили тільки незначні аспекти освітнього процесу (наприклад, рейтингову систему оцінювання), але при цьому не дали значних свобод при формуванні студентами свого власного графіка навчання [117]. Для впровадження трансформаційних процесів у вищу освіту в українських закладах вищої освіти слід вивчати досвід зарубіжних університетів.

1.4.Вимоги до рівня компетентності з програмування майбутніх бакалаврів комп'ютерних наук

В умовах інформатизації, комп'ютеризації та цифровізації освіти дуже важливим є питання підвищення рівня підготовки майбутніх бакалаврів комп'ютерних наук, який, своєю чергою, залежить від рівня компетентностей, яких вони набувають під час навчання та після його завершення. Дослідження базових понять «компетентність», «професійна компетентність», «професійна компетентність бакалавра комп'ютерних наук», «компетентність з програмування» представлено в **Додатку Б**.

Навчання майбутніх бакалаврів комп'ютерних наук здійснюється відповідно до освітньо-професійної програми та навчального плану, які, своєю чергою, будуються на основі стандарту підготовки бакалаврів комп'ютерних наук [24], [25].

Уточнюючи структуру професійної підготовки, ми констатуємо, що *компетентність з програмування бакалавра комп'ютерних наук є основою його цифрової компетентності*. Вона забезпечує перехід від репродуктивного використання вебресурсів до продуктивного створення інтелектуальних цифрових систем. В умовах стрімкого розвитку цифрових технологій та ШІ ця компетентність не обмежується знанням синтаксису мов (JavaScript, C++, Python), а включає здатність до алгоритмічного стилю мислення в межах веборієнтованих екосистем та навички парного програмування зі штучним інтелектом. [24].

Компетентність з програмування бакалаврів комп'ютерних наук – це динамічна здатність до використання веборієнтованих технологій, алгоритмічного стилю мислення, написання програмного коду різними мовами програмування [134] а також використання цифрових технологій в ЗВО в умовах дистанційного та змішаного навчання [135].

Компетентність з програмування *є основою цифрової компетентності*, оскільки вона забезпечує перехід від репродуктивної діяльності до продуктивної. Вона не обмежена вивченням синтаксису конкретних мов, а є процесом формування високорівневих когнітивних навичок.

Впровадження програмування як обов'язкового складника підготовки сучасного фахівця науково обґрунтовано потребою в адаптивності, системному мисленні та здатності до інноваційної діяльності в умовах глобальної цифровізації.

Компетентність з програмування майбутніх бакалаврів комп'ютерних наук має свої особливості, зокрема: майбутні бакалаври мають знати основи математичного аналізу, математичної статистики, чисельні методи, проєктувати і моделювати багаторівневі обчислювальні моделі тощо [134].

Крім того, у майбутніх бакалаврів комп'ютерних наук (Bachelor of Science in Computer Science) протягом усього часу підготовки формуються компетентності з програмування відповідно до міжнародних стандартів Європейської мережі із забезпечення якості інформатичної освіти (European

Quality Assurance Network for Informatics Education (EQANIE), міжнародної професійної організації для фахівців у галузі обчислювальної техніки та інформаційних технологій (Association for Computing Machinery (ACM) та Національного агентства із забезпечення якості вищої освіти України (НАЗЯВО), 2023–2025 рр.

Слід зазначити, що науковцем В. Кругликом не було враховано таку компетентність, як інформаційно-комунікаційна, як ключову компетентність.

Під інформаційно-комунікаційною компетентністю (ІКК) розуміємо підтверджену здатність особи автономно й відповідально використовувати на практиці ІКТ для задоволення власних індивідуальних потреб і розв'язування суспільно значущих, зокрема професійних задач, у певній предметній галузі або виді діяльності [26].

Володіння ІКК є необхідною умовою успішного навчання як у ЗВО, так і навчання впродовж життя, професійного розвитку та застосування можливостей використання ІКТ у навчальній та професійній діяльності вчителя в умовах сучасного інформаційного суспільства[137].

Формування ІКК студентів має здійснюватися з урахуванням спеціалізації їх підготовки, що потребує формування відповідного освітнього середовища. У ХХІ ст. таке середовище має всі ознаки синтетичного, про що зазначено в роботі групи авторів [24], [27].

Для підготовки бакалаврів комп'ютерних наук синтетичне освітнє середовище має бути веборієнтованим, щоб дати можливість студентам задіяти інноваційні засоби навчання, на що акцентує увагу науковець С. Литвинова у своїй роботі [28], а також має забезпечувати повсюдний доступ до основних засобів навчання, зокрема з програмування [15], [30].

Останнім часом, велика кількість науковців в своїх роботах приділяють увагу цифровій компетентності. Так, Л. Зубик визначає цифрову компетентність як здатність «працювати із цифровими носіями та впевнене й критичне використання технологій інформаційного суспільства для роботи, відпочинку та спілкування» впродовж життя [24], [131].

Дослідник С. Литвинова спрямовує розвиток цифрової компетентності студента за такими напрямками: інформаційна грамотність, робота з даними, комунікація, використання електронного контенту, відповідальність, розв’язання проблем[24], [30].

На підставі вивчення результатів наукових досліджень із проблеми компетентності фахівців галузі знань «Інформаційні технології» спеціальності бакалавра комп’ютерних наук було визначено, що під цифровою компетентністю бакалавра комп’ютерних наук (табл. 1.12) ми розуміємо здатність проводити теоретичні та експериментальні дослідження в галузі комп’ютерних наук; застосовувати математичні методи й алгоритмічні принципи в моделюванні, проектуванні, розробці та супроводі інформаційних технологій; здійснювати розробку, впровадження і супровід інтелектуальних систем аналізу й обробки даних організаційних, технічних, природничих і соціально-економічних систем [134].

Таблиця 1.12

Компетентності з програмування бакалавра з комп’ютерних наук

№	Складники компетентності	Зміст	Очікувані результати навчання бакалавра
1	Базові знання з програмування	Розуміння: алгоритмів та структур даних, конструкцій мов програмування, масивів, функцій(методів), файлів (JavaScript, C++, C#, Python, Java тощо)	Уміє створювати, налагоджувати та оптимізувати програми, використовуючи базові конструкції мов програмування
2	Об’єктно-орієнтоване програмування (ООП)	Знання парадигм програмування: абстракції інкапсуляції, наслідування, поліморфізму і вміння застосовувати їх у реальних проектах	Застосовує ООП для розробки інформаційних та програмних систем середнього рівня складності
3	Алгоритмічне мислення та аналіз ефективності алгоритмів	Знання класичних алгоритмів сортування, пошуку, графових обчислень та інших.	Оцінює ефективність програмних рішень, Виконує рефакторінг програмного коду , вибираючи оптимальні

№	Складники компетентності	Зміст	Очікувані результати навчання бакалавра
		Уміння аналізувати ефективність алгоритмів	структури даних і алгоритмів
4	Розробка веб-застосунків (Frontend, Backend)	Володіння сучасними технологіями створення веб-застосунків (HTML, CSS, JavaScript, React, Node.js, REST API)	Створює адаптивні вебзастосунки з використанням фреймворків і підходів клієнт–серверної архітектури
5	Бази даних і робота з інформаційними системами	Знання реляційних та нереляційних баз даних. Знання SQL, принципів нормалізації форм, проектування БД, роботи з ORM, транзакційності	Проектує та реалізує реляційні нереляційні бази даних; виконує запити оптимізації та керування даними
6	Командна розробка та використання систем контролю версій	Знання командної роботи, Git, GitHub/GitLab, основ інтеграційного тестування	Ефективно працює у команді розробників, використовує системи контролю версій для спільних проєктів
7	Забезпечення якості та тестування ПЗ	Уміння використання модульного, інтеграційного, системного тестування, тестування безпеки та продуктивності	Створює тести, виявляє та усуває помилки, оцінює якість програмного забезпечення
8	Інженерія програмного забезпечення	Розуміння життєвого циклу ПЗ, принципів Agile, Scrum, DevOps, документування та моделювання UML	Уміє планувати, проектувати та підтримувати ПЗ протягом усього життєвого циклу
9	Безпека програмних систем	Знання основ криптографії, захисту даних, управління доступом, виявлення вразливостей	Розробляє безпечні програмні рішення, дотримується принципів інформаційної безпеки
10	Професійна етика та відповідальність програміста	Знання етичного кодексу АСМ/IEEE, питань авторського права, етичного використання ІІІ та даних	Дотримується етичних норм і принципів професійної діяльності у сфері ІТ

Одним з ключових показників рівня фахових компетентностей майбутнього бакалавра комп'ютерних наук є вивчення переліку дисциплін циклів підготовки освітньо-професійної програми спеціальності «Комп'ютерні науки», які потрібно обов'язково щорічно переглядати, відповідно оновлювати робочі програми з переліком цих дисциплін, з урахуванням міжнародного досвіду та вимогами ІТ-галузі. З цією метою необхідно забезпечити достатню гнучкість освітніх програм, щоб вони могли оперативнo адаптуватися до змін, які відбуваються в ІТ-галузі та на ринку праці [24].

Тому для формування високого рівня компетентності майбутніх бакалаврів комп'ютерних наук необхідним є забезпечення тісної взаємодії між факультетами та кафедрами ЗВО й представниками ІТ-індустрії. Необхідність підготовки фахівців, які володіють не лише теоретичними знаннями, а й сучасними методами, технологіями та практичним досвідом роботи з інструментами, що використовується в професійній діяльності, зумовлює встановлення прямих контактів між ЗАВ та ІТ-компаніями вже на рівні етапу навчання. Основними формами такої співпраці можуть бути спільна організація практик, виконання курсових і дипломних проєктів під керівництвом фахівців ІТ-фірм, а також стажування викладачів в ІТ-компаніях. Зокрема компанія ЕРАМ щорічно проводить такі місячні програми [24].

Результатом навчання бакалаврів комп'ютерних наук в ЗВО має бути не засвоєний ними набір теоретичних знань, а підготовка їх до успішного виходу на ІТ-ринок праці, для чого знання повинні поєднуватися з практичним досвідом, навичками поведінки в професійному середовищі. Іншими словами, бакалавр комп'ютерних наук повинен володіти інтегральними, загальними та фаховими компетентностями. (див. розділ 3, підрозділ 3.1) [24].

Кінцевим показником якості освіти є високий рівень володіння фаховими компетентностями, зокрема компетентністю з програмування, затребуваність випускників роботодавців і їхнє кар'єрне зростання. У деяких зарубіжних ЗВО саме відсоток працевлаштування студентів-випускників є показником рівня якості отриманої освіти.

Наступним фактором, який впливає на рівень компетентності випускників є забезпечення якості викладання викладацького складу кафедр університетів. Це, передусім, рівень компетентностей і якість викладання науково-педагогічних та професорсько-викладацьких працівників, а також якість їхньої наукової діяльності та позитивна мотивація до професійного розвитку [24].

З метою підвищення рівня компетентності викладачів доцільно застосовувати різні форми професійного розвитку, зокрема: участь у тренінгах, майстер-класах, стажування в закладах вищої освіти України та за кордоном; взаємодію з роботодавцями; стажування в ІТ-компаніях із подальшим отриманням сертифікатів; залучення до викладання провідних фахівців ІТ-галузі; участь у всеукраїнських та міжнародних конференціях, семінарах і проєктах.

В процесі підготовки бакалаврів комп'ютерних наук, особливої уваги потребують ІКК викладача. Так, Н. Морзе, А. Кочарян досліджують розвиток понятійного апарату ІКК науково-педагогічного працівника університету, зокрема аналізують міжнародні документи, що стали підґрунтям для формування її змісту. Автори наголошують, що компетентність — це інтегрована характеристика особистості, яка охоплює знання, уміння, досвід діяльності та здатність ефективного застосовувати їх у професійній діяльності викладача [136].

Доповнюючи це визначення, О. Спирін зазначає, що інформаційно-комунікаційно-технологічна компетентність або ІКТ-компетентність є підтвердженою здатністю особистості практично використовувати інформаційно-комунікаційні технології для задоволення власних потреб і розв'язання суспільно значущих, зокрема професійних задач, у певній предметній галузі [12].

О. Кривонос констатує, що ІКК вчителя інформатики – це система знань, умінь, особистісних якостей вчителя інформатики, формування та розвиток яких дасть змогу розв'язувати типові професійні задачі, вирішувати проблеми,

котрі виникають у реальних ситуаціях педагогічної діяльності, з використанням усього різноманіття комп'ютерних засобів, а це передбачає здатність до фахового зростання в галузі інформаційно-комунікаційних технологій та до виконання ролі провідного фахівця в педагогічному колективі [137].

Водночас у наукових працях О. Овчарук [138] розглянуто міжнародні тенденції розвитку цифрової компетентності викладачів, зокрема в системі післядипломної освіти. Дослідниця аналізує такі стратегічні документи, як Рамка цифрової компетентності для громадян (DigComp) та Рамка цифрової компетентності для освітян (DigCompEdu), розроблені Європейським об'єднаним дослідницьким центром, і визначає їх ключові складові [138].

За основне визначення компетентності авторка пропонує взяти визначення, яке пропонують у своїй роботі автори Р. Вуорікарі (R. Vuorikari) [139], І. Пуні (Y. Punie) [138], С. Каррето Гомес S. Carretero Gomez) [139], Л. Ван ден Бранде (L. Van den Brande) [139]. Цифрову компетентність в означеному документі визначено як упевнене та ґрунтовне користування засобами інформаційно-комунікаційних технологій (ІКТ) у таких сферах, як робота (можливість працевлаштування), освіта, дозвілля, залучення та діяльність у житті суспільства, що є життєво необхідними для щоденного соціально-економічного життя [139].

Нині викладачі активно починають застосовувати такі нові форми навчання як: змішане та дистанційне навчання, віртуальний клас, роботу з яким ми можемо бачити у роботах науковця С. Г. Литвиної [110].

Наступним показником, який суттєво впливає на формування компетентності, є самостійна робота студентів-програмістів. Для забезпечення високого рівня успішності майбутнього бакалавра комп'ютерних наук очного та заочного форм навчання, отримання та систематизації навчальних відомостей, постійного підвищення професійного рівня майбутнього програміста викладачеві потрібно правильно організувати самостійну роботу студента. Це, насамперед, детальне складання плану самостійної роботи студента по вибраній дисципліні, обов'язкова розробка методичних вказівок до

виконання лабораторних та розрахунково-графічних робіт, курсового та дипломного проектування. При цьому студент повинен знати які теми призначені для самостійного вивчення, які види та форми самостійної роботи будуть застосовані, яка форма контролю і строки виконання[24].

Сучасні вимоги до бакалавра комп'ютерних наук включають обов'язковий складник етики та безпеки використання ШІ. Студент повинен не лише вміти генерувати код за допомогою ChatGPT чи GitHub Copilot, а й критично оцінювати пропозиції інтелектуальних агентів, усвідомлювати межі їх застосування та відповідальність за безпеку згенерованих рішень. Це формує цифрову професійну індивідуальність фахівця, здатного до ефективної праці в умовах розвитку інтелектуального вебу.

Розглянемо об'єднання стандарту F3, рамки DigComp 3.0 та веборієнтовані інструменти (табл. 1.13).

Таблиця 1.13

**Вимоги до компетентності бакалавра комп'ютерних наук у
веборієнтованому середовищі**

Група вимог	Напрямок DigComp 3.0	Конкретна вимога (Skill)	Веборієнтовані й інструментарій
Архітектурна	Створення контенту	Проектування хмарних систем та API	Netlify, Vercel, AWS
Співпраця	Комунікація	Управління версіями та командна розробка	GitHub, GitLab, Bitbucket
Інтелектуальна	Розв'язання проблем	Ефективний промпт-інжиніринг та дебагінг за допомогою AI	ChatGPT, Copilot, AI Tutors
Методологічна	Інформаційна грамотність	Робота з Big Data та інтерактивною документацією	Jupyter Notebooks, MDN Web Docs

Для успішного виконання самостійної роботи викладачі кафедр ознайомлюють майбутніх бакалаврів комп'ютерних наук, особливо першого курсу, з формами та методами організації навчального процесу в університеті, основами наукової організації праці, методикою самостійної роботи, критеріями оцінювання якості самостійної роботи; метою, засобами, трудомісткістю, строками виконання, формами контролю самостійної роботи майбутніх бакалаврів комп'ютерних наук [40].

Для організації самостійної роботи майбутнього бакалавра комп'ютерних наук необхідні такі умови: мотивація студента до самостійної роботи; наявність і доступність навчально-методичного забезпечення та довідкового матеріалу; наявність комп'ютерних класів; система регулярного контролю якості виконаної самостійної роботи; консультаційна допомога викладача [40].

Проведене дослідження підтверджує, що саме компетентність з програмування стає основою цифрової та професійної підготовки майбутніх бакалаврів комп'ютерних наук, забезпечуючи перехід від репродуктивного використання вебресурсів до продуктивного створення інтелектуальних систем.

У контексті вимог 2026 року та рамки DigComp 3.0 ця компетентність не обмежується лише знанням синтаксису мов, а охоплює здатність до алгоритмічного мислення, що дозволяє реалізувати восьмирівневу систему опанування навичками – від базових вправ у вебтренажерах до експертного проєктування архітектури хмарних застосунків, що є фундаментом формування цифрової професійної компетентності в умовах Веб 4.0 [24].

Водночас аналіз вітчизняного стану підготовки виявив суттєвий технологічний розрив між високою готовністю студентів до використання інновацій (зокрема ШІ та сучасних ІСР) та фрагментарністю освітнього середовища ЗВО. Переважання інструментального підходу, де вебресурси використовуються переважно для розміщення матеріалів, а не як дидактична основа для написання та тестування коду, гальмує підготовку фахівців, здатних задовольнити динамічні потреби ІТ-ринку. Виявлені суперечності та

необхідність подолання відірваності академічного навчання від промислових стандартів зумовлюють розроблення авторської моделі використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук.

Висновки до розділу 1

Проведене дослідження теоретико-методичних основ підготовки бакалаврів комп'ютерних наук дозволяє сформулювати низку концептуальних положень, які стають підґрунтям для подальшого дослідження проблеми, а саме:

- встановлено, що в умовах системної цифрової трансформації суспільства (2025–2026 рр.) компетентність з програмування трансформується з набору технічних навичок в основну цифрову компетентність фахівця;
- визначено, що компетентність з програмування базується на здатності до алгоритмічного мислення в межах інтелектуальних веборієнтованих систем, де фахівець діє у симбіозі зі штучним інтелектом (Веб 4.0);
- обґрунтовано доцільність дослідження процесу навчання програмування як поступового професійного зростання за 8-рівневою шкалою DigComp 3.0 – від репродуктивних вправ у браузері до експертного архітектурного проектування хмарних систем;
- встановлено трансформацію компетентності з програмування у базову цифрову компетентність фахівця в умовах Веб 4.0, що підтверджено порівняльний аналіз трьох світових моделей освіти (Атлантичної, Континентальної та Східноазійської) виявив глобальне зміщення ІТ-підготовки до хмаро орієнтованої парадигми (наприкладі Harvard CS50) та використання ШІ як повноцінного інтелектуального асистента (Stanford);
- доведено, що використання веборієнтованих середовищ розробки (ХО) та систем автоматизованої перевірки (Online Judges) є критичним фактором подолання технічних бар'єрів на початкових етапах навчання студентів;

– моніторинг сучасного стану підготовки у вітчизняних ЗВО засвідчив суперечність між високою готовністю студентів до інновацій (95,7% самостійно опанували ІІІ, 61,5% використовують VS Code) та фрагментарністю університетського середовища (розподіл між Moodle (45%) та Google Classroom (44%));

– підтверджено необхідність відмови від «інструментального» підходу (веб як сховище файлів) на користь створення цілісної дидактичної системи, де вебтехнології стають інтелектуальним середовищем для написання, аудиту та налаштуванню програмного коду;

– доведено, що для подолання розриву між академічною освітою та вимогами ІТ-ринку недостатньо механічного впровадження ІКТ, а потрібна глибока модернізація методики формування компетентності з програмування.

Отже, виявлені вимоги та технологічний розрив у вітчизняній освіті зумовлюють гостру наукову потребу в розробленні авторської моделі використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук, яка буде представлена в наступному розділі.

Список використаних джерел до Розділу 1

1. World Economic Forum. The Future of Jobs Report 2025, 2025. URL: <https://www.weforum.org/publications/the-future-of-jobs-report-2025/> (дата звернення: 27.03.26)

2. World Economic Forum. Future of Jobs Report 2025: 78 Million New Job Opportunities by 2030 but Urgent Upskilling Needed to Prepare Workforces, 2025. URL: https://www.weforum.org/press/2025/01/future-of-jobs-report-2025-78-million-new-job-opportunities-by-2030-but-urgent-upskilling-needed-to-prepare-workforces/?utm_source (дата звернення: 27.03.26).

3. European Training Foundation. Bridging the skills gap: embracing digital transformation. *ETF*, 2025. URL: <https://www.etf.europa.eu/en/document-attachments/bridging-skills-gap-embracing-digital-transformation> (дата звернення: 27.03.26).

4. Міністерство цифрової трансформації України. *Проекти*. URL: <https://thedigital.gov.ua/projects> (дата звернення: 27.03.26).

5. Концепція розвитку цифрової економіки та суспільства України на 2018–2020 роки та затвердження плану заходів щодо її реалізації : розпорядження Кабінету Міністрів України від 17 січ. 2018 р. № 67-р. URL: <https://zakon.rada.gov.ua/laws/show/67-2018-%D1%80#Text> (дата звернення: 27.03.26).

6. Кремень В. Г., Биков В. Ю., Ляшенко О. І., Литвинова С. Г., Луговий В. І., Мальований Ю. І., Пінчук О. П., Топузов О. М. *Науково-методичне забезпечення цифровізації освіти України: стан, проблеми, перспективи*. Вісник НАПН України, 2022. Т. 4, № 2. DOI: <https://doi.org/10.37472/v.naes.2022.4223> (дата звернення: 27.03.26).

7. Биков В. Ю., Буров О. Ю. Цифрове навчальне середовище: нові технології та вимоги до здобувачів знань. *Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців: методологія, теорія, досвід, проблеми*, 2020. Вип. 55. С. 11–22. DOI: <https://doi.org/10.31652/2412-1142-2020-55-11-22> (дата звернення: 27.03.26).

8. Биков В. Ю. Моделі організаційних систем відкритої освіти: монографія. Київ : Атіка, 2009. 684 с. URL: <https://lib.iitta.gov.ua/845/> (дата звернення: 27.03.2026).

9. Биков В. Ю. Проектний підхід і дистанційне навчання у професійній підготовці управлінських кадрів. *Кримські педагогічні читання : матеріали Міжнародної наукової конференції, 12–17 вересня 2001 р.* / за ред. С. О. Сисоєвої, О. Г. Романовського. Харків : НТУ «ХПІ», 2001. С. 30–50. URL: <https://lib.iitta.gov.ua/id/eprint/498/> (дата звернення: 27.03.2026).

10. Спирін О. М., Ляшенко О. І., Литвинова С. Г., Мальований Ю. І., Пінчук О. П. Цифрова трансформація освіти: штучний інтелект у сучасному освітньому просторі : наук.-аналіт. доповідь. Київ: ІЦО НАПН України, 2025. URL: <https://lib.iitta.gov.ua/id/eprint/747330/> (дата звернення: 27.03.26).

11. Спирін О. М., Вакалюк Т. А. Web-орієнтовані технології навчання основ програмування майбутніх учителів інформатики. *Математика та інформатика у вищій школі: виклики сучасності* : матеріали Всеукр. наук.-практ. конф., Вінниця, 2017. С. 61–65. URL: https://eprints.zu.edu.ua/25123/?utm_source=chatgpt.com (дата звернення: 27.03.26).
12. Спирін О. М. Інформаційно-комунікаційні та інформатичні компетентності як компоненти системи професійно-спеціалізованих компетентностей вчителя інформатики. Інформаційні технології і засоби навчання, 2009. Т.5 (13). URL: <https://journal.iitta.gov.ua/index.php/itlt/article/view/183/169> (дата звернення: 03.04.2026).
13. Спирін О. М., Вакалюк Т. А. Критерії добору відкритих Web-орієнтованих технологій навчання основ програмування майбутніх учителів інформатики. *Інформаційні технології і засоби навчання*, 2017. Т. 60, № 4. С. 275–287. DOI: <https://doi.org/10.33407/itlt.v60i4.1815> (дата звернення: 27.03.2026).
14. Vakaliuk T. A., Spirin O. M., Kontsedailo V. V. Criteria for selecting open web-oriented technologies for teaching the basics of programming to future software. *Educational Technology Quarterly*, 2021. С. 73–86. URL: <https://doi.org/10.55056/etq> (дата звернення 27.03.26)
15. Vakaliuk T. A., Osadchy V. V., Pinchuk O. P. Educational transformation in the digital age: artificial intelligence, digital competence, and innovation trends from DigiTransfEd 2025. *DigiTransfEd Workshop Proceedings*, 2025. Vol. 4096. P. 1-8. URL: <https://lib.iitta.gov.ua/id/eprint/747226/> (дата звернення: 27.03.26).
16. Осадчий В. В., Осадча К. П. Теорія і практика створення комп'ютерних програм навчального призначення. *Теорія та методика електронного навчання* : зб. наук. пр. Кривий Ріг, 2012. Вип. 3. С. 250–256.

URL: <https://www.researchgate.net/publication/359485465> (дата звернення: 27.03.26).

17. Проскура С. Л., Литвинова С. Г. Моніторинг використання WEB-орієнтованих технологій бакалаврами комп'ютерних наук в процесі вивчення програмування. *Інформаційно-цифровий освітній простір України: трансформаційні процеси і перспективи розвитку : матеріали методологічного семінару НАПН України (4 квітня 2019 р.)*. Київ : НАПН України, 2019. С. 211–219. URL: <https://lib.iitta.gov.ua/id/eprint/717123/> (дата звернення: 27.03.26).

18. Proskura S. L., Lytvynova S. H., Kronda O. P. The use of WEB-oriented technologies in the process of WEB-programming teaching for technical universities students. *Educational Dimension*, 2021. №5. URL: <https://lib.iitta.gov.ua/id/eprint/728607/1/Proskura-Lytvynova-Kronda-2021.pdf> DOI: <https://doi.org/10.31812/educdim.4724> (дата звернення: 27.03.26).

19. Литвинова С. Г. Віртуальний клас для організації індивідуального навчання учнів. *Інформаційні технології і засоби навчання*, 2011. № 1 (21). URL: https://www.journal.iitta.gov.ua/index.php/itlt/article/view/332?utm_source (дата звернення: 27.03.26).

20. Проскура С. Л., Литвинова С. Г. Особливості підготовки майбутніх бакалаврів комп'ютерних наук в університетах США. *Наукова молодь-2018 : матеріали VI Всеукр. Наук.-практ. конференції*. 2018. https://lib.iitta.gov.ua/id/eprint/717127/1/ОсобливПідготовкиБакалаврСША_NaukMolod2018-Тези%20молодь.pdf?utm_source (дата звернення: 27.03.26).

21. Проскура С. Л., Литвинова С. Г. Підготовка фахівців з інформаційних технологій у закладах вищої освіти: стан, проблеми і перспективи. *Інформаційні технології в освіті*, 2018. № 2(35). Р. 72–88. DOI: <https://doi.org/10.14308/ite000668> (дата звернення: 27.03.2026).

22. Lytvynova S. H., Rashevskaya N. V., Proskura S. L. The use of artificial intelligence in teaching students programming languages *Proceedings of the IX International Workshop on Professional Retraining and Life-Long Learning using*

ICT: Person-oriented Approach (3L-Person 2024), co-located with ICTERI 2024. CEUR Workshop Proceedings, 2024. Vol. 3781. P. 10–29. URL: <https://ceur-ws.org/Vol-3781/paper01.pdf> (дата звернення: 27.03.26).

23. Проскура С. Л., Сучасні підходи до викладання мов програмування в закладах вищої освіти/Проскура С.Л.*У Всеукраїнської науково-практичної конференції «Наукова молодь-2017»*. Інститут інформаційних технологій і засобів навчання НАПН України, 2017. С. 313-315. URL: https://lib.iitta.gov.ua/id/eprint/712028/1/Proskura_Nauk_mol2017.pdf (дата звернення: 27.03.26)

24. Проскура С. Л., Литвинова С. Г. Формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*, 2019. № 2 (20). С. 137–147. URL: https://lib.iitta.gov.ua/id/eprint/717302/1/PROFESSIONAL%20COMPETENCY_Proskura_Lytvynova_FMO.pdf (дата звернення: 27.03.26).

25. Проскура С. Л., Литвинова С. Г. Огляд компетентностей майбутніх бакалаврів комп'ютерних наук. Звітна наукова конференція Інституту інформаційних технологій і засобів навчання НАПН України, 2018. С.31 – 34 URL: <http://lib.iitta.gov.ua/711730/1/Збірник%20тез%20звітна%202018-output.pdf> (дата звернення: 27.03.26).

26. Литвинова С. Г. Модель використання системи комп'ютерного моделювання для формування компетентностей учнів з природничо-математичних предметів. *Фізико-математична освіта*, 2019. Т. 1(19). С. 108– 115. URL: <https://repository.sspu.edu.ua/items/4d7f10c8-e2be-4ea8-8aad-c255b14fb32e> (дата звернення: 27.03.26).

27. Пінчук О. П., Литвинова С. Г., Буров О. Ю. Синтетичне навчальне середовище – крок до нової освіти. *Інформаційні технології і засоби навчання*, 2017. Т. 60, № 4. С. 28– 45. URL: https://journal.iitta.gov.ua/index.php/itlt/article/view/1831?utm_source (дата звернення: 27.03.26).

28. Литвинова С. Г. Хмароорієнтоване навчальне середовище школи: від навчального кабінету до віртуальних методичних предметних об'єднань учителів. *Освітні технології та суспільство*, 2014. Vol. 17, No. 1. P. 457–468 URL: https://lib.iitta.gov.ua/id/eprint/6633/1/9.pdf?utm_source (дата звернення: 27.03.26).

29. Литвинова С. Г. On-line навчальне середовище вчителя-предметника загальноосвітнього навчального закладу. *Інформаційні технології і засоби навчання*, 2010. № 5(19). URL: <https://lib.iitta.gov.ua/id/eprint/195/1/356-1047-1-PB.pdf> (дата звернення: 27.03.26).

30. Литвинова С. Г. Віртуальний клас як комп'ютерно-орієнтоване навчальне середовище вчителя загальноосвітнього навчального закладу. *Інформаційні технології і засоби навчання*, 2011. № 2(22). URL: <https://journal.iitta.gov.ua/index.php/itlt/article/view/331/387> (дата звернення: 27.03.26).

31. Литвинова С. Г. Технології навчання учнів у хмаро орієнтованому навчальному середовищі загальноосвітнього навчального закладу. *Інформаційні технології і засоби навчання*, 2015, Том 47, №3. URL: <https://journal.iitta.gov.ua/index.php/itlt/article/view/1239/927> (дата звернення: 27.03.26).

32. Proskura S. L., Lytvynova S. H. The approaches to Web-based education of computer science bachelors in higher education institutions. *CTE Workshop Proceedings*, 2020. Vol. 7. P. 609–625. DOI: <https://doi.org/10.55056/cte.416> (дата звернення: 27.03.2026).

33. Литвинова С. Г. Хмаро орієнтоване навчальне середовище загальноосвітньої школи. *Хмарні технології в освіті : матеріали 5-го воркшопу CTE 2017*. Кривий Ріг, 2017. С. 7–12. (*CEUR Workshop Proceedings*, Vol. 2168). URL: <http://ceur-ws.org/Vol-2168/paper2.pdf> (дата звернення: 27.03.2026).

34. Литвинова С. Г. Поняття й основні характеристики хмаро орієнтованого навчального середовища середньоїшколи. *Інформаційні технології і засоби навчання*, 2014, Т. 40, №2.

<https://journal.iitta.gov.ua/index.php/itlt/article/view/970/756.pdf> (дата звернення: 27.03.2026).

35. Литвинова С. Г. Технологія «перевернутого» навчання у хмаро орієнтованому освітньому середовищі як складник розвитку медіаосвіти в середній школі. *Медіасфера і медіаосвіта: специфіка взаємодії в сучасному соціокультурному просторі*. Могильов, 2015.

36. Литвинова С. Г. Етапи, методичні підходи та принципи хмароорієнтованого навчального середовища навчального закладу *Комп'ютер у школі та сім'ї*, 2014. № 4(116). С. 5-11. URL: https://lib.iitta.gov.ua/id/eprint/6635/?utm_source (дата звернення: 27.03.26).

37. Proskura S. L., Lytvynova S. H., Kronda O. P. Students' academic achievement assessment in higher education institutions. ICTERI 2020: Proceedings of the 16th International Conference on ICT in Education, Research and Industrial Applications. CEUR Workshop Proceedings, 2020. Vol. 2732. P. 734–746 URL: <http://ceur-ws.org/Vol-2732/20200734.pdf> (дата звернення: 27.03.2026).

38. Проскура С. Л., Литвинова С. Г., Кронда О. П. Засоби організації дистанційного навчання в період карантину 2020 року в закладах вищої освіти України. Екстрене дистанційне навчання в Україні: монографія/ за ред. В. М. Кухаренка, В. В. Бондаренка. Харків : КП "Міська друкарня", 2020. С. 299–313. URL: https://duan.edu.ua/images/News/UA/Departments/Management/2020/monograph_ekstr_dyst_navch.pdf (дата звернення: 27.03.26).

39. Proskura S. L., Lytvynova S. H., Kronda O. P. Demeshkant N. Mobile learning approach as a supplementary approach in the organization of the studying process in educational institutions. ICTERI 2020: Proceedings of the 16th International Conference on ICT in Education, Research and Industrial Applications. CEUR Workshop Proceedings, 2020. P. 650–664. URL: <http://ceur-ws.org/Vol-2732/20200650.pdf> (дата звернення: 27.03.26).

40. Proskura S. L., Lytvynova S. H. Organization of independent studying of future bachelors in computer science within higher education institutions of

Ukraine. *ICTERI 2018 (3L-Person 2018)*, 2018. С. 348–358. URL: [http://ceur-
ws.org/Vol-2104/paper_160.pdf](http://ceur-
ws.org/Vol-2104/paper_160.pdf)

41. Проскура С. Л., Литвинова С. Г., Кронда О. П. Оцінювання рівня знань бакалаврів комп'ютерних наук у закладах вищої освіти. Сучасні тенденції та фактори розвитку педагогічних та психологічних наук в Україні та країнах ЄС : матеріали міжнародної науково-практичної конференції (Люблін, Польща), 2020.

URL: <http://www.baltijapublishing.lv/omp/index.php/bp/catalog/download/68/1534/3505-1> (дата звернення: 27.03.26).

42. Сороко Н. В. Використання вебтехнологій у професійній діяльності вчителів філологічної спеціальності. *Комп'ютер у школі та сім'ї*, 2014. № 1. С. 33–37. URL: http://nbuv.gov.ua/UJRN/komp_2014_1_9 (дата звернення: 27.03.26).

43. Круглик В. С. Веб-орієнтовані навчальні середовища у професійній підготовці майбутніх інженерів-програмістів. *Ukrainian Journal of Educational Studies and Information Technology*, 2017. Vol. 5, No. 2. Р. 19–22. URL: <http://www.ojs.mdpu.org.ua/index.php/itse/article/download/1857/2598> (дата звернення: 27.03.26).

44. Круглик В. С. Система підготовки майбутніх інженерів-програмістів: автореф. дис. докт. пед. наук, 2018. URL: http://phd.znu.edu.ua/page/aref/09/Kruglik_aref.pdf (дата звернення: 27.03.26).

45. Кухаренко В. М. Системний підхід до змішаного навчання. *Інформаційні технології в освіті*, 2015. № 24. С. 53–67. URL: <https://ite.kspu.edu/index.php/ite/article/download/147/156/272> (дата звернення: 27.03.2026).

46. Кухаренко В. М., Березенська С. М., Бугайчук К. Л., Олійник Т. О., Рибалко О. В. Теорія і практика змішаного навчання: монографія. Харків: НТУ «ХПІ», 2016. 284 с. URL: <http://repository.kpi.kharkov.ua/handle/KhPI-Press/23536> (дата звернення: 27.03.2026).

47. Триус Ю. В., Герасименко І. В. Комбіноване навчання як інноваційна освітня технологія у вищій школі. *Теорія і методика електронного навчання*. Кривий Ріг, 2012. Вип. 3. С. 299–308.

48. Семеріков С. О., Стрюк А. М. Моделі комбінованого навчання. *Вісник Дніпропетровського університету імені Альфреда Нобеля. Серія «Педагогіка і психологія»*, 2012. № 2 (4). С. 47–55.
URL: <https://lib.iitta.gov.ua/id/eprint/106652/1/Моделі%20комбінованого%20навчання.pdf> (дата звернення: 27.03.2026).

49. Стрюк К. М. Шляхи формування професійної компетентності майбутніх молодших спеціалістів із комп'ютерної інженерії. *Педагогічні науки: реалії та перспективи*, 2018. Вип. 63. С. 173–177.
URL: <https://www.chasopys.ps.npu.kiev.ua/archive/63-2018/63-2018.pdf> (дата звернення: 27.03.2026).

50. Лісецький К. А. Модель змішаного навчання в системі вищої освіти. *Міжкультурна комунікація у європейському мовному просторі : матеріали міжнар. наук.-практ. інтернет-кон.*, 2015.
URL: <https://ela.kpi.ua/handle/123456789/17816> (дата звернення: 27.03.2026).

51. Kovaliuk T., Kobets N. Integration of IT Education in Ukraine into the European Educational Space. *ICT in Education, Research and Industrial Applications: Integration, Harmonization and Knowledge Transfer. CEUR Workshop Proceedings*, 2019. Vol. 2387. P. 385–397
URL: https://elibrary.kubg.edu.ua/id/eprint/28116/1/T_Kovaliuk_N_Kobets_CEUR_WS_Vol_2387_paper_385-397.pdf (дата звернення: 27.03.2026).

52. Огнівчук Л. М. Оцінювання навчальних досягнень студентів вищих навчальних закладів на основі компетентнісного підходу. *Освітологічний дискурс*, 2014. № 3 (7). С. 154–166.
URL: https://elibrary.kubg.edu.ua/id/eprint/6002/?utm_source (дата звернення: 27.03.26).

53. Пометун О., Гупан Н. Таксономія Б. Блума і розвиток критичного мислення школярів. *Український педагогічний журнал*, 2019. № 3. С. 52–58.

DOI: <https://doi.org/10.32405/2411-1317-2019-3-50-58> (дата звернення: 27.03.2026).

54. Harris D. Web 2.0 Evolution Into the Intelligent Web 3.0. *Lulu.com*, 2008. Р. 148.

URL: https://books.google.com.ua/books/about/Web_2_0_Evolution_into_The_Intelligent_W.html?id=FZQ7gYAIUMoC&redir_esc=y (дата звернення: 27.03.26).

55. O'Reilly T. What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *International Journal of Digital Economics*, 2007. № 65. Р. 17–37. URL: <https://mpira.ub.uni-muenchen.de/4578/> (дата звернення: 27.03.26).

56. Calacanis J. Web. 3.0, the "official" definition. URL: <http://calacanis.com/2007/10/03/web-3-0-the-official-definition/> (дата звернення: 27.03.26).

57. Pollock J. T., Watson M. *Adaptive Information: Improving Business Through Semantic Interoperability, Grid Computing, and Enterprise Integration*. New Jersey: Wiley-Interscience, 2004. 440 с. URL: <https://surl.li/bzttnh> (дата звернення: 27.03.26).

58. Pollock T. *Semantic Web For Dummies*. Hoboken, NJ: John Wiley & Sons, 2009. 384 p. URL: <https://books.google.com/books?id=DDiS9KYNIqIC> (дата звернення: 27.03.2026).

59. Watson M. *Scripting Intelligence: Web 3.0 Information Gathering and Processing*. Springe,. 2009. 392 p.

60. Watson C., Frederick W. B. Li. Failure Rates in Introductory Programming Revisited *ACM Transactions on Computing Education*, 2014. DOI: <https://doi.org/10.1145/2591708.2591749> (дата звернення: 27.03.26).

61. Shanley N., Martin F., Hite N. et al. Teaching programming online: design, facilitation and assessment strategies and recommendations for high school teachers. *TechTrends*, 2022. Vol. 66. Р. 483–494. DOI: <https://doi.org/10.1007/s11528-022-00724-x> (дата звернення: 27.03.26).

62. Truong N. A. Web-based programming environment for novice programmers : PhD thesis. Queensland, 2007. 286 p. URL: <http://eprints.qut.edu.au/16471/> (дата звернення: 27.03.26).
63. Edwards S. H., Perez-Quinones M. A. Web-CAT: Automatically Grading Programming Assignments / S. H. Edwards, M. A. Perez-Quinones. ACM SIGCSE Bulletin, 2008. DOI: <https://doi.org/10.1145/1384271.1384371> (дата звернення: 27.03.26).
64. Paiva J. C., Leal J. P., Figueira Á. Automated Assessment in Computer Science Education: A State-of-the-Art Review. *ACM Transactions on Computing Education*, 2022. Vol. 22, Issue 3. Art. 34. P. 1–40. DOI: <https://doi.org/10.1145/3513140> (дата звернення: 27.03.2026).
65. Gong X., Xu W., Qiao A. Exploring undergraduates' computational thinking in prompt-assisted programming learning. *International Journal of Educational Technology in Higher Education*, 2025. V. 22, P. 51, DOI: <https://doi.org/10.1186/s41239-025-00552-y> (дата звернення: 27.03.26).
66. Lagakis P., Demetriadis S., Psathas G. Automated grading in coding exercises using large language models. *Lecture Notes in Networks and Systems*, 2024. 936 p. DOI: https://doi.org/10.1007/978-3-031-54327-2_37 (дата звернення: 27.03.26)
67. Tsai C. - W. et al. () The effects of online peer-facilitated learning and distributed pair programming. *Computers & Education*, 2023. V 203. Art. 104849. DOI: <https://doi.org/10.1016/j.compedu.2023.104849> (дата звернення: 27.03.26).
68. Baartman L. K. J., Quinlan K. M. Assessment and feedback in higher education reimaged. *International Journal of Leadership in Education*, 2023. Vol. 27, No. 1. P. 57–67. DOI: <https://doi.org/10.1080/13603108.2023.2283118>
69. Alfaleh M. Sustainable AI-Driven Assessment in Higher Education. *Sustainability*, 2026. Vol. 18, No. 2. Article 785. DOI: <https://doi.org/10.3390/su18020785> (дата звернення: 27.03.2026).

70. European Commission. *DigComp 3.0 Framework Overview and Implementation Guidance*. Luxembourg: Publications Office, 2023. URL: <https://surl.lu/umyhsg> (дата звернення: 27.03.26).

71. Проскура С. Л. Використання WEB-орієнтованих технологій в закладах вищої освіти. *Наукова молодь-2019* : збірник матеріалів VII Всеукр. наук.-практ. конф. молодих учених. Київ : ЦП «Компринт», 2019. С. 39-42. URL: <https://lib.iitta.gov.ua/id/eprint/718530/2/Збірник%20Наукова%20молодь%202019.pdf>.

72. Котяк В. В. Web-орієнтовані системи тестування навчальних досягнень. *Наукові записки Ніжинського державного університету ім. М. Гоголя. Психолого-педагогічні науки*. Ніжин, 2011. № 10. URL: <https://moodle.ndu.edu.ua/file.php/1/NaykZap2010N11/mtp/mtp8.pdf> (дата звернення: 27.03.26).

73. Baker C. J. O., Cheung K. - H. Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences. *Springer*, 2006. 450 p.

74. Online computer science courses for students. Code.org, 2025. URL: <https://code.org/cs/students/online-courses> (дата звернення: 27.03.26).

75. Billy B., L. Lim Teaching Web development technologies in CS/IS curricula. *Proceedings of the ACM SIGCSE Conference*, 1998. V. 30(1). P. 107-111. DOI: <https://doi.org/10.1145/274790.273172> (дата звернення: 27.03.26).

76. The best online learning platforms in 2026. *iSpring LMS*, 2026. URL: <https://www.ispring.com/knowledge-hub/best-online-learning-platforms> (дата звернення: 27.03.26).

77. Yağci M. A Web-based blended learning environment for programming languages: students' opinions / M. Yağci. *Journal of Education and Training Studies*, 2017. Vol. 5, No. 3. DOI: <https://doi.org/10.11114/jets.v5i3.2118> (дата звернення: 27.03.26).

78. Ogunlade O. Impact of web-based learning tools in instructional delivery of computer programming / O. Ogunlade та ін. *Universal Journal of Educational*

Research, 2022. V. 10(7). P. 476-487. DOI: <https://doi.org/10.13189/ujer.2022.100706> (дата звернення: 27.03.26).

79. Fonseca I., Martins N. , Lopes F. A web-based platform and a methodology to teach programming languages in electrical engineering education – evolution and student feedback. *32nd Annual Conference of the European Association for Education in Electrical and Information Engineering (EAEEIE)*, 2023. P. 1-3, <https://doi.org/10.1109/EAEEIE54893.2022.9820594> (дата звернення: 27.03.26).

80. Hwang W. - Y., Wang C. - Y., Huang Y. - M & Huang. S. A web-based programming learning environment to support cognitive development. *Interacting with Computers*, 2008. V. 6(20). P. 524-534. DOI: <https://doi.org/10.1016/j.intcom.2008.07.002> (дата звернення: 27.03.26).

81. Maryono D. et al. NgodingSeru.com: an adaptive e-learning system with gamification to enhance programming problem-solving skills. *Discover Educatio.*, 2025. V. 4. 157. DOI: <https://doi.org/10.1007/s44217-025-00581-9> (дата звернення: 27.03.26).

82. Pereira Cipriano B., Fachada N., Alves P. Drop Project: An automatic assessment tool for programming assignments. *SoftwareX*, 2022. 18, 101079, DOI: <https://doi.org/10.1016/j.softx.2022.101079> (дата звернення: 27.03.26).

83. GraJefe Feitosa Y., Silva M. A. G. Systematic mapping of problem solving in introductory programming courses, 2021. P. 358-367. DOI: <https://doi.org/10.5753/wei.2021.15927> (дата звернення: 27.03.26).

84. Соколова І. В. Управління вищою освітою у зарубіжних країнах. *Неперервна професійна освіта: теорія і практика*, 2014. Вип. 3-4. С. 98-105Р. URL: http://nbuv.gov.ua/UJRN/NPO_2014_3-4_21 (дата звернення: 27.03.26).

85. HarvardX: CS50's Introduction to Computer Scienc. Harvard University. HarvardX: CS50's Introduction to Computer Science : online course. URL: <https://www.edx.org/learn/computer-science/harvard-university-cs50-s-introduction-to-computer-science> (дата звернення: 27.03.26).

86. Massachusetts Institute of Technology. Introduction to Computer Science and Programming in Python : course 6.0001. – URL: <https://ocw.mit.edu/courses/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016/> (дата звернення: 27.03.26).
87. Massachusetts Institute of Technology. EECS is where the future is invented : official webpage. <https://www.eecs.mit.edu/> (дата звернення: 27.03.26).
88. Stanford University. CS106A: Programming Methodology : course page. <https://web.stanford.edu/class/cs106a/> (дата звернення: 27.03.26).
89. University of California, Berkeley. Data 8: The Foundations of Data Science : course website. URL: <https://data8.org/> (дата звернення: 27.03.26).
90. University of California, Berkeley. CS 61A: Structure and Interpretation of Computer Programs : course website URL: <https://cs61a.org/> (дата звернення: 27.03.26).
91. Jupyter Notebooks and JupyterHub. *UC Berkeley Data Science Curriculum Guide*. <https://curriculum-guide.datahub.berkeley.edu/technology/introduction-to-jupyter/> (дата звернення: 27.03.26).
92. Carnegie Mellon University. Autolab Project: Simplify Teaching : official site URL: <https://autolabproject.com/> (дата звернення: 27.03.26).
93. Педагогічна компаративістика і міжнародна освіта: навчально-методичний посібник для здобувачів вищої освіти за спеціальністю 011 «Освітні, педагогічні науки» та 013 «Початкова освіта». Укладачі: Сойчук Р.Л., Колупаєва Т.Є., Міщенья О.М. Рівне: РДГУ, 2020. 243 с. <http://repository.rshu.edu.ua/id/eprint/9095/1/Сойчук%20Р.%2С%20Колупаєва%20Т.%2С%20Міщенья%20О.pdf>
94. Virtual Programming Lab https://moodle.org/plugins/mod_vpl (дата звернення: 27.03.26).
95. Seoul Accord Graduate Attributes : official document. Seoul Accord Secretariat, 2015. 12 p.

URL: <https://www.seoulaccord.org/images/epost/ca327008a4b5cb655f336e8de83bc796.pdf> (дата звернення: 27.03.2026).

96. Рейтинг університетів 2025 — F3 «Комп'ютерні науки». (дата звернення: 27.03.26).

URL: <https://www.education.ua/vstup/university-ranking/kompiuterni-nauky/> (дата звернення: 27.03.26).

97. Education.ua підготував рейтинг закладів вищої освіти 2025. URL: <https://www.education.ua/news/2025/07/21/educationua-pidhotuvav-reitynh-zakladiv-vyshchoi-osvity-2025/> (дата звернення: 27.03.26).

98. Єдина державна електронна бази з питань освіти. <https://info.edbo.gov.ua/> (дата звернення: 27.03.26).

99. Вступна кампанія 2023: на «Комп'ютерні науки» найбільше заяв в Україні — майже 50 тисяч. URL: <https://dou.ua/lenta/articles/entrance-campaign-2023/> (дата звернення: 27.03.26).

100. Вступна кампанія 2024: на 30 % менше ІТ-першокурсників, ніж торік. URL: <https://dou.ua/lenta/articles/admission-campaign-2024/> (дата звернення: 27.03.26).

101. Вступна кампанія 2025: прохідний бал на ІТ суттєво знизився, на контракті бум заяв. URL: <https://dou.ua/lenta/articles/admission-campaign-2025/> (дата звернення: 27.03.26).

102. Про внесення змін до переліку галузей знань і спеціальностей, за якими здійснюється підготовка здобувачів вищої та фахової передвищої освіти : Постанова Кабінету Міністрів України від 30.08.2024 № 1021. URL: <https://zakon.rada.gov.ua/go/1021-2024-п> (дата звернення: 18.04.2026).

103. Галузі знань та спеціальності вищої та фахової передвищої освіти. URL: <https://osvita.ua/vnz/76723/> (дата звернення: 27.03.26).

104. Закон України «Про вищу освіту». URL: <https://zakon.rada.gov.ua/laws/show/1556-18> (дата звернення: 27.03.26).

105. Класифікатор професій. 2026. URL: <https://www.buhoblik.org.ua/kadry-zarplata/trudoustrojstvo/3978-klasifikator-profesij.html> (дата звернення: 27.03.26).

106. Наказ про затвердження стандарту вищої освіти за спеціальністю 122 «Комп'ютерні науки» для першого (бакалаврського) рівня вищої освіти. URL: <https://mon.gov.ua/static-objects/mon/sites/1/vishcha-osvita/zatverdzeni%20standarty/2019/07/12/122-kompyut.nauk.bakalavr-1.pdf> (дата звернення: 27.03.26).

107. Освітній процес в КПІ ім.Ігоря Сікорського. F3 Комп'ютерні науки. Освітньо-професійні програми першого (бакалаврського) рівня вищої освіти URL: <https://osvita.kpi.ua/F3> (дата звернення: 27.03.26).

108. Факультет інформаційних технологій та математики. Київський столичний університет імені Бориса Грінченка. Освітні програми. URL: <https://fitm.kubg.edu.ua/struktural/kafedry/kafedra-kompiuternykh-nauk/osvitni-prohramy.html#osvitnia-prohrama-ta-navchalnyi-plan> (дата звернення: 27.03.26).

109. Каталог освітніх програм Національного університету «Запорізька політехніка». URL: <https://catalogop.zp.edu.ua/EduProgs.php> (дата звернення: 27.03.26).

110. Запорізький національний університет. Освітня програма «Комп'ютерні науки». URL: <https://www.znu.edu.ua/ukr/pk/4362/bakalavr/12399> (дата звернення: 27.03.26).

111. Львівський національний університет». Освітньо-професійна програма «Комп'ютерні науки». URL: <https://lpnu.ua/sap/osvitni-prohramy> (дата звернення: 27.03.26).

112. Oleksandr Burov, Evgeniy Lavrov, Svitlana Lytvynova, Olha Pinchuk, Svitlana Proskura, Oleksii Tkachenko, Natalia Kovalenko, Yana Chybiriak & Yana Dolgikh. Cognitive and Perceptual Reliable Performance: Comparison of Psychophysiological Limitations. *HCI International 2025 Posters. Communications*

in *Computer and Information Science*. Cham: Springer, 2025. Vol. 2523. P. 3-13. DOI: https://doi.org/10.1007/978-3-031-94153-5_1 (дата звернення: 27.03.26).

113. Ковалюк Т. В. Проєктно-орієнтований підхід до розвитку ІТ-освіти. *Управління розвитком складних систем*, 2013. Вип. 15. С. 140–143. URL: http://nbuv.gov.ua/j-pdf/Urss_2013_15_28.pdf (дата звернення: 27.03.26)'

114. Oudshoorn M. Curricula Guidelines and Computer Science Accreditation. *Proceedings of the 2024 International Symposium on Accreditation of Engineering and Computing Education (ICACIT)*, 2024. URL: [IEEE Xplore URL: https://ieeexplore.ieee.org/document/10788590?utm_source](https://ieeexplore.ieee.org/document/10788590?utm_source) (дата звернення: 17.03.2026).

115. Кривонос О. М. Формування інформаційно-комунікаційних компетентностей майбутніх учителів інформатики в процесі навчання програмування: автореф. дис. ... канд. пед. наук: 13.00.02. Київ, 2013. URL: <http://enpuir.npu.edu.ua/handle/123456789/4698> (дата звернення: 27.03.26).

116. Дані аналітики EPAM Campus: як рівень освіти ІТ-фахівців впливає на кар'єрне зростання. IT Ukraine Association; EPAM Ukraine, 2024. URL: <https://itukraine.org.ua/dani-analitiki-epam-campus-yak-riven-osviti-it-fahivtsiv-vplivaye-na-kar-yerne-zrostannya/> (дата звернення: 27.03.26).

117. Молчановський О. Порівняння української та американської вищих освіт: КПІ та Georgia Tech. Частина 1. URL: https://web.archive.org/web/20151114041455/http://oim.asu.kpi.ua:80/2014/03/24/kpi_vs_gatech_part_1 (дата звернення: 27.03.26).

118. МОН України. Стратегія розвитку вищої освіти в Україні на 2022–2032 роки. 2022. URL: <https://mon.gov.ua/osvita-2/vishcha-osvita-ta-osvita-doroslikh/strategiya-rozvitku-vishchoi-osviti-v-ukraini-na-2022-2032-roki> (дата звернення: 27.03.26).

119. Манелюк А. В. Формування змісту підготовки бакалаврів із комп'ютерних наук в аспекті компетентнісного підходу. *Педагогіка вищої та середньої школи*, 2015. Вип. 49. DOI: <https://doi.org/10.31812/pedag.v49i0.1165> . (дата звернення: 27.03.26).

120. Яременко В. В., Сліпушко О. М. *Новий тлумачний словник української мови*. Т. 1. Київ : Аконіт, 2008.

121. Освіта.ua. Компетентність як ключ до оновлення змісту освіти. URL: <https://osvita.ua/school/method/381/> (дата звернення: 27.03.26).

122. European e-Competence Framework (e-CF). URL: https://www.consortio-cini.it/index.php/en/labcf-home/labcf-areas-of-research/cfc-digital-competences/cfc-the-european-e-competence-framework?utm_source (дата звернення: 27.03.26).

123. Ковалюк Т. В. Узгодження вимог професійних та освітніх ІТ-стандартів. *Вісник НУ «Львівська політехніка»*, 2017. № 872. С. 229–240. URL: https://ena.lpnu.ua/bitstreams/11e5f47b-bd10-46c9-8b1b-a4984e0848bc/download?utm_source (дата звернення: 27.03.26).

124. Захаров Р. Г. Інформаційна технологія формування компетентностей здобувачів освіти : дис. ... д-ра філософії : 122 Комп'ютерні науки / Р. Г. Захаров. – Київ : Державний торговельно-економічний університет, 2024. – URL: <https://ur.knute.edu.ua/handle/123456789/10660> (дата звернення: 27.03.26).

125. Vitello S., Greatorex J. What is competence? A shared interpretation of competence to support teaching, learning and assessment. *Cambridge University Press & Assessment*, 2022. URL: https://www.cambridge.org/news-and-insights/insights/What-is-competence-A-shared-interpretation-of-competence-to-support-teaching-learning-and-assessment?utm_source (дата звернення: 03.04.2026).

126. Laura H. Salganik, Dominique S. Rychen, Urs Moser, John W. Konstant. Projects on Competencies in the OECD Context. *Analysis of Theoretical and Conceptual Foundations*, SFSO, OECD, ESSI, Neuchatel, 1999. 51 p. URL: https://www.orientamentoirreer.it/sites/default/files/materiali/1999%20progetti%20competenze%20OECD.pdf?utm_source (дата звернення: 27.03.26).

127. Овчарук О. В. Компетентнісний підхід в освіті: загальноєвропейські підходи.

URL: https://lib.iitta.gov.ua/id/eprint/744809/1/Овчарук_2009ooveea.pdf (дата звернення: 06.05.2026).

128. Кравченко С. О. Науково-педагогічний аналіз понять «компетенція» та «компетентність» *Імідж сучасного педагога*, 2017. № 8 (177). С. 41–47. URL: <https://isp.pano.pl.ua/article/view/121203/117166> (дата звернення: 27.03.26).

129. Пометун О. І. Формування громадянської компетентності: погляд з позиції сучасної педагогічної науки. *Вісник програм шкільних обміні*, 2005. – № 23. С. 18–24. https://ojs.uem.edu.ua/index.php/vpo/article/view/102?utm_source

130. Tuning Academy. Tuning Educational Structures in Europe, 2014. URL: http://tuningacademy.org/wp-content/uploads/2014/02/TuningEUI_Final-Report_EN.pdf (дата звернення: 27.03.26).

131. Zubyk L. V. Formation of Professional of Future Bachelor's Degree Students in Information Technologies: ... PhD dissertation in Pedagogical Scienc, 2016. URL: <https://nrat.ukrintei.ua/searchdoc/0416U005356/> (дата звернення: 27.03.26).

132. Морозова Т. Вища ІТ-освіта в Україні (системне дослідження): монографія Східноукр. нац. ун-т ім. В. Даля. - Луганськ : вид-во СНУ ім. В. Даля, 2010. - 287 с. : рис., табл. - Бібліогр.: С. 260-287. URL: <https://surl.li/lijvvm> (дата звернення: 27.03.26).

133. Спеціальність F3 "Комп'ютерні науки" (Бакалавр). https://nubip.edu.ua/spetsialnist-f3-kompyuterni-nauky-bakalavr?utm_source (дата звернення: 27.03.26).

134. Проскура С. Л. Модель формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*, 2019. № 3 (21). С. 104–112. URL: <https://fmo-journal.fizmatsspu.sumy.ua/publ/4-1-0-522> (дата звернення: 27.03.2026).

135. Kronda O. P., Proskura S. L. The level of digital technologies use in higher education institutions in the conditions of distance and blended learning. *Наукова молодь-2020* : матеріали VIII Всеукр. наук.-практ. конф.,

м.Київ,

2020.

С. 129-131.

URL: https://lib.iitta.gov.ua/id/eprint/722327/1/ЗБІРНИК%20НАУКОВА%20МОЛЮДЬ%202020_1.pdf

136. Морзе Н. В., Кочарян А. Б. Інформаційно-комунікаційна компетентність науково-педагогічних працівників університету. *Педагогічна освіта: теорія і практика*. Психологія. Педагогіка, 2015. № 24. С. 20–31. URL: <https://pedosvita.kubg.edu.ua/index.php/journal/article/view/67> (дата звернення: 03.04.2026)

137. Ktyvonos O. M. (). Formation of IKT IKT- Competencies of Future Computer Science Teachers Candidate of Pedagogical Sciences Dissertation 2014 URL: http://eprints.zu.edu.ua/11625/2/08_dis_Kryvonos.pdf (дата звернення: 27.03.26)

138. 137Овчарук О. В. Цифрова компетентність учителя: міжнародні тенденції та рамки. *Нова педагогічна думка*, 2019. № 4 (100). С. 52–55. DOI: <https://doi.org/10.37026/2520-6427-2019-100-4-52-55> (дата звернення: 03.04.2026).

139. Vuorikari R., Punie Y., Carretero Gomez S., Van den Brande G. DigComp 2.0: The Digital Competence Framework for Citizens. Update Phase 1: The Conceptual Reference Model. Luxembourg. *Publications Office of the European Union*. 2016. EUR 27948 EN. P. 44. URL: <https://surl.li/umeupv> (дата звернення: 03.04.2026).

У розділі були використані праці автора: [17], [18], [20], [21]. [22], [23], [24], [25], [32]. [37], 38], [39], [40], [41], [71], [134].

РОЗДІЛ 2. МОДЕЛЮВАННЯ НАВЧАННЯ ПРОГРАМУВАННЯ З ВИКОРИСТАННЯМ ВЕБОРІЄНТОВАНИХ ТЕХНОЛОГІЙ

2.1 Модель використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук

Сучасна система підготовки програмістів зазнає суттєвих трансформацій під впливом цифровізації освіти та інформатизації суспільства. Одним із провідних напрямів її модернізації є використання веборієнтованих технологій і впровадження веборієнтованого навчання – інноваційного підходу, що передбачає інтеграцію ІКТ та інтернет-ресурсів в освітній процес з метою підвищення його ефективності, інтерактивності й практичної спрямованості.

Підготовка бакалаврів комп'ютерних наук здійснюється переважно, в технічних закладах вищої освіти. В системі підготовки таких фахівців мають використовуватися як актуальні мови програмування, так і технології, що відповідають сучасному стану розвитку ІТ-галузі в Україні, європейських і східних державах [1].

Важливим чинником у підготовці майбутніх бакалаврів комп'ютерних наук є наявність продуманої, відпрацьованої та працездатної моделі формування компетентності з програмування. У межах такої моделі забезпечення доступу студентів-програмістів до інноваційних ресурсів має здійснюватися на засадах веборієнтованих технологій, як потужного інструменту повсюдного доступу до інформаційних і програмних систем [1].

Слід зауважити, що розроблення цієї моделі має здійснюватися з урахуванням затвердженого Стандарту вищої освіти за спеціальністю F3 «Комп'ютерні науки» галузі знань F «Інформаційні технології» для першого (бакалаврського) рівня вищої освіти, а саме [1]:

- визначення принципів і процедур забезпечення якості вищої освіти;
- здійснення моніторингу та періодичного перегляду освітніх програм;

- щорічне оцінювання здобувачів вищої освіти, науково-педагогічних і педагогічних працівників вищого закладу освіти та регулярне оприлюднення результатів таких оцінювань на офіційному вебсайті ЗВО;
- забезпечення підвищення кваліфікації педагогічних, наукових і науково-педагогічних працівників;
- забезпечення наявності необхідних ресурсів для організації освітнього процесу, зокрема самостійної роботи студентів за освітньою програмою «Комп'ютерні науки»;
- забезпечення наявності інформаційних систем для ефективного управління освітнім процесом;
- забезпечення публічної інформації про освітні програми, ступені вищої освіти та кваліфікації;
- розвиток практики академічної доброчесності та забезпечення ефективної системи запобігання та виявлення академічного плагіату у наукових працях працівників вищих закладах освіти і здобувачів вищої освіти [1].

У рамках цього дослідження розглянемо та уточнимо такі основні поняття, як «модель», «моделювання», «компетентність з програмування бакалавра комп'ютерних наук».

Термін «модель» має багато тлумачень, причому залежно від контексту в нього вкладається різний зміст. Слово «модель» походить від латинського «modulus», що означає «зразок», «норма», «міра», «мірило». У найширшому розумінні під словом «модель» розуміють деякий образ об'єкта (зокрема, умовний або уявний), який становить інтерес для дослідника, або, навпаки, як прообраз певного об'єкта чи системи об'єктів. У нашому дослідженні модель – це деякий об'єкт-замінник об'єкта-оригіналу (освітнього процесу), який забезпечує вивчення деяких істотних, з погляду дослідника, властивостей оригіналу, а саме процесу формування компетентності з програмування у бакалаврів комп'ютерних наук на засадах використання веборієнтованих технологій [1].

Заміщення одного об'єкта іншим із метою здобуття інформації про найважливіші властивості об'єкта-оригіналу за допомогою об'єкта-моделі називається моделюванням [2]. Моделювання (фр.- зразок, прообраз) – це відтворення характеристик певного об'єкта на іншому об'єкті, спеціально створеному для їх вивчення. Модель є ніби мостом між теорією та практикою [1], [3].

Так, у своїх працях Л. Зубик вважає, що одним із основних сучасних методів дослідження є моделювання систем, яке зазвичай передбачає створення концептуальної моделі об'єкта дослідження, її формалізацію та перетворення на математичну або комп'ютерну модель з подальшою перевіркою адекватності й дослідженням за допомогою аналітичних чи чисельних методів і сучасних комп'ютерних технологій. Учений також зазначає, що метод моделювання є предметом широкого використання в сучасних науково-педагогічних дослідженнях: В. Гриньова, О. Дубасенюк, О. Карпенко, С. Сисоєва, В. Чайка, Л. Ядвіршіс та інші [1], [4].

Моделювання процесу підготовки ІТ-фахівців в умовах університетської освіти має здійснюватися інтегровано, із поєднанням змісту технічного та управлінського напрямів підготовки. Гнучка підготовка фахівців, що відповідає сучасним змінам в освітній сфері та ринку праці, повинна реалізовуватись узгоджено й циклічно на всіх освітньо-професійних рівнях [1].

Таким чином, моделювання полягає в заміні реального об'єкта його моделлю з метою отримання інформації про властивості, структуру або особливості функціонування цього об'єкта шляхом проведення досліджень та експериментів над створеною моделлю. Якщо результати моделювання підтверджуються практикою і можуть бути використані для прогнозування процесів, що відбуваються в об'єкті-оригіналі, модель вважається адекватною цьому об'єкту. Адекватність моделі визначається метою моделювання та критеріями, обраними для оцінювання її відповідності реальному об'єкту. У межах цього дослідження метою моделювання є теоретичне обґрунтування та

дослідження використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук [1].

У своїх працях К. Стрюк визначає компетентність як інтегральну характеристику особистості майбутнього фахівця, що поєднує знання, уміння, навички у галузі інформатики та обчислювальної техніки, а також професійно важливі якості, необхідні для ефективного здійснення професійної діяльності. Учений наголошує, що для формування у майбутніх фахівців фахової компетентності, зокрема компетентності з програмування, необхідно враховувати типові професійні завдання та виробничі функції [5].

Залучення до моделі веборієнтованих технологій надає у наше розпорядження потужний інструментарій для забезпечення ефективності навчально-виховної роботи у технічному ЗВО [1].

У результаті проведення дисертаційних досліджень, одним із аспектів яких є аналіз рівня викладання дисциплін спеціальності F3 «Комп'ютерні науки» та аналіз рівня засвоєння теоретичних та практичних знань та навичок студентів-програмістів, було створено модель використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук (рис. 2.1).

Актуальність розроблення моделі зумовлена стрімким розвитком веборієнтованих технологій, цифрових платформ, хмарних сервісів і систем ІІІ, які стали невід'ємною складовою професійної діяльності сучасного програміста. У зв'язку з цим підготовка бакалаврів комп'ютерних наук потребує оновлення змісту, методів, засобів і форм навчання відповідно до вимог цифрової трансформації освіти та ІТ-індустрії.

Основною метою моделі є використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук. Запропонована модель відображає цілісну та логічно структуровану систему взаємопов'язаних компонентів, кожен із яких виконує визначену функцію у процесі організації навчання програмування майбутніх бакалаврів комп'ютерних наук із застосуванням веборієнтованих технологій.

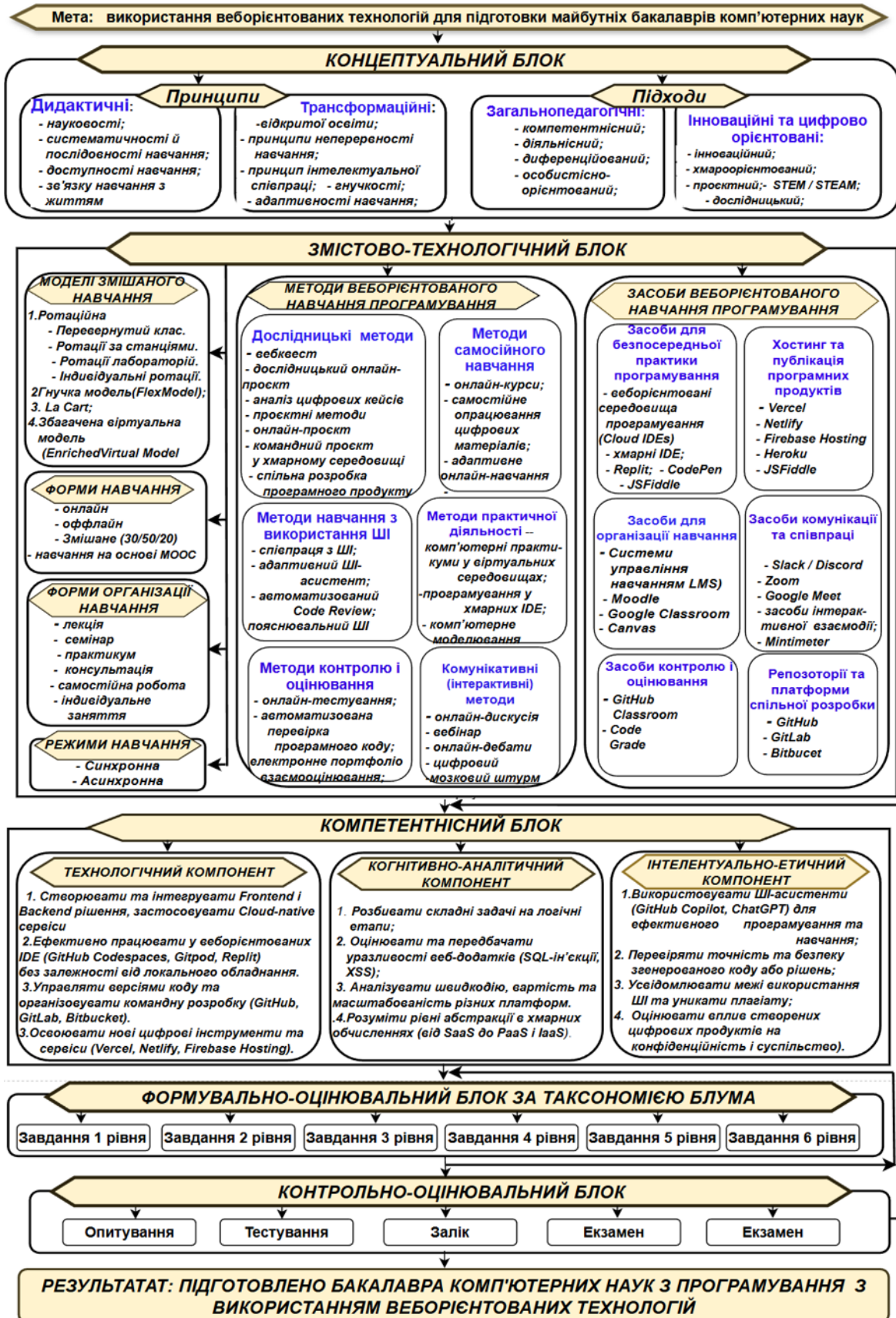


Рис. 2.1. Модель використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук

Структура моделі включає такі блоки:

1. концептуальний блок;
2. змістово-технологічний блок;
3. компетентнісний блок;
4. формувально-оцінювальний блок за таксономією Блума;
5. контрольнo-оцінювальний блок.

Концептуальний блок відображає сукупність принципів і підходів, що визначають теоретико-методологічні засади використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук. До його складу входять принципи (дидактичні та трансформаційні) і підходи (загальнопедагогічні, інноваційні та цифроорієнтовані).

До *дидактичних принципів* належать принципи:

– *науковості* передбачає забезпечення зв'язку між наукою і навчальним предметом для засвоєння майбутніми бакалаврами науково обґрунтованих положень, для формування науковими методами глибоких ідейних переконань, забезпечення єдності діяльності і свідомості. Практика сучасних ЗВО сформувала низку правил успішної реалізації вимог цього принципу: систематично інформувати студентів про нові досягнення в різних галузях науки, культури, виробництва суспільного життя, які пов'язані з розробкою ПЗ; розкривати студентам методи наукових досліджень, а також залежність результатів дослідження від методів; застосовувати найновішу наукову термінологію, що вводиться з розвитком мов програмування і задач, що вирішуються з їх допомогою; розкривати перед студентами головні ідеї наукових досягнень, приділяти увагу ключовим науковим проблемам, привчати їх стежити за науковими результатами (розробками); розвивати науковий світогляд [1];

– *систематичності й послідовності навчання* ґрунтується на закономірності психологічної науки про те, що під час дотримання логічних зв'язків навчальний матеріал запам'ятовується в більшому обсязі і більш

міцно. Навчальний матеріал має подаватися послідовно, системно, відповідно до вікових особливостей студентів: від простого - до складного, від нижчого рівня складності - до вищого рівня, від попереднього – до наступного. Практика сучасних ЗВО виробила низку правил успішної реалізації вимог принципу систематичності й послідовності навчання, а саме: застосовувати структурно-логічні схеми, опорні конспекти, інфографіку, схеми, зокрема інтелект-карти для ефективного засвоєння бакалаврами системи знань; розділяти зміст навчання на логічно завершені частини – блоки, модулі; використовувати яскраві факти з життя, літератури, кіно, телебачення, наводити приклади зарубіжного досвіду; поєднувати зміст, що вивчався раніше, з новим матеріалом на засадах побудови нової системи зв'язків; приділення особливої уваги поясненню студентам щодо важливості самостійної роботи у навчанні, тим самим створюючи умови для розв'язання більш складних завдань, систематичного аналізу помилок, як основного робочого інструментарію майбутнього бакалавра комп'ютерних наук [1];

– *доступності навчання* передбачає, з одного боку, готовність студентів до сприйняття й опанування нового навчального матеріалу, а з іншого створення організаційно-педагогічних умов, які забезпечують доступ до освітніх ресурсів незалежно від місця перебування та часу. Реалізація принципу доступності навчання ґрунтується на таких правилах: урахувати попередній досвід студентів, їхні інтереси та освітні потреби; застосувати інноваційні технології навчання; широко використовувати сучасне ПЗ для підвищення ефективності освітнього процесу; застосовувати аналогію, порівняння, зіставлення та протиставлення, як засоби кращого розуміння навчального матеріалу; стимулювати творчу діяльність студентів, зокрема шляхом виконання індивідуальних проєктів[1], [6], [7];

– *зв'язку навчання з життям* здійснюється через зміст навчання та організацію освітнього процесу, спрямовані на формування в студентів здатності застосовувати набуті знання, уміння й навички для розв'язання практичних завдань. Форми і методи навчання залежать від вікових та

індивідуальних можливостей студентів (мають враховуватися необхідні умови для студентів з особливими потребами). Основними засадами реалізації принципу зв'язку навчання з життям є: урахування життєвого та навчального досвіду студентів; використання конкретних прикладів із освітньої діяльності, результатів наукових досліджень, даних інформаційних ресурсів, аналітичних звітів ІТ-компаній; формування розуміння того що розвиток науки і техніки зумовлюється практичними потребами суспільства; організація навчальної діяльності студентів із використанням новітніх технологій, сучасних форм і методів праці; орієнтація освітнього процесу на актуальні тенденції розвитку професійного середовища та сучасних виробничих відносин

Трансформаційні принципи реалізуються через принципи:

- *відкритої освіти* передбачає відкритий доступ до широкого спектру навчальних і освітніх матеріалів та інструментів не тільки для публікації та зберігання навчальних матеріалів, а й розвинутого комплексу засобів колективної роботи з цими матеріалами за чітко визначеними критеріями в рамках освітніх систем, як в самих ЗВО, так і поза ними [8].

- *неперервності навчання* передбачає основні підходи до побудови адаптивного освітнього середовища через індивідуалізацію та персоналізацію навчання, а також безперервне навчання у різних середовищах, зокрема аудиторному, дистанційному [9]

- *інтелектуальної співпраці* полягає у використанні AI-технологій для підвищення ефективності навчання програмування, зокрема адаптації задач до рівня знань студентів, а також застосування ШІ для аналізу, класифікації задач з програмування в освітньому процесі, автоматизації процесу класифікації задач, що враховує їхню складність, тематику та тип. [10].

- *гнучкості навчання* передбачає новітні освітні технології, що пов'язані з персоналізацією навчання та адаптацією його змісту до індивідуальних потреб студентів та групової роботи через адаптивні технології та доповнену реальність [11].

– *адаптивності навчання* визначає практичні аспекти впровадження ІІІ як інструменту адаптивного навчання, здатного забезпечити персоналізовані рекомендації, пояснення навчального матеріалу, автоматизовану перевірку знань і підтримку в режимі 24/7 [12].

Таким чином, системна інтеграція класичних дидактичних положень із принципами *неперервності, інтелектуальної співпраці та гнучкості* створює необхідний теоретико-методологічний фундамент моделі, що забезпечує використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук.

Серед *підходів*, які можна застосувати в освітньому процесі, доцільно використовувати такі : *загальнопедагогічні* [8], *інноваційні та цифроорієнтовні підходи*.

Свою чергою, *загальнопедагогічні підходи* ґрунтуються на таких взаємодоповнювальних підходах:

– *Компетентнісний підхід* є свідченням необхідності переходу від традиційної «знаннєвої» парадигми до орієнтації на формування низки професійних і цифрової компетентності. Реалізація цього підходу передбачає не лише передачу знань, а й використання інноваційних освітніх технологій, а також створення цифрового освітнього середовища в ЗВО для підвищення якості підготовки та розвитку цифрової грамотності здобувачів [13].

– *Діяльнісний підхід* передбачає активну участь студентів в освітньому процесі, зокрема використання ігрових симуляторів для формування професійних компетентностей майбутніх інженерів-програмістів [14].

– *Диференційований підхід* формує умови для підвищення мотивації до навчання, врахування індивідуальних особливостей студентів [15].

– *Особистісно орієнтований підхід* спрямований на сучасне навчання, яке має відповідати здібностям, можливостям, мотивації та інтересам кожної особистості; передбачати діалогічність, діяльнісно-творчий характер; бути спрямованим на підтримку індивідуальних запитів здобувача освіти; передбачати альтернативи стандартним шляхам вирішення проблем [16].

Інноваційні та цифрово орієнтовані підходи класифікуються на наступних підходах:

– *Інноваційний підхід* розглядає використання онлайн-платформ, цифрових технологій та сучасних методик у навчанні програмування, визначає ключові переваги їх застосування: індивідуалізацію навчання, посилення мотивації, розвиток метакогнітивних навичок, аналізує інтерактивне навчання, розвиток алгоритмічного мислення та цифрової компетентності [17].

– *Хмаро орієнтований підхід* передбачає формування хмаро орієнтованого освітнього середовища, зокрема для STEM-освіти, використання ІКТ-аутсорсингу в проектуванні освітніх систем, віртуалізації, застосування ХО ІСР, сервісів спільної розробки та хмарних платформ у підготовці програмістів [18].

– *Проектний підхід* передбачає використання проєктного навчання, інтерактивних технологій та групових форм роботи під час підготовки студентів з програмування [19].

– *STEM / STEAM підхід* - це інтеграція програмування, інженерії, математичного моделювання та творчих елементів під час створення цифрових продуктів [20].

– *Дослідницький підхід* розглядає програмування як форму дослідницької діяльності, де студенти експериментують із кодом та аналізують результати [21].

Змістово-технологічний блок включає: моделі змішаного навчання, форми навчання, форми організації навчання, режими навчання, методи веборієнтованого навчання програмування, засоби веборієнтованого навчання програмування.

Моделі змішаного навчання передбачають поєднання традиційного, дистанційного та веборієнтованого навчання з метою підвищення гнучкості, доступності та ефективності підготовки бакалаврів комп'ютерних наук [22]. До найбільш поширених моделей змішаного навчання належать:

– *Ротаційна модель* передбачає чергування різних форматів навчальної діяльності, зокрема аудиторної роботи, онлайн-навчання та самостійної діяльності студентів. До різновидів ротаційної моделі належать «перевернутий клас» («Flipped Classroom»), ротація за станціями (Station Rotation), лабораторна ротація (Lab Rotation, індивідуальна ротація (Individual Rotation) [22].

– *Flex Model (гнучка модель)* надає студентам можливість гнучко поєднувати різні форми навчальної діяльності відповідно до їхніх індивідуальних потреб. При цьому викладачі забезпечують консультативну та навчально-методичну підтримку за потреби, тоді як студенти самостійно опановують значну частину навчального матеріалу. Така модель забезпечує високий рівень автономності та контролю студентів над власною освітньою траєкторією [23].

– *La Carte Model* дозволяє студентам обирати окремі онлайн-курси для доповнення основної освітньої програми навчання, що сприяє індивідуалізації навчання та розширенню освітніх можливостей [24].

– *Enriched Virtual Model* (збагачена віртуальна модель) передбачає поєднання дистанційного навчання з періодичними очними заняттями, під час яких студенти взаємодіють із викладачами та одногрупниками, отримують консультації та виконують практичні завдання. [25].

При підготовці майбутніх бакалаврів комп'ютерних наук значна увага приділяється таким формам навчання як:

– Онлайн навчання реалізується із використання цифрових технологій та мережі інтернет і передбачає проведення відеолекцій, вебінарів, онлайн консультації, виконання комп'ютерних практикумів у віртуальному освітньому середовищі, а також організацію самостійної роботи студентів із використанням електронних освітніх ресурсів [26].

– Офлайн навчання реалізується у формі аудиторних занять, комп'ютерних практикумів, семінарів і консультацій, що передбачають

безпосередню взаємодію викладача та студентів у навчальному середовищі ЗВО [26].

– Змішане навчання поєднує онлайн та офлайн формати навчання. Наприклад, співвідношення навчальної діяльності може становити 30% аудиторної роботи, 50% дистанційного (онлайн) навчання та 20% проєктної діяльності студентів. Такий підхід дає змогу поєднати переваги очного та дистанційного навчання, забезпечити гнучкість освітнього процесу, підвищити рівень самостійності бакалаврів комп'ютерних наук та ефективніше використовувати цифрові освітні ресурси. Змішане навчання вважається однією з найефективніших моделей сучасної освіти, оскільки сприяє розвитку цифрової компетентності, навичок самоосвіти та готовності студентів до професійної діяльності в ІТ-компаніях.

– *Навчання на основі MOOC* (Massive Open Online Courses) передбачає використання масових відкритих онлайн-курсів, розміщених на платформах Udemy [27], Coursera [28], Prometheus [29] та ін., як додаткового джерела навчальних матеріалів, знань та практичних навичок. Також можливе використання онлайн-платформ на основі MOOC для організації проєктної та самостійної діяльності студентів спеціальності «Комп'ютерні науки» [30]

В ході дисертаційного дослідження були розглянуті такі поняття як технологія навчання, веборієнтовані технології навчання, форми веборієнтованих технологій навчання (див. Розділ 1).

Форми організації навчання призначені для організації освітнього процесу в очному, дистанційному та змішаному форматах відповідно до умов цифрового освітнього середовища У процесі використання веборієнтованих технологій, на різних етапах навчання можна реалізувати такі *форми організації навчання*:

– *Лекція* – це систематичний, послідовний виклад навчального матеріалу, будь-якого питання, теми, розділу, предмета, методів науки. Лекції бувають навчальними (одна з основних форм навчального процесу й один з основних методів викладання у вузі) і публічними (одна з основних форм

пропаганди й поширення наукових знань). Головні вимоги до лекцій: науковість, доступність, єдність форми й змісту, емоційність викладу, органічний зв'язок з іншими видами навчальних занять - семінарами, практичними заняттями тощо [31].

– *Семінар* – це один з основних видів навчальних практичних занять студентів ЗВО. Полягає в самостійному вивченні студентами за завданнями педагога окремих питань і тем лекційного курсу з наступним оформленням матеріалу у вигляді реферату, доповіді тощо. Розрізняють три основні типи семінарів: семінари, що сприяють поглибленому вивченню певного систематичного курсу; семінари по вивченню окремих основних або найважливіших тем курсу і семінари (спецсемінари) дослідницького характеру з незалежною від лекцій тематикою [31].

– *Практикум* – це один з видів лабораторних робіт. Практикум проводиться після вивчення великих розділів курсу й має повторювальний і узагальнюючий характер. Під час практикуму зазвичай даються складніші й трудомісткіші роботи, ніж під час фронтальних комп'ютерних практикумів. [31].

– *Консультація* – це форма організації освітнього процесу, що передбачає надання викладачем роз'яснень, рекомендації і методичну допомогу студентам з окремих питань навчальної дисципліни. Консультації проводяться у формі співбесіди індивідуально або з групи студентів у позанавчальний час відповідно до встановленого графіка чи за потреби. Вони можуть організовуватися як після завершення вивчення окремого розділу навчальної програми так і в процесі його опанування а також під час підготовки студентів до підсумкового контролю та іспитів

– *Самостійна робота* - це різноманітний вид індивідуальної і колективної освітньої діяльності студентів, яка здійснюється ними на навчальних заняттях або дома за завданнями викладача, під його керівництвом, однак без його безпосередньої участі. Реалізація цих настанов вимагає від студентів активної розумової діяльності, самостійного виконання різних

пізнавальних завдань, застосування раніше засвоєних знань. Самостійні роботи можна поділити на підготовчі, спрямовані на засвоєння нових знань, тренувальні, узагальнюючеповторювальні й контрольні. Для перевірки знань, умінь та навичок студентів використовуються різноманітні контрольні роботи. Мають самостійний характер усі види творчих робіт. Завдання для самостійних робіт можуть бути фронтальними та індивідуальними [31].

– *Індивідуальні заняття* - є новою формою організації навчального процесу у ЗВО. Вони передбачають створення умов для якнайповнішої реалізації творчих можливостей студентів, які виявили здібності до навчання, науково- дослідної роботи та творчої діяльності [31]. *Індивідуальні заняття* проводяться, як правило, у позанавчальний час за окремим графіком, складеним кафедрою з урахуванням потреб і можливостей студента. Організація та проведення індивідуальних занять доручається найбільш кваліфікованим викладачам. Індивідуальні заняття на молодших курсах спрямовуються здебільшого на поглиблення вивчення студентами окремих навчальних дисциплін, на старших вони мають науково-дослідний характер і передбачають безпосередню участь студента у виконанні наукових досліджень та інших творчих завдань [32].

Підготовка бакалаврів комп'ютерних наук може відбуватись у двох *режимах навчання*: синхронному та асинхронному.

– *Синхронний режим* передбачає одночасну взаємодію викладача та студентів у реальному часі. Це можуть бути: онлайн-конференції; вебінари; відеолекції; чат-заняття. Перевагами синхронного навчання є: оперативний зворотний зв'язок; можливість дискусії; висока залученість студентів; контроль навчальної діяльності [33].

– *Асинхронний режим* не потребує одночасної присутності учасників освітнього процесу. Студенти самостійно працюють із матеріалами у зручний час. Основними засобами асинхронного навчання є: електронні курси; відеозаписи лекцій; форуми, тести; електронні бібліотеки. Перевагами асинхронного режиму є: гнучкість у виборі часу навчання; можливість

повторного перегляду матеріалів; індивідуальний темп навчання; доступність для великої кількості студентів [33].

Методи веборієнтованого навчання програмування містять наступні складники: дослідницькі методи, методи самостійного навчання, методи навчання з використанням ІІІ, методи практичної діяльності, методи контролю оцінювання, комунікативні інтерактивні методи.

Дослідницькі методи передбачають організацію навчальної діяльності студентів, спрямованої на самостійне дослідження алгоритмів, програмних рішень та інформаційних технологій. Використання таких методів сприяє формуванню здатності аналізувати складні програмні задачі, знаходити оптимальні рішення та розвивати алгоритмічне мислення. Застосування *дослідницьких методів* у навчанні програмування орієнтовано на розвиток аналітичного мислення, дослідницьких навичок та здатності до самостійного пошуку ефективних програмних рішень [1]. До них належать такі методи:

- *Вебквест* – це інноваційна технологія навчання, розглядається як розвиток дослідницьких навичок. Це метод організації навчально-дослідницької діяльності, у межах якого здобувачі освіти виконують проблемні завдання, використовуючи вебресурси. У процесі вебквесту студенти аналізують інформацію, добирають інструменти програмування, створюють власні рішення та презентують результати. Метод сприяє розвитку критичного мислення, навичок пошуку інформації та цифрової грамотності [34].

- Дослідницький онлайн-проект передбачає організацію дослідницьких онлайн-проектів у процесі навчання програмування, використання цифрових платформ, дослідження сучасних технологій та презентацію результатів студентських проектів [35].

- *Аналіз цифрових кейсів* – це використання цифрових інструментів, симуляторів та ігрових платформ у навчанні програмування. Аналізуються цифрові сценарії та кейси використання програм, зокрема Scratch, BlocklyGames, CodeMonkey для розвитку алгоритмічного мислення та мотивації студентів [36].

– *Проектні методи* інтерпретують реальні професійні ситуації, дозволяють шукати альтернативи, вибирати найбільш доцільні рішення професійних проблем [37]. Здобувачі освіти проходять усі етапи розроблення: планування, програмування, тестування та презентацію результату. Метод сприяє інтеграції теоретичних знань із практичною діяльністю.

– *Онлайн-проект* передбачає собою форму проектної діяльності, що реалізується повністю у вебсередовищі. Для організації взаємодії використовуються хмарні сервіси, системи керування проектами та онлайн-платформи. Метод забезпечує розвиток цифрової комунікації та навичок дистанційної співпраці [38].

– *Командний проект у хмарному середовищі* передбачає колективну розробку програмного продукту із застосуванням хмарних платформ, систем контролю версій та сервісів спільної роботи. Реалізація проекту з використанням GitHub, Replit, Gitpod, Notion та інших інструментів [39].

– *Спільна розробка програмного продукту* – це така форма організації навчання, за якої студенти спільно створюють ПЗ, використовуючи інструменти колективного програмування. Метод сприяє розвитку навичок спільного програмування(взаємного рецензування (peer-review), взаємоконтролю та професійної комунікації [39].

Методи самостійного навчання охоплюють: онлайн-курси, самостійне опрацювання, цифрових матеріалів, адаптивне онлайн навчання.

– *Онлайн-курси* дають інноваційні можливості впровадження онлайн-платформ, зокрема платформ: LeetCode, HackerRank, CheckiO та Project Euler для навчання програмування студентам-програмістам. Розглядаються теоретичні засади використання таких платформ, зокрема теорії конструктивізму, зони проксимального розвитку та когнітивного навантаження [40].

– *Самостійне опрацювання цифрових матеріалів* передбачає самостійне вивчення відеолекцій, інтерактивних ресурсів, навчальних

платформ, електронних підручників. Метод забезпечує поглиблення теоретичних та практичних знань і розвиток навичок самоосвіти [41].

– *Адаптивне онлайн навчання* – це метод навчання, за якого цифрова система автоматично підлаштовує зміст, складність і темп навчального матеріалу відповідно до індивідуальних потреб студента. Сприяє персоналізації освітнього процесу та підвищенню ефективності засвоєння знань [42].

Методи навчання з використанням ІІІ – це методи інтеграції штучного інтелекту в навчання програмування, який дозволяє підвищити ефективність освітнього процесу та забезпечити персоналізацію навчання. До них відносяться методи: співпраця з ІІІ, адаптивний ІІІ, автоматизований Code Review, пояснювальний ІІІ.

– *Співпраця з ІІІ* – це метод, який передбачає використання інтелектуальних систем для підтримки процесу програмування: генерації ідей, аналізу коду, пошуку помилок та отримання рекомендацій [43].

– *Адаптивний ІІІ-асистент* – це метод, який передбачає використання інтелектуального помічника та надає персоналізовані рекомендації, пояснення і навчальні завдання. Метод забезпечує індивідуалізацію навчання програмування [44].

– *Автоматизований Code Review* – це метод автоматичної перевірки коду за допомогою спеціалізованих сервісів або ІІІ-систем. Дозволяє виявляти програмного синтаксичні, логічні, та стилістичні помилки, оцінювати якість коду та формувати навички професійного програмування [45].

– *Пояснювальний ІІІ* – це метод використання систем штучного інтелекту, здатних не лише генерувати відповіді, а й пояснювати логіку виконання алгоритмів, принципи роботи коду та причини помилок. Сприяє глибшому розумінню програмування [46].

Методи практичної діяльності спрямовані на формування практичних навичок програмування розробки програмних проектів і моделювання професійної діяльності майбутніх фахівців шляхом безпосереднього виконання програмних завдань. До цієї групи належать такі методи:

– *Комп'ютерні практикуми у віртуальних середовищах* - це організація комп'ютерних практикумів у спеціалізованих онлайн середовищах або віртуальних лабораторіях. Студенти виконують завдання з програмування, тестують програмний код і моделюють програмні процеси без необхідності локального налаштування програмного забезпечення [47].

– *Програмування у хмарних ICP(IDE)* - це метод навчання, що базується на використанні веборієнтованих інтегрованих середовищ розроблення. Дає можливість писати, тестувати, запускати програмний код через браузер, забезпечуючи доступність і мобільність навчання [48].

– *Комп'ютерне моделювання* – метод, який передбачає створення цифрових моделей процесів, алгоритмів або систем із використанням мов програмування та спеціалізованих сервісів. Метод сприяє розвитку алгоритмічного мислення та практичних навичок програмування [49].

Методи контролю оцінювання призначені для перевірки рівня сформованості знань умінь і навичок студентів, а також для моніторингу результатів навчання, а саме:

– *Онлайн тестування* - це метод перевірки знань за допомогою цифрових платформ і тестових систем, які автоматично оцінюють студентські завдання з програмування та забезпечують автоматизований аналіз успішності студентів [50].

Автоматизована перевірка програмного коду – передбачає використання спеціалізованих систем для тестування правильності виконання програмних завдань метод дозволяє швидко оцінювати функціональність ефективність і коректність програмного коду [51].

– *Електронне портфоліо* являє собою цифрову добірку навчальних, практичних і проєктних робіт студента, що відображає рівень сформованості його професійних компетентностей. Такий метод використовується для моніторингу індивідуального освітнього прогресу, оцінювання результатів навчання та відстеження динаміки професійного розвитку майбутнього [52].

– *Взаємооцінювання* – це метод, за допомогою якого студенти аналізують та оцінюють роботи одне одного за визначеними критеріями. Сприяє розвитку рефлексії критичного мислення та навичок професійної комунікації. Успішно застосовується в різних академічних дисциплінах і вважається ефективним для розвитку вищих когнітивних навичок студентів. [53].

Комунікативні (інтерактивні) методи – це структуроване навчання, яке зосереджено на наставництві, керівництві та розвитку технічних, комунікативних та соціальних навичок як у студентів, які отримують допомогу, так і у студентів, які виконують викладацьку роботу. Призначені для розвитку цифрової комунікації, співпраці, командної взаємодії та активації навчально-пізнавальної діяльності студентів[54].

– *Онлайн дискусія* – це метод організації обговорення проблемних питань програмування у веб-середовищі студенти аргументують власну позицію аналізують рішення інших учасників і формують навички професійного спілкування[55].

– *Вебінар* – це метод призначений для синхронного онлайн навчання, під час якого викладач здійснює подання теоретичного матеріалу, демонструє практичні приклади та відповідає на запитання студентів у режимі реального часу через Інтернет у віддалених географічних точках [56], [57].

– *Онлайн-дебати* – це інтерактивний метод цифрового обговорення проблемних питань у вебсередовищі, під час якого учасники аргументують власну позицію, аналізують різні точки зору та розвивають критичне мислення і навички командної взаємодії Використання цього методу може проводитися синхронно (у режимі реального часу через відеоконференції) [58] або асинхронно (на форумах, у системах дистанційного навчання, соціальних мережах) [59].

– *Цифровий мозковий штурм* – це груповий метод генерування ідей із використанням цифрових платформ і спільних онлайн дошок та вебсервісів для колективної роботи. Цей метод застосовується для пошуку алгоритмічних

рішень, планування програмних проєктів та обговорення інноваційних підходів у програмуванні [59]. «Мозковий штурм» – це метод, при якому всі учасники можуть вільно (без критики й оцінювання з боку ведучого та інших учасників) висловлювати будь-які думки, навіть абсурдні та фантастичні, щодо поставленого питання або проблеми [60].

Засоби веборієнтованого навчання програмування охоплюють цифрові інструменти, сервіси та платформи, що забезпечують організацію, підтримки та реалізацію процесу навчання програмування у вебсередовищі. До його структури входять засоби для безпосередньої практики програмування засоби, хостингу та публікації програмних продуктів, засоби для організації навчання, засоби комунікації та співпраці, засоби контролю і оцінювання, а також репозиторії та платформи спільної розробки. До цієї групи відносяться наступні засоби:

- *Засоби для безпосередньої практики програмування* призначені для виконання практичних завдань, пов'язаних з написанням, редагуванням, запуском та налагодженням програмного коду в онлайн або локальному середовищі.

- *Веборієнтовані середовища програмування* - це онлайн середовища розробки, які дають змогу створювати редагувати та запускати програмний код через браузер. Такі засоби забезпечують доступ до навчання програмування незалежно від технічної характеристики локального пристрою [61].

- *Хмарні ІСР* – це ХО ІСР, що забезпечують можливість програмування без встановлення спеціального програмного забезпечення на комп'ютер. Вони підтримують автоматичне збереження коду та інтеграцію з репозиторіями [62].

- *Replit* – це онлайн-платформа для написання, тестування та запуску програм різними мовами програмування. Сервіс підтримує командну роботу, спільне редагування програмного коду та швидке розгортання навчальних проєктів [63].

– *CodePen* – це вебсервіс для створення та тестування веб-застосунків із використанням HTML CSS і Javascript. Дає можливість миттєво переглядати результати програмування та ділитися власними розробками [64]

– *JSFiddle* – це онлайн редактор для роботи з програмним вебкодом, що дозволяє тестувати та налагоджувати фрагменти програм у реальному часі. Засіб активно використовується для демонстрації прикладів коду та експериментування з веб-технологіями [65]

Хостинг та публікація програмних продуктів – використовується для розміщення зберігання та публічного доступу до програмних продуктів і застосунків забезпечують демонстрацію результатів роботи, спільну розробку та розгортання програмного забезпечення у мережі Інтернет.

– *Versel* – це хмарна платформа для розгортання сучасних вебзастосунків і фронтенд-проектів. Забезпечує швидке розміщення програмного коду та автоматичне оновлення після внесення змін до коду [66].

– *Netlify* – це сервіс для автоматизованого хостингу веб-проектів із підтримкою безпосереднього розгортання. Дає можливість швидко публікувати вебзастосунки та інтегрувати їх із репозиторіями коду [67].

– *Firebase Hosting* – це платформа для розміщення вебзастосунків із підтримкою безпечного хостингу та інтеграції з іншими сервісами Firebase. Використовується для швидкого розгортання статичних і динамічних вебпроектів[68].

– *Xeroku* – це хмарний сервіс для запуску підтримки та масштабування вебзастосунків дозволяє студентам працювати з реальними процесами розгортання програмного забезпечення [69].

– *JSFiddle* – це онлайн сервіс, який також може використовуватись для публікації та демонстрації окремих фрагментів програмного коду. Забезпечує швидкий доступ до результатів програмування через вебзастосунки [65].

Засоби організації навчання – призначені для управління освітнім процесом розміщення навчальних матеріалів організації курсів контролю

виконання завдань оцінювання та взаємодії між викладачем і здобувачами освіти.

Система управління навчанням (LMS) - це цифрові платформи, які призначені для організації адміністрування та підтримки освітнього процесу. Вони забезпечують доступ до навчальних матеріалів, завдань тестів і засобів комунікації [70].

Moodle – це навчальна платформа для управління навчальною діяльністю студентів та створення дистанційних курсів. Дозволяє організовувати тестування, обмін матеріалами та моніторинг успішності [71].

Google Classroom – це вебсервіс для організації змішаного та дистанційного навчання. Забезпечує швидку взаємодію між викладачем і студентами через систему завдань, повідомлень і цифрових матеріалів [72].

Canvas - це система управління навчанням, яка підтримує інтеграцію навчального контенту, оцінювання та комунікацію учасників освітнього процесу. Платформа орієнтована на персоналізацію навчання та зручну організацію курсів [73].

Засоби комунікації та співпраці – забезпечують онлайн-спілкування, проведення відеоконференцій, групову роботу та взаємодію учасників освітнього процесу. Сприяють командній діяльності, обговоренню проєктів і швидкому обміну інформацією. До них можна віднести:

- *Slack/Discord* – це платформи для командної комунікації, що забезпечують обмін повідомленнями, файлами та організацію тематичних каналів. Використовуються для координації командної роботи та підтримки навчальної взаємодії [74], [75].

- *Zoom* – це онлайн-платформа для відеоконференцій, що забезпечує аудіо- та відеозв'язок, текстовий чат, а також інструменти для проведення онлайн-зустрічей і навчальних занять [76].

- *Google Meet* – це вебплатформа для проведення синхронних онлайн-занять і відеозустрічей. Забезпечує доступність дистанційного навчання та інтеграцію з сервісами Google Workspace [77].

– *Засоби інтерактивної взаємодії* – це цифрові інструменти для організації опитувань обговорень і колективної роботи у вебсередовищі (Mentimeter, Kahoot!, Quizizz, Slido, Google Forms). Вони сприяють активізації навчальної діяльності студентів [78].

– *Mentimeter* – це онлайн-сервіс для створення інтерактивних презентацій, опитувань і голосувань. Дає можливість отримувати миттєвий зворотній зв'язок та залучати студентів до активної взаємодії [78].

Засоби контролю і оцінювання використовують для перевірки знань математичного тестування програмного коду оцінювання виконання завдань та моніторингу здобувачів освіти.

– *GitHub Classroom* – це веборієнтований сервіс, створений на базі GitHub для організації навчання програмування та управління практичними завданнями студентів. Основним призначенням сервісу є автоматизація процесу видачі, виконання та перевірки програмних робіт у межах навчального курсу [79]. За допомогою GitHub Classroom викладач може створювати навчальні завдання та автоматично генерувати окремі репозиторії для кожного студента або команди. Це дозволяє організувати індивідуальну чи групову роботу над програмним кодом у структурованому середовищі. Студенти отримують доступ до власного репозиторію, у якому може писати програмний код, вносити зміни, зберігати історію редагування та працювати над проектом спільно з іншими учасниками [79].

Однією з ключових можливостей GitHub Classroom є підтримка системи контролю версії Git. Завдяки цьому студенти навчаються використовувати професійні інструменти розробки програмне забезпечення, створювати коміти, працювати з гілками, об'єднувати зміни та вирішувати конфлікти прпограмного коду. Такий підхід сприяє формуванню навичок колективної розробки програмного забезпечення (collaboration development) та підготовці до реальних умов роботи в ІТ сфері.

Сервіс також дозволяє викладачу відстежувати активність студентів переглядати історію зміну у коді та аналізувати процес виконання завдань це

забезпечує можливість оцінювання не лише кінцевого а його самого процесу розробки програмного продукту.

– *CodeGrade* – це платформа для автоматизованого оцінювання програмного коду та начальних робіт, швидке надання зворотного зв'язку студентам [80].

Репозиторії та платформи спільної розробки призначені для зберігання програмного коду, керування версіями проектів та організації колективної роботи над програмним забезпеченням. Дозволяють відстежувати зміни, координувати роботу команди.

– *GitHub* – це вебплатформа для розміщення та управління репозиторіями програмного коду, що функціонує на основі системи контролю версій Git. Платформа забезпечує інструменти для спільної розробки програмних проектів, управління версіями коду, проведення Code Review та автоматизації процесів розгортання програм [81].

– *GitLab* – це платформа для управління життєвим циклом програмного забезпечення, яка поєднує систему контролю версій Git із засобами безперервної інтеграції, тестування та розгортання програмних застосунків. Використання GitLab у навчанні програмування сприяє формуванню у студентів навичок командної розробки програмного забезпечення [82].

– *Bitbucket GitHub* – це це онлайн сервіс для роботи з репозиторіями коду та організації командної взаємодії, де є можливість керувати версіями програмного забезпечення та підтримувати спілкування між учасниками освітнього процесу [83].

Компетентнісний блок відображає сукупність компетентностей, які формуються у майбутніх бакалаврів комп'ютерних наук у процесі веборієнтованого навчання програмування. *Компетентнісний блок моделі* представлено через три взаємопов'язані складники: технологічний, когнітивно-аналітичний та інтелектуально-етичний компоненти.

Технологічний компонент пов'язаний зі створенням, інтеграцією, розгортанням і підтримкою вебзастосунків у сучасному хмарному середовищі, а також із використанням сучасних інструментів розробки, автоматизації та командної роботи.

Створювати та інтегрувати Frontend і Backend рішення, застосовувати XO services GitHub – це передбачає здатність розробляти сучасні вебзастосунки, що складаються з клієнтської (Frontend) та серверної (Backend) частин, а також інтегрувати їх із сучасними хмарними сервісами. У Frontend рішення включено: створення та реалізація адаптивного інтерфейсу, роботу з HTML, CSS, JavaScript використання сучасних фреймворків і бібліотек (React View, Angular), забезпечення інтерактивності вебсторінок. Бекенд розробка передбачає: створення серверної логіки, реалізацію REST API, роботу з базами даних, налаштування авторизації та автентифікації. обробку клієнтських запитів, реалізацію механізмів безпеки.

– *Ефективно працювати у веборієнтованих ICP (IDE, GitHub Codespaces, Gitpod, Replit)* – передбачає здатність використовувати хмарні середовища, а саме: створювати та налаштувати хмарні робочі середовища, запускати сервери без локально встановлення ПЗ, використовувати інтегровані термінали, автоматично налаштувати залежності проєкту, синхронізувати середовища між різними пристроями, працювати з онлайн редакторами коду.

– *Управляти версіями коду та організовувати командну розробку (GitHub, GitLab, Bitbucket)* – це здатність застосовувати системи контролю версій і координувати спільну розробку програмного забезпечення, а саме: використовувати Git, створювати та клонувати репозиторії, фіксацію змін у коді, роботу з гілками, об'єднання змін, вирішення конфліктів версій. Організація команди розробки передбачає :розподіл ролей у команді, управління задачами, проведення перевірки програмного коду, документування змін, використання запитів на об'єднання змін, автоматизацію процесів безперервної інтеграції та безперервного розгортання. Практичні навички охоплюють: введення історії проєкту, контроль якості коду, роботу із системою

відстеження завдань і помилок, налаштування автоматичного тестування, організацію колективного доступу до проєкту.

– *Освоювати нові цифрові інструменти та сервіси (Vercel, Netlify, Firebase Hosting)* – відображає готовність до безперервного професійного розвитку та адаптації до швидких змін цифрових технологій, а саме: здатність швидко вивчати нові платформи, аналіз функціональних можливостей сервісів, адаптацію до оновлень технологій, самостійне планування документації, інтеграція нових інструментів у робочий процес. Практичні аспекти включають: налаштування вебзастосунків, налаштування хостингу, автоматичне розгортання через Git, роботу з доменами, інтеграцію серверless-функції.

Когнітивно-аналітичний компонент охоплює розвиток алгоритмічного мислення, здатності аналізувати програмні системи та вирішувати складні інженерні задачі.

– *Розбивати складні задачі на логічні етапи* – здатність здійснювати декомпозицію складних програмних завдань для їх ефективного розв’язання і побудови структурованого процесу розробки. Включає: аналіз вимог, визначення підзадач, побудову алгоритму розв’язання, планування етапів реалізації, визначення залежностей між компонентами. Практичні аспекти: створення архітектури застосунку, модульний підхід до програмування, поділ функціоналу на окремі сервіси, пріоритизація задач, оцінка складності виконання

– *Оцінювати та передбачати уразливості вебдодатків (SQL-ін’єкції, XSS)* – характеризує здатність аналізувати ризики інформаційної безпеки та забезпечувати захист вебзастосунків. Вона включає: розуміння типів кіберзагроз, аналіз потенційних уразливостей, виявлення небезпечних ділянок коду, застосування методів захисту, дотримання принципів подвійного програмування.

– *Аналізувати швидкодію, вартість та масштабованість різних платформ* – пов’язана зі здатністю оцінювати ефективність цифрових рішень із

технічною та економічною точок зору. Вона включає: аналіз продуктивності систем, оцінку навантаження, визначення витрат на інфраструктуру, дослідження можливості масштабування, порівняння хмарних платформ. Практичні аспекти: оптимізація запитів, кешування даних, використання CDN, балансування навантаження, моніторинг продуктивності. *Розуміти рівні абстракції в хмарних обчисленнях (від SaaS до PaaS і IaaS)* передбачає: розуміння принципів організації хмарної інфраструктури та моделей надання цифрових послуг, а саме: розуміння моделей хмарних сервісів, знання архітектури хмарних систем, усвідомлення меж відповідальності користувача та провайдера, вибір відповідної моделі для конкретних задач.

Інтелектуально-етичний компонент визначає безпечне використання веборієнтованих технологій й і систем штучного інтелекту

– *Використовувати ІІІ-асистенти (GitHub Copilot, ChatGPT) для ефективного програмування та навчання* – це застосування ІІІ для підтримки процесу розроблення програмного коду, підтримка самостійного навчання програмування, допомога у виконанні комп'ютерних практикумів й проєктів, автоматизація окремих завдань, пояснення алгоритмів і синтаксису мов програмування, отримання теоретичних та практичних пояснень, пошук і виправлення помилок у коді, отримання рекомендацій щодо отримання рефакторінгу коду, створення тестів та документації.

– *Перевіряти точність та безпеку згенерованого коду або рішень* – це здатність аналізувати програмний код, створеного автоматично, зокрема за допомогою ІІІ й генераторів коду, з метою виявлення помилок та відповідності поставленим вимогам; виконувати перевірку правильності роботи алгоритму; тестувати функціональність програми; аналізувати логічні та синтаксичні помилки; виявляти небезпечні конструкції в коді; перевіряти безпечну роботу застосунків.

– *Усвідомлювати межі використання ІІІ та уникати плагіату* – це здатність дотримуватися принципів академічної доброчесності під час використання ІІІ, розуміти можливості помилковості згенерованої інформації,

перевіряти правильність згенерованого програмного коду, самостійно доопрацьовувати результати роботи ШІ, посилаючись на використані джерела та інструменти, не допускати копіювання чужих матеріалів без зазначення авторства, дотримуватись етичних норм використання цифрових технологій і здійснювати критичне оцінювання відповідей ШІ-систем.

– *Оцінювати вплив створених цифрових продуктів на конфіденційність і суспільство* – це здатність враховувати етичні, соціальні та правові аспекти розроблення програмного забезпечення, як програмне забезпечення, цифрові сервіси або ШІ-системи можуть впливати на захист персональних даних, аналіз ризиків витоку інформації, дотримання принципів конфіденційності та кібербезпеки, запобігання дискримінації та маніпуляції у цифрових системах.

Формувально-оцінювальний блок за таксономією Блума – це частина освітнього процесу підготовки майбутніх бакалаврів комп'ютерних наук, що направлена на оцінювання теоретичного рівня засвоєння знань, практичних умінь і навичок студентів відповідно до шести рівнів когнітивної сфери, які виділив Б. Блум.

Як правило цей блок містить завдання на кожен рівень, який оцінюється кількістю балів, які наведені у таблиці 2.1.

Таблиця 2.1

**Оцінювання рівня знань бакалаврів комп'ютерних наук за
модифікованою таксономією Блума**

Рівень	Рівні модифікованої таксономії Блума	Бали
6	Рівень створення Creating	95-100
5	Рівень оцінювання Evaluating	85-94
4	Рівень аналізування (Analysing)	75-84
3	Рівень застосування (Applying)	70-74
2	Рівень усвідомлення (Understanding)	65-69
1	Рівень пригадування (Remembering)	60-64

Майбутній бакалавр комп'ютерних наук має можливість вибирати та виконувати завдання відповідно до рівня власної підготовки, що сприяє його саморозвитку та формуванню індивідуальної траєкторії навчання. На цьому етапі здобувач освіти може підвищити свій професійний рівень, повертатися до виконання завдань підвищеної складності або повторно опрацьовувати окремі аспекти розв'язання завдань нижчих рівнів для закріплення необхідних умінь та навичок.

У процесі реалізації авторської моделі у бакалаврів комп'ютерних наук формуються фахові компетентності, зокрема компетентність з програмування, що у подальшому дасть можливість майбутньому випускнику ЗВО бути конкурентоспроможним на IT-ринку праці.

В процесі формування фахової компетентності з програмування майбутніх бакалаврів комп'ютерних наук дуже важливо оцінювати рівень знань не тільки в кінці курсу (іспит, заліки лабораторних робіт), але під час вивчення навчальної дисципліни. Оцінювання рівня знань студентами-програмістами можливо за допомогою таксономії Блума, яка містить 6 рівнів складності. Кожне практичне завдання буде відповідати своєму рівню з визначеною кількістю балів (табл. 2.1). Критерії оцінювання практичного завдання прописуються для кожної теми за таксономією Блума. Таким чином, використовуючи рівні таксономії Блума, ми можемо відслідковувати рівень знань майбутніх бакалаврів комп'ютерних наук протягом всього навчального курсу.

2.2 Модель веборієнтованого навчання майбутніх бакалаврів комп'ютерних наук

Входження України в Болонський процес стимулює формування нових принципів та підходів до розвитку вищої освіти. Це своєю чергою потребує удосконалення форм, методів та засобів, спрямованих на створення цілісної системи безперервної освіти [84].

В умовах цифрової трансформації проектування моделі веборієнтованого навчання майбутніх бакалаврів комп'ютерних наук базується на переході до парадигми Веб 4.0, де веборієнтовані технології стають інтелектуальним середовищем співпраці людини та ШІ. Це вимагає впровадження принципу неперервного навчання, що забезпечує неперервність освітнього досвіду студента незалежно від часу та місця перебування [84].

Обґрунтування поняття «веборієнтоване навчання» представлено в **Додатку В**.

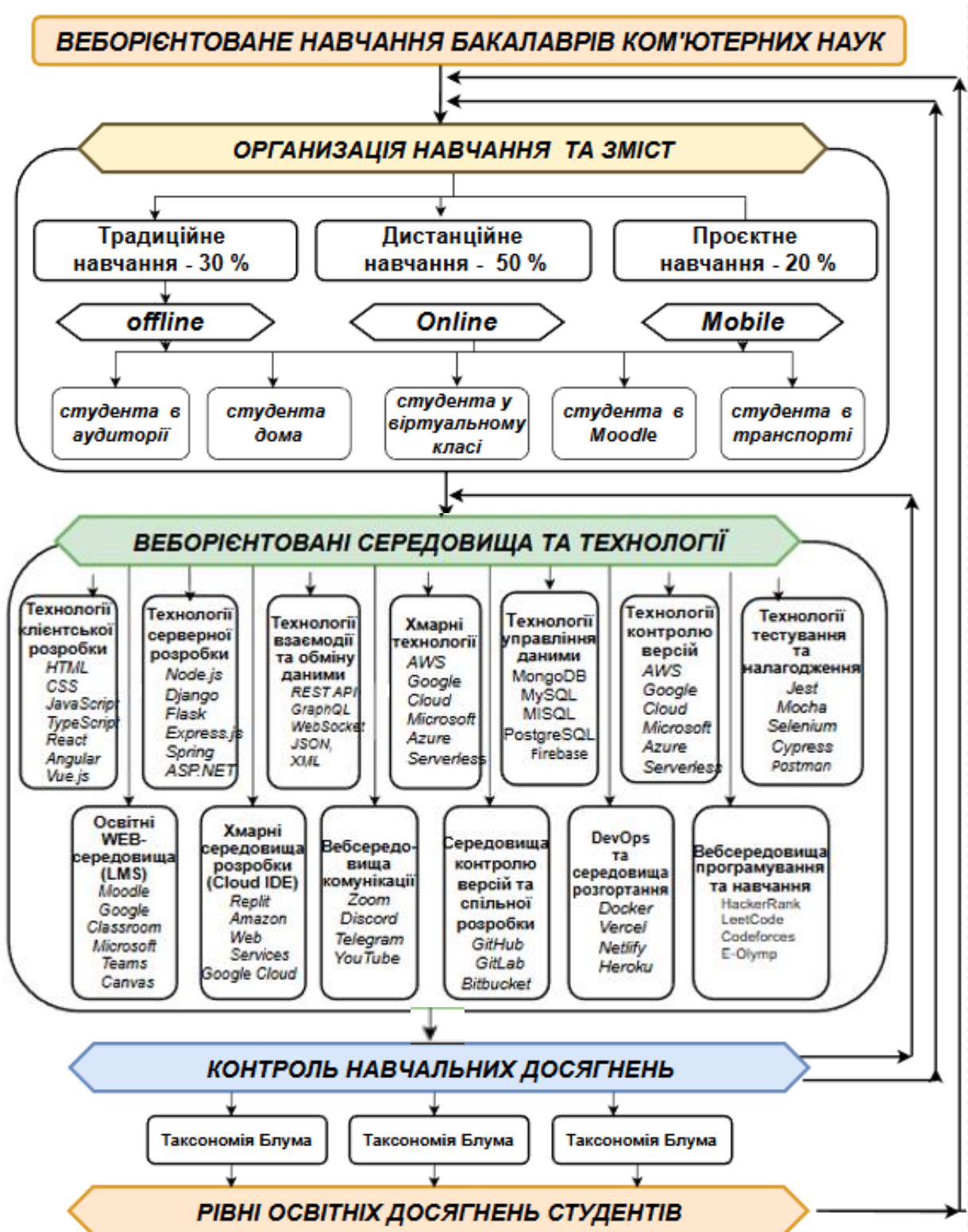
Вивчаючи підходи до організації веборієнтованого навчання у дослідженнях зарубіжних та відчизняних науковців, а також аналізуючи власний багаторічний досвід викладання мов програмування і перевіряючи результати навчання студентів на практиці, ми дійшли висновку, що у веборієнтованому навчанні доцільно поєднувати 30 % технологій традиційного навчання, 50 % технологій інноваційного навчання та 20 % технологій проєктного навчання. До проєктного навчання входить спільна робота викладача зі студентами, командна і проєктна (розробка міні-проєктів) діяльності студентів [84].

В науковій та педагогічній літературі обґрунтовано моделі інтегрованого навчання, перевернутого класу, адаптивного навчання та ін., проте дотепер модель веборієнтованого навчання не отримала цілісного теоретичного обґрунтування в наукових публікаціях вчених.

Тому виникла необхідність розробити та обґрунтувати модель веборієнтованого навчання бакалаврів комп'ютерних наук. На рисунку 2.2 представлено авторську структурно-функціональну модель, що поєднує різні форми організації навчання, сучасні вебтехнології, цифрові середовища та засоби контролю результатів навчання. Вона демонструє взаємозв'язок між організацією освітнього процесу, технологічним забезпеченням та формуванням освітніх досягнень студентів.

Метою моделі є забезпечення ефективної підготовки бакалаврів комп'ютерних наук шляхом інтеграції традиційного, дистанційного та

проектного навчання із використанням сучасних веборієнтованих технологій, хмарних сервісів та цифрових платформ. Модель орієнтована на формування компетентностей з програмування майбутніх бакалаврів комп'ютерних наук у галузі інформаційних технологій.



. Рис. 2.2. Модель веборієнтованого навчання програмування бакалаврів комп'ютерних наук

Модель складається з чотирьох взаємопов'язаних блоків: організація навчання та зміст; веборієнтовані середовища та технології; контроль навчальних досягнень; рівні освітніх досягнень студентів

Блок «Організація навчання та зміст» визначає форми організації освітнього процесу та пропорції використання різних форматів навчання.

Традиційне навчання (30%) передбачає безпосередню взаємодію викладача і студентів в аудиторії. Така форма використовується під час проведення: лекцій; лабораторних робіт; практичних занять; консультацій; контролю знань. Навчання здійснюється в режимі *offline*, де студенти працюють безпосередньо в навчальних аудиторіях ЗВО.

Основними перевагами є: безпосередня комунікація; оперативний зворотний зв'язок; можливість колективного обговорення; контроль навчальної діяльності студентів.

Дистанційне навчання (50%) є основною складовою моделі та реалізується через веборієнтовані освітні середовища. Навчання відбувається в режимі *online*, що дозволяє студентам працювати: вдома, у віртуальному класі, у будь-якому місці з доступом до мережі Інтернет. Дистанційне навчання забезпечує: доступ до навчальних матеріалів 24/7; виконання комп'ютерних практикумів; проведення тестування; організацію відеоконференцій; асинхронну та синхронну взаємодію.

Проектне навчання (20%) спрямоване на розвиток практичних навичок програмування та командної роботи. Навчання реалізується переважно через мобільні (Mobile), та вебтехнології, що дозволяє студентам працювати: у Moodle; під час пересування; з мобільних пристроїв; у хмарних середовищах. Проектне навчання включає: розробку вебзастосунків; створення програмних продуктів; командні проекти; використання методологій Agile та Scrum; розв'язання практичних професійних завдань.

— *Блок «Веборієнтовані середовища та технології»* являється центральним елементом моделі, який забезпечує технологічну основу підготовки майбутніх фахівців.

– *Технології клієнтської розробки* призначені для створення інтерфейсів користувача вебзастосунків. Включають: HTML; CSS; JavaScript; TypeScript; React; Angular; Vue.js. Студенти набувають компетентностей зі створення адаптивних та інтерактивних вебінтерфейсів.

– *Технології серверної розробки* забезпечують реалізацію бізнес-логіки вебзастосунків. Використовуються: Node.js; Django; Flask; Express.js; Spring; ASP.NET. Під час навчання студенти опановують принципи побудови серверних застосунків, API та взаємодії з базами даних.

– *Технології взаємодії з базами даних* Забезпечують зберігання та обробку інформації. Вивчаються: PostgreSQL; MySQL; MongoDB; Firebase; Redis; JSON; XML. Формуються навички проєктування структур даних та організації інформаційних систем.

– *Хмарні технології* дозволяють розгортати вебзастосунки у сучасній хмарній інфраструктурі. Використовуються: AWS; Google Cloud; Microsoft Azure; Cloud Servers. Студенти вивчають принципи хмарних обчислень, контейнеризації та масштабування застосунків.

– *Технології управління даними* забезпечують адміністрування та обробку інформаційних ресурсів. Приклади: MongoDB; MySQL; PostgreSQL; Firebase. Особлива увага приділяється питанням безпеки, резервування та оптимізації даних.

– *Технології контролю версій* Забезпечують організацію командної розробки програмного забезпечення. Використовуються :Git; GitHub; GitLab; Bitbucket. Студенти навчаються: працювати з репозиторіями; організовувати спільну розробку; виконувати злиття гілок; вести історію змін проєкту. *Технології тестування* забезпечують контроль якості програмного забезпечення. Використовуються: Jest; Mocha; Selenium; Cypress; Postman. Формуються навички: модульного тестування; інтеграційного тестування; автоматизованого тестування; тестування API.

– *Освітні вебсередовища (LMS)* забезпечують організацію освітнього процесу. Використовуються: Moodle; Google Classroom; Microsoft Teams;

Canvas. Функції: розміщення навчальних матеріалів; оцінювання; тестування; комунікація.

- *Хмарні середовища розробки* надають можливість програмувати без локального встановлення програмного забезпечення. Приклади: XO ICP (Cloud IDE); Replit; Amazon Web Services; Google Colab. Такі середовища забезпечують доступ до навчання з будь-якого пристрою.

- *Вебсервіси для комунікації* забезпечують взаємодію між учасниками освітнього процесу. Використовуються: Zoom; Discord; Telegram; YouTube. З їх допомогою проводяться лекції, консультації та обговорення проєктів.

- *Середовища контролю версій і спільної розробки* дозволяють організовувати колективну діяльність студентів. Основні платформи: GitHub; GitLab; Bitbucket. Вони підтримують: командну розробку; контроль версій; перевірку коду; управління проєктами.

- *DevOps та середовища розгортання* призначені для автоматизації розгортання програмних продуктів. Використовуються: Docker; Vercel; Netlify; Heroku. Студенти знайомляться з принципами безперервної інтеграції та безперервного розгортання (CI/CD).

- *Вебсередовища програмування та навчання* сприяють розвитку професійних компетентностей. Приклади: HackerRank; LeetCode; Codeforces; E-olymp. Ці ресурси використовуються для розвитку алгоритмічного мислення та підготовки до професійної діяльності.

Блок «Контроль навчальних досягнень» представляє контроль результатів навчання, яке здійснюється на основі таксономії Блума. Оцінювання спрямоване на перевірку рівнів: запам'ятовування; розуміння; застосування; аналізу; оцінювання; створення. Контроль реалізується через: тести; комп'ютерні практикуми; проєкти; презентації; захисти програмних продуктів; портфоліо студентів.

Контроль навчальних досягнень у межах моделі спрямований на відстеження динаміки професійного зростання студента за шкалою DigComp 3.0. Використання веборієнтованих систем автоматизації

перевірки коду, таких як E-olymp, у поєднанні з аналізом «цифрового сліду» студента в GitHub має змогу об'єктивно оцінювати перехід від базового володіння мовними конструкціями (рівні 1-2) до експертного проєктування архітектури складних хмарних систем.

Блок «Рівні освітніх досягнень студентів» є кінцевим результатом функціонування моделі є формування різних рівнів освітніх досягнень студентів. Вони характеризуються: рівнем засвоєння теоретичних знань; сформованістю компетентності з програмування; здатністю розробляти вебзастосунки; умінням працювати в команді; навичками використання сучасних вебтехнологій; готовністю до професійної діяльності в ІТ-галузі.

Таким чином, запропонована модель забезпечує комплексне поєднання традиційного, дистанційного та проєктного навчання з використанням сучасних веборієнтованих технологій, хмарних сервісів, цифрових платформ та інструментів розробки програмного забезпечення, що сприяє формуванню високого рівня професійної підготовки бакалаврів комп'ютерних наук та їх готовності до роботи в умовах сучасної цифрової економіки. Якщо студента-програміста не влаштовує рівень навчальних досягнень, він має змогу виконати додаткові завдання і підвищити свій рівень.

Використання цієї моделі на практиці суттєво впливає на ефективність навчання студентів програмістів [84].

Для підвищення професійних компетентностей з програмування бакалаврів комп'ютерних наук в освітньому процесі використовують хмарні сервіси та хмарні платформи (наприклад, Google Engineering), які є складовими хмароорієнтованого навчального середовища. Як зазначає С. Литвинова, під хмароорієнтованим навчальним середовищем (ХОНС) розуміємо спеціально створене середовище, що охоплює будь-які аспекти використання хмарних обчислень в організації навчання студентів усіх категорій за різними формами і моделями навчання [84], [100].

Формування ІКК студентів пропонується на засадах створення синтетичного веборієнтованого освітнього середовища для надання

можливостей студентам задіяти інноваційні засоби навчання, зокрема з програмування [84], [104].

Такі додатки як MS Office 365, Google Doc, Google-сайт, Google Classroom полегшують спільну роботу викладача зі студентами. Командна робота студентів реалізуються через використання веборієнтованих сервісів: компіляторів, інтелект-карт, автоматизованих систем перевірки знань з програмування. Контроль знань студентів здійснюється за допомогою тестів (форми Office 365 або Google Apps) [84].

Центральним елементом веборієнтованого середовища в моделі є складник інтелектуальної підтримки. ШІ інтегрується в модель не як засіб пошуку відповідей, а як інтелектуальний асистент у режимі AI Pair Programming. Він виконує функції пояснення логіки алгоритмів, допомога у виправленні помилок 24/7 та інструменту автоматизованого Code Review, що дозволяє студенту отримувати миттєвий зворотний зв'язок ще до перевірки роботи викладачем.

Важливу роль в організації діяльності студентів з веборієнтованими технологіями відіграють так звані віртуальні класи (Zoom, Teams, Google Meet, Etutorium, Cisco та ін.). Ця модель максимально наближена до навчання у реальному класі online. Віртуальний клас ми розуміємо як особливе навчальне середовище, у якому навчання здійснюється в реальному часі, інтегруючи Інтернет й інформаційно-комунікаційні технології й об'єднує спільними освітніми цілями і задачами студента і викладача [105], [106], Студенти, які відсутні за будь-яких причин, можуть засобами віртуального класу в реальному часі слухати лекцію викладача і приймати активну участь в обговоренні навчального матеріалу [84].

Використання комплексного підходу до організації веборієнтованого навчання бакалаврів дозволить студенту застосовувати так звану освітню мобільність, яка дозволить йому приймати активну участь у навчальному процесі свого закладу, будучи при цьому поза межами цього закладу. Наприклад, студент знаходиться на стажуванні в іншій країні, або будь-де в

іншому місці, але при цьому має можливість бути присутнім на лекціях у віртуальному класі, користуватись сервісами хмаро орієнтованого середовища, і використовувати веборієнтовані технології для навчання; користуватися освітнім контентом, виконувати домашні завдання і самостійну роботу, спілкуватися в викладачем тощо. Уточнимо, що академічна мобільність базується на міжнародній програмі, за якою студент одного ЗВО за контрактом навчається у ЗВО іншої країни [84].

Запит суспільства на розроблення об'єктів віртуальної та доповненої реальності, потребує від студентів-програмістів уміння використовувати новітні технології та інструменти, зокрема платформу Vuforia, фреймворк BeyondAR, проєкт Beyond Reality Face Nxt, технології 3D-трекінга для розпізнавання образів, відстежування та геолокації AR для мобільних пристроїв, планшетів та смарт-окулярів та ін.

З кожним роком збільшується сфера діяльності IT-індустрії, з'являються нові мови програмування, нові методології та технології в галузі комп'ютерних наук, удосконалюються управління проєктами. це своєю чергою збільшує обсяг навчальних відомостей, який студент-програміст повинен опанувати. На сьогоднішній день існує велика кількість методик для інформації. Найбільш вагомим є використання веборієнтованих інтелект-карт. «Саме застосування методу інтелект-карт стає новим інструментарієм, який забезпечує структурування, систематизацію, конкретизацію та ефективним засобом впливає для запам'ятовування відомостей студентом для подальшого використання» [107]. Цей метод не тільки візуалізує і систематизує великі обсяги навчальних відомостей, але стає мотиватором для подальшого навчання [108],[84].

Аналізуючи досвід застосування веборієнтованого навчання в різних країнах, ми робимо висновок, що як в закордонних університетах так і в українських ЗВО активно застосовуються різні підходи до навчання бакалаврів, зокрема майбутніх бакалаврів ком'ютерних наук. Різниця полягає в

національних традиціях, підходах до організації освітнього процесу та змісту вищої освіти.

Підходи до організації веборієнтованого навчання, зокрема змішаного навчання бакалаврів комп'ютерних наук, таких як підходи за сценаріями «перевернутого навчання», La Carte, індивідуальної ротації, є поширеними серед ЗВО Європи, США, Азії. В Україні набули широкого використання системи Moodle та «перевернуте навчання».

Аналізуючи підходи до організації веборієнтованого навчання у дослідженнях закордонних та вітчизняних науковців, враховуючи результати опитування студентів-програмістів, дійшли висновку, що доцільно використовувати модель веборієнтованого навчання бакалаврів комп'ютерних наук, за якої пропонується здійснення освітнього процесу у відношеннях 20 % проєктного навчання, 30 % традиційного і 50 % дистанційного навчання.

У запропонованій моделі 50 % складника, що припадає на дистанційне навчання на засадах веборієнтованих технологій, реалізується через Cloud-native парадигму. На відміну від традиційного використання локальних ICP, ми пропонуємо перенесення всього циклу розробки у хмару (GitHub, Codespaces, Replit), що усуває технічні бар'єри налаштування середовища та забезпечує ідентичність умов навчання для всіх студентів.

Проєктне навчання (20%) при цьому орієнтоване на створення життєздатного продукту з обов'язковою публікацією коду в репозиторіях для формування цифрового портфоліо.

До недоліків веборієнтованого навчання можна віднести те, що викладачам потрібно присвячувати чимало часу на підготовку цікавих та зрозумілих навчальних матеріалів (відеолекції, методичних вказівок до виконання комп'ютерних практикумів, і т.д.) Також дуже важливо, щоб стратегія веборієнтованого навчання підтримувалася адміністрацією закладів вищої освіти

.3 Система оцінювання навчальних досягнень майбутніх бакалаврів комп'ютерних наук з програмування

Оцінювання навчальних досягнень студентів ЗВО є однією з найбільш актуальних задач, що постають перед викладачами як організаторами освітнього процесу. Проблема формування коректного та об'єктивного рейтингового балу в оцінюванні результатів вивчення дисципліни є ключовим моментом завершального етапу опанування тематичних та модульних частин навчального матеріалу. Рейтинговий бал формується на основі оцінювання окремих навчальних дій студента. Сприйняття об'єктивності оцінювання студентами відрізняється залежно від типу засобів контролю [109].

На сьогодні в Україні поки не існує єдиного інструментарію, що дозволяє вимірювати і оцінювати результати навчання та рівень сформованості набутих компетенцій, хоча на міжнародному рівні такі інструменти застосовуються (наприклад, TIMSS, PISA, CIVIC Education Project), якими українські студенти за бажанням можуть скористатися [110]. Водночас удосконалення форм і методів, спрямованих на створення цілісної системи безперервної освіти, дає змогу сформувати єдину систему оцінювання динаміки підготовки бакалаврів комп'ютерних наук у навчальній діяльності [109].

Однак особливої уваги потребують питання процесуального, технологічного, мотиваційного забезпечення оцінювання навчальних досягнень студентів закладів вищої освіти (ЗВО) як цілісної педагогічної системи, що враховує індивідуальні інтереси, здібності і схильності. [109].

У цьому підрозділі дисертаційного дослідження проаналізуємо праці українських та зарубіжних авторів, присвячені вивченню питання контролю та оцінювання результатів навчальної діяльності студентів ЗВО; висвітлимо принципи формування оцінювання та рівні пізнавальних, емоційних та рухових цілей, що мають комплексно охоплювати навчально-інформаційний простір; розглянемо особливості формування оцінювальних завдань в умовах дистанційного навчання [109].

Фахові компетентності майбутніх бакалаврів комп'ютерних наук, зокрема компетентність з програмування, визначається системою оцінювання рівня результатів навчальної діяльності студентів і має представляти собою цілісну структуру, що описує, вимірює, записує та передає результати досягнень студентів, підпорядковуючись чітко визначеним критеріям [109].

Підходи до оцінювання навчальних досягнень студентів закордонних та вітчизняних ЗВО, теоретичні та практичні питання організації системи оцінювання навчальних досягнень студентів, висвітлені в роботах В. Кухаренко [95], С. Литвинова [109], Л. Огнівчук [110], О. Калініної [111], А. Турчина [113], Ж. Пайва (J. Paiva) [114], Л.Ваннер (L. Wanner) [115], С. Бергін (S. Bergin) та А. Муні (A. Mooney) [116], С. Аламр (S. Alamr) [117], М. Уррутія (M. Urrutia) та Л. Карр (L. Carr) [117], Л. Бартман (L. Baartman) та К. Квінлан (M. Quinlan) [118], К. Нгуєн (Q. Nguyen), Б. Рієнтіс (B. Rienties), Л. Тойтенель (L.Toetenel) та Р. Фергюсон (R. Ferguson) [119], Д. Вайтлок (D. Whitelock) [119], Л. Баруа (L. Barua) [120], Б. Локкі (B. Lockee) [120],

Так, науковець О. Калініна дослідила, що міжнародна практика оцінювання рівня знань студентів ґрунтується на трьох основних підходах, які сформувалися незалежно один від одного, спочатку в США (поведінковий підхід), потім у Великобританії (функціональний підхід) і в останню чергу у Франції й Німеччині (багатомірний підхід) [109], [111].

Поведінковий підхід має за основу постановку критеріїв очікуваної поведінки студента у визначених ситуаціях, яка, при цьому, спрогнозована сукупністю таких процесів у навчанні, як постановка питання або проблеми, пошук відповіді або рішення студентом, та реакції викладача, яка спрямована на побудову рефлексивного поведінкового ефекту. Саме поведінковий підхід, основною методикою якого є орієнтація результатів навчання на відповідність рівню практичного застосування знань, став основою для формування поняття «компетенція», введеного американським ученим Р. Уайтом для опису тих

особистісних характеристик, які «тісно пов'язані з ідеальним виконанням роботи й високою мотивацією» [109], [111].

Функціональний підхід ми спостерігаємо у дослідженнях університетів Великої Британії, де вчені підходять до визначення оцінювання рівня знань студентів, виділяючи наступні критерії: рівність горизонтальна (між предметами в різних закладах), рівність вертикальна (в межах одного закладу), простота підходу, прозорість, послідовність, правомірність, цільова обґрунтованість, адміністративна ефективність, прийнятність для всіх зацікавлених сторін [112].

Багатомірний підхід уособлює Німеччина, яка виступає лідером Європейської співдружності у багатьох питаннях, і прийняла Болонську систему кредитів для оцінювання рівня досягнень студентів з 2007 року, проте залишила за німецькими університетами «право вибору» підготовки студентів як за класичним, так і за новим європейським варіантом підготовки бакалаврів і магістрів [113].

Дослідник А. Турчин. у своїх дослідженнях проаналізував виконання програмних завдань Болонського процесу в сфері контролю й оцінювання навчальних досягнень студентів у німецьких ЗВО та зазначив, що контроль – невід'ємний складник процесу навчання, головною його метою, разом із визначенням та оцінюванням рівня сформованості навичок і вмінь, є управління ним. [113]. Варто зазначити, що український науковець В.М. Кухаренко, який присвятив багато досліджень темі оцінювання навчальних досягнень студентів, особливої уваги надає формуючому оцінюванню Німеччини [95].

У своїй дослідницькій роботі Х. Пайва (J. Paiva) [114], Х. Ліал (J. Leal) [114], А. Фігейра (Á. Figueira) [114] розглядають автоматизоване оцінювання знань спеціальності «Комп'ютерні науки», як ключовий інструмент у викладанні програмування, алгоритмічного мислення, масових онлайн-курсів (МООС). Виділяють наступні типи автоматизованого оцінювання: перевірка тестами – найпоширеніший метод (unit tests, input/output); статичний аналіз

коду – оцінка стилю, структури, якості; динамічний аналіз – виконання програм у різних умовах; оцінювання з використанням моделей і шаблонів; AI/ML-підходи – аналіз рішень студентів, передбачення помилок [114].

Перевагами автоматизованого оцінювання вважають: масштабованість (велика кількість студентів), швидкий зворотний зв'язок, об'єктивність оцінювання, зменшення навантаження на викладача. Але своєю чергою підкреслюють такі важливі виклики, як складність оцінювання креативних рішень, ризик «підгонки під тести», обмежена здатність оцінювання якості коду, питання академічної доброчесності (плагіат), технічна складність розробки систем [114].

Своєю чергою Л. Ваннер (L. Wanner) [115] наголошує, що використання автоматизованого оцінювання знань надасть можливість безперервного оцінювання знань бакалаврів, завдяки використанню програмного забезпечення для автоматизованого оцінювання. Такий підхід дозволить оперативно обробляти результати великої кількості студентів, забезпечить швидкий зворотний зв'язок і буде сприяти своєчасному виявленню прогалин у знаннях.

У роботі автор описує процес створення тестових завдань, організацію автоматизованого оцінювання, а також аналіз результатів навчання та відгуки студентів. Отримані результати свідчать про позитивний вплив використання автоматизованих інструментів оцінювання на якість навчання, підвищення мотивації студентів та ефективність освітнього процесу. Застосування регулярних тестів (квізів) підвищує активність студентів і стимулює систематичне опрацювання навчального матеріалу. Аналіз результатів та відгуків студентів підтверджує, що така модель оцінювання позитивно впливає на успішність і якість навчання. Водночас автоматизація зменшує навантаження на викладача та робить процес оцінювання більш масштабованим і об'єктивним [115].

Застосування регулярних тестів підвищує активність студентів і стимулює систематичне опрацювання навчального матеріалу. Аналіз результатів та відгуків студентів підтверджує, що така модель оцінювання

позитивно впливає на успішність і якість навчання. Водночас автоматизація зменшує навантаження на викладача та робить процес оцінювання більш масштабованим і об'єктивним [115].

Цікаві дослідження провели С. Бергін (S. Bergin) [116], А. Муні (A. Mooney [116] в університеті Мейнута (Ірландія, місто Мейнут). Вони досліджували впровадження інноваційного підходу до оцінювання в умовах виконання реального групового проєкту з розроблення програмного забезпечення. Особливу увагу приділялось проблемам оцінювання результатів навчання студентів у командній роботі, зокрема нерівномірному внеску учасників, недостатній прозорості оцінювання та складності визначення індивідуальних досягнень. Як результат, автори пропонують комплексну модель оцінювання, яка поєднує взаємооцінювання, самооцінювання та оцінювання викладачем. Такий підхід спрямований на підвищення об'єктивності, відповідальності студентів та їх активної участі у навчальному процесі, а також на узгодження оцінювання з практиками реальної інженерії програмного забезпечення [116].

Результати дослідження свідчать про підвищення мотивації студентів, більш активну участь у груповій роботі та точніше відображення індивідуального внеску кожного учасника. Крім того, підхід сприяє формуванню професійних компетентностей, зокрема компетентності з програмування, таких як командна робота, комунікація та рефлексія[116].

В дослідженнях закордонних авторів значна увага приділяється електронному оцінюванню. Так, С. Аламр (S. Alamr) [117], М. Уррутія (M. Urrutia) [117], Л. Карр (L. Carr) [117] у своїх роботах досліджують ключові фактори ефективного впровадження електронного оцінювання у процесі підготовки фахівців з комп'ютерних наук. Автори вважають, що електронне оцінювання відіграє важливу роль у підвищенні ефективності освітнього процесу завдяки таким перевагам, як покращення досвіду навчання та академічної успішності студентів, а також оптимізація викладацьких зусиль [117].

Це дослідження визначає фактори успіху використання методів електронного оцінювання з позиції викладачів, які здійснюють оцінювання завдань з програмування в університетах Великої Британії та Саудівської Аравії. У результаті виокремлено ключові фактори успішного впровадження електронного оцінювання, серед яких: автентичність завдань, надійність і валідність оцінювання, гнучкість систем, доступність та масштабованість.

Запропоновано узагальнену концептуальну модель, що може бути використана для вдосконалення систем електронного оцінювання у вищій освіті. Результати дослідження свідчать про недостатній рівень підготовки викладачів та недостатність спеціалізованих автоматизованих інструментів для оцінювання програмного коду. Також зазначається, що існуючі засоби електронного оцінювання не забезпечують належного рівня автоматизованого зворотнього зв'язку та неповною мірою відповідають вимогам викладачів, щодо гнучкості, доступності, масштабованості, практичності, валідності, надійності та автентичності [117].

Нідерландські дослідники Л. Бартман (L. Baartman) [117] та К. Квінлан (M. Quinlan) [117] стверджують, що практику оцінювання та зворотній зв'язок у вищій освіті необхідно трансформувати, щоб краще відповідати трьом цілям: сприянню навчанню, забезпеченню надійності ретельності оцінювання та інформуванню студентів про їхню працевлаштування. Вони провели дослідження у якому довели, що традиційні моделі оцінювання у вищій освіті є обмеженими, оскільки вони орієнтовані на підсумкові результати та не забезпечують належної підтримки процесу навчання. Програмне оцінювання розглядається як ефективна альтернатива, що дозволяє інтегрувати оцінювання в освітній процес [118].

У статті авторів К. Нгуєн (Q. Nguyen), Б. Рієнтіс (B. Rienties), Л. Тойтенель (L. Toetenel), Р. Фергюсон (R. Ferguson), Д. Вайтлок (D. Whitelock) [119], досліджується вплив розроблення комп'ютерного оцінювання на залученість студентів, їх задоволеність освітнім процесом та рівень успішності. Аналізуються різні моделі організації

оцінювання в курсах вищої освіти, зокрема частоту проведення тестувань, типи завдань і характер зворотного зв'язку [119].

На основі емпіричних даних встановлено, що регулярні формувальні оцінювання, своєчасний зворотний зв'язок та педагогічно обґрунтована розроблення завдань сприяють підвищенню активності студентів і покращенню результатів навчання. Дослідження підкреслює важливість інтеграції ефективних стратегій оцінювання у цифрові освітні середовища. [119].

Своєчасний і змістовний зворотний зв'язок є ключовим чинником підвищення задоволеності студентів та якості засвоєння навчального матеріалу. Виявлено, що поєднання різних типів завдань і можливість повторного виконання сприяють покращенню результатів навчання та зростанню показників успішності. Загалом доведено, що комп'ютерне оцінювання є ефективним інструментом підвищення якості освіти за умови його інтеграції в цілісну педагогічну стратегію [119].

У статті Л. Баруа (L. Barua) [120] та Б. Локі (B. Lockee) [120] розглядається концепція гнучкого оцінювання у вищій освіті як відповідь на виклики цифровізації та різноманітності освітніх потреб студентів. Дослідження має оглядовий характер і систематизує сучасні підходи, стратегії та практики організації оцінювання [120]. Автори аналізують різні форми гнучкості, зокрема варіативність форматів оцінювання, адаптацію термінів виконання завдань, персоналізацію оцінювальних траєкторій і використання цифрових технологій. Особливу увагу приділено ролі інклюзивності та студенторієнтованого підходу. У роботі підкреслюється, що гнучке оцінювання сприяє підвищенню мотивації студентів, їх залученості до навчання та формуванню відповідальності за власні результати [120].

У дослідженні авторів К. Гонсалвес (C. Gonsalves) та З. Лін (Z. Lin) розглядається поняття прозорості оцінювання бакалаврів комп'ютерних наук у закладів вищої освіти Великої Британії [121]. Дослідження ґрунтується на якісному аналізі інституційних положень оцінювання та виявляє розрив між формальними вимогами до прозорості та їх реальним сприйняттям студентами.

Автори показують, що положення часто орієнтовані на формальне дотримання стандартів, але не забезпечують фактичної зрозумілості процедур оцінювання. Доведено, що ефективна прозорість передбачає не лише відкритість інформації, але й її інтерпретованість, доступність і педагогічну доцільність. Важливу роль відіграє активне залучення студентів до процесу оцінювання та чітке пояснення критеріїв і процедур [121].

У статті М. Альфалех (M. Alfaleh) [122] обґрунтовано доцільність використання технологій ШІ для оцінювання результатів навчання у ЗВО Саудівської Аравії. Встановлено, що впровадження AI-орієнтованих систем сприяє підвищенню ефективності оцінювання, забезпечує оперативний зворотний зв'язок та знижує навантаження на викладачів. Доведено, що застосування ШІ відкриває можливості для персоналізації оцінювання відповідно до індивідуальних освітніх потреб здобувачів, а також підвищує об'єктивність оцінювання за рахунок мінімізації впливу суб'єктивних чинників. Водночас акцентовано увагу на необхідності врахування етичних аспектів використання AI, зокрема забезпечення прозорості алгоритмів, захисту персональних даних та запобігання алгоритмічній упередженості.

Обґрунтовано, що ефективне впровадження AI-орієнтованого оцінювання потребує відповідного технологічного забезпечення, підготовки педагогічних кадрів і нормативного регулювання. Підкреслено доцільність поєднання інструментів ШІ з традиційними підходами до оцінювання, що забезпечує комплексність і педагогічну доцільність освітнього процесу. Таким чином, AI-орієнтоване оцінювання розглядається як перспективний напрям модернізації системи вищої освіти за умови його збалансованого, етично виваженого та науково обґрунтованого впровадження.

Цікаву пропозицію, щодо оцінювання у своїй дослідницькій роботі, С. Хан (S. Khan), М. Алам пропонують новий підхід, а саме систему оцінки якості вищої освіти, яка оцінює ефективність університетів на основі успішності студентів, що закінчили навчання. Було виявлено, що успішність студентів після завершення курсу є результатом освітнього процесу.

Наголошують, що ЗВО повинні нести відповідальність за успішність своїх студентів навіть після завершення ними курсу. Включення цього виміру до оцінки якості дозволить оцінювати освітні установи за цими ознаками. Для впровадження та тестування системи було використано хмарну технологію великих даних BigQuery для проведення аналітики [123].

У своїй роботі, Л. Огнівчук запропонувала систему оцінювання навчальних досягнень студентів ВЗО на основі компетентнісного підходу. Оцінювання навчальних досягнень студентів ВНЗ на основі компетентнісного підходу є актуальним питанням педагогіки, психології і методики та потребує проведення досліджень різних підходів та пошуку ефективних методів проведення такого оцінювання [110]. Авторка розглядає математичну модель та метод оцінювання рівня сформованості компетенцій у студентів ЗВО на основі апарату нечіткої логіки, доводить доцільність її використання, описує особливості проведення аналізу та обробки даних, визначає подальші напрями дослідження цього питання. [110].

На думку В. Бочарнікової оцінювання, що здійснюється в ході контролю навчальних досягнень студентів, стимулює їх до активної навчально-пізнавальної діяльності, а отже, виступає навчаючим фактором, що зумовлює позитивні результати навчального процесу [124].

Український науковець В. Кухаренко, за результатами багаторічних досліджень, прийшов до висновку, що на етапі розвитку наукових підходів до оцінювання навчальних досягнень студентів особливої уваги потребує формулює оцінювання [95]. Особливої уваги, за висновками В. Кухаренка, заслуговує досвід Німеччини щодо впровадження формульовального оцінювання, яке акцентує на персональному прогресі студента, а не на оцінці його особистості [95].

У своїх дослідженнях С. Литвинова зазначає, що за допомогою формульовального оцінювання можна підвищити ефективність контролю та оцінювання знань студентів. Є погодження з її думкою, що формулює оцінювання застосовується викладачами для отримання даних щодо поточного

стану засвоєння знань студентами конкретної теми і визначення найближчих кроків щодо їх покращення [125],[109].

Особливості формувального оцінювання полягають в тому, що оцінюється робота студента в досягненні цілей навчання, а не його особистість; пропонується чіткий алгоритм визначення оцінки, зрозумілий студенту; увага акцентується на персональному прогресі студента, а не на оцінці. Форми проведення формувального оцінювання учнів пропонуються такі: рефлексивні техніки (сигнали рукою, карточками) для з'ясування і виявлення складних питань; уточнюючі питання; аналітичні питання; міні-тести; перевірка творчих робіт з метою виявлення помилок тощо [125]. Серед функцій виділяють: навчальну, стимулюючу, контролюючу [109].

Останнім часом, для ефективного оцінювання рівня знань студентів ЗВО, набуває поширення так звана таксономія Блума. «Таксономія» позначає систематизацію об'єктів, які розташовуються за ієрархією, послідовно, за наростаючою складністю. Цей термін у перекладі з грецької мови і означає «розміщення за порядком», «закон» [109].

У 50-х роках минулого сторіччя, колектив американських науковців під керівництвом Б. Блума, розробив систему впорядкованих педагогічних цілей та результатів, яка згодом була висвітлена у спеціальному виданні «Таксономія» [126]. Розгорнута система Б. Блума представлена у таблиці 2.2. Перші три (знання, розуміння, застосування) є результатами нижчого порядку (потребують від студентів мисленнєвих операцій низького рівня), а наступні три (аналіз, синтез, оцінювання) – вищого (вимагають застосування мислення високого рівня).

Таблиця 2.2

Таксономія навчальних цілей та результатів Б. Блума

Навчальна мета/рівень	Результат, якого досягають здобувачі освіти	Дії, що демонструють ці результати. Чи можуть здобувачі освіти
Знання	запам'ятовування інформації: факти, поняття, терміни та теорії	відтворити раніше засвоєний матеріал на-пам'ять
Розуміння	сприймання інформації й передача в іншій формі (інші слова, інша мова, зображення і т. д.)	пояснити сенс інформації, «переклавши» її своїми словами і прикладами, перевести з мови слів у графічну, математичну мову, викласти коротко чи розширено, екстраполювати
Застосування	використання без зовнішньої підказки в новому контексті, у життєвій ситуації засвоєні знання, теорії	застосувати вивчене (правило, формулу, теорію) у нових умовах чи ситуаціях, застосувати поняття, правила та ін. у житті чи навчанні
Аналіз	розділення інформації, явища на окремі складники, порівняння, встановлення зв'язків, логіки, визначення структури	виокремлювати частини та встановлювати зв'язок між ними, структуру, логіку, схожість чи відмінність
Синтез	творче поєднання частин або елементів у нове ціле	скомбінувати елементи об'єкта по-новому для отримання нового продукту з іншими властивостями
Оцінка	формлювання кількісних або якісних оцінок, ціннісних суджень	оцінити значення, ефективність, вчинки людей, концепцію, аргумент, виходячи з певних критеріїв, підтримати або спросту

В 2001 році авторами Л. Андерсоном та Д.Кратволом була переглянута таксономія Блума, результатом якої стала поява так званої «переглянутої» таксономії Блума [127]. На нашу думку сутнісний зміст таксономії не змінився.

У таблиці 2.3 автори О.Пометун, Н.Гупан показують шість рівнів мисленнєвої діяльності студентів (учнів) за переглянутою таксономією [127].

Таблиця 2.3

Шість рівнів мисленнєвої діяльності студентів за переглянутою таксономією [102]

Навчальна мета/рівень	Результат, якого досягають студенти	Дії, що демонструють ці результати: Чи можуть здобувачі освіти
Пригадування	пригадують, визначають, ідентифікують	відтворити інформацію, яку вони прочитали чи почули
Усвідомлення	пояснюють поняття, ідеї чи концепції, про які вони читали чи чули	пояснити своїми словами, прикладами, віднести до певної групи, спрогнозувати послідовність, наступний крок
Застосовування	застосовують набуті знання, уміння, навички в іншій ситуації, контексті	використовувати інформацію (формулу, правило, алгоритм тощо) для розв'язання іншого завдання
Аналіз	визначають, виокремлюють і порівнюють частини, функції, структуру	розділити інформацію або явище, предмет на складники і порівняти їх, виявити зв'язки
Оцінювання	оцінюють результати дослідження з точки зору його завдань, формулюють оцінні судження	оцінити достовірність і значення отриманої інформації за критеріями, результати аналізу, перспективи діяльності
Створення	розробляють щось нове на основі отриманих знань, умінь, навичок	створити, сконструювати, сформулювати, спланувати щось нове, використавши відоме, засвоєне

Враховуючи позитивний досвід викладачів Національного технічного університету "Харківський політехнічний інститут", зокрема В. Кухаренка, щодо проведення експерименту із застосування таксономії Блума для оцінювання навчальних досягнень студентів, застосуємо цей підхід для розроблення авторської методики оцінювання навчальних досягнень майбутніх бакалаврів комп'ютерних наук [108].

Слід зауважити, що саме таксономічний підхід Блума був покладений в основу *контрольно-оцінювального блоку* моделі використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук. Оцінювання рівня знань студентів-програмістів здійснюється за допомогою таксономії Блума, яка містить 6 рівнів складності [109].

У цій моделі розглядається контрольно-оцінювальний блок, за допомогою якого можна визначити рівень компетентності студента, не тільки в кінці курсу (іспит, залік, комп'ютерного практикуму), а також протягом всього курсу навчальної дисципліни використовувати рівні таксономії Блума, які пов'язані з системою оцінювання ECTS та рейтинговим балом. Критерії оцінювання комп'ютерного практикуму прописуються для кожної теми за 6 рівнями складності. Кожний виконаний комп'ютерний практикум відповідає своєму рівню з визначеною кількістю балів [109].

Студент може вибирати і виконувати завдання відповідно до вибраного рівня його знань. Це спонукає його до саморозвитку, формування стилю навчання. На цьому етапі він завжди може підвищити свій професійний рівень, повернутися для відпрацювання більш складного завдання або закріпити складні моменти в розв'язанні завдань нижчих рівнів і т.п.

Кожний комп'ютерний практикум буде відповідати своєму рівню з визначеною кількістю балів.

При формуванні контрольно-оцінювального блоку цієї моделі, слід пом'ятати, що роботодавець ставить високі вимоги до працівників, а рівень знань, умінь і навиків профільних випускників не відповідає цим вимогам, оскільки носить здебільшого теоретичний характер. Це, у свою чергу, вимагає постійної корекції освітньо-професійних програм, навчальних планів і навчальних дисциплін, що вивчаються в ЗВО, регулярної перепідготовки кадрів, удосконалення системи не тільки оцінювання рівня знань майбутніх бакалаврів комп'ютерних наук програмістів, але здатності студентів-програмістів знаходити правильні рішення в різних проблемних ситуаціях [109].

При оцінюванні рівня знань, велику увагу приділяється контролю за результатами самостійної роботи майбутнього бакалавра комп'ютерних наук, як невід'ємної складової частини навчально-виховного процесу, що має на меті забезпечити зворотній зв'язок «студент-викладач» [109].

У контексті вебпрограмування, таксономія Блума, яка характеризує рівні пізнавальної діяльності бакалаврів комп'ютерних наук, допомагає структурувати вивчення мов програмування, фреймворків, інтегрованих середовищ розробки, оцінити, який рівень розуміння та вміння може продемонструвати студент-програміст — від базового запам'ятовування синтаксису до здатності створювати нові архітектурні рішення програмного забезпечення, складні інформаційні системи.

Формування завдань здійснюється у відповідності до сфер пізнавальних (Cognitive Domain), емоційних (Affective Domain) та рухових (Psychomotor Domain) цілей. Пізнавальні цілі охоплюють усе, що пов'язано з набуттям знань і розвитком розумових навичок. Емоційні цілі містять у собі всі завдання, пов'язані з формуванням цінностей, відношень, розвитком емоційного самоконтролю студентів. До рухових цілей належить розвиток рухових навичок, фізичної витривалості [109].

При складанні комп'ютерних практикумів особлива увага приділяється пізнавальним цілям навчання, що структуровані відповідно до шести рівнів таксономії Блума: пригадування (відтворення фактичної інформації); усвідомлення (розуміння змісту та зв'язків навчального матеріалу); застосування (використання знань у нових ситуаціях); аналіз (виділення складових і встановлення зв'язків між ними); оцінювання (критичне оцінювання та обґрунтування рішень); створення (синтез і генерація нового продукту або рішення), що є найвищим рівнем пізнавальної діяльності.

Ця класифікація є класифікацією мислення, організованого за рівнями складності й дає викладачам і студентам можливість вчитися та діяти в інформаційно-освітньому просторі, забезпечує просту структуру для багатьох видів питань. [109].

2.3.1 Оцінювання комп'ютерних практикумів за таксономією Блума

Розглянемо зміст рівнів таксономії Блума із визначенням відповідних видів навчальної діяльності (дієслівних форми) для проєктування практичних завдань з дисципліни «Веборієнтовані технології. Frontend-розробка» спеціальності «Комп'ютерні науки», яка вивчається здобувачами освіти третього курсу підготовки бакалаврів комп'ютерних наук [109] (табл. 2.4).

Таблиця 2.4

Зміст діяльності студента під час виконання практичних завдань відповідно до рівнів таксономії Блума

Рівень	Когнітивна дія	Очікувані навчальні результати	Приклади завдань
1. Пригадування (Запам'ятовування)	Відтворити, перерахувати, визначити	- знає HTML-теги, CSS-селектори, базові концепції JavaScript	- перелічити основні CSS-селектори
		- пам'ятає HTTP-методи, статус-коди, структуру URL	- назвати типи подій у DOM
		- розрізняє типи баз даних, основні команди Git	- визначити різницю між GET і POST.
2. Усвідомлення (Розуміння)	Пояснити, описати, інтерпретувати	- пояснює стилізацію (CSS), призначення DOM-дерева	- пояснити, чому відбувається reflow у браузері
		- описує конструкції та масиви мови JS, цикл подій JS,	- описати принцип роботи fetch API

Рівень	Когнітивна дія	Очікувані навчальні результати	Приклади завдань
3. Застосування	Використовувати, реалізовувати, демонструвати	принципи SPA/SSR	
		- розуміє клієнт-серверну взаємодію; використовує HTTP-запити	- інтерпретувати структуру відповіді JSON
		- створює адаптивні веб-застосунки за допомогою HTML/CSS/JS	- створити компонент галереї на JS
4. Аналіз	Розбирати, порівнювати, діагностувати	- пише інтерактивні елементи, валідацію форм, працює з API	- реалізувати форму логіну з валідацією
		- розробляє прості запити на сервери на	- написати REST-ендпоінт
		- аналізує структуру DOM, CSS-спадкування, продуктивність JS	- знайти помилку в асинхронному коді
5. Оцінювання		- відстежує помилки через DevTools, логіку маршрутизації	- діагностувати проблеми у відповідях API
		- визначає причини уповільнень у фронтенді	- порівняти два способи стану у React (Context vs Redux).
		- обирає відповідні	-провести code review

Рівень	Когнітивна дія	Очікувані навчальні результати	Приклади завдань
	Оцінити, аргументувати, обирати	фреймворки, бібліотеки та архітектури	
		- оцінює якість коду, безпечність застосунку (XSS, CSRF)	- обґрунтувати вибір SPA чи SSR
		- аналізує ефективність алгоритмів і технологій	- оцінити ризики в системі авторизації
6. Створення	Проектувати, розробляти, інтегрувати	- створює повноцінні алаптивні веб-frontend додатки	- створити веб-додаток із реєстрацією й особистим кабінетом
		- застосовує REST API	- реалізувати власний React-хук або компонентну бібліотеку
		- впроваджує реєстрацію авторизацію (JWT/OAuth)	- спроектувати й реалізувати API для мобільного застосунку
		- налаштовує CI/CD, Docker, деплой	

Поєднання таксономії Блума з класичною бальною системою оцінювання знань студентів (див. табл. 2.5) дає змогу оцінювати рівень сформованості компетентностей бакалаврів комп'ютерних наук як на проміжних етапах навчання, так і на завершальному етапі підготовки. Виконуючи завдання

різного рівня складності, студент може визначити власний рівень підготовки та усвідомити напрями подальшого професійного розвитку [109].

Таблиця 2.5

Таксономія Блума і бально-рейтингова система в оцінюванні досягнень студентів

Рівень	Назва рівня за таксономією Блума	ECTS	Бали	Оцінка	Рівень за DigComp 3.0
VI	Створення	A	95-100	відмінно	7–8 (Експертний)
V	Оцінювання	B	85-94	дуже добре	5-6 (Просунутий)
IV	Аналіз	C	80-84	добре	
III	Застосування	D	70-79	дуже задовільно	4 (Середній)
II	Усвідомлення	Dx	65-69	задовільно	1–3 (Базовий)
I	Пригадування	E	60-64	достатньо	

Запропонована методика оцінювання на основі таксономії Блума є *універсальною* та може бути адаптована до різних освітніх компонентів. Її апробація на прикладі Frontend-розробки (табл. 2.5) та подальша реалізація через практичні завдання з мови JavaScript (див. підрозділ 3.5) демонструють доцільність застосування такого підходу до дисциплін циклу професійної підготовки бакалаврів і комп'ютерних наук.

Підходи до реалізації підсумкового оцінювання навчальних досягнень студентів з програмування обґрунтовано в Додатку Г.

Висновки до розділу 2

У цьому підрозділі аналізуються роботи українських та зарубіжних авторів, щодо питань контролю та оцінювання результатів навчальної діяльності студентів закладів вищої освіти. Висвітлюються принципи формуального оцінювання та рівні пізнавальних, емоційних та рухових цілей.

Проведене моделювання процесу веборієнтованого навчання програмування майбутніх бакалаврів комп'ютерних наук дозволило сформулювати такі висновки:

- Обґрунтовано теоретико-методологічні засади моделювання, що базуються на синтезі класичних дидактичних принципів та сучасних парадигм цифрової трансформації (Веб 4.0). Встановлено, що перехід від інструментального використання вебресурсів до формування інтелектуальних навчальних систем потребує реалізації принципів неперервності навчання та інтелектуальної взаємодії студента зі штучним інтелектом.

- Розроблено авторську модель веборієнтованого навчання, яка забезпечує оптимальне співвідношення навчальної діяльності: 30 % традиційної, 50 % дистанційної (на засадах Cloud-native) та 20 % проєктної. Ключовою особливістю моделі є перенесення повного циклу розробки у хмарні середовища (Cloud IDE, GitHub), що усуває технічні бар'єри локального ПЗ та гарантує рівні умови для всіх учасників освітнього процесу.

- Удосконалено модель формування компетентності з програмування, інтегруючи складник штучного інтелекту в режимі інтелектуальної взаємодії студента зі штучним інтелектом. Визначено, що сучасна фахова компетентність бакалавра охоплює не лише знання синтаксису, а й навички ефективного промпт-інжинірингу, критичного аналізу безпеки згенерованого коду та дотримання норм академічної доброчесності в умовах Веб 4.0.

- Спроектовано комплексну систему оцінювання навчальних досягнень студента, яка поєднує таксономію Блума, бально-рейтингову систему та рамку DigComp 3.0. Запропоновано методику, у якій рівні інтелектуальної діяльності за Блумом корелюють із рівнями цифрової компетентності (від базового 1–3 до експертного 7–8), що забезпечує об'єктивний моніторинг професійного зростання студента.

- Впроваджено механізми об'єктивізації та автоматизації контролю, зокрема через аналіз «цифрового сліду» студента у репозиторіях GitHub та

інтеграцію результатів автоматизованих систем перевірки алгоритмів (Online Judges) безпосередньо у навчальний рейтинг. Це мінімізує суб'єктивізм викладача та надає студенту миттєвий зворотний зв'язок для самокорекції.

– Визначено ефективність використання ІІІ-асистентів для викладача, зокрема автоматизованого Code Review та аналітики труднощі/затримки у процесі програмування. Така інтеграція дозволяє суттєво оптимізувати менеджмент освітнього процесу та спрямувати додатковий час на підвищення ефективності засвоєння навчального матеріалу.

Список використаних джерел до Розділу 2

1. Проскура С. Л. Модель формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*, 2019. № 3 (21). С. 104–112. URL: <https://fmo-journal.fizmatsspu.sumy.ua/publ/4-1-0-522> (дата звернення: 27.03.2026).

2. Поняття «моделі» та «моделювання». URL: <https://buklib.net/books/24846/> (дата звернення: 27.03.2026).

3. Аніщенко О. В., Яковець Н. І. Сучасні педагогічні технології : курс лекцій : навч. посіб. Ніжин : Вид-во НДУ ім. М. Гоголя, 2005. 198 с. URL: <https://www.scribd.com/document/995740523/> (дата звернення: 27.03.2026).

4. Зубик Л. Модель формування професійної компетентності ІТ-спеціалістів під час вивчення фахових дисциплін. *Науковий вісник Миколаївського національного університету ім. В. О. Сухомлинського. Серія: Педагогічні науки*, 2016. № 1. С. 83–89. URL: http://nbuv.gov.ua/UJRN/Nvmdup_2016_1_18 (дата звернення: 27.03.2026).

5. Стрюк К. М. Шляхи формування професійної компетентності майбутніх молодших спеціалістів із комп'ютерної інженерії. *Педагогічні науки: реалії та перспективи*, 2018. Вип. 63. С. 173–177. URL: <https://www.chasopys.ps.npu.kiev.ua/archive/63-2018/63-2018.pdf> (дата звернення: 27.03.2026).

6. Литвинова С. Г. Технології навчання учнів у хмаро орієнтованому навчальному середовищі загальноосвітнього навчального закладу. *Інформаційні технології і засоби навчання*, 2015, Том 47, №3. URL: https://journal.iitta.gov.ua/index.php/itlt/article/view/1239/927.pdf?utm_source (дата звернення: 27.03.26).
7. Литвинова С. Г. On-line навчальне середовище вчителя-предметника загальноосвітнього навчального закладу. *Інформаційні технології і засоби навчання*, 2010. № 5(19). URL: <https://lib.iitta.gov.ua/id/eprint/195/1/356-1047-1-PB.pdf> (дата звернення: 27.03.26).
8. Йосі Т., Кумара В. Відкрита освіта: колективний розвиток освіти через відкриті технології, відкритий контент та відкрите знання / пер. з англ. А. Іщенка, О. Насіка. Київ : Наука, 2009. 256 с. URL: http://library.nlu.edu.ua/POLN_TEXT/KNIGI_2009_2/vidkryta_osvita_2009.pdf (дата звернення: 27.03.2026).
9. Медведєв Р. П. Адаптивне навчання у фаховій передвищій освіті в умовах інклюзії та цифровізації *Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців: методологія, теорія, досвід, проблеми*, 2025. Вип. 76. С. 105–113. DOI: <https://doi.org/10.31652/2412-1142-2025-76-105-113> (дата звернення: 27.03.2026).
10. Величко В. Є., Ганієв О. С., Жадан С. С. Штучний інтелект як інструмент аналізу і класифікації задач з програмування в освітньому процесі *Технології електронного навчання*, 2024. № 1. DOI: <https://doi.org/10.31865/2709-840082024316949> (дата звернення: 27.03.2026).
11. Marienko M., Nosenko Y., Shyshkina M. Personalization of learning using adaptive technologies and augmented reality. *arXiv*, 2020. URL: <https://arxiv.org/abs/2011.05802> (дата звернення: 27.03.2026).
12. Корнієнко О. С. Використання штучного інтелекту для персоналізації дистанційного навчання студентів спеціальності «Інженерія програмного забезпечення». *Вісник Херсонського національного технічного*

університету, 2025. № 2(93). DOI: <https://doi.org/10.35546/kntu2078-4481.2025.2.2.20> (дата звернення: 27.03.2026).

13. Мозгальов А. А., Кізім С. С. Компетентісний підхід як методологічна основа підготовки фахівців з інформаційних технологій. *Науковий альманах мистецтва та освіти*, 2025. № 1. DOI: <https://doi.org/10.31652/3041-1017-SAAE-2025.1.14> (дата звернення: 27.03.2026).

14. Концедайло В. В. Розробка моделі використання ігрових симуляторів для формування професійних компетентностей майбутніх інженерів-програмістів. *Журнал Житомирського державного університету імені Івана Франка. Педагогічні науки*, 2018. № 1(92). С. 90–96. DOI: [https://doi.org/10.35433/pedagogy.1\(92\).2018.90-96](https://doi.org/10.35433/pedagogy.1(92).2018.90-96) (дата звернення: 27.03.2026).

15. Дегтярьова Н. Петренко С., Вернидуб Г., Тутова Н., Мигаль В. Реалізація диференційованого підходу при навчанні програмуванню мовою Python здобувачів загальної середньої освіти. *Сучасні інформаційні технології та інноваційні методи навчання в підготовці фахівців, методологія, теорія, досвід, проблеми*, 2024. № 72. С. 53–61. DOI: <https://doi.org/10.31652/2412-1142-2024-72-53-61> (дата звернення: 27.03.2026).

16. Фіялка С. Б., Гончарук Т. Г. Особистісно орієнтований підхід під час підготовки студентів спеціалізації «Видавнича справа та редагування». *Обрії друкарства*, 2023. № 1(13). С. 120–131. DOI: [https://doi.org/10.20535/2522-1078.2023.1\(13\).287503](https://doi.org/10.20535/2522-1078.2023.1(13).287503) (дата звернення: 27.03.2026).

17. Павленко М., Павленко Л., Павленко Є., Молчанов В. Інноваційні можливості впровадження онлайн-платформ для навчання програмування майбутніх вчителів інформатики. *Молодь і ринок*, 2024. № 5. С. 299. DOI: <https://doi.org/10.24919/2308-4634.2024.309871> (дата звернення: 27.03.2026).

18. Shyshkina M. The Problems of Personnel Training for STEM Education in the Modern Innovative Learning and Research Environment. *arXiv:1007.08562*, 2018. URL: <https://arxiv.org/abs/1807.08562> (дата звернення: 27.03.2026).

19. Ткачук Г. В. Методика використання різних форм та методів навчання під час вивчення програмування. *Молодь і ринок*. 2024. № 7. С.230
DOI: <https://doi.org/10.24919/2308-4634.2024.314170> (дата звернення: 27.03.2026).

20. Gasnas A. A STEAM project model within the object-oriented programming discipline. *Acta et Commentationes, Sciences of Education*, 2024. Vol. 35(1). P. 35–46. DOI: <https://doi.org/10.36120/2587-3636.v35i1.35-46> (дата звернення: 27.03.2026).

21. Litherland K., Kluge A. Learning to program as empirical inquiry: using a conversation perspective to explore student programming processes. *Computer Science Education*, 2023. Vol. 33(4). P. 495–519. DOI: <https://doi.org/10.1080/08993408.2023.2290410> (дата звернення: 27.03.2026).

22. Station Rotation. Blended Learning Universe. Christensen Institute. URL: <https://www.blendedlearning.org/models/> (date of access: 27.03.2026)

23. Flex. URL: <https://www.blendedlearning.org/models/#flex> (дата звернення: 27.03.2026).

24. A La Carte. <https://www.blendedlearning.org/models/#ala> (дата звернення: 27.03.2026).

25. Enriched Virtual. <https://www.blendedlearning.org/models/#enrich>

26. Pashchenko O., Koroviaka Y., Mamaikin O., Kozhushkina T., Rastsvietaiev V. *Effectiveness of blended learning in the informatics course: analysis of online and offline formats* *Молодь і ринок*, 2025. DOI: <https://doi.org/10.24919/2308-4634.2025.334056> (дата звернення: 27.03.2026).

27. Udemy. *Online Learning Platform*. URL: <https://www.udemy.com/>

28. Coursera. *Online Learning Platform* URL: <https://www.coursera.org/>

29. Prometheus/ *Online Learning Platform*
URL: <https://prometheus.org.ua/>

30. Vakaliuk T. A., Chyzhmotria O. V., Chyzhmotria O. H., Didkivska S. O., Kontsedailo V. V. The use of massive open online courses in teaching the fundamentals of programming to software engineers. *Educational Technology Quarterly*, 2022. DOI: <https://doi.org/10.55056/etq.37> (дата звернення: 27.03.2026)
31. Гончаренко С. У. Український педагогічний словник / Семен Гончаренко ; гол. ред. С. Головка. Київ : Либідь, 1997. 376 с. URL: <https://lib.iitta.gov.ua/id/eprint/106820/1/Гончаренко.%20Педагогічний%20словник%20%281%29.pdf> (дата звернення: 27.03.2026)
32. Семінарське заняття, індивідуальне заняття. Педагогіка : підручники для студентів онлайн. URL: https://pidru4niki.com/88627/pedagogika/seminarske_zanyattya (дата звернення: 11.05.2026).
33. Бересток О. В. Синхронний та асинхронний режими електронного навчання: стратегії, методи, завдання *Інженерні та освітні технології*, 2021. Т. 9, № 1. С. 18–32. DOI: <https://doi.org/10.30929/2307-9770.2021.09.01.02> (дата звернення: 27.03.2026).
34. Кулінка Ю. С. Методологічні аспекти впровадження веб-квест-технології як виду проектної діяльності для підвищення якості освіти. *Педагогіка вищої та середньої школи*, 2014. Том 43. С. 145–151. DOI: <https://doi.org/10.31812/educdim.v43i0.2778> (дата звернення: 27.03.2026).
35. Козолуп Є. В. Практичний кейс навчання учнів програмування шляхом реалізації науково-дослідницьких проєктів *Відкрите освітнє е-середовище сучасного університету*, 2025. DOI: <https://doi.org/10.28925/2414-0325.2025.185> (дата звернення: 27.03.2026).
36. Глазова В. В., Глазова В. А., Касьянова Т. М. Інтеграція ігрових елементів та цифрових інструментів у сучасні методи навчання програмуванню. *Технології електронного навчання*, 2024. Т.8. С.24-29. DOI: <https://doi.org/10.31865/2709-840082024316932> (дата звернення: 27.03.2026).

37. Бабкін В. Прошкін В. Проектні методи навчання як тренди фахової підготовки майбутніх фахівців ІТ *Фізико-математична освіта*, 2021. № 3(29). С. 45–50. DOI: <https://doi.org/10.31110/2413-1571-2021-029-3-006>
38. Zawacki-Richter Olaf, Marin Victoria I., Bond Melissa, Gouverneur Franziska. *Systematic review of research on artificial intelligence applications in higher education – where are the educators?* International Journal of Educational Technology in Higher Education, 2019. Vol. 16, Article 39. DOI: <https://doi.org/10.1186/s41239-019-0171-0>.
39. Шаклеїна І. Хмарні сервіси для підтримки командної роботи у вивченні об'єктно-орієнтованого програмування. *Молодь і ринок*, 2025. Т.№7-8. С.239-240 DOI: <https://doi.org/10.24919/2308-4634.2025.336063>
40. Revelo-Sánchez O., Collazos C. A., Barón A. A. **Collaborative** Modelling of Activities to Support the Teaching of Programming: An Empirical Study. *Interaction Design and Architecture(s)*, 2024. № 63, P.131–155. DOI: <https://doi.org/10.55612/s-5002-063-007> (дата звернення: 10.05.2026).
41. Стеценко Н. В., Ткачук Г. О. Особливості організації самостійної роботи учнів з використанням цифрових ресурсів, *Молодь і ринок*, 2024. № 11(231). С. 28-35. DOI: <https://doi.org/10.24919/2308-4634.2024.316380> URL: <https://mir.dspu.edu.ua/article/view/316380> (date of access: 27.03.2026).
42. Kose U., Deperlioglu O. Intelligent learning environments within blended learning for ensuring effective C programming course. *arXiv*, 2012. DOI: <https://doi.org/10.48550/arXiv.1205.2670> (date of access: 10.05.2026).
43. Liu J., Li S. Toward Artificial Intelligence-Human Paired Programming: A Review of the Educational Applications and Research on Artificial Intelligence Code-Generation Tools. *Journal of Educational Computing Research*, 2024. Vol. 62(5) P.1165 – 1195. DOI: <https://doi.org/10.1177/07356331241240460> (date of access: 27.03.2026).
44. Elnaffar S., Rashidi F., Abualkishik A. Teaching with AI: A Systematic Review of Chatbots, Generative Tools, and Tutoring Systems in Programming

Education. *arXiv*, 2025. URL: <https://arxiv.org/abs/2510.03884> (date of access: 27.03.2026).

45. Li Z., Lu S., Guo D. et al. Automating Code Review Activities by Large-Scale Pre-training. *arXiv*, 2022. URL: <https://arxiv.org/abs/2203.09095> (date of access: 27.03.2026).

46. Sarsa S., Denny P., Hellas A., Leinonen J. Automatic Generation of Programming Exercises and Code Explanations using Large Language Models. *arXiv*, 2022. URL: <https://arxiv.org/abs/2206.11861> (date of access: 27.03.2026).

47. Олексюк В. П. Досвід організації віртуальних лабораторій на основі технологій хмарних обчислень. *Інформаційні технології в освіті*, 2014. № 20. С.128-138. DOI: <https://doi.org/10.14308/ite000503> URL: <https://ite.kspu.edu/index.php/ite/article/view/206/223> (date of access: 27.03.2026).

48. Gurzhii A., Glazunova O., Voloshyna T., Korolchuk V., Yakobchuk O. Cloud resources and services for training of future specialist of information technologies: selection criteria, case studies *Journal of Information Technologies in Education*, 2019. № 38. DOI: <https://doi.org/10.14308/ite000699> (date of access: 27.03.2026).

49. Волк М. О., Лабазов В. Г., Кипаренко Д. О. та ін. Розподілене комп'ютерне моделювання в системах хмарних обчислень *Вісник Херсонського національного технічного університету*, 2024. № 1. DOI: <https://doi.org/10.35546/kntu2078-4481.2024.1.29>

50. Douce C., Livingstone D., Orwell J. Automatic test-based assessment of programming: A review *Journal on Educational Resources in Computing*, 2005. Vol. 5(3). DOI: <https://doi.org/10.1145/1163405.1163409> (date of access: 27.03.2026).

51. Combéfis S. Automated Code Assessment for Education: Review, Classification and Perspectives on Techniques and Tools *Software*, 2022. Vol. 1(1). P. 3–30. DOI: <https://doi.org/10.3390/software1010002> (date of access: 27.03.2026).

52. Cain A., Woodward C. J. Examining student reflections from a constructively aligned introductory programming unit. *Computing Education 2013* :

Proceedings of the 15th Australasian Computing Education Conference (ACE 2013). Adelaide, Australia, 2013. P. 127–136. URL: https://research.monash.edu/en/publications/examining-student-reflections-from-a-constructively-aligned-intro/?utm_source (date of access: 27.03.2026).

53. Sitthiworachart J., Joy M. Effective peer assessment for learning computer programming. *Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE 2004)*. Leeds, UK, 2004. P. 122–126. DOI: <https://doi.org/10.1145/1007996.1008030> (date of access: 27.03.2026).

54. Leyk T., McInvale R., Chen L. Structured Peer Learning Program – An Innovative Approach to Computer Science Education *arXiv preprint:1703.04174*, 2017. URL: https://arxiv.org/abs/1703.04174?utm_source (date of access: 27.03.2026).

55. Mihail R. P., Rubin B., Goldsmith J. Online discussions: Improving education in CS?. *Proceedings of the 45th ACM Technical Symposium on Computer Science Education (SIGCSE 2014)*. Atlanta, USA, 2014. P. 409–414. DOI: <https://doi.org/10.1145/2538862.2538898> (date of access: 27.03.2026).

56. Gegenfurtner A., Bedenlier S., Ebner C., Keskin Ö., Händel M. *Designing effective synchronous online learning. Designing Effective Digital Learning Environments*. Routledge, 2024. P. 241–254. DOI: <https://doi.org/10.4324/9781003386131-23> (date of access: 27.03.2026).

57. Богачков Ю. М., Царенко В. О. Методика застосування вебінар орієнтованих платформ у навчальному процесі з інформатики старшої школи. *Інформаційні технології в освіті*, 2013. № 14. С. 42–47. URL: https://lib.iitta.gov.ua/id/eprint/892/?utm_source (дата звернення: 27.03.2026).

58. Li J., Durmus E., Cardie C. Exploring the Role of Argument Structure in Online Debate *Persuasion.arXiv preprint*, 2020. https://arxiv.org/abs/2010.03538?utm_source

59. DeSantis J. Accelerating Student Engagement with Asynchronous Online Debates. *College Teaching*, 2022. Vol. 71, No. 2. P. 103–111. DOI: <https://doi.org/10.1080/87567555.2022.2027857> (date of access: 27.03.2026).
60. Кочерга Є. В. «Мозковий штурм» як ефективний метод навчання. *Вісник Науково-методичного центру навчальних закладів сфери цивільного захисту*, 2017. № 27. С. 35–38. URL: <https://repositsc.nuczu.edu.ua/bitstream/123456789/7036/1/Kocherha%2035-38.pdf> (дата звернення: 27.03.2026).
61. Chikwiriro H. Chaka, P. Mavhemwa P. The Impact of Cloud Based IDEs on Programming High Level Languages for Educational Purposes on Mobile Devices . *International Journal of Engineering Research & Technology*, 2014. Vol. 3(4) URL: https://www.ijert.org/the-impact-of-cloud-based-ides-on-programming-high-level-languages-for-educational-purposes-on-mobile-devices?utm_source (date of access: 27.03.2026).
62. Fylaktopoulos G., Goumas G., Skolarikis M., et al. (2016). An overview of platforms for cloud based development. *SpringerPlus*, 2016. Vol. 5, P. 38. DOI: <https://doi.org/10.1186/s40064-016-1688-5> (date of access: 27.03.2026).
63. URL: <https://replit.com/> (date of access: 27.03.2026).
64. URL: <https://codepen.io/> (date of access: 27.03.2026).
65. URL: <https://jsfiddle.net/> (date of access: 27.03.2026).
66. Official Website. URL: <https://vercel.com/> (date of access: 27.03.2026).
67. Official Website. URL: <https://www.netlify.com> (date of access: 27.03.2026).
68. Official Website. URL: <https://firebase.google.com/> (date of access: 27.03.2026).
69. Official Website. <https://www.heroku.com/> (date of access: 27.03.2026).
70. Ataboyeva S., Bekchanov B. Transformation of learning management systems (LMS) in online education and their role in the modern world: analysis and comparison. *Topical Issues of Technical Sciences*, 2025. <https://doi.org/10.47390/ts-v3i7y2025N1> (date of access: 27.03.2026).

71. Gamage, S. H. P. W., Ayres, J. R., & Behrend, M. B. A systematic review on trends in using Moodle for teaching and learning. *International Journal of STEM Education*, 2022.Vol. 9. N. 9. URL: https://link.springer.com/article/10.1186/s40594-021-00323-x?utm_source (date of access: 27.03.2026).
72. Auliasari M. M., Pratama A. D. Efektivitas e-learning pada pendidikan tinggi dengan menggunakan LMS Moodle dan Google Classroom. *Jurnal Inovasi Akademik*, 2023.Vol 2. No 1. URL: https://journal.ugm.ac.id/v3/jinovak/article/view/11719?utm_source (date of access: 27.03.2026).
73. Натхнення для викладачів цікаві матеріали для учнів. URL: https://www.canva.com/uk_ua/osvita/ (дата звернення: 27.03.2026).
74. Slack is where the future works. URL: <https://slack.com/what-is-slack>
75. Official Website. URL: <https://discord.com> (date of access: 27.03.2026).
76. Lifewire. What Is Zoom? The Essential Guide to Understanding Features, History, and Usage. URL: <https://www.lifewire.com/what-is-zoom-11708920> (date of access: 27.03.2026).
77. Official Website. URL: <https://meet.google.com> (date of access: 27.03.2026).
78. Explore menti features. URL: <https://www.mentimeter.com/features> (date of access: 27.03.2026).
79. Bennedsen J., Böjtjter T., Tola D. Using GitHub Classroom in Teaching Programming. *Proceedings of the 18th International CDIO Conference*. Reykjavik University, Iceland, 2022. P. 690–702. URL: https://www.cdio.org/knowledge-library/documents/using-github-classroom-teaching-programming?utm_source (date of access: 24.05.2026)
80. CodEv: An Automated Grading Framework Leveraging Large Language Models for Consistent and Constructive Feedback. *arXiv*, 2025. URL: [CodEv Automated Grading Framework](#) (date of access: 27.03.2026).

81. Bounegru L. The platformisation of software development: Connective coding and platform vernaculars on GitHub. *New Media & Society*, 2024. Vol. 30, Issue 6. URL: <https://journals.sagepub.com/doi/abs/10.1177/13548565231205867> (date of access: 27.03.2026).
82. Cortés Ríos J. C., Kopec-Harding K., Eraslan S., Page C., Haines R., Jay C., Embury S. M. A Methodology for Using GitLab for Software Engineering Learning Analytics. *arXiv*, 2019. URL: <https://arxiv.org/abs/1903.06772> (date of access: 27.03.2026).
83. Dmytrenko D., Alieksieieva H., Khomenko S. BITBUCKET as a Solution for Organizing Teamwork on IT Projects // *ScienceRise: Pedagogical Education*, 2023. URL: https://journals.uran.ua/sr_edu/article/view/277945 (date of access: 24.05.2026).
84. Proskura S. L., Lytvynova S. H. The approaches to Web-based education of computer science bachelors in higher education institutions. *CTE Workshop Proceedings*, 2020. Vol. 7. P. 609–625. DOI: <https://doi.org/10.55056/cte.416> (дата звернення: 27.03.2026).
85. Закон України «Про освіту» : Закон України від 05.09.2017 № 2145-VIII. URL: <http://zakon5.rada.gov.ua/laws/show/2145-19> (дата звернення: 27.03.2026).
86. Нові освітні стандарти в Україні: що змінить реформа. URL: https://24tv.ua/zakon_pro_osvitu_2017_priynyali_shho_zminit_reforma_osviti_ukrayini_n861156 (дата звернення: 27.03.2026).
87. Биков В. Ю. Моделі організаційних систем відкритої освіти : монографія, Київ : *Аміка*, 2009. 684 с. URL: <https://lib.iitta.gov.ua/845/> (дата звернення: 27.03.2026).
88. Биков В. Ю. Проектний підхід і дистанційне навчання у професійній підготовці управлінських кадрів. *Кримські педагогічні читання : матеріали Міжнародної наукової конференції*, 12–17 вересня 2001 р. / за ред. С. О. Сисоєвої, О. Г. Романовського. Харків : НТУ «ХПІ», 2001. С. 30–50. URL: <https://lib.iitta.gov.ua/id/eprint/498/> (дата звернення: 27.03.2026).

89. Спирін О. М., Вакалюк Т. А. Веборієнтовані технології для навчання основ програмування майбутніх учителів інформатики. *Математика і інформатика у вищій школі: виклики сучасності : матеріали Всеукраїнської науково-практичної конференції*, 2017. С. 61–65.
URL: <https://eprints.zu.edu.ua/25123/> (дата звернення: 27.03.2026).

90. Литвинова С. Г. Хмаро орієнтоване навчальне середовище загальноосвітньої школи. *Хмарні технології в освіті : матеріали 5-го воркшопу CTE 2017*. Кривий Ріг, 2017. С. 7–12. (*CEUR Workshop Proceedings*, Vol. 2168).
URL: <http://ceur-ws.org/Vol-2168/paper2.pdf> (дата звернення: 27.03.2026).

91. Bykov Valeriy, Shyshkina Mariya. *The conceptual basis of the university cloud-based learning and research environment formation and development in view of the open science priorities*. Information Technologies and Learning Tools. 2018. Vol. 68(6). P. 1–19. DOI: <https://doi.org/10.33407/itlt.v68i6.2609> (дата звернення: 27.03.2026).

92. Семеріков С. О., Стрюк А. М. Моделі комбінованого навчання. *Вісник Дніпропетровського університету імені Альфреда Нобеля. Серія «Педагогіка і психологія»*, 2012. № 2 (4). С. 47–55.
URL: <https://lib.iitta.gov.ua/id/eprint/106652/1/Моделі%20комбінованого%20навчання.pdf> (дата звернення: 27.03.2026).

93. Ткачук Г. В. Зарубіжний досвід реалізації змішаного навчання. *Фізико-математична освіта*, 2018. Вип. 1 (15). С. 98–102. DOI: <https://doi.org/10.31110/2413-1571-2018-015-1-016>

94. Кухаренко В. М. Системний підхід до змішаного навчання. *Інформаційні технології в освіті*, 2015. № 24. С. 53–67.
DOI: <https://doi.org/10.14308/ite000550>.
URL: <https://ite.kspu.edu/index.php/ite/article/download/147/156/272> (дата звернення: 27.03.2026).

95. Кухаренко В. М., Березенська С. М., Бугайчук К. Л., Олійник Т. О., Рибалко О. В. Теорія і практика змішаного навчання : монографія. Харків : НТУ

«ХІІІ», 2016. 284 с. URL: <http://repository.kpi.kharkov.ua/handle/KhPI-Press/23536> (дата звернення: 27.03.2026).

96. Триус Ю. В., Герасименко І. В. Комбіноване навчання як інноваційна освітня технологія у вищій школі. *Теорія і методика електронного навчання*. Кривий Ріг, 2012. Вип. 3. С. 299–308.

97. Лісецький К. А. Модель змішаного навчання в системі вищої освіти. *Міжкультурна комунікація у європейському мовному просторі : матеріали міжнар. наук.-практ. інтернет-конф.*, 2015. URL: <https://ela.kpi.ua/handle/123456789/17816> (дата звернення: 27.03.2026).

98. Blended learning. URL: <https://www.christenseninstitute.org/resources/topic/education/blended-learning/> (дата звернення: 27.03.2026).

99. What is Blended Learning? URL: <https://www.christenseninstitute.org/video/what-is-blended-learning-2/>

100. Staker H., Horn M. B. Classifying K–12 Blended Learning. Christensen Institute, 2012. URL: <https://www.christenseninstitute.org/publication/classifying-k-12-blended-learning-2/> (date of access: 27.03.2026)

101. Основи програмування CS50 2019. URL: https://courses.prometheus.org.ua/courses/course-v1:Prometheus+CS50+2019_T1/about (дата звернення: 27.03.2026).

102. Литвинова С. Г. Технологія «перевернутого» навчання у хмаро орієнтованому освітньому середовищі як складник розвитку медіаосвіти в середній школі. *Медіасфера і медіаосвіта: специфіка взаємодії в сучасному соціокультурному просторі*. Могильов, 2015.

103. Чередніченко Г. А., Шапран Л. Ю. Модель змішаного навчання і її використання у викладанні іноземних мов. *Moodle Moot Ukraine 2015*. Київ : КНУБА, 2015. С. 13. URL: <https://dspace.nuft.edu.ua/handle/123456789/20579> (дата звернення: 27.03.2026).

104. Proskura S. L., Lytvynova S. H., Kronda O. P. The use of WEB-oriented technologies in the process of WEB-programming teaching for technical universities

students. *Educational Dimension*, 2021. №5.
 URL: <https://lib.iitta.gov.ua/id/eprint/728607/1/Proskura-Lytvynova-Kronda-2021.pdf> DOI: <https://doi.org/10.31812/educdim.4724> (дата звернення: 27.03.2026).

105. Проскура С. Л., Литвинова С. Г. Формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*, 2019. № 2 (20). С. 137–147.
<https://cyberleninka.ru/article/n/formuvannya-profesiynoyi-kompetentnosti-maybutnih-bakalavriv-kompyuternih-nauk/viewer>

106. Литвинова С. Г. Віртуальний клас для організації індивідуального навчання учнів. *Інформаційні технології і засоби навчання*. 2011. № 1 (21)

107. Проскура С. Л. Застосування інтелект-карт для підвищення якості та ефективності навчання студентів програмувальних курсів закладів вищої освіти. *Актуальні питання природничо-математичної освіти*, 2017. № 1 (9). С. 129–137. URL: http://fizmatsspu.sumy.ua/Konferencii/sbor/appmo/appmo_v7-8_2016.pdf#page=220 (дата звернення: 27.03.2026).

108. Проскура С. Л., Литвинова С. Г. Моніторинг використання WEB-орієнтованих технологій бакалаврами комп'ютерних наук в процесі вивчення програмування. *Інформаційно-цифровий освітній простір України: трансформаційні процеси і перспективи розвитку : матеріали методологічного семінару НАПН України (4 квітня 2019 р.)*. Київ : НАПН України, 2019. С. 211– 219. URL: <https://lib.iitta.gov.ua/id/eprint/717123/> (дата звернення: 27.03.26).
https://lib.iitta.gov.ua/id/eprint/717123/1/NEW_Metod%20Semenar%207.04.19%20Proskura%20Lytvynova.pdf

109. Proskura S. L., Lytvynova S. H., Kronda O. P. Students' academic achievement assessment in higher education institutions. *ICTERI 2020: Proceedings of the 16th International Conference on ICT in Education, Research and Industrial Applications. CEUR Workshop Proceedings*, 2020. Vol. 2732. P. 734–746
 URL: <http://ceur-ws.org/Vol-2732/20200734.pdf> (дата звернення: 27.03.2026).

110. Огнівчук Л. М. Оцінювання навчальних досягнень студентів вищих навчальних закладів на основі компетентнісного підходу. *Освітологічний дискурс*, 2014. № 3 (7). С. 154–166. URL: https://od.kubg.edu.ua/index.php/journal/article/view/128?utm_source (дата звернення: 27.03.2026).
111. Калініна О. Г. Система управління якістю освіти у вищих навчальних закладах США : дис. ... канд. пед. наук : 13.00.06. Київ, 2015. 259 с. URL: https://luguniv.edu.ua/wp-content/uploads/2015/12/d29.053.03_17.12.15_kalinina_dis.pdf (дата звернення: 27.03.2026).
112. Measuring and recording student achievement : report of the Scoping Group chaired by Professor Robert Burgess. London : Universities UK, 2004. 56 p. URL: <https://paperzz.com/doc/8900140/universities-uk-report---measuring-and-recording-student-> (дата звернення: 27.03.2026).
113. Турчин А. І. Система контролю й оцінювання навчальних досягнень студентів у ВНЗ Німеччини. *Наукові записки ТНПУ*. 2010. № 1. URL: http://nbuv.gov.ua/UJRN/NZTNPU_ped_2010_1_35 (дата звернення: 27.03.2026).
114. Paiva J. C., Leal J. P., Figueira Á. Automated Assessment in Computer Science Education: A State-of-the-Art Review. *ACM Transactions on Computing Education*, 2022. Vol. 22, Issue 3. Art. 34. P. 1–40. DOI: <https://doi.org/10.1145/3513140> (дата звернення: 27.03.2026).
115. Wanner L. Integrating Continuous Assessment into Undergraduate Computer Architecture using Automated Grading. *International Journal of Computer Architecture Education*, 2024. Vol. 13, No. 1. P. 24–32. DOI: <https://doi.org/10.5753/ijcae.2024.5341> (дата звернення: 27.03.2026).
116. Bergin S., Mooney A. Using an innovative assessment approach on a real-world group based software project, 2016. DOI: <https://doi.org/10.48550/arXiv.1609.07899> (дата звернення: 27.03.2026).
117. Alamr S., Urrutia M., Carr L. Success factors of e-assessment in computer science for higher education: perspectives from instructors. *Frontiers in*

Education. 2025. Vol. 10. Article 1595171. DOI: <https://doi.org/10.3389/feduc.2025.1595171> (дата звернення: 27.03.2026).

118. Baartman L. K. J., Quinlan K. M. Assessment and feedback in higher education reimagined. *International Journal of Leadership in Education*, 2023. Vol. 27, No. 1. P. 57–67. DOI: <https://doi.org/10.1080/13603108.2023.2283118>

119. Nguyen Q., Rienties B., Toetenel L., Ferguson R., Whitelock D. Examining the designs of computer-based assessment. *Computers in Human Behavior*, 2017. Vol. 76. P. 703–714. DOI: <https://doi.org/10.1016/j.chb.2017.03.028> (дата звернення: 27.03.2026).

120. Barua L., Lockee B. Flexible Assessment in Higher Education. *TechTrends*, 2025. Vol. 69. P. 301–309. DOI: <https://doi.org/10.1007/s11528-025-01039-3>

121. Gonsalves C., Lin Z. Transparency of assessment practices. *Assessment & Evaluation in Higher Education*, 2024. Vol. 49, No. 9. P. 1454–1470. DOI: <https://doi.org/10.1080/03075079.2024.2381124> (дата звернення: 27.03.2026).

122. Alfaleh M. Sustainable AI-Driven Assessment in Higher Education. *Sustainability*, 2026. Vol. 18, No. 2. Article 785. DOI: <https://doi.org/10.3390/su18020785> (дата звернення: 27.03.2026).

123. Khan S., Alam M. Outcome-Based Quality Assessment Framework for Higher Education, 2017. DOI: <https://doi.org/10.48550/arXiv.1703.06063>

124. Бочарнікова В. М. Стимулююча функція контролю знань, умінь і навичок студентів вищої школи : автореф. дис. ... канд. пед. наук : 13.00.01. Київ, 1999. 23 с. (дата звернення: 27.03.2026).

125. Литвинова С. Г. Модель використання системи комп'ютерного моделювання для формування компетентностей учнів з природничо-математичних. *Фізико-математична освіта*, 2019. Т. 1(19). С.108–115 URL: <https://repository.sspu.edu.ua/items/4d7f10c8-e2be-4ea8-8aad-c255b14fb32e> (дата звернення: 27.03.2026).

126. Benjamin S. Bloom (Ed.). Taxonomy of educational objectives: The classification of educational goals. *Handbook I: Cognitive domain*. David McKay. 1956.

127. Пометун О., Гупан Н. Таксономія Б. Блума і розвиток критичного мислення школярів. *Український педагогічний журнал*, 2019. № 3. С. 52–58. DOI: <https://doi.org/10.32405/2411-1317-2019-3-50-58> (дата звернення: 27.03.2026).

У розділі були використані праці автора: [1], [84], [104], [105]. [107], [108], [109].

РОЗДІЛ 3. МЕТОДИКА ВИКОРИСТАННЯ ВЕБОРІЄНТОВАНИХ ТЕХНОЛОГІЙ У НАВЧАННІ ПРОГРАМУВАННЯ МАЙБУТНІХ БАКАЛАВРІВ КОМП'ЮТЕРНИХ НАУК

3.1 Цільовий та змістовий складники методики використання веборієнтованих технологій

Сучасний етап цифрової трансформації вищої освіти зумовлює необхідність переосмислення традиційних підходів до організації підготовки майбутніх бакалаврів комп'ютерних наук. Інтеграція України до європейського освітнього простору та реалізація принципів Болонського процесу передбачають розвиток безперервної освіти, академічної мобільності й упровадження інноваційних цифрових технологій в освітній процес закладів вищої освіти.

У контексті розвитку інтелектуального вебу освітнє середовище набуває інтерактивності, адаптивності та можливостей інтелектуальної підтримки навчальної діяльності. Це зумовлює необхідність розроблення нових методичних підходів до навчання програмування, які враховують потенціал веборієнтованих технологій, хмарних обчислень і систем штучного інтелекту.

Одним із теоретичних підґрунтів дослідження є поняття методики навчання. За визначенням професора С. Гончаренка, методика навчання розглядається як галузь педагогічної науки, що досліджує закономірності, принципи, методи, форми та засоби організації освітнього процесу з метою забезпечення ефективного засвоєння змісту освіти [1].

Учений зазначає: «Головне завдання методики як науки – вибрати з відповідних галузей знань доступні для студентів положення, які слід увести до навчального предмета чи курсу, обґрунтувати необхідність їх вивчення, визначити засоби і методи навчання, що забезпечать їх засвоєння. Завдання методики в узагальненому вигляді містить три основні компоненти: для чого навчати? чого навчати? як навчати?» [1].

Методика навчання програмування має охоплювати такі основні компоненти освітнього процесу:

- *Мету навчання:* формування у студентів компетентностей, зокрема компетентності з програмування у процесі вивчення відповідної дисципліни, а також закріплення теоретичних знань і розвиток практичних умінь та навичок.
- *Зміст навчання:* систему знань, умінь і навичок, представлену у вигляді розділів, тем, теоретичних матеріалів і комп'ютерних практикумів, запропонованих студентам для вивчення та опрацювання.
- *Методи та прийоми викладання:* способи подання навчального матеріалу, організації лекційних занять і комп'ютерних практикумів, використання відеозаписів лекцій, презентацій та інших дидактичних матеріалів.
- *Засоби навчання:* навчально-методичні матеріали, електронні підручники, апаратне та програмне забезпечення, інтернет-ресурси, цифрові інструменти та інші засоби, що забезпечують ефективну організацію навчальної діяльності.
- *Форми організації навчання:* лекції, семінари, комп'ютерні практикуми, індивідуальні та командні проєкти, самостійну роботу студентів, що сприяють формуванню відповідних компетентностей.
- *Оцінювання результатів навчання:* визначення рівня сформованості компетентностей, зокрема компетентності з програмування, шляхом проведення тестування, контрольних-модульних робіт, а також захисту комп'ютерних практикумів, індивідуальних та командних проєктів.

Запропонована методика ґрунтується на сучасних педагогічних підходах до цифрового навчання, зокрема на забезпеченні безперервності освітнього процесу в різних навчальних середовищах, взаємодії студента із системами ІІІ та організації навчальної діяльності в хмарному цифровому середовищі.

На відміну від традиційного інструментального підходу, за якого вебтехнології розглядаються лише як сукупність програмних засобів, у запропонованій методиці вони формують цілісну дидактичну ХО систему

навчання, що інтегрує середовища програмування, системи контролю версій, сервіси розгортання програмних продуктів та інструменти інтелектуальної підтримки навчання.

Авторська методика навчання програмування з використанням веборієнтованих технологій передбачає виокремлення основних дидактичних складників, які традиційно розглядаються через систему взаємопов'язаних запитань: «для чого навчати?», «чого навчати?» та «як навчати?».

– *Метою навчання* є формування компетентності з програмування у майбутніх бакалаврів комп'ютерних наук, що передбачає здатність проєктувати, розробляти та підтримувати вебзастосунки з використанням сучасних технологій Frontend-розробки. Водночас формування такої компетентності пов'язане з розвитком цифрової компетентності на рівнях 7–8 Рамки цифрових компетентностей DigComp 3.0, які характеризуються здатністю створювати складні цифрові продукти та розв'язувати нестандартні професійні завдання.

– *Зміст навчання (чого навчати)* охоплює теоретичні та практичні компоненти підготовки студентів у галузі програмування, зокрема вивчення основ HTML5 і семантичної розмітки вебсторінок, стилізації інтерфейсів засобами CSS3, програмування клієнтської логіки мовою JavaScript, створення інтерактивних користувацьких інтерфейсів. Практична складова реалізується через виконання системи комп'ютерних практикумів і проєктних завдань, спрямованих на розроблення вебінтерфейсів різного рівня складності.

Методи навчання (як навчати) передбачають поєднання традиційних та інноваційних методів навчання, серед яких важливе місце посідає Git-центричний підхід, що забезпечує організацію спільної роботи над програмними проєктами, контроль версій коду та доступ до навчального середовища незалежно від технічних характеристик пристрою, який використовується студентом.

До таких засобів належать: хмарні середовища програмування (GitHub Codespaces, Replit тощо), системи контролю версій (Git), сервіси розгортання вебзастосунків (Vercel, Netlify) (див. підрозділ 2.1).

Форми організації навчання та оцінювання: передбачає поєднання індивідуальної та командної роботи студентів над програмними проєктами. Оцінювання результатів навчання здійснюється з використанням таксономії Блума, що дає змогу об'єктивно визначити рівень сформованості компетентності з програмування – від відтворення знань до створення власних програмних продуктів.

На рисунку 3.1 представлено структурно-логічну схему цієї методики навчання, яка містить основні складники:

- Блок 1. *Цільовий складник* спрямований на формування сучасних компетентностей, що забезпечують здатність студентів застосовувати вебтехнології у реальних проєктах. Це передбачає формування цифрової професійної ідентичності та досягнення 7–8 рівнів за DigComp 3.0, а також інтегральної, загальної і фахових компетентностей, визначених стандартом спеціальності F3 «Комп'ютерні науки».
- Блок 2. *Змістовий складник* базується на вивченні HTML, CSS, JavaScript (ES6+), DOM-моделі та бібліотеки React. Зміст охоплює три логічні розділи: від базових принципів клієнтських розробок до створення складних адаптивних інтерфейсів.
- Блок 3. *Процесуальний складник* передбачає використання авторської пропорції змішаного навчання (30% традиційне, 50% дистанційне, 20% проєктне) [2]. Серед методів навчання виділяються програмування в реальному часі (live coding) та розв'язання проблемно-пошукових задач.
- Блок 4. *Технологічний складник* базується на використанні інтегрованої хмарної (Cloud-native) екосистеми, що включає LMS Moodle для організації навчання, хмарні середовища розробки (GitHub Codespaces, Replit) як основне робоче місце студента, систему контролю версій GitHub для реалізації Git-центричного підходу та сервіси автоматичного розгортання

(Vercel, Netlify) для публікації працездатних релізів застосунків (див. підрозділ 2.1).

– Блок 5. Складник інтелектуальної підтримки включає методику використання ШІ у процесі розробки програмного коду, використання адаптивних ШІ-асистентів та автоматизованого перегляду коду (Code Review) для підвищення якості коду.

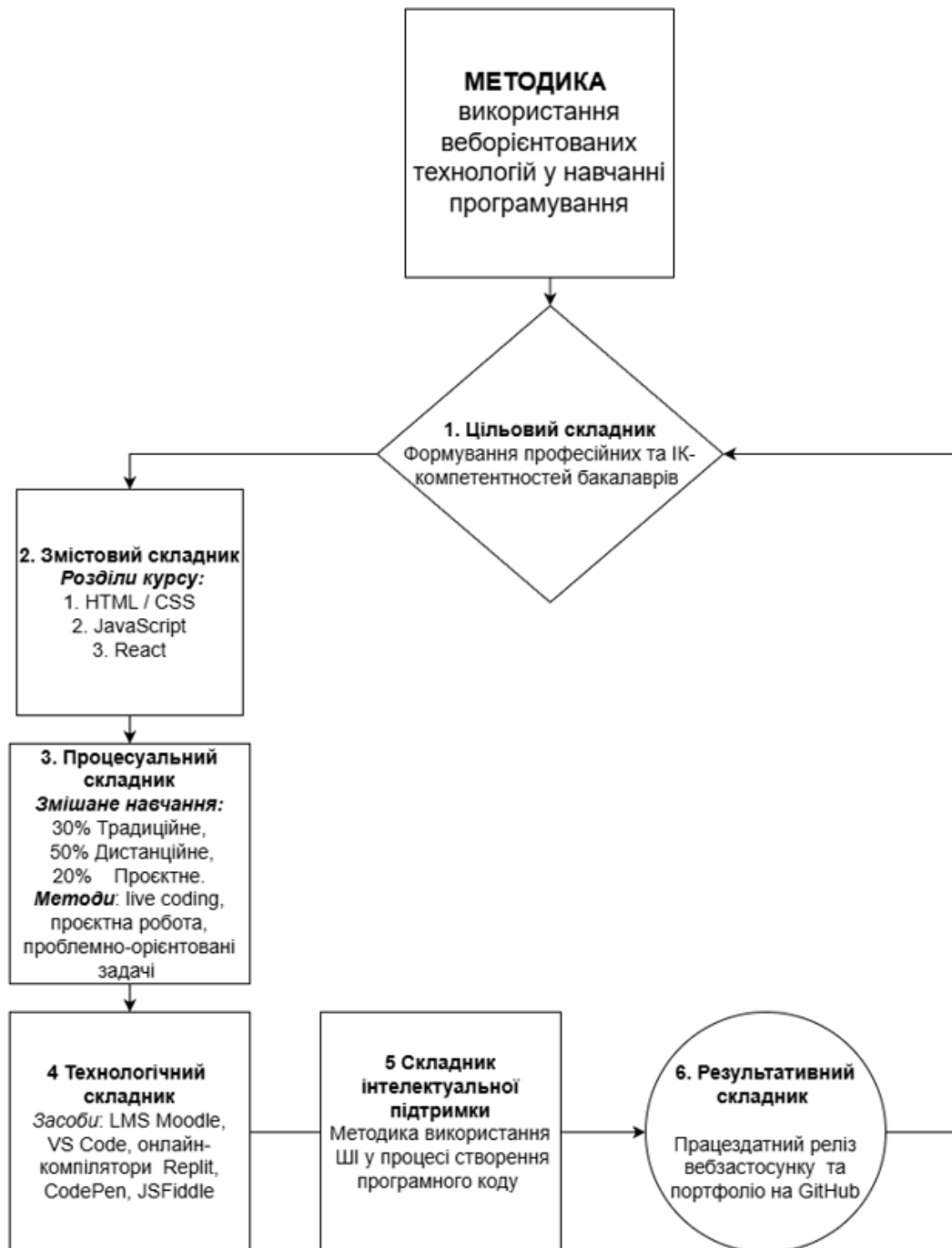


Рис. 3.1 Структурно-логічна схема методики навчання програмування бакалаврів комп'ютерних наук

– Блок 6. *Результативний складник* відображає кінцеві результати навчання, що полягають у створенні працездатного релізу вебзастосунку під час кожного комп'ютерного практикуму, а також у формуванні публічного GitHub-портфолію, яке містить реалізовані проєкти застосунків та історію змін («цифровий слід») студента.

Побудова такої методики подібна до розробки сучасного програмного застосунка, де цільовий блок – це технічне завдання, змістовий – архітектура та створення програмного коду, а результативний – фінальний запуск продукту, який доводить його працездатність.

Методика навчання програмування з використанням веборієнтованих технологій бакалаврів комп'ютерних наук охоплює:

1. Мету навчання:

- Формування цілісних знань про архітектуру та принципи роботи вебзастосунків.
- Оволодіння сучасними інструментами, бібліотеками та фреймворками Frontend-веброзробки.
- Розвиток навичок створення Frontend-частини застосунків.
- Забезпечення готовності до роботи в команді та участі у реальних проєктах.

2. Зміст навчання.

Теоретичний компонент:

- Основи вебтехнологій: HTTP/HTTPS, клієнт–серверна модель, REST, WebSocket.
- HTML, CSS, JavaScript (ES6+), DOM.
- Основи UX/UI дизайну.
- User Experience(UX)/ User Interface(UI).

Практичний компонент:

- Розробка Frontend-частини (HTML, CSS, Native JavaScript, React).
- Використання REST API.

- Система контролю версій (Git, GitHub).

3. *Методи та прийоми навчання:*

- Пояснювально-ілюстративний метод (демонстрація прикладів програмного коду).
- Комп'ютерні практикуми з покроковим створенням вебзастосунку.
- Проблемно-пошуковий метод (вирішення завдань з помилками у коді, рефакторинг коду).

4. *Засоби навчання:*

- Хмарні середовища розробки (GitHub Codespaces, Replit) як віртуальні робочі місця студента.
- Сервіси автоматичного розгортання та публікації вебресурсу в мережі Інтернет (Vercel, Netlify).
- Інтелектуальні засоби підтримки (GitHub Copilot, ChatGPT) як персональний інтелектуальний помічник студента для дебагінгу та пояснення алгоритмів.

Форми організації:

- Лекції з демонстраціями.
- Комп'ютерні практикуми (практичне кодування).
- Командні, курсові та дипломні проєкти.

5. *Оцінювання результатів:*

- Виконання комп'ютерних практикумів (оцінка якості коду, адаптивності, оптимізації).
- Тестування теоретичних знань (тести, усне опитування).
- Оцінка проєктів за критеріями: функціональність, безпека, дизайн, документація.

Активність у командній роботі та дотримання дедлайнів

Методика навчання вебтехнологій має бути максимально практико-орієнтованою, передбачати активне застосування сучасних інструментів, технологій і реальних кейсів, для забезпечення не лише засвоєння теоретичних

знань, а й формування практичних умінь, навичок і фахових компетентностей, зокрема компетентності з програмування у сфері веброзробки.

У процесі навчання студенти повинні набувати досвіду розроблення повноцінних веборієнтованих застосунків, опановувати актуальні фреймворки та бібліотеки, системи контролю версій, засоби роботи з базами даних і технології командної розробки програмного забезпечення. Такий підхід сприятиме підготовці конкурентоспроможних фахівців, здатних виконувати практичні завдання реальних ІТ-проектів і ефективно працювати в сучасних ІТ-компаніях.

Деталізуємо етапи створення програмного продукту (рис. 3.2).

Етап 1: *Проектування (UI/UX)*. Цей етап відповідає першому розділу дисципліни, у межах якого студенти опановують основи розроблення клієнтської розробки. Робота над проектом розпочинається з аналізу предметної галузі та опису бізнес-логіки майбутнього вебзастосунку. На основі проведеного аналізу студент створює UML-діаграму прецедентів, що дає змогу визначити взаємодію користувачів із системою та окреслити основний функціонал застосунку.

Надалі розробляється візуальний інтерфейс у середовищі Figma, де створюються макети сторінок, проектується структура інтерфейсу, навігація, розташування елементів і загальний візуальний стиль застосунку. Особлива увага приділяється дотриманню принципів UI/UX-дизайну, забезпеченню зручності використання, доступності та адаптивності інтерфейсу для різних типів пристроїв.

Практична реалізація цього етапу завершується верстанням створеного макету із застосуванням технологій HTML5 та CSS3. Для побудови гнучкої структури сторінок використовуються сучасні засоби CSS3, зокрема технологія Flexbox та медіа-запити, що забезпечують адаптивність вебзастосунку й коректне відображення його інтерфейсу на персональних комп'ютерах, планшетах і мобільних пристроях. У результаті студенти набувають

практичного досвіду створення сучасних адаптивних користувацьких інтерфейсів відповідно до актуальних вимог веброзробки.

Етап 2: *Надання інтерфейсу динамічності*. Зміст другого розділу дисципліни зосереджений на вивченні програмних засобів мови JavaScript для створення динамічних інтерфейсів.

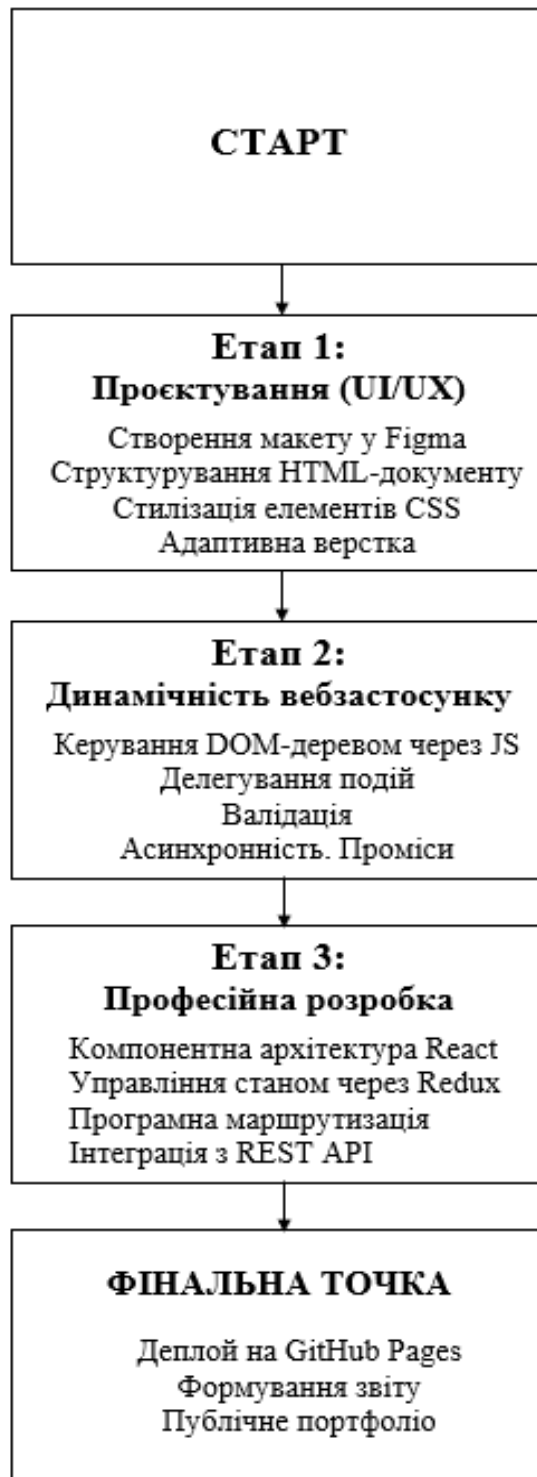


Рис. 3.2. Етапи створення програмного продукту

Студенти опановують методи маніпулювання DOM-деревом, роботу з масивами об'єктів, а також механізми деструктуризації. Важливим складником цього етапу є перехід до асинхронного програмування, що передбачає використання промісів, а також виконання HTTP-запитів до серверів за допомогою REST API.

Етап 3: *Професійна розробка*. Третій розділ дисципліни присвячений вивченню сучасних підходів до професійної Frontend-розробки із використанням бібліотеки React. На цьому етапі студенти переходять від створення статичних вебсторінок до розроблення повноцінних інтерактивних вебзастосунків із компонентною архітектурою.

В процесі навчання студенти опановують принципи побудови функціональних компонентів, роботу JSX, механізми передавання даних між компонентами, а також використання хуків React для керування станом і життєвим циклом компонентів.

Особлива увага приділяється використанню хуків `useState`, `useEffect`, `useContext` та інших інструментів, що забезпечують ефективну організацію логіки застосунку. Для реалізації централізованого керування станом застосунку студенти вивчають технологію Redux Toolking, яка дає змогу організувати зберігання, оновлення та передачу даних між компонентами у великих програмних проєктах. У межах комп'ютерних практикумів студенти навчаються створювати глобальні сховища даних, працювати з асинхронними запитами та інтегрувати вебзастосунки із зовнішніми API.

Застосування сучасних підходів Frontend-розробки дає змогу створювати складні інтерактивні інтерфейси, які працюють із даними в режимі реального часу, підтримують масштабованість програмного проєкту та відповідають сучасним вимогам професійної веброзробки. У результаті студенти набувають практичного досвіду створення SPA-застосунків (Single Page Application) наближеного до умов роботи над реальними програмними проєктами.

Фінальна точка. Результатом навчання є виконання шести комп'ютерних практикумів, кожен з яких передбачає поетапне розширення функціональності

єдиного цілісного вебпроекту. Такий підхід забезпечує послідовне формування практичних умінь і навичок, а також дає змогу студентам усвідомити взаємозв'язок між окремими етапами розроблення вебзастосунку.

У процесі виконання практикумів студенти проходять усі ключові етапи створення сучасного вебресурсу: від проектування інтерфейсу та реалізації клієнтської частини до інтеграції із серверними сервісами, керування станом застосунку та організації командної роботи з використанням систем контролю версій. Логічним завершенням технологічного циклу є автоматизоване розгортання вебресурсу на платформах Vercel або Netlify безпосередньо з репозиторію GitHub (див. підрозділ 2.1). Студенти налаштовують процес безперервного розгортання (Continuous Deployment), що забезпечує автоматичне оновлення вебзастосунку після внесення змін до репозиторію.

Такий підхід сприяє опануванню повного життєвого циклу розроблення ПЗ (Software Development Life Cycle, SDLC), який охоплює етапи проектування, реалізації, тестування, керування версіями та розгортання програмного продукту. У результаті студенти отримують можливість презентувати результати власної діяльності у вигляді функціонального мінімального життєвого продукту (Minimum Viable Product, MVP), доступного за публічними гіперпосиланнями, що максимально наближує освітній процес до реальних умов професійної діяльності в IT-сфері.

Обґрунтуємо підходи до використання веборієнтованих технологій у процесі створення програмного продукту (рис. 3.3).

Хмара (Infrastructure). Використовується для розгортання проєктів (Netlify, AWS) та версіонування коду (GitHub). GitHub також виступає платформою для формування публічного портфоліо студента.

Навчальне середовище (LMS). Moodle та Google Classroom є пріоритетними сервісами для розміщення навчально-методичних матеріалів (45% та 44% відповідно за результатами опитування) [3]. Вони забезпечують структурування курсу за темами та контроль виконання завдань.

Розробка (Dev Tools). Включає як повноцінні ІСР (VS Code, що є найпопулярнішим серед студентів – 61,5%) [4], так і онлайн-компілятори (Replit, CodePen, JSFiddle), які дозволяють програмувати в браузері без встановлення локального ПЗ.

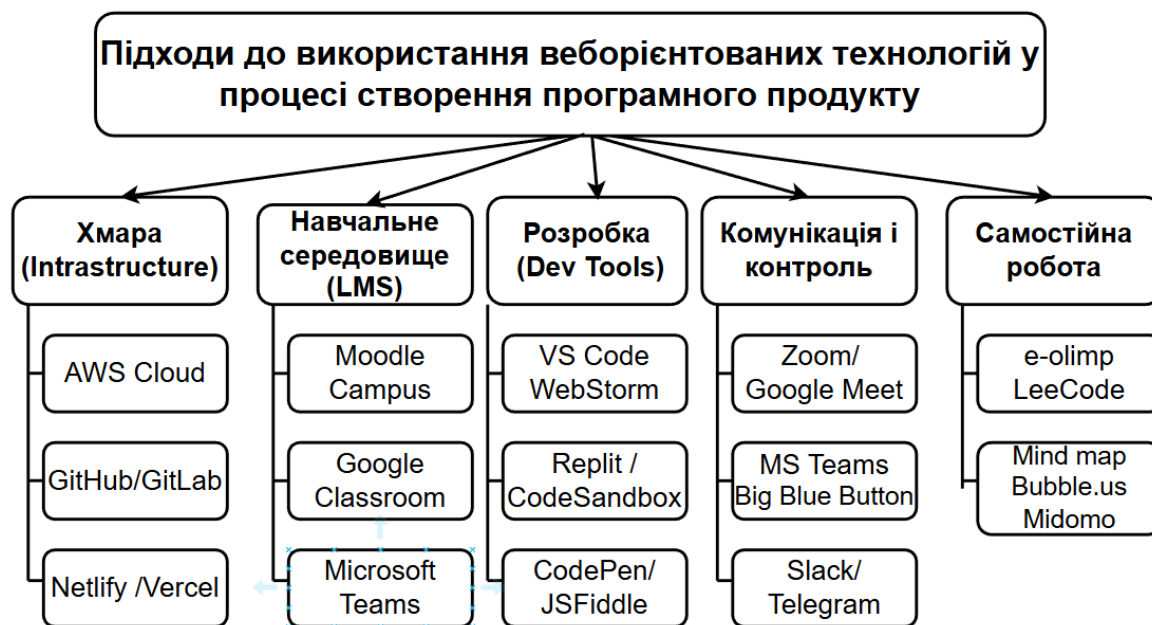


Рис. 3.3 Підходи до використання веборієнтованих технологій у процесі створення програмного продукту

Комунікація та контроль. Базується на використанні віртуальних класів для проведення лекцій та вебінарів у режимі реального часу. За результатами дослідження, найбільш поширеним сервісом для організації відеоконференцій є Zoom, який використовують 75% опитаних викладачів [3]. Для оперативної комунікації між учасниками освітнього процесу також широко застосовується Telegram, яким користуються 67% студентів [3].

Самостійна робота. Включає автоматизовані системи перевірки знань, серед яких найбільшого поширення набули платформи для автоматизованої перевірки програмного коду, зокрема E-olymp, яку обрали 75% респондентів [5]. Також важливу роль відіграють інтелект-карти (mind maps) з найбільш популярними сервісами Bubbl.us та Mindomo та ін. для візуалізації складних концепцій та структурування коду [5].

Практичне застосування авторської методики представлено в Додатку Д.

Значний обсяг навчальної інформації, який опрацьовують студенти, потребує її систематизації, структурування та застосування ефективних засобів опрацювання та зберігання. Одним із таких методів є *інтелект-карти*.

У науковій літературі та в мережі Інтернет для позначення цього методу використовуються різні терміни, зокрема: Mind Map, Mind Mapping, ментальні карти, карти знань, карти пам'яті, концептуальні карти, когнітивні карти, карти аргументів. Для забезпечення термінологічної узгодженості в цьому дисертаційному дослідженні використовується термін «інтелект-карти».

З огляду на те, що останні дослідження з проблем застосування інтелект-карт у викладанні технічних дисциплін не містять достатньої кількості фундаментальних праць, вивчення цього питання здійснюється на основі наукових статей українських та зарубіжних авторів.

У 1960-х роках XX століття професор Корнелльського університету Дж. Новак (Joseph D. Novak) уперше запропонував концепт-карти як інструмент візуалізації знань і генерування нових ідей і понять [7], [8].

У своїх наукових працях психолог Т. Б'юзен описує технологію створення та застосування інтелект-карт у різних сферах діяльності, зокрема в науці та освіті, підготовці презентацій, бізнесі, плануванні та під час проведення мозкових штурмів. Практична ефективність інтелект карт сприяла їх широкому поширенню в різних сферах інтелектуальної діяльності [7], [9].

Так, український учений М. Бирка [10] в своїй праці наводить порівняльну характеристику інтелект-карт та опорних схем В. Шаталова [10]. Опорні схеми являють собою сукупність схематичних зображень, формул, ключових понять і фраз, за допомогою яких навчальна інформація подається у стислій структурованій формі. Представлення навчального матеріалу може здійснюватися за допомогою моделей, які мають лінійну, ієрархічну, мережну або матричну структуру [7].

Професор Бирка обґрунтовує актуальність використання інтелект-карт у професійній діяльності вчителів природничо-математичних дисциплін, а також розкриває основні принципи та етапи їх створення [10].

Н. Терещенко у своїй науковій праці розглядає інтелект-карти як сучасну інноваційну технологію навчання, що використовується в освітньому процесі [11]. Питання застосування інтелект-карт в освітньому процесі також активно досліджують Ю. Тулашвілі [12] та Н. Олексів [12], які висвітлюють особливості підготовки інженерів-педагогів комп'ютерного профілю за допомогою цього методу, як засобу когнітивної візуалізації [7].

Інтелект-карти – це графічне відображення процесів багатовимірного мислення. Вони є потужним візуальним методом, що розкриває потенціал людського мислення. Інтелект-карти можуть застосовуватися в різних сферах життя, де виникає потреба у розвитку та вдосконаленні інтелектуальних здібностей особистості, а також у розв'язанні різноманітних завдань і проблем [7], [11].

Метод «ментальна карта» призначений для візуального представлення слів, ідей і завдань, пов'язаних між собою та розташованих радіально навколо ключового поняття або ідеї. Він використовується для генерування, візуалізації, структурування та класифікації ідей, а також як засіб підтримки під час вивчення й організації навчального матеріалу, розв'язання задач і прийняття рішень.

Метод реалізується у вигляді діаграми, яка відображає семантичні та інші зв'язки між окремими елементами інформації. Елементи діаграми розташовуються інтуїтивно відповідно значущості понять і організуються у групи, підрозділи та тематичні зони. Ментальні карти широко використовуються викладачами для пояснювання навчального матеріалу інноваційним способом, а також під час проведення лекційних занять. Ментальні карти створюються швидко та сприяють кращому запам'ятовуванню інформації завдяки наочності. [13].

Зазначимо, що у своїх дослідженнях британський психолог і автор концепції інтелект-карт Т. Б'юзен підкреслював, що людський мозок не завжди ефективно засвоює великі обсяги інформації виключно у послідовній та логічній формі. Нова інформація краще засвоюється тоді, коли вона

пов'язується з уже відомими образами, поняттями чи явищами, тобто через механізми асоціативного та образного мислення. Такі ментальні образи зберігаються в пам'яті та слугують основою для формування нових асоціацій, що забезпечує створення розгалуженої мережі зв'язків між поняттями [9].

Отже, в основі методу інтелект-карти лежать асоціативне мислення, візуалізація та цілісне сприйняття інформації (гештальт-підхід). Саме таким чином людський мозок сприймає навколишній світ і прагне організувати отриману інформацію у вигляді ієрархічних структур. Тому одним із найефективніших способів структурування та запам'ятовування інформації є подання навчального матеріалу у вигляді деревоподібної структури. Такі структури широко застосовуються в різних сферах діяльності, де виникає потреба структурно і наочно представити значний обсяг даних [7].

Однією із переваг використання інтелект карт є поєднання різних способів опрацювання інформації, що активізує роботу обох півкуль головного мозку. Ліва півкуля відповідає за логічне мислення (послідовність, робота з текстом, числами), тоді як права півкуля – за образне та просторове сприйняття (цілісність сприйняття, сприйняття кольору, ритму, просторова орієнтація). Таким чином, застосування інтелект-карт позитивно впливає на всебічний розвиток студентів та підвищує ефективність навчальної діяльності

При побудові інтелект-карт доцільно дотримуватись таких основних правил [7]:

- головний об'єкт в інтелект-карті завжди розміщується в центрі та має бути яскравим, доповненим малюнком або графічним образом;
- від головного об'єкту відходять гілки-асоціації першого рівня, на яких розміщуються ключові питання чи поняття, пов'язаною з центральною темою;
- на гілках-асоціаціях розміщуються ключові слова, словосполучення, короткі фрази або графічні образи. Від них відходять гілки другого рівня, що відображають похідні ідеї, від яких, своєю чергою, розгалужуються гілки третього рівня;

– гілки асоціацій утворюють ієрархічну структуру, яку в математиці називають графом.

Інтелект-карти є ефективною альтернативою традиційним способам конспектування, запам'ятовування значних обсягів навчальної інформації з метою її подальшого використання.

Запам'ятовування навчального матеріалу є одним із найважливіших чинників, що впливають на якість навчання студентів. Нині обсяги навчальних матеріалів, які пропонуються викладачами для опрацювання, а також вимоги до рівня їх засвоєння, суттєво зросли. Для студента ефективно запам'ятовування навчального матеріалу має ґрунтуватись на загальних розумових діях і операціях, до яких належать [7]:

- структурування (мисленнєва діяльність, спрямована на встановлення зв'язків між поняттями, у процесі якої формується структура знань);
- систематизація (мисленнєва діяльність, спрямована на встановлення віддалених зв'язків між поняттями та судженнями, у результаті чого вони організуються в єдину систему);
- конкретизація (застосування набутих знань на практиці); варіювання (зміна несуттєвих ознак понять, їх властивостей та фактів тощо за незмінності суттєвих ознак); доведення (логічне обґрунтування) [7].;
- формування висновків (поступове спрощення теоретичних або практичних положень з метою отримання певного результату);
- пояснення (виокремлення та осмислення найважливіших зв'язків і закономірностей); класифікація (розподіл понять на взаємопов'язані групи або класи за суттєвими ознаками);
- аналіз (мисленнєва операція, що полягає у виокремленні ознак, властивостей і відношень понять, а також у виявленні їхніх спільних і відмінних характеристик; синтез (мисленнєва операція, що полягає в об'єднанні окремих елементів, ознак або частин у єдине ціле; є процесом протилежним аналізу;
- порівняння (мисленнєва операція, спрямована на виявлення спільних і відмінних ознак, властивості та характеристик понять, явищ або об'єктів);

- абстрагування (мисленнєва діяльність, що передбачає виокремлення суттєвих ознак понять шляхом відмежування від несуттєвих характеристик);
- узагальнення (мисленнєва операція, спрямована на виділення спільних істотних ознак і властивостей, характерних для кількох понять, явищ або об'єктів.

Усі перелічені процеси свідчать про значний потенціал людського мозку щодо обробки великих обсягів інформації. Проте для ефективного розвитку цих можливостей необхідно постійно їх тренувати та організовувати подання навчального матеріалу таким чином, щоб його структура максимально відповідала за значним інтелектуальними операціям. [7].

Саме тому використання інтелект карт дає змогу не лише візуалізувати систематизувати значні обсяги інформації, а й підвищувати мотивацію студентів до активнішої навчальної діяльності [7].

Щодня студенти стикаються з великими обсягами інформації, отриманої під час лекцій, самостійної роботи, участі в конференціях і семінарах, підготовки доповідей, а також із різноманітних інтернет-ресурсів. Основними формами подання такої інформації є тексти, таблиці, діаграми та списки. Водночас традиційні способи її фіксації та опрацювання мають низку недоліків, серед яких: складність запам'ятовування матеріалу, поданого у вигляді традиційного конспекту; труднощі з відтворенням інформації під час підготовки до іспиту; ускладнене виявлення ключових ідей та взаємозв'язків між поняттям; значні витрати часу на пошук необхідної інформації в конспектах і навчальних матеріалах; обмежені можливості для творчого підходу та генерування нових ідей під час розв'язання нових завдань.

Викладач, під час підготовки, наприклад, до вступної лекції з дисципліни «Веборієнтовані технології. Frontend-розробка» (рис.3.4-3.5), може візуально представити сутність та зміст навчального курсу.

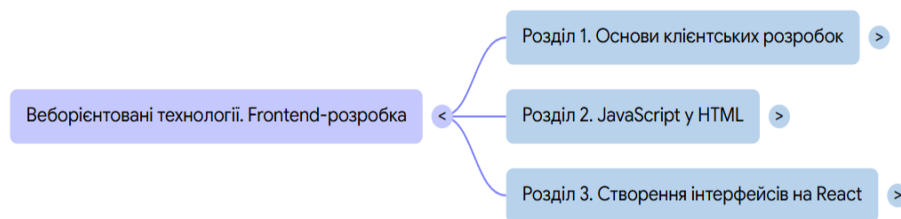


Рис.3.4. Інтелект-карта розділів вступної лекції з дисципліни
«Веборієнтовані технології. Frontend-розробка»

Це особливо важливо на початковому етапі ознайомлення студентів із дисципліною, коли необхідно сформувані чітке уявлення про обсяг навчального матеріалу, предметна область і межі наукового знання, а також про коло понять і проблем, що розглядаються в межах курсу.

Саме застосування методу інтелект-карт є ефективним інструментом, що забезпечує структурування, систематизацію, конкретизацію інформації, а також сприяє її кращому запам'ятовуванню студентами для подальшого використання.

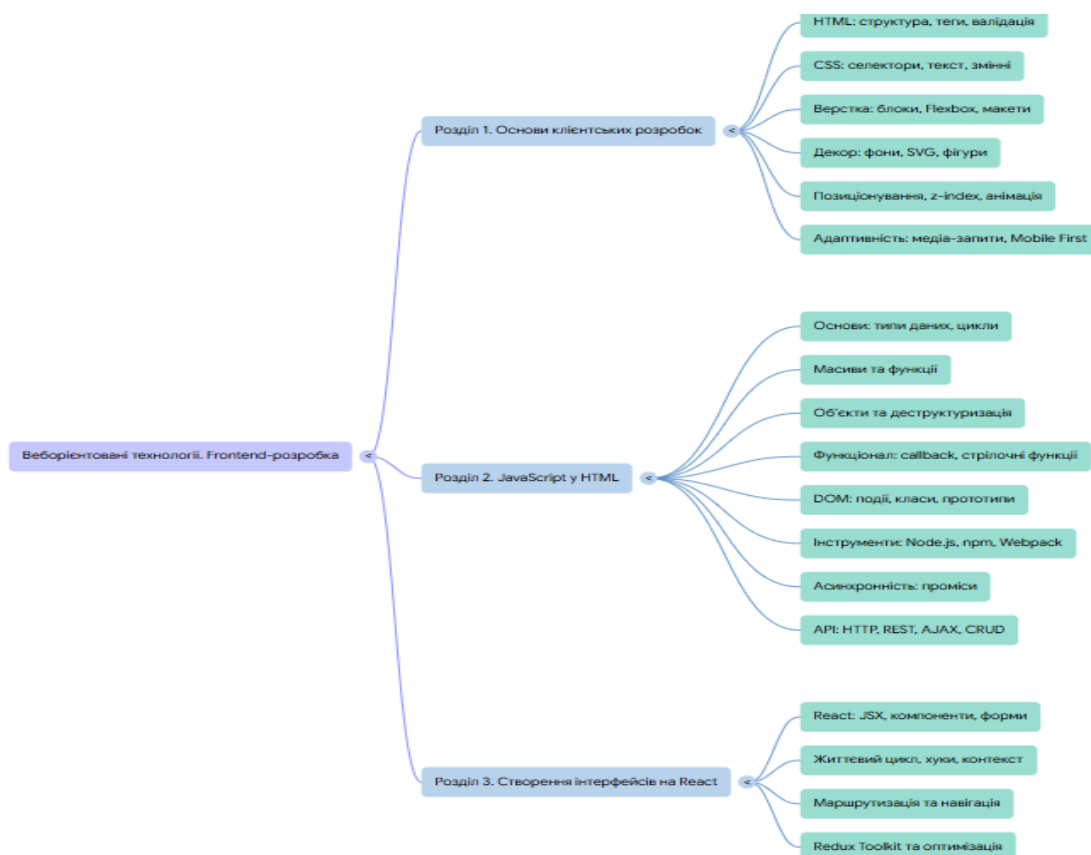


Рис.3.5. Інтелект-карта розділів та тем вступної лекції з дисципліни
«Веборієнтовані технології. Frontend-розробка»

Отже, Інтелект-карти перетворюють процес навчання програмування з лінійного процесу (читання документації) на мережевий, що сприяє глибшому розумінню принципів функціонування складних технічних систем.

3.2 Трансформація форм і засобів навчання програмування в умовах хмаро орієнтованого середовища

Форми організації процесу навчання розкриваються через способи взаємодії педагога зі студентами в процесі здобуття знань і практичних навичок під час розв'язання дидактичних завдань.

Форма організації навчання – це певна структурно-організаційна та управлінська конструкція навчального заняття, що визначається його дидактичними цілями, змістом, а також особливостями діяльності суб'єктів освітнього процесу [14]. Для підготовки майбутніх бакалаврів комп'ютерних наук особливо важливим є поєднання теоретичних знань із практичними вправами, а також адаптація методики навчання до вимог ІТ-індустрії (див. табл. 3.1).

Таблиця 3.1

Аналіз відповідності форм навчання бакалаврів знанням рівням та інструментарію

Форма навчання	Ключові веборієнтовані інструменти	Роль ІІІ-підтримки	Очікуваний результат (DigComp 3.0)
Лекція	GitHub Codespaces, Zoom/Teams, CodePen, Google Slides	Автоматична візуалізація схем та алгоритмів	Рівень 1-2 (Обізнаність)
Практикум	GitHub Codespaces, VS Code Cloud, Git, Vercel	Промпт-інжиніринг, дебагінг, пояснення коду	Рівень 3-4 (Самостійність)
Проектна робота	Vercel, Netlify, Figma, GitHub,	Automated Code Review, експертиза безпеки	Рівень 5-6 (Професійність)

Форма навчання	Ключові веборієнтовані інструменти	Роль ШІ-підтримки	Очікуваний результат (DigComp 3.0)
Самостійна робота	LMS Moodle, ChatGPT, e-olymp, FreeCodeCamp	Персоналізована траєкторії, адаптивний асистент	Рівень 5 (Аналіз та оцінка)

Ефективне навчання програмуванню передбачає використання різних форм організації освітнього процесу. Кожна з них має свої особливості, переваги та доцільність застосування залежно від мети навчання, рівня підготовки студентів та навчального середовища. Розглянемо більш детально ключові форми організації навчання програмування для бакалаврів комп'ютерних наук.

1. Лекційна форма.

Лекція є базовою формою організації навчального процесу, метою якої є подання студентам систематизованої теоретичної інформації (див. підрозділ 2.1). Під час лекцій з програмування розглядаються базові поняття, синтаксис мов програмування, основні парадигми (процедурна, об'єктно-орієнтована, функціональна), а також архітектурні підходи.

У межах розробленої методики лекційна форма навчання зазнає трансформації, переходячи від традиційного формату передачі статичної теоретичної інформації за допомогою слайд-презентацій до інтерактивних сесій живого кодування (live-coding сесій) [15]. Традиційна лекція, основною метою якої є систематизований виклад базових понять та синтаксису, в умовах розвитку інтелектуального вебу доповнюється інтерактивними формами навчання. Такий підхід передбачає використання хмаро орієнтованих середовищ, таких як CodePen [16] або GitHub Codespaces [17], у яких викладач демонструє процес написання коду в реальному часі безпосередньо у браузері. Ключовою зміною є трансформація ролі студента: замість пасивного сприйняття інформації він стає активним учасником навчального процесу через копіювання репозиторію або кодової бази викладача у власне робоче

середовище в один клік. Це дає змогу студентам одразу експериментувати з кодом паралельно з поясненнями викладача, що сприяє засвоєнню когнітивних рівнів «запам'ятовування» та «розуміння» відповідно до таксономії Блума безпосередньо під час заняття.

Роль інтелектуальної підтримки (ШІ) під час лекції полягає в автоматичній генерації пояснювальних схем, коментарів до складних алгоритмів, а також у візуалізації структур даних, наприклад DOM-дерева або станів React-компонентів, у реальному часі. Лекція у форматі live-coding [15] сприяє подоланню розриву між теоретичними знаннями та практичним застосуванням, оскільки будь-яка концепція (цикли, масиви, асинхронність) одразу підкріплюється робочим прикладом, який студент може модифікувати та тестувати у власному хмарному середовищі («пісочниці»).

Використання веборієнтованих інструментів, таких як Zoom, Google Meet або Microsoft Teams (див. підрозділ 2.1), забезпечує синхронну комунікацію, у межах якої викладач може оперативно відповідати на запитання та аналізувати фрагменти коду, що виникають у процесі спільної роботи. Таким чином, лекція перестає бути лише джерелом знань і перетворюється в інтелектуальне освітнє середовище ініціації професійної діяльності, у якому формується базис інформаційної та дата-грамотності згідно до рамки DigComp 3.0.

Приклад. Лекція з теми «Основи JavaScript», де викладач демонструє базові типи даних, цикли, умови, приклади з live-coding [15] у браузері або в CodePen [16].

2. Комп'ютерний практикум.

Комп'ютерний практикум передбачає виконання індивідуальних завдань, пов'язаних із розробкою фрагментів програмного забезпечення, інтерфейсів або інтеграції технологій. У процесі навчання використовуються ПЗ, ICP (IDE), фреймворки, онлайн-компілятори (див. п. 2.1).

Комп'ютерний практикум за авторською методикою трансформується з формату формальної перевірки домашніх завдань у простір інтенсивної

професійної взаємодії в режимі парного програмування із залученням інтелектуальної підтримки ШІ. Студент працює у XO ICP (GitHub Codespaces [17]), де системи ШІ, такі як GitHub Copilot [18], [19] або спеціалізовані чат-боти на базі GPT [20], [21], виконують роль інтелектуальних асистентів. Основний акцент зміщується з механічного написання коду на промпт-інжиніринг та критичну оцінку рішень, запропонованих ШІ, який допомагає студентові у процесі налагодження коду, поясненні логіки асинхронних процесів та пропонує варіанти оптимізації алгоритмів, що відповідає рівням «Застосування» та «Аналіз» за таксономією Блума.

Критично важливим складником методики оцінювання практикуму стає аналіз «цифрового сліду» студента. Викладач більше не перевіряє лише фінальний статичний файл. Об'єктом оцінювання стає історія фіксацій змін у репозиторії GitHub [22]. Методика передбачає наявність ітеративної історії розробки (наприклад, щонайменше 5 фіксацій змін (комітів) для одного завдання), що дає змогу відстежити еволюцію проєкту та забезпечити дотримання принципів академічної доброчесності. Такий підхід унеможливорює плагіат через одноразове завантаження готового коду і стимулює студента до систематичної роботи.

Викладач на практикумі виконує роль ментора та архітектурного аудитора, здійснюючи автоматизований аналіз коду (Automated Code Review) [23], [24] із використанням інструментів ШІ для виявлення вразливостей і порушень стандартів кодування. Завершальним етапом практикуму є автоматичне розгортання релізу на платформах Vercel або Netlify, що забезпечує студенту миттєвий результат у вигляді діючого URL-посилання на власну розробку. Це формує у бакалавра комп'ютерних наук навички управління повним життєвим циклом ПЗ (Software Development Life Cycle, SDLC [25]) та відповідає рівням 7–8 Європейської рамки цифрової компетентності DigComp 3.0

Приклад. Створення вебзастосунку на React, інтеграція з API, тестування компонента.

3. Семінарське заняття.

Семінар у веборієнтованій парадигмі – це не просто перевірка теоретичних знань, а форма організації колективної інтелектуальної діяльності, яка спрямована на розвиток критичного мислення, навичок аргументації та здатності до прийняття технічних рішень (див. підрозділ 2.1).

У традиційному підході семінар часто дублює лекцію у форматі опитування. У запропонованій методиці семінар трансформується у дискусійну платформу, де студенти аналізують підходи до розв’язання алгоритмічних задач, розглядають типові помилки та формулюють власну позицію щодо вибору архітектурних патернів. Це відповідає рівням «Аналіз» (IV) та «Оцінювання» (V) за таксономією Блума.

Розглянемо види семінарських занять.

Онлайн-дискусії. Структуроване обговорення проблем програмування (наприклад, порівняння архітектур REST та GraphQL) у цифровому середовищі (див. підрозділ 2.1).

Цифрові дебати. Аргументоване обговорення альтернативних рішень, де студенти мають захистити обраний стек технологій (наприклад, React і Vue) (див. підрозділ 2.1).

Цифровий мозковий штурм. Колективне генерування ідей щодо розв’язання складних програмних задач за допомогою інтерактивних дошок типу Miro або Padlet (див. підрозділ 2.1).

Аналіз цифрових кейсів. Розгляд реальних або змодельованих ситуацій у сфері програмної інженерії, що дозволяє студентам зануритися у професійний контекст (див. підрозділ 2.1)..

Веборієнтований інструментарій семінарів. Ефективність семінару забезпечується використанням таких сервісів:

- комунікаційні платформи (Zoom, Microsoft Teams або Google Meet (див. підрозділ 2.1) для синхронної взаємодії;

- інструменти асинхронної взаємодії (GitHub Discussions [26], [27]), де студенти можуть вести професійні дискусії безпосередньо в репозиторіях проєктів;
- віртуальні кімнати, що реалізуються з використанням середовищ, зокрема Mozilla Hubs [28], для проведення зустрічей у форматі метавсесвіту, що забезпечує глибше занурення в освітній процес.
- ШІ на семінарах виступає як експерт та аналітичний інструмент. ШІ-агенти можуть автоматично підсумовувати результати дискусій, виокремлювати найбільш обґрунтовані аргументи або генерувати пояснювальні схеми до обговорюваних алгоритмів у режимі реального часу.

На семінарах студенти обговорюють підходи до розв'язання алгоритмічних задач, аналізують типові помилки, а також формують і аргументують власну позицію щодо архітектурних рішень. Це сприяє розвитку критичного мислення.

Приклад. Порівняння підходів до реалізації системи маршрутизації у SPA-застосунках.

4. Дистанційна (онлайн) форма.

Набуває дедалі більшої популярності завдяки своїй гнучкості та доступності. Студенти можуть навчатись у власному темпі за допомогою онлайн-платформ, відеоуроків та інтерактивних тренажерів. Викладач здійснює контроль за прогресом через системи управління навчанням (LMS). У межах запропонованої методики дистанційна форма навчання трансформується у парадигму неперервного навчання, де використання хмарних ХО ІСР, зокрема GitHub Codespaces [17] усуває технічний розрив між аудиторним комп'ютером та персональним пристроєм студента. Це забезпечує однаковість робочого оточення та безперервність «цифрового сліду» студента незалежно від місця перебування (див. підрозділ 2.1).

Приклад. Онлайн-курс «JavaScript: From Basics to Advanced» на платформі Coursera, виконання завдань у CodeSandbox [29], захист проєктів у Zoom.

5. Проектна форма.

Передбачає розробку студентами повноцінних програмних продуктів індивідуально або в групах. У цьому форматі важливим є не лише технічні аспекти розробки, а й планування роботи, командна взаємодія та презентація результатів. У процесі навчання розвиваються практичне мислення та м'які навички (Soft Skills) [30], [31].

Проектна форма стає завершальним етапом, на якому студенти у складі розподілених команд створюють мінімально життєздатний продукт (MVP). Використання систем контролю версій (Git) та платформ автоматичного розгортання (Vercel, Netlify) дає змогу моделювати повний життєвий цикл розробки ПЗ (SDLC) [25]. Це забезпечує досягнення експертних 7–8 рівнів Європейської рамки цифрової компетентності DigComp 3.0, оскільки студент презентує не лише файл із кодом, а й функціональний вебресурс із публічним URL-посиланням.

Приклад. Розробка SPA-застосунку для онлайн-магазину, робота з репозиторієм GitHub [22], Trello [32], презентація мінімально життєздатного продукту (MVP) [33].

6. Індивідуальна форма.

Індивідуальна форма навчання розглядається як одна з найважливіших форм навчальної діяльності, що стимулює активність, пізнавальний інтерес і створює передумови для навчання впродовж життя. У підготовці бакалаврів комп'ютерних наук вона посідає важливе місце, оскільки значна частина освітньої діяльності пов'язана із самостійним опрацюванням матеріалів, виконанням комп'ютерних практикумів та реалізацією індивідуальних проєктів.

Замість епізодичного виконання індивідуальних завдань у позаурочний час студент залучається до неперервного професійного саморозвитку у хмаро орієнтованому середовищі розробки. Використання XO ICP (IDE) (наприклад, GitHub Codespaces [17]) забезпечує безперервність освітнього процесу та мінімізує відмінності між аудиторною та позааудиторною роботою. Студент

може продовжити розробку з того самого етапу, на якому її було призупинено, незалежно від використаного пристрою.

Ключовими складниками індивідуального підходу є:

- можливість самостійно обирати темп навчання, формат роботи та рівень складності завдань, що відповідає сценаріям «A La Carte» або «індивідуальної ротації»;
- індивідуальна діяльність фіксується у вигляді «цифрового сліду» в GitHub [22]. Викладач оцінює не лише кінцевий результат, а й історію фіксацій змін, що відображає ітеративність процесу розробки та дотримання принципів академічної доброчесності;
- створення власного працездатного релізу (мінімально життєздатного продукту), який публікується на платформах Vercel або Netlify;
- використання методу інтелект-карт (Mind Maps), який сприяє візуалізації складних концепцій програмування, структуруванню великих обсягів інформації та плануванню індивідуальної освітньої траєкторії.

Попри високу ефективність для запам'ятовування та структурування інформації, результати опитувань свідчать, що лише 10% студентів систематично використовують цей інструмент, що вказує на значний потенціал його подальшого впровадження [5].

Основними функціями ІІІ в індивідуальній роботі є:

- формування евристичних запитань замість надання готового коду, що стимулює студента до самостійного пошуку розв'язання поставленої задачі;
- трансформація складних теоретичних концепцій (наприклад, замикань в JS) у зрозумілі аналогії або персоналізовані практичні вправи;
- миттєве отримання автоматизованого звіту про якість програмного коду за допомогою інструментів автоматизованої перевірки коду (Automated Code Review) [23],[24], що дає змогу своєчасно виявляти та виправляти помилки перед поданням виконаної роботи.

Індивідуальна робота передбачає виконання студентом навчальних завдань у власному темпі, з можливістю отримання індивідуальних

консультацій. Така форма організації навчання дає змогу поглибити знання з окремих тем і реалізувати складніші проєкти.

Приклад. Створення власного вебпортфоліо, інтеграція REST API та використання сучасних бібліотек.

До спеціалізованих форм індивідуальної діяльності належать олімпіади та самоосвіта.

Олімпіада є формою роботи з обдарованою молоддю (10–15% студентів), у межах якої вони мають змогу застосовувати поглиблені знання в умовах, наближених до реальної професійної діяльності. Участь в олімпіадах сприяє розвитку не лише фахових навичок (Hard Skills) [31], а й важливих соціальних компетентностей, зокрема комунікативності, здатності до командної роботи та емоційної стійкості

Студенти активно доповнюють університетську освіту навчанням на платформах Coursera [34], Udemy [35], Prometheus [36], FreeCodeCamp [37]. Особливе місце займає курс CS50 від Гарвардського університету [38], який часто розглядається як один із найбільш успішних прикладів організації хмаро орієнтованого навчання. Результати опитування студентів КПП свідчать, що сервісом Coursera користуються 45,6% респондентів, що підтверджує високий рівень мотивації до самостійного розширення професійних знань [3].

Важливим складником індивідуальної підготовки є використання автоматизованих систем перевірки завдань. Міжнародний освітній онлайн-портал *E-olymp* є одним із найбільш популярних серед українських студентів (75% використання), оскільки містить понад 7000 задач та забезпечує оперативний зворотний зв'язок щодо правильності виконання програмного коду [5]. Запропонована методика передбачає інтеграцію результатів на цій платформі до персонального портфоліо студента на GitHub.

Упровадження методики навчання програмування ґрунтується на концепції неперервності освітнього процесу незалежно від часу, місця перебування студента чи характеристик використовуваного пристрою. Реалізація цієї концепції забезпечується завдяки хмаро орієнтованим

середовищам розробки, які усувають технічний розрив між навчанням в аудиторії та самостійною роботою вдома. Замість складного локального налаштування середовища розробки та інсталяції необхідних програмних засобів, студент отримує доступ до попередньо налаштованого робочого середовища безпосередньо через веббраузер.

Завдяки хмаро орієнтованому підходу весь програмний код, налаштування та історія розробки зберігаються у хмарному сховищі. Це дає змогу студенту розпочати роботу на університетському комп'ютері та продовжити її на іншому пристрою без втрати даних і необхідності додаткового налаштування. Такий підхід забезпечує однаковість навчального середовища для всіх учасників освітнього процесу та усуває проблему несумісності ПЗ, відому серед розробників як ситуація «на моєму комп'ютері не працює», яку дослідники розглядають як один із бар'єрів початкового етапу навчання програмування.

Використання хмарних інструментів також сприяє підвищенню освітньої мобільності, дозволяючи студентам брати активну участь в освітньому процесі під час стажувань або перебування поза межами ЗВО без втрати якості взаємодії з викладачем та групою. Таким чином, веборієнтоване середовище трансформується з інструменту доставки контенту на цілісну інфраструктуру формування цифрової професійної ідентичності бакалавра комп'ютерних наук.

Форми організації навчання мають доповнювати одна одну. Найбільш ефективним є поєднання теоретичної, практичної та проєктної складових із використанням сучасних інструментів, зокрема онлайн-компіляторів, систем контролю версій (Git), ICP (IDE) та хмарних сервісів, що сприяє глибшому засвоєнню знань і набуттю практичних фахових навичок.

Навчання програмуванню в умовах цифрової трансформації освіти потребує сучасних підходів до організації освітнього процесу. Особливу роль у підготовці бакалаврів комп'ютерних наук відіграє застосування

веборієнтованих технологій, які забезпечують інтерактивність, адаптивність і практичну спрямованість навчання.

Форми навчання – це організаційні моделі освітнього процесу, які визначають, як, де і в яких умовах відбувається здобуття знань, умінь і навичок. У контексті підготовки бакалаврів комп'ютерних наук вони мають бути адаптовані до специфіки галузі, зокрема практичної орієнтованості, постійної взаємодії з ІТ-середовищем та використання актуальних технологій.

Форми навчання повинні відповідати меті підготовки ІТ-фахівців, забезпечуючи формування не лише знань, а й практичних умінь, здатності до командної роботи, гнучкості та адаптивності в умовах цифрового середовища. Використання сучасних вебтехнологій суттєво розширює можливості реалізації цих форм.

Формування компетентності з програмування за розробленою методикою здійснюється поетапно відповідно до логіки когнітивного зростання студента. Розглянемо етапи формування компетентності з програмування на прикладі навчальної дисципліни «Веборієнтовані технології. Frontend-розробка».

Перший етап. Знаннєва адаптація (Розділ 1). Фокусується на рівнях «Знання» та «Розуміння» за таксономією Блума. Студенти опановують семантичну розмітку HTML5 та стилізацію CSS3 у процесі проектування інтерфейсів у Figma. На цьому етапі веборієнтовані технології використовуються як інструменти візуалізації. Студенти навчаються структурувати інформацію, що закладає основу інформаційної та дата-грамотності (сфера 1 DigComp 3.0). Результатом етапу є створення статичного макета вебсторінки, що відповідає принципам Mobile First та Accessibility.

Другий етап. III як інтелектуальний партнер (Розділ 2). Відповідає рівням «Застосування» та «Аналіз». Центром навчання стає мова JavaScript та маніпуляції з DOM-деревом. На цьому етапі впроваджується III як інтелектуальний партнер: студенти використовують генеративні моделі для пояснення асинхронних процесів та оптимізації алгоритмів. Критично

важливим є перехід від копіювання коду до його аудиту. Студент має вміти виявити логічні помилки в коді, згенерованому ШІ, що сприяє розвитку критичного мислення та навичок розв'язання складних проблем (сфера 5 DigComp 3.0).

Третій етап: Професійна реалізація (Розділ 3). Орієнтований на рівні «Створення» та «Оцінювання». Студенти переходять до професійної розробки на базі бібліотеки React та управління станом через Redux Toolkit [39]. Веборієнтоване середовище тут постає як цілісна інфраструктура розробки, де обов'язковим є налаштування CI/CD (Continuous Integration /Continuous Deployment) [40] процесів через GitHub Actions [41]. Студент діє як архітектор хмарного застосунку, самостійно обираючи технологічний стек для реалізації функціоналу MVP [33]. Фінальною точкою є публічне портфоліо на GitHub, яке слугує підтвердженням формування цифрової професійної ідентичності бакалавра та його готовності до виходу на глобальний IT-ринок.

Вивчення та застосування вебтехнологій напряду пов'язані з веборієнтованими технологіями навчання. Під *веборієнтованими технологіями навчання* ми розуміємо форми, методи і веборієнтовані засоби навчання[42].

Аналізуючи підходи до організації веборієнтованого навчання у дослідженнях зарубіжних і вітчизняних науковців, а також узагальнюючи власний багаторічний досвід викладання мов програмування та результати навчальної діяльності студентів, ми дійшли висновку, що у веборієнтованому навчанні доцільним є поєднання 30% технологій традиційного навчання 50% технологій інноваційного навчання та 20% технологій проєктного навчання. Проєктне навчання передбачає спільну роботу викладача зі студентами, а також командну та проєктну діяльність студентів, зокрема розробку мініпроєктів.

Розглядаючи комплексний підхід до організації веборієнтованого навчання, що поєднує традиційне (30%), дистанційне(50%) та проєктне (20%) навчання, із використанням хмаро орієнтованих та веборієнтованих

технологій” [2], слід зазначити, що значна увага у ЗВО приділяється питанням контролю та оцінювання результатів навчальної діяльності студентів, принципам формування оцінювання, а також рівням пізнавальних, емоційних та психомоторних цілей, які мають забезпечувати комплексне охоплення освітнього процесу, зокрема в умовах дистанційного навчання [43].

Актуальними сьогодні є дослідження та впровадження веборієнтованих технологій навчання у підготовці бакалаврів у ЗВО України зокрема за спеціальністю «Комп'ютерні науки». Це зумовлює необхідність наукового обґрунтування мотиваційного, технологічного та процесуального забезпечення освітнього процесу як цілісної педагогічної системи, що враховує індивідуальні інтереси, здібності та схильності студентів, а також сучасні стратегії навчання в XXI сторіччі

Проблема організації веборієнтованого навчання бакалаврів, зокрема комп'ютерних наук є актуальною для ЗВО. ІТ-індустрія висуває нові вимоги до підготовки майбутніх фахівців, що зумовлює необхідність модернізації освітнього процесу: уточнення змісту, оновлення форм, методів і засобів навчання відповідно до потреб соціально-економічного розвитку суспільства [2].

Так, за результатами досліджень Центру цифрової освіти (“The Center for Digital Education”, CDE) 90% опитаних викладачів використовують креативні підходи до викладання навчальних дисциплін із застосуванням технологій веборієнтованого навчання [44]. Водночас в Україні такі технології перебувають на етапі активного впровадження та поступово поширюються в освітньому процесі ЗВО.

Ефективність запропонованої методики забезпечується інтегрованим використанням трьох груп засобів, що формують синтетичне навчальне середовище, наближене до умов сучасної професійної ІТ-діяльності, завдяки чому досягається підвищення рівня професійної підготовки випускників.

Інфраструктурні засоби. На відміну від традиційного навчання, орієнтованого на локальне встановлення ПЗ, запропонована методика базується

на хмаро орієнтованому підході. Використання GitHub [22] як центрального елементу інфраструктури дає змогу організувати неперервний моніторинг діяльності студента. Платформи автоматичного розгортання (Vercel, Netlify) забезпечують миттєву апробацію результатів кодування, надаючи студенту відчуття реальності створеного продукту. Це відповідає вимогам щодо масштабованості та доступності ресурсів незалежно від апаратних потужностей пристрою студента.

Середовища розробки. Основним робочим місцем бакалавра стає GitHub Codespaces або Replit. Ці інструменти усувають проблему «технічного розриву», оскільки всі налаштування (контейнеризація середовища) зберігаються у хмарі. Студент може розпочати роботу в університетській лабораторії і продовжити її вдома у тому ж стані браузера, що реалізує принцип неперервності (Seamless Learning [45]). Хмарні IDE також дозволяють викладачу підключатися до сесії студента у реальному часі для надання оперативної консультативної допомоги.⁵

Інтелектуальні засоби підтримки. ШІ інтегрується у методику через використання GitHub Copilot та спеціалізованих чат-ботів. Вони виконують роль адаптивних асистентів, які забезпечують пояснення програмного коду 24/7, допомагають виявляти синтаксичні помилки та пропонують варіанти рефакторингу. Важливим складником методики є залучення засобів автоматизованої перевірки коду, що дає змогу студенту отримати первинний зворотний зв'язок щодо якості та безпеки програмного коду ще до подання роботи викладачеві. Це сприяє розвитку навичок самоконтролю, відповідального ставлення до результатів власної діяльності та професійної відповідальності (напрям 4 та 5 DigComp 3.0).

Аналізуючи засоби навчання, зазначимо, що засоби Frontend-орієнтованих технологій – це засоби розробки інтерфейсу веборієнтованих застосунків, з якими безпосередньо працює користувач. До них належать мови програмування, бібліотеки, фреймворки, інтегровані середовища розробки та

інші програмні засоби, що забезпечують створення зручного, функціонального та привабливого користувацького інтерфейсу.

Для створення користувацького інтерфейсу у Frontend-розробці використовуються такі основні мови:

- HTML (HyperText Markup Language) – це мова розмітки, яка використовується для структурування вебсторінок (заголовки, параграфи, зображення, посилання). HTML5 розширив можливості мови завдяки впровадженню семантичних тегів (<header>, <article>, <footer>) і підтримки мультимедійного контенту.

- CSS (Cascading Style Sheets) – це мова стилізації вебсторінок, що визначає їх зовнішній вигляд, відповідає за їх стиль та оформлення (кольори, шрифти, відступи, адаптивність); надає сучасні можливості використання Flexbox та Grid Layout для складних макетів, CSS-анімації та трансформації, змінних CSS (Custom Properties).

- JavaScript (JS) – це мова програмування, яка забезпечує інтерактивність і динамічну поведінку вебсторінок. Вона використовується для реалізації модальних вікон, слайдерів, валідації форм, роботи з API та іншого динамічного функціоналу. Сучасний JavaScript також підтримує асинхронне програмування, модульну архітектуру та роботу з фреймворками і бібліотеками.

Особлива увага в розробленій методиці приділяється формувальному оцінюванню. З урахуванням позитивного досвіду В. Кухаренка щодо застосування таксономії Блума в ЗВО. Цей підхід було адаптовано до процесу підготовки майбутніх бакалаврів комп'ютерних наук. Формувально-оцінювальний блок передбачає виконання завдань шести рівнів складності відповідно до таксономії Блума. Як приклад розглянемо комп'ютерний практикум (мова програмування JavaScript) по темі «Прототипи. Класи. DOM. Події. Делегування подій. Throttle і Debounce» для бакалаврів спеціальності «Комп'ютерні науки»

Таблиця 3.2

Приклад комп'ютерного практикума за таксономією Блума

№	Рівень таксономії Блума	Практичне завдання
1	1 рівень Запам'ятовування (Remember) – 5 балів	Завдання Створіть HTML-сторінку та JavaScript-файл. Виведіть у консоль: 1. Всі елементи сторінки за допомогою: – document.body – document.head – querySelector() – querySelectorAll() 2. Створіть об'єкт student та виведіть його властивості. 3. Створіть клас Student та створіть один екземпляр класу. Результат У консолі браузера відображаються знайдені DOM-елементи та об'єкт класу. Оцінювання: 5 балів
2	2 рівень Розуміння (Understand) – 5 балів	Завдання Створіть сторінку з двома кнопками: <button id="btn1">Властивість</button> <button id="btn2">Атрибут</button> При натисканні: – перша кнопка змінює властивість textContent; – друга кнопка змінює атрибут title. У консолі виведіть різницю між: – element.textContent – element.getAttribute() Результат Студент демонструє різницю між властивостями та атрибутами DOM. Оцінювання: 10 балів
3	3 рівень 3. Застосування (Apply) – 15 балів	Завдання Розробіть форму входу. Поля: – Логін – Пароль – Кнопка «Увійти» Необхідно: 1. Отримати дані через DOM. 2. Обробити подію submit.

№	Рівень таксономії Блума	Практичне завдання
		<p>3. Заборонити стандартне перезавантаження сторінки.</p> <p>4. Якщо всі поля заповнені:</p> <ul style="list-style-type: none"> – вивести повідомлення «Вхід виконано». <p>5. Якщо хоча б одне поле порожнє:</p> <ul style="list-style-type: none"> – показати повідомлення про помилку. <p>Результат Працююча форма з використанням об'єкта події. Оцінювання: 15 балів</p>
4	<p>4 рівень</p> <p>4. Аналіз (Analyze) –20 балів</p>	<p>Завдання Створіть сторінку каталогу товарів: <pre><div class="products"> <div class="product">Ноутбук</div> <div class="product">Планшет</div> <div class="product">Смартфон</div> </div></pre> Реалізуйте: 1. При кліку на товар вивести: <ul style="list-style-type: none"> – target – currentTarget – текст товару. 2. Пояснити різницю між: <ul style="list-style-type: none"> – event.target – event.currentTarget Результат Студент аналізує механізм поширення подій у DOM. Оцінювання: 15 балів</p>
5	<p>5 рівень</p> <p>5. Оцінювання (Evaluate) – 25 балів</p>	<p>Завдання <i>Створіть список із 50 елементів.</i> Варіант А Для кожного елемента окремо призначте: element.addEventListener(...) Варіант Б Використайте делегування подій через один обробник на батьківському контейнері. Виміряйте: <ul style="list-style-type: none"> – кількість створених обробників; – обсяг коду; – зручність підтримки. Зробіть висновок: <ul style="list-style-type: none"> – який спосіб ефективніший; </p>

№	Рівень таксономії Блума	Практичне завдання
		<p>– чому.</p> <p>Результат Письмовий висновок та два варіанти реалізації.</p> <p>Оцінювання: 15 балів</p>
6	<p>6 рівень</p> <p>6. Створення (Create) – 30 балів</p>	<p>Завдання <i>Розробити вебзастосунок «Керування списком покупок»</i> Функціонал</p> <p>1. Клас Product Створіть клас: <pre>class Product { constructor(name, quantity) { this.name = name; this.quantity = quantity; } }</pre> </p> <p>2. Додавання товарів Користувач вводить: <ul style="list-style-type: none"> – назву товару; – кількість. Після натискання кнопки: «Додати» товар з'являється у списку.</p> <p>3. Події форм Використати: <ul style="list-style-type: none"> – submit; – input. </p> <p>4. Події клавіатури Додавання товару клавішею: Enter</p> <p>5. Делегування подій Кожен товар містить кнопки: <ul style="list-style-type: none"> – ✓ Куплено – ✗ Видалити Обробка всіх кнопок повинна виконуватися через один addEventListener() на контейнері списку.</p> <p>6. Debounce Для поля пошуку товарів: <ul style="list-style-type: none"> – реалізувати пошук; – запускати фільтрацію через 500 мс після завершення введення. </p> <p>7. Throttle</p>

№	Рівень таксономії Блума	Практичне завдання
		<p>Під час прокручування сторінки:</p> <ul style="list-style-type: none"> – відображати поточну позицію скролу; – оновлювати інформацію не частіше ніж один раз на 300 мс. <p>8. Робота з DOM</p> <p>Використати:</p> <ul style="list-style-type: none"> – createElement() – append() – remove() – classList – dataset

3.3 Використання веборієнтованих технологій у процесі навчання бакалаврів комп'ютерних наук.

Використання веборієнтованих технологій відіграє важливу роль у навчанні програмування майбутніх бакалаврів комп'ютерних наук у ЗВО. Насамперед це комплексне поєднання інтерактивних ресурсів, цифрових платформ, веборієнтованих і хмарних інструментів, яке забезпечує формування у студентів поглиблених теоретичних знань, практичних умінь і професійних навичок у сфері сучасної розробки ПЗ.

Водночас підвищення ефективності навчання програмування із застосуванням веборієнтованих технологій можливе лише за умови створення сприятливого освітнього середовища та забезпечення відповідних педагогічних умов [46].

Педагогічні умови – це система факторів і обставин, які створюються в освітньому середовищі з метою забезпечення ефективного виховання та навчання студентів. Вони впливають на процес засвоєння знань, формування навичок, розвиток особистості та соціалізацію[46].

Приділення уваги педагогічним умовам навчання з використанням веборієнтованих технологій для студентів інформаційних технологій має низку важливих переваг і обґрунтованих причин. Розглянемо їх детальніше.

Актуальність полягає у забезпеченні студентів сучасними технологіями, що відповідають їхнім освітнім потребам та актуальним вимогам ринку праці. Розвиток вебтехнологій є невід'ємною частиною індустрії інформаційних технологій. На сучасному етапі всі технічні університети, зокрема НТУУ «Київський політехнічний інститут імені Ігоря Сікорського», активно співпрацюють з ІТ-компаніями з метою підвищення якості фахової підготовки бакалаврів комп'ютерних наук.

Сучасні вебтехнології, що використовуються ІТ-компаніями, викладачі ЗВО намагаються впроваджувати в освітній процес. Як свідчать результати опитування, на 4 курсі 76,7 % студентів–випускників вже влаштовані в ІТ-компаніях як штатні співробітники, що свідчить про конкурентоспроможність випускників КПІ ім. Ігоря Сікорського на сучасному ІТ-ринку праці [4].

Практичні навички формуються завдяки використанню веборієнтованих технологій у навчанні, що сприяє підготовці студентів до професійної діяльності в індустрії. Протягом навчання в університеті студенти удосконалюють свої практичні навички написання програмного коду в таких ІСР, як Visual Studio Code[47], WebStorm[48], Sublime Text [49], Visual Studio Community [50], Eclipse[51], NetBeans [52], та інші. При цьому використовуються такі веборієнтовані фреймворки і бібліотеки, зокрема Node.js[53], для Backend-розробки, а також React, Angular, Vue.js та інші для створення Frontend-інтерфейсів. Додатково застосовуються мови розмітки HTML, стилізації (CSS), мови програмування JavaScript, Java, C#, Python та інші. Це дає змогу студентам безпосередньо застосовувати отримані знання і практичні навички у віртуальних та реальних проєктах.

– *Гнучкість та дистанційність* – використання вебтехнологій дає змогу організовувати освітній процес у дистанційному форматі, що особливо актуально для студентів, які можуть перебувати в різних місцях. Для організації дистанційного навчання ефективним є використання відеоконференцій, як альтернативи аудиторним заняттям. Для проведення відеоконференцій можна виділити наступні популярні платформи: Zoom[54], Google Meet [55], My Own

Conference [56], OpenMeeting [57], Microsoft Teams [58]. Також набувають популярності платформи для віртуальних конференцій як Hopin, vFairs, Demio, WebinarJam, Airmeet, Livestorm, Accelevents, Zoho Meeting, Bevy, Cvent. [59].

– *Активна участь та спільна робота* – веборієнтовані технології надають можливості для активної участі студентів у віртуальних проєктах, спільній роботі та обміні досвідом. Насамперед це використання системи контролю версій Git, а також вебсервісу для спільної командної розробки ПЗ GitHub і клієнтського застосунку GitHub Desktop [60].

– *Індивідуалізація та адаптивність* – вебплатформи дають змогу адаптувати матеріали та завдання відповідно до індивідуальних потреб та рівня підготовки кожного студента.

– *Збагачення освітнього процесу* – використання інтерактивних відеоуроків, віртуальних лабораторій та інших цифрових засобів урізноманітнює освітній процес і робить його більш цікавим для студентів. Так, для поглиблення знань із дисциплін, пов'язаних із веборієнтованими технологіями, викладачі рекомендують студентам проходити онлайн-курси на платформах Coursera [34], Udemy [35], Prometheus [36], CyberBionic Systematics [61] тощо.

– *Розвиток критичного мислення та самоосвіти* – веборієнтовані технології сприяють формуванню навичок критичного аналізу і самостійного опрацювання інформації в інтернеті.

Загальна мета педагогічних умов полягає у створенні оптимального освітнього середовища, що сприяє всебічному розвитку особистості студента. Основними складовими педагогічних умов є методи навчання, зміст навчання, організація освітнього середовища, система оцінювання, комунікація та взаємодія, індивідуалізація навчання та використання інноваційних технологій.

Педагогічні умови використання веборієнтованих технологій у підготовці бакалаврів з інформаційних технологій та систем мають низку особливостей, зокрема:

– *інтерактивність та онлайн-ресурси* – це забезпечення доступу до онлайн-ресурсів, що сприяють засвоєнню навчального контенту студентами; використання відеолекцій, вебінарів, онлайн-курсів для підвищення ефективності навчання та комунікації;

– *використання вебплатформ* – це застосування вебплатформ для дистанційного навчання, де студенти мають змогу здійснювати завдання, отримувати зворотний зв'язок та взаємодію з викладачами; використання електронних журналів для моніторингу успішності студентів та надання зворотного зв'язку;

– *активна залученість* – це створення віртуальних лабораторій і практичних завдань, що дають змогу застосовувати теоретичні знання на практиці; залучення студентів до командної роботи над проєктами за допомогою онлайн-інструментів, форумів та інших засобів віртуальної взаємодії.

– *оцінювання та зворотний зв'язок* – це використання автоматизованих систем для оцінювання завдань та тестів; забезпечення оперативного зворотного зв'язку через електронні коментарі та онлайн-консультації;

– *індивідуалізація навчання* – це використання адаптивних систем навчання, які враховують потреби та рівень підготовки кожного студента; організація індивідуальних консультацій через відеозв'язок;

– *розвиток навичок самоосвіти* – це заохочення студентів до самостійного навчання та дослідницької діяльності з використанням онлайн-ресурсів; формування навичок критичного мислення та аналізу інформації з інтернету;

– *забезпечення технічної інфраструктури* – це гарантування доступу студентів до необхідних технічних засобів та програмного забезпечення для ефективного вивчення.

Ці педагогічні умови сприяють ефективній інтеграції веборієнтованих технологій у процес навчання, роблячи його більш доступним та цікавим для

студентів з інформаційних технологій, зокрема для бакалаврів комп'ютерних наук.

Основними перевагами веборієнтованих технологій у вивченні програмування є: можливість навчатися в зручний час, а також забезпечення швидкого доступу до навчальних матеріалів: Moodle [62], GitHub[22], Google Classroom [63],); використання онлайн-компіляторів (Replit [64], CodePen [16], JSFiddle [65]; широкий вибір курсів з програмування в інтернеті, зокрема на платформах Coursera [34], Udemy [35], тощо; використання документованих API, специфікацій (MDN, W3Schools); підтримка мобільного навчання.

Складниками веборієнтованих технологій є: мови програмування, мови вебпрограмування, інтегровані середовища розробки, компілятори та інтерпретатори, інтелект-карти, а також технології ІІІ.

Мови програмування – це алгоритмічні мови, призначені для опису алгоритмів і структур даних, що виконуються комп'ютером або система позначень для точного опису алгоритмів, що реалізуються за допомогою комп'ютера [66].

Мови програмування можна класифікувати за різними критеріями: за призначенням, рівнем абстракції, парадигмами програмування, та способом виконання (табл. 3.3-3.7).

Таблиця 3.3

Класифікація мов програмування. за призначенням

Призначення	Приклади	Де використовуються
Системні мови	C, C++, Rust, Go	Для створення операційних систем (ОС), драйверів, низькорівневих програм.
Веброзробка (Frontend)	HTML, CSS, JavaScript, TypeScript	Для створення користувацького інтерфейсу (UI) адаптивних веборієнтованих застосунків (сайтів).
Веброзробка (Backend)	Node.js, Python, Ruby, Java, PHP	Для створення серверної частини веборієнтованих

Призначення	Приклади	Де використовуються
		застосунків, які пов'язані з серверами, API, базами даних.
Мови для науки та ІІІ	Python, R, Julia, MATLAB	Data Science, ML, AI.
Мобільна розробка	Swift (iOS), Kotlin/Java (Android), Dart (Flutter)	Створення додатків для планшетів смартфонів.
Ігрова розробка	C++, C#, Lua	Unity, Unreal Engine, ігрові рушії.
Спеціалізовані	SQL (бази даних), Verilog/VHDL (мікросхеми)	Робота з даними, апаратним забезпеченням.

Таблиця 3.4

Класифікація мов програмування. за рівнем абстракції

Тип	Приклад	Особливості
Мови низького рівня	Assembly, Machine code	Близькі до апаратної частини ПК, швидкі, але складні у використанні.
Мови середнього рівня	C, C++	Поєднують роботу з пам'яттю та абстракції високого рівня.
Мови високого рівня	Python, Java, JavaScript, PHP	Більш абстрактні, простіші для людини, автоматизують управління пам'яттю.

Таблиця 3.5

Класифікація мов програмування. за парадигмами програмування

Парадигма	Приклад	Опис
Процедурні	C, Pascal, Fortran	Програма складається з процедур (функцій), які викликаються одна за одною.

Парадигма	Приклад	Опис
Об'єктно-орієнтовані (ООР)	Java, C++, C#, Python	Програмування через об'єкти та класи.
Функціональні	Haskell, Lisp, Elixir, Scala	Фокус на математичних функціях, без змінних стану.

Таблиця 3.6

Класифікація мов програмування за способом виконання

Тип	Приклади	Особливості
Компілюємі	C, C++, Rust, Go, Swift	Перекладаються у машинний код перед виконанням → дуже швидкі.
Інтерпретовані	Python, JavaScript, PHP, Ruby	Виконуються рядок за рядком → зручніші, але повільніші.
Bytecode + VM	Java, C#, Kotlin	Код компілюється у проміжний байткод і запускається у віртуальній машині (JVM, CLR).

Таблиця 3.7

Класифікація мов програмування за поколінням

Покоління	Позначення
1 покоління	(1GL) – машинні коди (0 і 1)
2 покоління	(2GL) – асемблер (Assembly)
3 покоління	(3GL) – високорівневі мови (C, Java, Python)
4 покоління	(4GL) – спеціалізовані мови (SQL, MATLAB)
5 покоління	(5GL) – мови для ІІІ та логічного програмування (Prolog, Mercury)

Однією із складових веборієнтованих технологій є мови вебпрограмування, які поділяються на дві групи: серверні і клієнтські. Програми, написані серверними мовами вебпрограмування, виконуються на

стороні сервера, шляхом обробки запитів клієнта та взаємодії із системами керування базами даних. Результат обробки запиту передається мережею до клієнта у вигляді файлів різних форматів, зокрема html, php, asp, aspx, perl, xml, ssi, dhtml, xhtml тощо. Водночас програми, створенні з використанням клієнтських мов програмування, виконуються браузером на стороні клієнта.

В межах цього дисертаційного дослідження розглядалися такі технології Frontend-розробки:

- HTML – мова розмітки вебсторінок;
- CSS – мова стилізації (шрифти, анімації, кольори) ;
- JavaScript – мова програмування, що забезпечує динамічність та інтерактивність веборієнтованих застосунків;
- React, Vue.js, Angular – сучасні JS-бібліотеки та фреймворки.

В 2025 році було проведено опитування студентів галузі знань F “Інформаційні технології”, на факультеті Інформатики та обчислювальної техніки (ФІОТ), у якому взяли участь 105 студентів-респондентів 4 курсу (87%), 3 курсу (10%) та інших курсів (3%) [4].

Метою опитування було з'ясування того, які мови програмування студенти старших курсів самостійно обирають для розробки програм у межах освітньо-професійних програм факультету, а також наскільки ці засоби корелюють із програмними технологіями, що використовуються практикуючими розробниками під час реалізації проєктів в ІТ-компаніях. Зокрема, досліджувалися мови вебпрограмування, інтегровані середовища розробки, платформи, фреймворки, технології та бібліотеки.

Також було проведено порівняльний аналіз використання мов програмування та програмного забезпечення за такими напрямками:

- до навчання в університеті;
- під час навчання в університеті;
- додаткове(самостійне) вивчення ПЗ;.
- використання мов програмування та ІСР в реальних проєктах.

– рекомендації студентів щодо добору ПЗ для посилення змістової складової навчальних курсів.

На рисунку 3.6 представленні мови програмування, якими студенти володіли ще до початку навчання в університеті. З'ясувалось, що 36,2% респондентів узагалі не вивчали жодної мови, 36,2% вивчали мову Pascal, а 23,8% програмували мовою C++ [4].

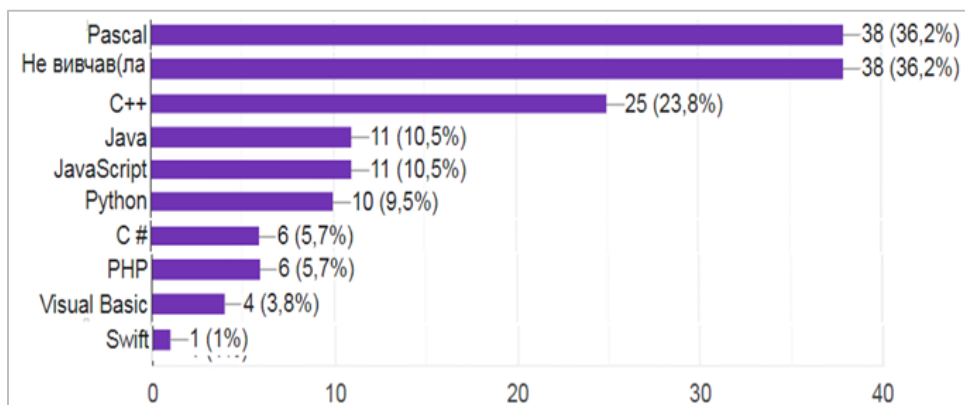


Рис. 3.6 Мови програмування, які вивчались студентами до початку навчання в університеті [4].

Результати опитування, щодо вивчення мов програмування під час навчання в університеті показали, що на першому місці за частотою використання перебуває мова програмування JavaScript (18,1%), на другому місці Java (14,6%), на третьому C# (14,4%), (рис. 3.7) [4].

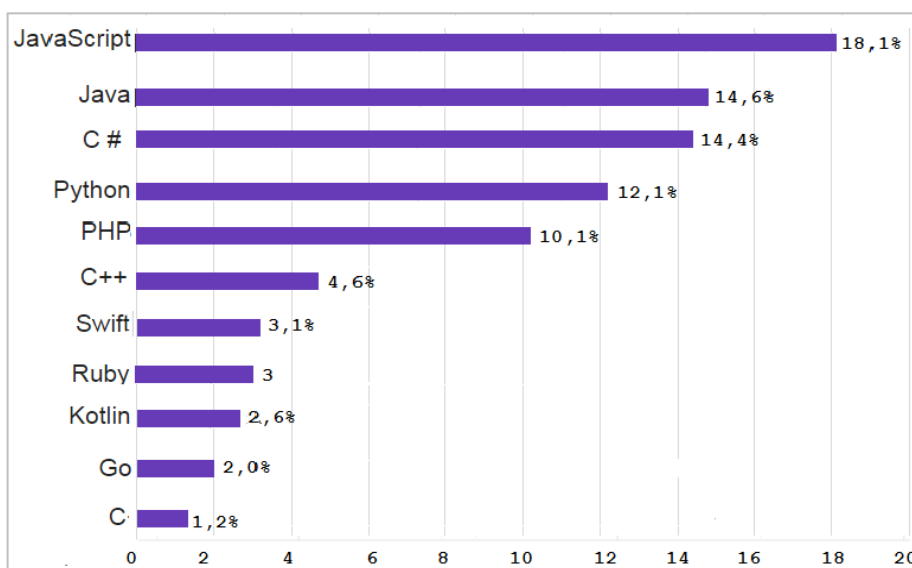


Рис. 3.7 Мови програмування, які вивчались в межах освітніх програм в університеті ФІОТ університету в 2025 році [4].

Протягом усього навчання, поряд із лекціями, семінарами, комп'ютерними практикумами, значна увага приділяється самостійній роботі бакалаврів комп'ютерних наук, починаючи з першого курсу. Організація самостійної роботи сприяє розвитку у студентів самостійності, відповідальності і організованості та творчого підходу до розв'язання навчальних і професійних завдань [67]. На факультеті інформатики та обчислювальної техніки студенти й викладачі приділяють цьому питанню особливу увагу.

Діаграма на рисунку 3.8 ілюструє відсоткове співвідношення вивчення додаткового ПЗ студентами-програмістами, які під час навчання в університеті вдосконалювали свої навички або самостійно опановували нові мови програмування.

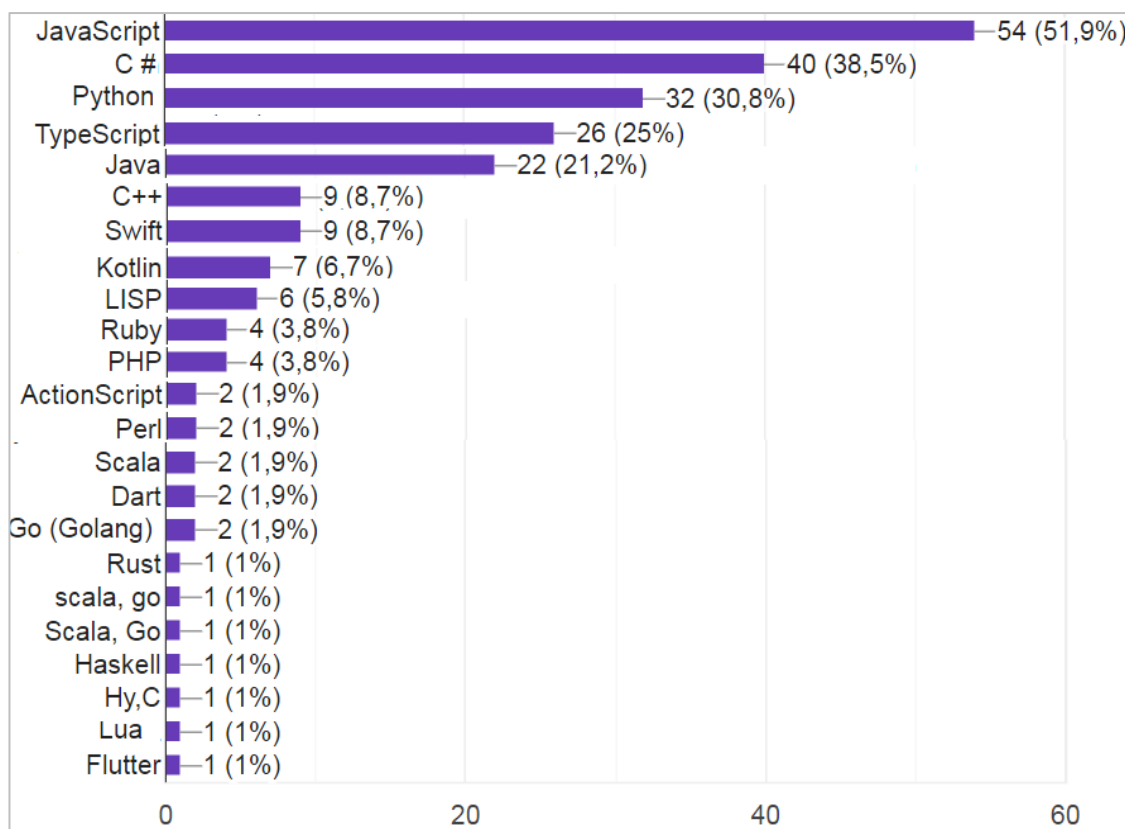


Рис. 3.8. Мови програмування, які студенти удосконалюють або вивчають самостійно [4].

Використання мов програмування в реальних проєктах в ІТ-компаніях студентами 4 курсу, представлено на рисунку 3.9.

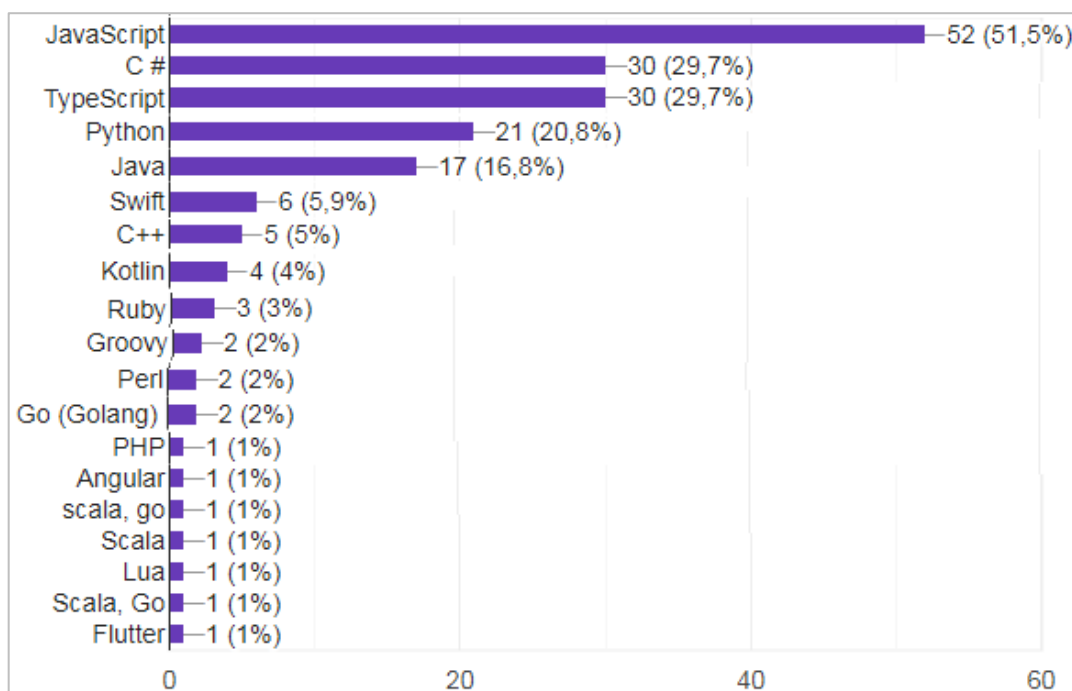


Рис. 3.9 Мови програмування, які студенти 4 курсу використовують при написанні реальних проєктів в ІТ-фірмах [4].

Як свідчать результати опитування, мова програмування JavaScript посідає перше місце за рівнем популярності серед студентів-респондентів (51,5 %). Другу позицію поділяють мови C# (29,7%) і TypeScript (29,7%), популярність якої останніми роками суттєво зростає у сфері розроблення ПЗ. Трійку лідерів завершує мова Python (20,8%) [4].

Важливим аспектом також є врахування рекомендацій студентів-випускників щодо добору ПЗ з метою вдосконалення змістового складника навчальних дисциплін. Результати опитування, пов'язаного з такими рекомендаціями, наведено на рисунку 3.10.

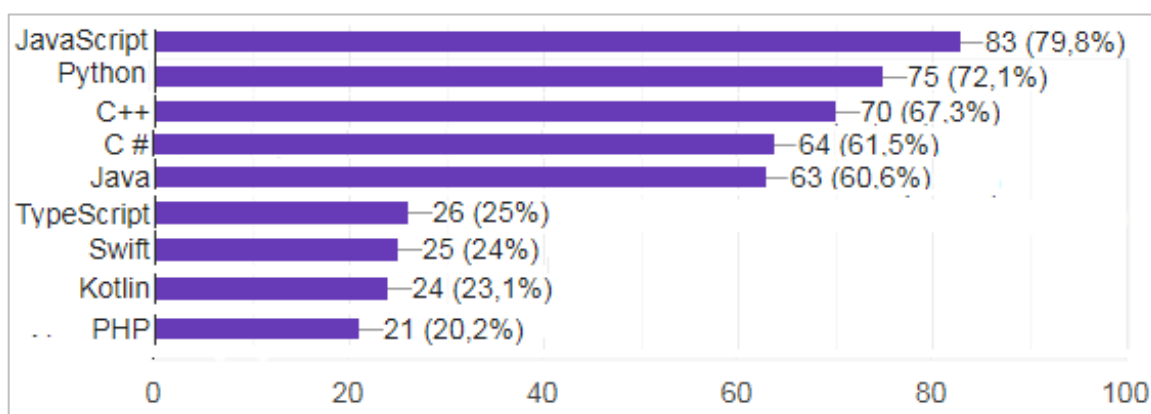


Рис. 3.10. Рекомендації студентів-випускників щодо добору мов програмування

Аналіз результатів опитування щодо використання мов програмування студентами факультету інформатики та обчислювальної техніки засвідчив, що мова програмування JavaScript є найбільш поширеною серед студентів як під час виконання реальних проєктів, так і в процесі самостійного опанування нових мов програмування[4].

Наступним складником веборієнтованих технологій, якому приділялась значна увага під час опитування, є вибір і використання ІСР (Integrated Development Environment (IDE)) для написання програмного коду. В інтернет-джерелах наведено велику кількість визначення понять ІСР. Розглянемо декілька з них [4].

Інтегроване середовище розробки (ІСР) – це програмний застосунок, який надає веброзробникам інструменти, необхідні для створення програмного забезпечення. Зазвичай воно містить текстовий редактор, засоби автоматизації, інструменти компіляції програмного коду та відладчик (debugger)

Науковець Harish Rajora у своїх статтях визначає, що ІСР – це потужні інструменти, які оснащені потужними функціями, такими як автозаповнення, перевірка синтаксису, відладчик, налаштування, перегляд вебсторін в реальному часі для кращого розуміння результатів роботи [68].

Він визначає, що інтегроване середовище розробки (ІСР) – це програмний засіб, який сприяє розробці застосунків. ІСР призначені для об'єднання всіх етапів програмування в межах одного програмного середовища. Тому вони надають єдиний інтерфейс, що містить всі інструменти, необхідні розробнику: редактор коду, компілятор, відладчик та засоби автоматизації налаштування. Крім того, деякі ІСР можуть також включати браузер класів, переглядач об'єктів, діаграма і ієрархію класів [69]

За основу визначення ІСР в цьому дослідженні взято наведені вище визначення, які взаємно доповнюють одне одного. Залежно від кількості підтримуваних мов програмування ІСР поділяються на одномовні та багатомовні.

Залежно від платформ, для яких створюються програми, а також від платформ на яких функціонує сама ІСР, середовища розробки поділяються на кросплатформні та платформозалежні. Кросплатформні ІСР підтримують роботу з різними операційними системами, зокрема Microsoft Visual Studio Code, Eclipse, NetBeans, Embarcadero RAD Studio. Платформозалежні функціонують в межах однієї платформи, зокрема Visual Basic, Delphi, Dev-C++ [4].

Результати опитування щодо використання ІСР під час написання програмного коду в процесі навчання в університеті засвідчили таке: найбільш популярним серед студентів є середовище розробок Microsoft Visual Code (61,5%). Друге місце посів Android Studio (57,7 %), а третє – IntelliJ IDEA (35,6%). Такий самий показник має Microsoft Visual Community (35,6 %) (рис.3.11) [4].

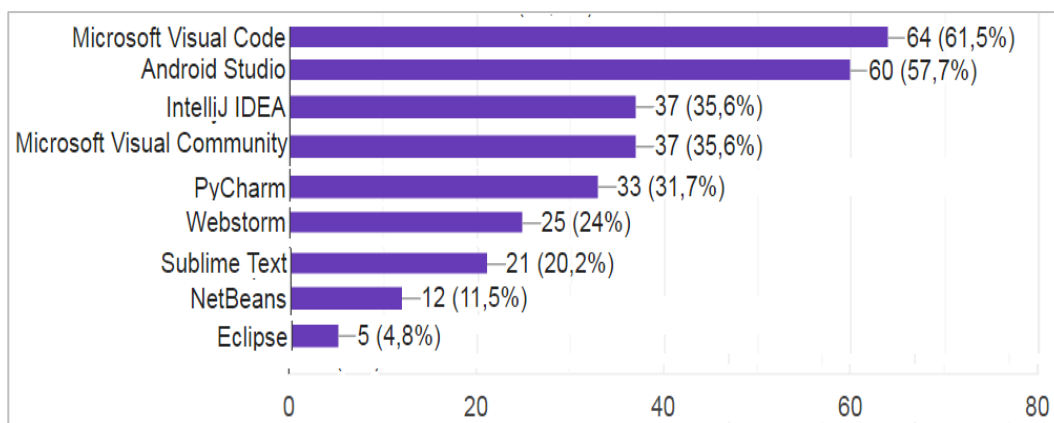


Рис. 3.11. Використання ІСР при написанні програм протягом навчання в університеті [4].

Щодо використання ІСР у реальних проєктах в ІТ-компаніях, результати опитування свідчать, що найбільш затребуваними є Microsoft Visual Code (45,1%), IntelliJ IDEA (19,6%), WebStorm (17,6%), PyCharm (15,7%), Microsoft Visual Community (15,7%) (рис 3.12).

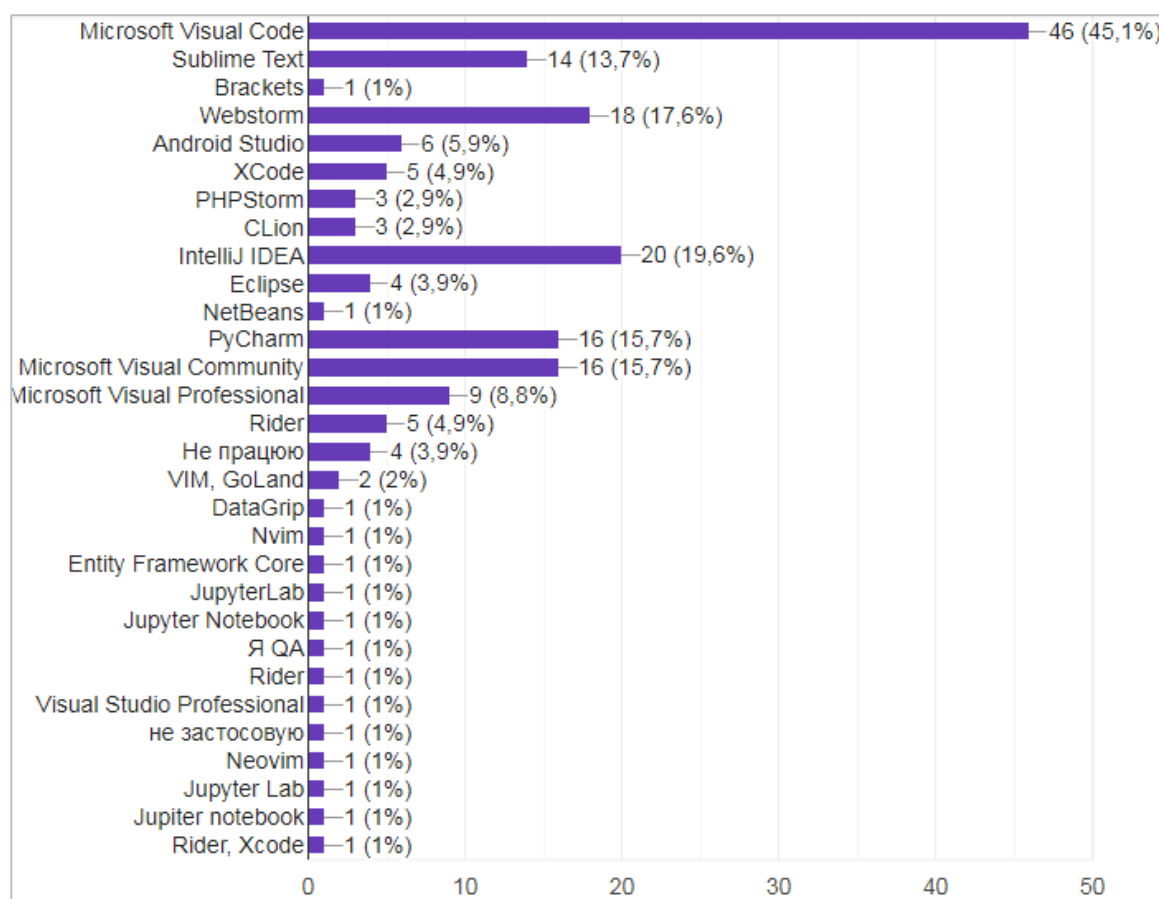


Рис. 3.12. Використання ІСР при написанні програм в реальних проектах ІТ-фірм (за межами університету) [4].

У своїх рекомендаціях, щодо вивчення ІСР в межах освітніх програм університету, студенти також віддають перевагу у наступній послідовності: Microsoft Visual Code(55,8%), Android Studio 45,2%), Microsoft Visual Community (40,4%), IntelliJ IDEA (40,4%), PyCharm (28,8%) тощо (рис 3.13) [4].

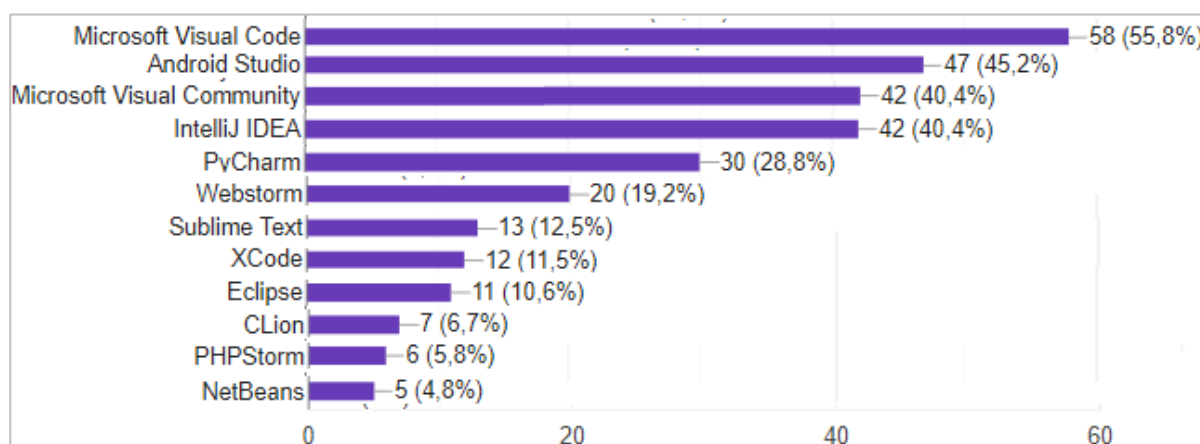


Рис. 3.13. Рекомендації студентів-випускників щодо добору мов програмування [4].

Результати опитування щодо використання ІСР показали, що студенти найчастіше застосовують Microsoft Visual Code, як під час навчання так і під час роботи над реальними проєктами. Далі посідають IntelliJ IDEA, Microsoft Visual Community, WebStorm, PyCharm [4].

Продуктивність використання ІСР підвищується завдяки прискоренню виконання завдань розробки, скороченню часу на налаштування середовища, регулярному оновленню функціональних можливостей та стандартизації процесу розробки ПЗ. Умовно ІСР поділяються на три категорії: ІСР для Frontend-розробок, ІСР для Backend-розробок та хмарні ІСР. Для прикладу розглянемо особливості, переваги та недоліки представників кожної категорії, а саме Visual Studio Code (Frontend-розробка), IntelliJ IDEA (Backend-розробка), Visual Studio Codespaces(хмарне середовище розробки) (табл. 3.8) [4].

Таблиця 3.8

Переваги та недоліки інтегрованих середовищ розробки

IDE	Тип розробки	Підтримувані мови	Операційні системи	Переваги	Недоліки
Visual Studio Code	Фронтенд / Бекенд / Mobile	JavaScript / TypeScript, Python, PHP, Java, C#, інші	Windows, macOS, Linux	Безкоштовна, гнучка, з відкритим вихідним кодом, кросплатформна, популярна, тому легко знайти довідку або розширення, велика кількість розширень	Потребує великої кількості налаштувань
WebStorm	Фронтенд / Node.js	JavaScript / TypeScript	Windows, macOS, Linux	Кросплатформеність, платформа з відкритим кодом, потужна підтримка JavaScript, зручна навігація	Платна; споживає багато ресурсів

IDE	Тип розробки	Підтримувані мови	Операційні системи	Переваги	Недоліки
IntelliJ IDEA	Бекенд	Java, Kotlin	Windows, Linux, macOS,	Багато інструментів, які полегшують розробку, професійна для Java/Spring, розумний автокомпліт Створено на платформі з відкритим кодом	Ultimate-версія платна; вимоглива до апаратної частини ПК
PyCharm	Бекенд	Python	Windows, macOS, Linux	Зручна для Django/Flask, потужний дебагер	Professional-версія платна; велике споживання пам'яті
PhpStorm	Бекенд	PHP	Windows, macOS, Linux	Оптимізована для Laravel, зручна робота з БД	Платна; потребує потужного ПК
Visual Studio	Бекенд / Mobile	C#, .NET	Windows, macOS	Потужна для ASP.NET, хороші інструменти для Xamarin	Великий розмір; складна для новачків
Android Studio	Mobile	Kotlin, Java	Windows, macOS, Linux	Офіційна для Android, вбудований емулятор	Дуже ресурсомістка
Xcode	Mobile	Swift, Objective-C	macOS	Офіційна для iOS, повна інтеграція з Apple	Працює лише на macOS; великі оновлення
Eclipse	Бекенд	Java	Windows, macOS, Linux	Безкоштовна, розширювана	Застарілий інтерфейс; складне налаштування

Третій важливий аспект у процесі навчання мов програмування – онлайн-компілятори (табл.3.9).

Таблиця 3.9

Перелік онлайн-компіляторів

Платформа	Які мови підтримує	Призначення	Посилання
Replit	Python, C++, JavaScript	Загальне програмування	https://replit.com/
CodePen	HTML, CSS, JS	Фронтенд-розробка, макети	https://codepen.io
JSFiddle	HTML, CSS, JS	Тестування фрагментів JavaScript	https://jsfiddle.net
CodeSandbox	React, Vue, TypeScript	Професійна веб-розробка, SPA	https://codesandbox.io
PlayCode	HTML, CSS, JS	Навчання, початківці	https://playcode.io
W3Schools TryIt	HTML, CSS, JS	Інтерактивні приклади, довідник	https://w3schools.com

Онлайн-компілятори – це вебплатформи, які дають змогу писати, запускати, тестувати та поширювати програмний код без необхідності встановлення ІСР на комп’ютер (див. підрозділ 2.1), а саме:

- *доступність*: робота безпосередньо у веббраузері, без потреби додаткового встановлення ПЗ;
- *швидкий старт*: є можливість відразу писати код – зручно для демонстрацій і практики;
- *зворотний зв’язок*: можливість миттєво отримувати результати;
- *співпраця*: можливість ділитися програмним кодом з викладачем або студентами;
- *кросплатформеність*: здатність ПЗ коректно функціонувати на різних операційних системах і платформах без необхідності суттєвої модифікації програмного коду.

Типові сценарії використання онлайн-компіляторів при вивченні програмування:

- *Вивчення синтаксису*: студенти мають можливість швидко перевірити, як працює певна конструкція мови (умова, цикл, функція).
- *Практичні завдання*: викладач задає вправу, а студенти виконують її в онлайн-компіляторі (наприклад, Replit) і надсилають посилання на результат.
- *Демонстрація коду під час лекції* : викладач демонструє код у реальному часі, змінює його і показує результат.
- *Проектне навчання*: студенти можуть працювати над міні-проєктами в командах (наприклад, CodeSandbox для React) онлайн.
- *Домашнє завдання з автоперевіркою*: деякі платформи (наприклад, JSFiddle) дозволяють перевіряти, чи працює вебкод правильно.

Розглянемо найбільш поширені онлайн-компілятори, що використовуються при вивченні програмування в ЗВО.

Replit (від "read-eval-print loop")[73] – це *інтегроване середовище розробки (ICP)*, яке дозволяє програмувати прямо в браузері без необхідності встановлювати середовище на свій комп'ютер.

Replit надає *можливість*:

- писати код на понад 50 мовах програмування (Python, JavaScript, C++, Java, HTML/CSS/JS, Bash, тощо);
- запускати програми;
- співпрацювати з іншими користувачами в реальному часі (як Google Docs для коду) ;
- розгортати вебзастосунки (навіть сервери) ;
- використовувати AI-помічника (Replit Ghostwriter) ;
- створювати боти, ігри, API, бази даних тощо.

Основні функції Replit:

- *Онлайн-середовище розробки*: Повноцінний редактор коду в браузері, з підсвіткою синтаксису, автодоповненням, налагоджувачем.
- *Консоль та термінал*: вивід результатів, термінал Linux, підтримка Bash-команд;

- Бібліотеки: є можливість встановлювати пакети через *pip* (для Python), *npm* (для JS) тощо.
- Співпраця (multiplayer): є можливість писати код одночасно з іншими студентами.
- Хостинг: розгортає проєкти як вебсайти чи сервери 24/7.
- Ghostwriter (AI): помічник, який пише код, пояснює помилки, пропонує рішення. (коштовно).
- Проєкти (Repls): кожен проєкт – окрема “пісочниця” з файлами, терміналом та середовищем.
- База даних: вбудована Replit DB для зберігання даних. Підходить для простих застосунків.

ICP Replit має свої переваги і недоліки. *Переваги Replit:*

- Доступність з будь-якого пристрою – потрібен лише браузер.
- Не потрібно встановлювати ПЗ – (все працює онлайн).
- Кросплатформеність – (підтримує майже 50 мов програмування).
- Ідеально для навчання – особливо для новачків.
- Швидкий старт – буквально 2 кліки для початку написання коду.
- Підтримка GitHub – можна імпортувати репозиторії.
- Безкоштовний тариф – з базовим функціоналом.
- Командна робота – в реальному часі.

Недоліки Replit:

- Обмежені ресурси на безкоштовному тарифі: менше оперативної пам’яті, обмежений час роботи проєкту (sleep mode), немає постійного хостингу (на free-акаунтах проєкт "засинає" без активності).
- Коштовний Ghostwriter – AI доступний тільки на коштовному тарифі.
- Не ідеально для великих проєктів – підходить радше для навчання, прототипів і MVP.
- Менш гнучке середовище, ніж локальне ICP типу VSCode або PyCharm.

– Може бути повільним, особливо на складних проєктах або при повільному інтернеті.

JSFiddle – це одна з найпопулярніших онлайн-платформ для швидкого тестування HTML, CSS та JavaScript-кодів, який дозволяє:

- писати HTML, CSS та JavaScript у браузері,
- миттєво переглядати результат роботи (Live Preview),
- ділитися кодом через посилання (без реєстрації),
- імпортувати бібліотеки (наприклад, jQuery, Vue, React),
- тестувати ідеї, демонструвати баги, або створювати приклади.

При відкритті JSFiddle, на екрані відображається головне вікно, що містить інтерфейс із чотирма основними секціями:

- *HTML* – призначена для створення структури та розмітки вебсторінки ;
- *CSS* – використовується для стилізації та оформлення елементів вебсторінки;
- *JavaScript* – забезпечує реалізацію логіки поведінки вебзастосунку (можливість вибору Vanilla JS, Babel або TypeScript);
- *Result* – відображає результат виконання коду в режимі реального часу.

Основні функції Replit:

- *Live Preview*: Автоматично (або вручну) відображає результат.
- *Title / Description*: Можна дати назву та короткий опис.
- *Save / Fork*: Зберегти власну версію або створити копію чужого проєкту (форк).
- *Frameworks & Extension*: Підключити зовнішні бібліотеки: jQuery, Vue, React, Bootstrap тощо.
- *Settings*: Налаштування рендерингу, скриптів, вкладень, інтерпретаторів (Babel, TypeScript).

– *Collaborate (обмежено)*: Спільна робота в реальному часі (тільки через сторонні сервіси).

Приклад роботи у середовищі JSFiddle.net за авторською методикою представлено в Додатку Е.

Наступний складник веборієнтованих технологій у процесі навчання є штучний інтелект.

У сучасному світі програмування III стає основною навичкою для багатьох професій і потребує сучасного підходу до організації навчання мов програмування студентів ЗВО, оскільки традиційні методи навчання не завжди можуть забезпечити необхідний рівень підготовки студентів в умовах швидкого розвитку технологій. Водночас використання III в навчанні програмування є актуальним з кількох причин: підвищення конкурентоспроможності майбутніх бакалаврів комп'ютерних наук на сучасному ринку праці; впровадження інноваційних підходів в освітній процес, що сприяють зростанню навчальної мотивації студентів; підвищення ефективності навчання програмування завдяки персоналізації освітньої траєкторії, оперативному зворотному зв'язку та підтримці самостійної діяльності.

Штучний інтелект (III) все більше впроваджується в різні галузі, зокрема в освіту. Особливого значення набуває його використання у навчанні програмування в ЗВО. В умовах стрімкого розвитку інформаційних технологій зростає потреба у фахівцях, які володіють сучасними мовами програмування та здатні ефективно використовувати цифрові інструменти у професійній діяльності. Нині існує значна кількість прикладів застосування III в освітньому процесі. Такі платформи, як Codecademy (<https://www.codecademy.com>), Coursera (<https://www.coursera.org>) та edX (<https://www.edx.org/>), що забезпечують самоосвіту студентів, активно використовують алгоритми III для адаптації навчальних матеріалів під рівень знань студентів. Вони аналізують прогрес кожного користувача, надають

індивідуальні рекомендації та корегують навчальну програму в режимі реального часу.

Незважаючи на те, що технології ШІ розвиваються протягом тривалого часу, їх масове впровадження в освітню діяльність суттєво прискорилося після поширення генеративних моделей ШІ. У зв'язку з цим міжнародною, європейською та вітчизняною науково-освітньою спільнотою вже напрацьовано низку нормативних документів і рекомендацій щодо використання технологій штучного інтелекту в освіті як на міжнародному, так і національному рівні так і на рівні окремих закладів вищої освіти.

Так на міжнародному рівні Генеральна Асамблея ООН 21 березня 2024 року ухвалила знакову резолюцію щодо сприяння «безпечним, захищеним і надійним» системам ШІ, які також мають сприяти сталому розвитку для всіх [70]. Проект резолюції щодо ШІ розміщений в інтернеті для обговорення [71]. Хоча резолюція не має обов'язкової юридичної сили, її підтримали понад 120 країн, зокрема Китай, а також вона була схвалена без голосування всіма 193 державами-членами ООН. Віце-президент США Камала Гарріс високо оцінила ухвалення цього документа, заявивши, що ця «резолюція, ініційована США та співавторами понад 100 країн є історичним кроком до встановлення чітких міжнародних норм для штучного інтелекту та розвитку безпечних, захищених і надійних систем штучного інтелекту» [72].

Якщо розглядати ШІ в Україні, то в цій галузі було схвалено Концепцію розвитку ШІ. Цією Концепцією визначаються мета, принципи та завдання розвитку технологій штучного інтелекту в Україні як одного з пріоритетних напрямів у сфері науково-технологічних досліджень [73].

В національному технічному університеті «Київський політехнічний інститут імені Ігоря Сікорського», одним із перших було затверджено та введено в дію наказом від 29.12.2023 р. № НОН/393/2023 документ «Політика використання штучного інтелекту для академічної діяльності в КПІ ім. Ігоря Сікорського». Відповідно до цього документу університет підтримує раціональне використання генеративних інструментів ШІ за умови

дотримання ключових вимог, зокрема щодо інформаційної безпеки, конфіденційності даних, дотримання авторського права та принципів академічної доброчесності [74]. Крім того, університет забезпечує підтримку науково-педагогічних працівників і здобувачів вищої освіти у підвищенні рівня обізнаності та цифрової грамотності щодо використання ІІІ в освітньому процесі.

Таким чином, впровадження ІІІ в освіту на засадах дотримання нормативних документів має стати стимулом викладачам до активного впровадження інновацій, а у викладанні мов програмування має відкрити нові можливості для індивідуалізації та підвищення ефективності та якості навчання. Однак, це також ставить перед університетами низку викликів, які потребують системного підходу та комплексних рішень, зокрема готовності викладачів і студентів до його використання.

Останні дослідження у сфері використання ІІІ в освіті показали, що його застосування в освітньому процесі ЗВО є доцільним для оцінювання навчальних досягнень студентів, підвищення їх мотивації до навчання, посилення зворотного зв'язку між студентами та викладачами, а також для покращення якості освітніх послуг в ЗВО, зумовленого змінами, що відбуваються в суспільстві.

Деякі дослідження засвідчили, що використання викладачами платформ ІІІ надало їм можливість виконувати різноманітні адміністративні функції, зокрема організовувати більш ефективну перевірку та оцінювання завдань студентів, що сприяє підвищенню якості викладацької діяльності. Оскільки платформи ІІІ базуються на методах машинного навчання та адаптивних алгоритмах, навчальні програми та зміст дисциплін можуть бути налаштовані та персоналізовані відповідно до потреб студентів. Це сприяє більш якісному засвоєнню та запам'ятовуванню навчального матеріалу що, у свою чергу, покращує навчальні досягнення студентів і загальну якість освітнього процесу [75], [76], [77], [78].

Впровадження технологій III в систему вищої освіти та чат-ботів сприятиме не тільки підвищенню залученості студентів до процесу навчання, збільшуючи їх навчальні можливості [79], а й сприятиме якісному управлінню процесами навчання в ЗВО, формуючи комплекс інструментів для організації навчання та оцінювання навчальних досягнень [80].

Одним із світових трендів системи вищої освіти є мобільність студентів і бажання отримати високоякісні знання у провідних закладах освіти, які організовують процес онлайн-навчання для іноземних студентів. Для вирішення багатьох питань організації навчання таких студентів доцільно використовувати III для 1) організації персоналізованого навчання із створеним та налаштованим навчальним контентом, який відповідає здібностям студента; 2) адаптивне тестування дозволить зрозуміти здібності студентів і покращити зворотний зв'язок; 3) прогностична або навчальна аналітика може показати як викладачі розуміють і підтримують академічну успішність студентів; 4) аналітика для визначення факторів, які сприяють успіху іноземних студентів [81].

Основною парадигмою XXI століття є навчання впродовж життя, одним із шляхів реалізації якої є інтеграція технологій III в освітній процес, побудови відповідної моделі організації навчання в системі вищої освіти на його основі, складовими якої можна вважати [82]:

- створення умов для когнітивного навчання студентів;
- підтримка процесу навчання, за умови, що студенти працюють у співавторстві зі III;
- розширення можливостей організації процесу навчання, що створює умови свободи дій для студентів.

Для реалізації зазначених складових необхідно [83]:

- адаптувати навчальний матеріал під процес навчання для досягнення вищих результатів навчання;
- в процесі навчання застосовувати інноваційні методи навчання та технології оцінювання рівнів навчальних досягнень студентів. Оцінювання

результатів навчання на основі використання ШІ може здійснюватися на етапах формуючого та підсумкового оцінювань, самооцінювання та взаємооцінювання, а отримані результати надають можливість здійснювати аналіз ефективності навчального процесу;

- змінити підхід до організації спілкування між студентом і викладачем через зворотній зв'язок для корегування навчання, викладання та подолання навчальних втрат.

В дослідженні, проведеному у Сполучених Штатах Америки, метою якого було вивчення впливу оцінювання та зворотного зв'язку на результати та успішність студентів у системі вищої освіти, було обрано чотири платформи штучного інтелекту: I-FCN, ANN, XG Boost та SVM. Автори дослідження виявили позитивні результати різних практик оцінювання та зворотного зв'язку, які можуть покращити навчальний досвід і результати студентів. А найкращою платформою, на їх думку, є I-FCN, яка узагальнює більшість методів штучного інтелекту, машинного навчання та аналітики навчання для оцінювання та надання якісного та інтелектуального зворотного зв'язку студентам [84].

Незважаючи на позитивні результати впровадження ШІ для оцінювання навчальних досягнень студентів та організації зворотного зв'язку, слід відзначити і деякі проблеми:

- студенти ЗВО не завжди розуміють переваги такого оцінювання над традиційним і не сприймають його позитивно [85], що може бути обумовлене не адаптованістю закладу освіти до інновацій;

- наукові дослідження, проведені шляхом опитування молоді у світі та в Україні показали, що в Україні молодь не володіє інформацією про сутність штучного інтелекту, про можливі його переваги та недоліки, в них не сформовані практичні навички використання ШІ в проблемних життєвих ситуаціях, вирішення яких можливе за умови залучення додаткових інформаційних ресурсів для вирішення питань; засвідчена відсутність теоретичних знань про можливості ШІ для його застосування в освітньому

процесі, зокрема використання систем пошуку, оброблення та представлення результатів навчально-пізнавальної діяльності здобувачів освіти [86].

Як було зазначено вище, не зважаючи на те, що ІІІ став доступним з 2023 року, перші спроби введення ІІІ до системи навчання студентів відбулися у 2018-2019 навчальному році [85] з вибіркою із 130 студентів різних факультетів Туреччині і полягало лише у виявленні стану готовності студентів використовувати ІІІ в процесі навчання. Дослідження показало, що в цілому студенти мало обізнані про ІІІ і в більшості мають негативне ставлення до таких технологій.

Пізніше в університетах світу почали з'являтися різноманітні курси, які спочатку були спрямовані на підвищення загальної грамотності використання ІІІ: візуальна, медійна, комп'ютерна, цифрова грамотності та грамотність в інформаційно-комунікаційних технологіях та технологіях штучного інтелекту, а пізніше для впровадження ІІІ в навчальні курси з програмування.

В роботі [87] описано експериментальне дослідження, проведене в університетах Гонконгу. Одним із досліджуваних питань було формування вміння розвинути концептуальне розуміння штучного інтелекту шляхом його впровадження в навчальний курс з програмування. Даний курс охоплював такі основні поняття як штучний інтелект, машинне навчання, контрольоване та неконтрольоване навчання. Результати дослідження засвідчили, що учасники обох статей із різним початковим освітнім рівнем можуть розуміти концепції машинного навчання, контрольованого навчання, регресії, класифікації, неконтрольованого навчання та кластеризації.

Проведене фінальне тестування показало, що курс надав учасникам можливість досягти статистично значущих покращень у розумінні концепцій штучного інтелекту, у самосвідомій грамотності та розширенні можливостей ІІІ. Крім того, курс успішно зменшив гендерний розрив щодо грамотності та розширення можливостей ІІІ. Тестування також показало, що попередні знання з програмування не були передумовою розвитку грамотності зі ІІІ.

Учасники повідомили про позитивне сприйняття процесу навчання та його результатів з точки зору підвищення їхньої грамотності ШІ [87].

Грунтуючись на результати дослідження [88], в яких було виокремлено вісім компонентів використання ШІ для освіти: інтелектуальна система навчання, оцінювання студентів, рух студентів, настрої, рекомендації, моніторинг групи, персоналізоване навчання, прогнозування успішності, автори [88] виокремили на кожен критерій шість підпунктів три з яких включали позитивне мислення, а три – негативне. Дані критерії було узгоджено з точки зору психології і надавали можливість підійти індивідуально до кожного студента. Така шкала дослідження отримала назву «Уявлення студентів про ШІ в освіті» (SCAIES).

Результати дослідження показали, що інтелектуальні системи навчання створюють умови для якісної підготовки студентів, дозволяють отримувати зворотній зв'язок для подолання розривів у навчанні, надають студентам можливість отримувати індивідуальний підхід відповідно до його навчальних досягнень. В той же час є і ряд не затребуваних критеріїв, таких як моніторинг групи, а також використання чат-ботів у процесі навчання, як таких, що не були ще представлені на той момент на загал. Не можна відкидати і частину студентів, що мають негативне ставлення до ШІ у зв'язку з можливістю порушень правил академічної доброчесності.

Дослідження з питання використання чат-ботів в системі вищої освіти Китаю було проведено у 2023 році показали, що в цілому таке застосування ШІ має позитивний вплив на навчальні досягнення студентів [89]. Але ще більш цікавим виявилось дослідження впровадження роботів у процес навчання програмуванню. Результати дослідження показують, що студенти університетів мають високу тенденцію використовувати роботів на основі ШІ для навчальних цілей, налаштовані вдосконалювати свої навички та бути в курсі останніх розробок у технологіях штучного інтелекту.

Роботи на основі ШІ пропонують винятковий інструмент навчання як для студентів, так і для викладачів, відтворюючи досвід глибокого навчання

для вивчення предметів у захоплюючий спосіб. Це дає студентам шанс досліджувати та здобувати нові знання, вибудовуючи персональну траєкторію навчання, оскільки використання роботів в системі освіти надають можливість створити безпечне та привітне навчальне середовище, в якому студенти почуваються невимушено [90].

Аналізуючи публікації дослідників ІІІ було виявлено низку проблем, що уповільнюють впровадження інноваційних технологій в освітню практику викладачів закладів вищої освіти, зокрема необхідність постійної адаптації освітніх програм, недостатня гнучкість закладів вищої освіти до впровадження інновацій.

Основною проблемою є адаптація ІІІ для ефективного навчання студентів з різним рівнем підготовки та стилями навчання. Студенти, які вивчають мови програмування, мають різний рівень знань та навичок. ІІІ може допомогти в цьому питанні, використовуючи адаптивні алгоритми для налаштування навчальних матеріалів відповідно до потреб кожного студента. Однак розробка таких алгоритмів є складною задачею, яка вимагає глибокого розуміння педагогіки та методів програмування навчання [90].

Ще одна проблема виникла у створенні ефективних інтерактивних інструментів для навчання. ІІІ може бути використаний для розробки інтерактивних середовищ, де студенти можуть писати код, отримувати миттєві відгуки та бачити результати своїх дій у реальному часі. Проте, створення таких середовищ вимагає значних ресурсів та технологічної експертизи. Крім того, необхідно забезпечити, щоб ці інструменти були інтуїтивно зрозумілими та доступними для студентів з різним рівнем підготовки та рівнем навчання.

Значним викликом є також забезпечення якості та актуальності навчальних матеріалів. ІІІ може аналізувати великий обсяг даних та автоматично оновлювати навчальні програми, але це потребує постійного моніторингу та коригування з боку викладачів. Важливо, щоб навчальні матеріали відповідали сучасним стандартам та вимогам промисловості.

Існує також проблема етичного характеру. Використання ІІІ в навчанні вимагає збереження конфіденційності даних студентів та забезпечення їх безпеки. Це включає захист персональних даних та запобігання неправомірному використанню особистої інформації [90].

Важливим аспектом є також інтеграція ІІІ в існуючу навчальну інфраструктуру. Це потребує відповідної підготовки викладачів, які мають розуміти, як ефективно використовувати ІІІ у своєму навчальному процесі. Необхідні також інвестиції в технологічну інфраструктуру та постійна підтримка

Отже, наукова проблема використання ІІІ в навчанні студентів мовам програмування є багатогранною і вимагає комплексного підходу. Вирішення цих викликів може суттєво покращити якість освіти та підготовку майбутніх фахівців у галузі програмування.

У цьому дослідженні ми розглядаємо такі аспекти, як: готовність студентів до використання ІІІ в навчанні мов програмування; ставлення студентів до інноваційного інструменту; рівень використання студентами ІІІ під час виконання практичних завдань; обмеження щодо використання ІІІ; формування переліку завдань, які можна виконувати з використанням ІІІ.

У дослідженні представлено результати аналізу стану використання ІІІ студентами ЗВО та їх готовності до використання ІІІ в навчанні мовам програмування. В опитуванні взяли участь 281 студент 1-4 курсу НТУУ «Київський політехнічний інститут імені Ігоря Сікорського» факультету Інформатики та обчислювальної техніки кафедри інформаційних систем та технологій. У процесі дослідження було застосовано моніторинг виконання практичних завдань студентами ЗВО; розроблено анкету з метою з'ясування стану використання та готовності до навчання мовам програмування з використанням ІІІ [90].

Опитування проводилося протягом березня 2024 року. Студенти не були залежні від результатів іспитів чи заліків. На анкету відповідали онлайн в зручний для них час.

В опитуванні взяли участь студенти 1 курсу – 41,3%, студенти 2-курсу – 21,0%, студенти 3 курсу – 31,3%, та незначна кількість студентів 4 курсу – 6,0% (рис. 3.14).

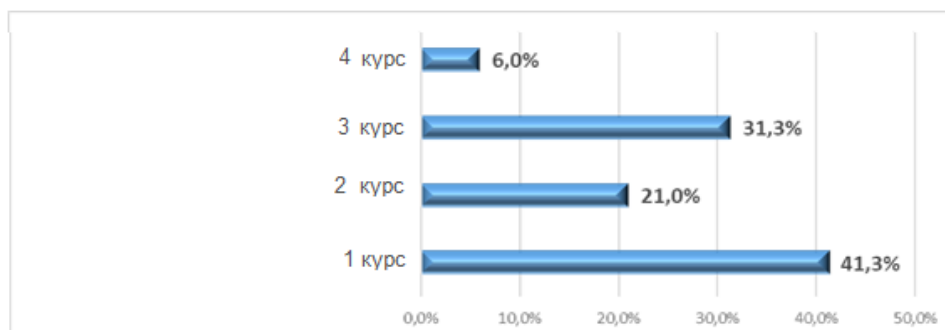


Рис. 3.14. Кількісний склад студентів, які пройшли тестування [90].

Результати опитування також показали, що найактивнішими є студенти віком від 16 до 19 років – 67,3%, менш активними студенти 20-24 років – 24,2%. У порівнянні з 25-29-річними (2,1%) та старшими за 30 років більш активними виявилися останні. Їх відсоток склав – 6,4% (рис. 3.15).

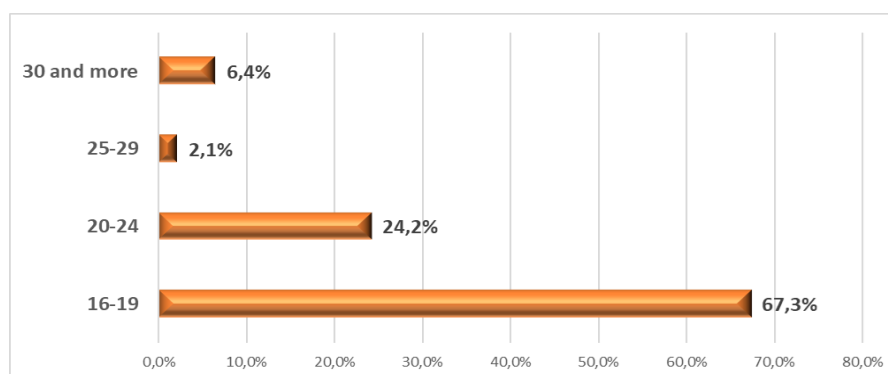


Рис. 3.15. Віковий склад студентів [90].

За результатами опитування найбільш активними, а отже, і більш вмотивованими для сприймання інновацій виявилися студенти першого курсу.

Розглянемо кілька аспектів, що впливають на використання ІІІ в навчанні студентів мовам програмування, а саме: готовність студентів до використання інноваційного інструменту, підходи та рішення щодо використання ІІІ у процесі навчання програмування.

Штучний інтелект – це здатність автоматизованої програмної системи виконувати дії чи завдання, які раніше були доступні лише людині: розумне міркування, аналіз і обробка інформації, набуття досвіду та оптимізація. ШІ використовує алгоритми машинного навчання для отримання та обробки інформації і відіграє ключову роль в освітньому процесі.

Використання ШІ в освітньому процесі створює умови для реалізації таких освітніх інновацій [90].

- *Індивідуалізація та персоналізація навчання.* ШІ може адаптувати навчальний матеріал до індивідуальних потреб кожного студента, забезпечуючи більш ефективне навчання, що відповідає їхнім інтересам та рівню знань;
- *Підвищення мотивації студентів до навчання.* Використання ШІ може допомогти студентам у засвоєнні навчального матеріалу, шляхом уведення інтерактивних завдань, віртуальних вчителів та індивідуального підходу до навчання;
- *Прогнозування успішності студентів протягом всього циклу навчання.* ШІ на основі аналізу даних може спрогнозувати ризики в навчанні студентів та запропонувати шляхи їх вирішення;
- *Підвищення якості освіти (навчання, викладання)* шляхом підбору оптимальних навчальних курсів та програм для подолання навчальних розривів;
- *Автоматизація оцінювання навчальної діяльності студентів.* ШІ може автоматично оцінювати завдання, зменшуючи людський фактор та забезпечуючи об'єктивні результати навчання студентів.

Не зважаючи на зазначені переваги використання ШІ в освітньому процесі, слід підкреслити і слабкі сторони [44]: етичні норми – загроза академічної доброчесності, демократизація плагіату; зменшення когнітивних здібностей студентів; соціальна ізоляція – втрата своєї ідентифікації як студента в зв'язку із зменшенням спілкування між однолітками та викладачами; втрата творчості та творчого підходу до розв'язання задач.

У сучасному світі програмування стає основною навичкою для багатьох професій і потребує сучасного підходу до організації процесу навчання програмуванню студентів ЗВО, оскільки традиційні методи навчання не завжди можуть забезпечити необхідний рівень підготовки студентів в умовах швидкого розвитку технологій. Саме тому використання ІІІ в навчанні мовам програмування є актуальним з кількох причин: розвиток ринку праці, інноваційні процеси в освіті, ефективність та економія часу, вплив глобальних трендів.

Розвиток ринку праці. Програмісти з навичками роботи зі ІІІ є більш затребуваними на ринку праці, оскільки можуть розробляти імплементації ІІІ, створювати інтелектуальні системи та оптимізувати процеси.

Інноваційні процеси в освіті. Використання ІІІ в освітньому процесі ЗВО надає можливість впроваджувати нові активні методи навчання, створювати інтерактивні платформи та розширювати навчальні можливості студентів.

Ефективність та економія часу. використання ІІІ надає можливість автоматизувати багато освітніх процесів, що дозволяє вчителям та студентам ефективніше використовувати час [90]..

Вплив глобальних трендів. ІІІ є однією з ключових технологій майбутнього, і використання її в освіті є необхідністю для підготовки конкурентоспроможних майбутніх фахівців.

Використання штучного інтелекту в навчанні студентів закладів вищої освіти мовам програмування охоплює кілька різних галузей. Опишемо більш ґрунтовно кожен із галузей та розглянемо приклади деяких платформ для їх забезпечення

– *Інтелектуальні навчальні системи (ІНС).* ІНС використовують ІІІ для створення персоналізованих навчальних планів, адаптують навчальні матеріали під потреби кожного студента. До таких систем відносять: *Coursera* – платформа, яка використовує алгоритми ІІІ для аналізу прогресу студентів і пропонування відповідних курсів; *edX* – надає персоналізовані рекомендації на основі даних про успішність студентів.

– Автоматизоване оцінювання (АО). ІІІ-системи для автоматичного оцінювання написаних програм можуть аналізувати код на предмет правильності, ефективності та стилю програмування. Наприклад, CodeSignal (<https://codesignal.com>) використовує алгоритми для перевірки коду та аналізу помилок; HackerRank (<https://www.hackerrank.com>) забезпечує автоматичне оцінювання кодових завдань та надає зворотний зв'язок.

– Інтерактивні навчальні платформи (ІНП). Інтерактивні платформи використовують ІІІ для створення симуляцій, віртуальних лабораторій та ігор, що робить навчання більш захоплюючим і ефективним. Такими платформами є: Codecademy (<https://www.codecademy.com>) пропонує інтерактивні вправи з програмування з автоматичним зворотним зв'язком; SoloLearn (<https://www.sololearn.com>) містить інтерактивні курси з підтримкою ІІІ для навчання мовам програмування [90].

– Репетиторські системи (РС). ІІІ-репетитори забезпечують підтримку в режимі реального часу, відповідаючи на питання студентів та пояснюючи складні концепції. Такими системами можуть бути: IBM Watson Tutor (<https://developer.ibm.com>), яка може взаємодіяти зі студентами, надаючи відповіді на їхні запитання та пояснюючи матеріал; Socratic by Google використовує ІІІ для допомоги студентам у вирішенні проблем з програмуванням.

– Аналіз освітніх даних (АОД). ІІІ аналізує великі обсяги освітніх даних для виявлення тенденцій і рекомендацій щодо покращення навчальних програм. Наприклад, BrightBytes (<https://www.brightbytes.net>) збирає та аналізує дані для покращення освітніх стратегій; Knewton (<https://www.wiley.com/en-us/education/alta>) використовує дані для адаптації навчальних програм до індивідуальних потреб студентів.

– Мобільні застосунки для навчання (МЗН). Мобільні додатки використовують ІІІ для надання персоналізованих навчальних матеріалів і оцінки прогресу студентів у зручному форматі. Такими додатками є: Mimo (<https://mimo.org>) пропонує інтерактивне навчання програмуванню з

використанням ІІІ; Grasshopper застосунок, що допомагає вивчати основи програмування за допомогою інтерактивних уроків (<https://rhino3d.online/uk/product/grasshopper>) [90]..

– Віртуальні лабораторії (ВЛ). Віртуальні лабораторії надають студентам можливість виконувати практичні завдання та експерименти у віртуальному середовищі, що керується ІІІ: AWS Educate (<https://aws.amazon.com/ru/education/awseducate>) надає доступ до хмарних лабораторій, де студенти можуть практикувати навички програмування; Google Cloud Platform (<https://cloud.google.com>) пропонує лабораторії для навчання і роботи з реальними хмарними інструментами .

– Штучний інтелект у створенні навчальних матеріалів. ІІІ використовують для автоматичного створення навчальних матеріалів, таких як підручники, відеоуроки та тести. Навчальні матеріали можна створити за допомогою: Content Technologies, який використовує ІІІ для створення персоналізованих підручників; Querium, що генерує адаптивні тести та навчальні матеріали на основі аналізу даних про студентів.

Перераховані галузі демонструють широкий спектр можливостей для використання ІІІ в навчанні мовам програмування, кожна з яких спрямована на підвищення ефективності та якості освіти [90].

Попри численні переваги, використання ІІІ у навчанні мовам програмування має і свої обмеження та особливості:

– *Технічні обмеження:* Алгоритми ІІІ поки що не завжди можуть точно інтерпретувати креативні або нестандартні рішення, що може призвести до неправильних оцінок.

– *Етичні питання:* Використання ІІІ передбачає збір і аналіз великої кількості даних про студентів, що може викликати питання конфіденційності та безпеки даних.

– *Вартість впровадження:* Розробка та впровадження ІІІ-систем в освітній процес вимагає значних фінансових інвестицій, що не завжди доступно для всіх закладів освіти [90].

Враховуючи всі переваги та недоліки використання ШІ в навчанні студентів програмуванню виокремимо в структурі організації такого навчання основні складники: вступ до програмування зі ШІ; ШІ як партнер для ідеї; навігація алгоритмами з допомогою ШІ; ШІ як партнер для налагодження програми (рис. 3.16).



Рис. 3.16. Структура навчання студентів програмування з використанням ШІ [90].

Для дослідження впливу штучного інтелекту на освітню діяльність студентів, були проаналізовані завдання, що пропонуються студентам під час навчання теоретичного та практичного змісту. Також було виокремлені завдання, які можна виконати з використанням або підтримкою штучного інтелекту:

- пошук теоретичних відомостей;
- написання первинного коду;
- перевірка правильності написання коду;
- оптимізація коду;
- генерація тестових випадків;
- автоматичне тестування програмного забезпечення;
- статистичний аналіз коду для виявлення помилок або недоліків;
- написання документації до коду;
- рефакторинг коду;

- визначення ефективності алгоритмів;
- прогнозування продуктивності програмного забезпечення;
- автоматична генерація звітів про помилки;
- застосування машинного навчання для розв’язання специфічних завдань;

- побудова і тренування моделей машинного навчання;
- аналіз та обробка великих даних;
- розробка чат-ботів або інших інтелектуальних агентів;
- створення рекомендаційних систем;
- візуалізація даних;
- аналіз поведінки користувачів;
- розробка систем розпізнавання образів або мовлення;
- робота з природною мовою (NLP);
- автоматичне виправлення помилок у коді [90];
- інтеграція штучного інтелекту в існуючі програмні продукти;
- моніторинг і аналіз роботи серверів або мереж;
- розробка ігор з використанням алгоритмів штучного інтелекту;
- застосування AI для кібербезпеки;
- автоматизація рутинних завдань програмування;
- розпізнавання та обробка зображень;
- проєктування та симуляція систем;
- генерація ідеї для нових проєктів або продуктів.

Кожен студент ЗВО повинен прагнути бути конкурентоспроможним на світовому ринку праці, а для цього необхідно завжди підвищувати свою кваліфікацію шляхом удосконалення знань та умінь.

Для підвищення власних знань та удосконалення умінь, студенти, які навчаються програмуванню, повинні використовувати [90].:

1. *Мови програмування:*

- Python: популярна мова для розробки веб-додатків, наукових обчислень, штучного інтелекту та багатьох інших сфер;
 - JavaScript: використовується для розробки зовнішнього інтерфейсу веб-додатків та взаємодії з користувачем;
 - Java: застосовується в багатьох великих корпоративних проєктах та мобільних додатках;
 - C++: використовується для системного програмування, графіки та ігор;
 - C# - використовується для створення динамічних веб-застосунків на основі потужного фреймворку ASP.NET від Microsoft
- 2. Фреймворки та бібліотеки:
- Django: Python-фреймворк для розробки веб-додатків;
 - React: JavaScript-бібліотека для розробки зовнішнього інтерфейсу;
 - Angular – фреймворк JavaScript для розробки складних та високонавантажених веб-систем;
 - Vue.js – фреймворк JavaScript для розробки користувацьких інтерфейсів;
 - Spring: Java-фреймворк для побудови корпоративних додатків.
3. Інструменти розробки:
- Visual Studio Code (VS Code): популярний текстовий редактор для програмістів;
 - Git: система контролю версій для спільної роботи над програмним кодом;
 - Jupyter Notebook: інтерактивне середовище для аналізу даних та наукових обчислень.
4. Інші технології:
- Docker: контейнеризація додатків для зручного розгортання;
 - SQL та NoSQL бази даних: наприклад, MySQL, PostgreSQL, MongoDB.
 - Штучний інтелект та машинне навчання: TensorFlow, PyTorch, Scikit-learn;

- API технології: REST (JSON, XML)- Архітектурний стиль для створення веб-сервісів.

Для організації ефективної роботи зі штучним інтелектом, студентам-програмістам також варто ознайомитися з наступними програмами та курсами, що нададуть їм можливість поглибити свої знання та навички:

- «IBM Introduction to Artificial Intelligence (AI)» – цей курс надає введення в алгоритми, прикладне машинне навчання, нейронні мережі, комп'ютерне зору та інші ключові аспекти ШІ (Coursera);
- «AI For Everyone» – цей курс допоможе зрозуміти основи даних, глибокого навчання та бізнес-трансформації (DeepLearning.AI);
- «Explore free artificial intelligence courses and more» – цей курс зі штучного інтелекту та інші можна знайти edX;
- «Machine Learning, Computational Thinking, Deepfakes» – онлайн-курси, які допоможуть розширити знання зі штучного інтелекту (MIT OpenCourseWare).
- «AI For Business» – цей курс охоплює машинне навчання, аналіз даних, алгоритми та ін. (University of Pennsylvania) [90]..

Як зазначають учені та викладачі-практики, ШІ може значно покращити якість і ефективність навчання, персоналізувати підхід до кожного студента та зробити процес засвоєння матеріалу більш інтерактивним і цікавим. Вже існують такі інструменти, як CodeSignal, LeetCode та HackerRank, що використовують ШІ для автоматичного оцінювання написаних програм, аналізу помилок і надання зворотного зв'язку. Це надає студентам можливість швидше зрозуміти свої помилки та вдосконалювати свої навички програмування.

CodeSignal, LeetCode та HackerRank – це онлайн-платформи для практики кодування та підготовки до співбесід з технічного працевлаштування. В результаті порівняльного аналізу було визначено функціональні критерії зазначених платформ (табл. 3.10).

До спільних рис зазначених платформ відносять:

– *кодування*: усі три платформи пропонують широкий спектр задач з кодування, що охоплюють різні теми, різні рівні та мови програмування. Такий широкий спектр задач створює умови для відпрацювання практичних навичок з кодування та надає можливість вивчати нові алгоритми та концепції;

– *підтримка декількох мов програмування*: C++, Java, Python, JavaScript та багато інших;

– *інтерактивний інтерфейс*: усі три платформи мають інтерактивний інтерфейс, який надає можливість писати та запускати код прямо в браузері, що економить час і робить процес вирішення задач більш зручним;

Таблиця 3.10

Онлайн-платформи для кодування з підтримкою ШІ

	CodeSignal	LeetCode	HackerRank
Критерій	https://codesignal.com/	https://leetcode.com/	https://www.hackerrank.com/
Фокус	підготовка до співбесід	підготовка до співбесід	конкурентне програмування, навчання
Складність задач	висока	висока	середня
Задачі	оригінальні, різнопланові за складністю та оновлені	широкий спектр завдань, включаючи ті, що використовуються у реальних компаніях	різноманітні задачі, включаючи тематичні конкурси
Інтерфейс користувача	сучасний, зручний, простий та інтуїтивно зрозумілий	більш складний, але з більшою кількістю функцій	простий, сучасний та елегантний
Ціна	безкоштовний та платний плани	тільки платний план	безкоштовний та платний плани
Організація навчання	інтерактивні курси та підказки	детальні пояснення та статті	навчальні посібники, відео та сертифікації
Спільнота	менша, але активна	велика та активна	велика та глобальна
Переваги	автоматичне генерування тестів,	велика кількість задач, активна спільнота	широкий спектр задач, конкурси, сертифікації

	детальний аналіз результатів		
Недоліки	менш широкий спектр задач, ніж у HackerRank	менш зручний інтерфейс, ніж у CodeSignal	задачі можуть бути не такими складними, як на CodeSignal та LeetCode
Функціонал	готуватися до співбесід з технічного працевлаштування	готуватися до співбесід з технічного працевлаштування, практикувати кодування	практикувати кодування, брати участь у конкурсах, отримувати сертифікати
Додаткові функції	автоматизоване тестування; візуалізація виконання коду; режими навчання та практики	дискусійні форуми; змагання; можливість створення власних завдань	код-редактор; інструменти для командної роботи; можливість розміщення власних конкурсів
Можливості вибору	хороший вибір для початківців, які хочуть покращити свої навички кодування та підготуватися до співбесіди	підходить для досвідчених розробників, які хочуть практикувати вирішення складних завдань та змагатися з іншими	відмінний варіант для тих, хто шукає комплексне рішення для навчання, практики та сертифікації

– *система оцінювання* вирішених задач відбувається автоматично, що допомагає відстежувати прогрес і визначати області, які потребують покращення;

– *спільнота*: всі три платформи мають активні спільноти користувачів, де можна спілкуватися з іншими програмістами, ділитися знаннями та отримувати допомогу у вирішенні задач [90]..

Отже, кожна із наведених платформ допоможе у вирішенні конкретних задач, тому вибір більш зручної і функціональної є суто персоналізованим.

Впровадження штучного інтелекту (ШІ) у сферу програмування знаменує собою нову еру в розробці програмного забезпечення. ШІ відкриває перед студентами безліч можливостей для автоматизації завдань, покращення

продуктивності та створення більш інноваційних рішень. Однак, для ефективного використання ШІ у програмуванні, студенти повинні володіти не лише знаннями мов програмування, але й розумінням принципів роботи ШІ, а також навичками інтеграції ШІ у свої проєкти.

У цьому розділі ми представили результати дослідження за такими напрямками: готовність студентів до використання ШІ та проаналізували фактори, що впливають на цю готовність, такі як знання про ШІ, навички програмування з ШІ, сприйняття ШІ як асистента для розробки програмного забезпечення.

У відповідь на питання «Чи маєте досвід використання штучного інтелекту?» понад 45% студенти використовують рішення, запропоновані штучним інтелектом; 27% використовують епізодично і майже 23% використовують готові рішення на професійній основі (рис. 3.17) [90]..

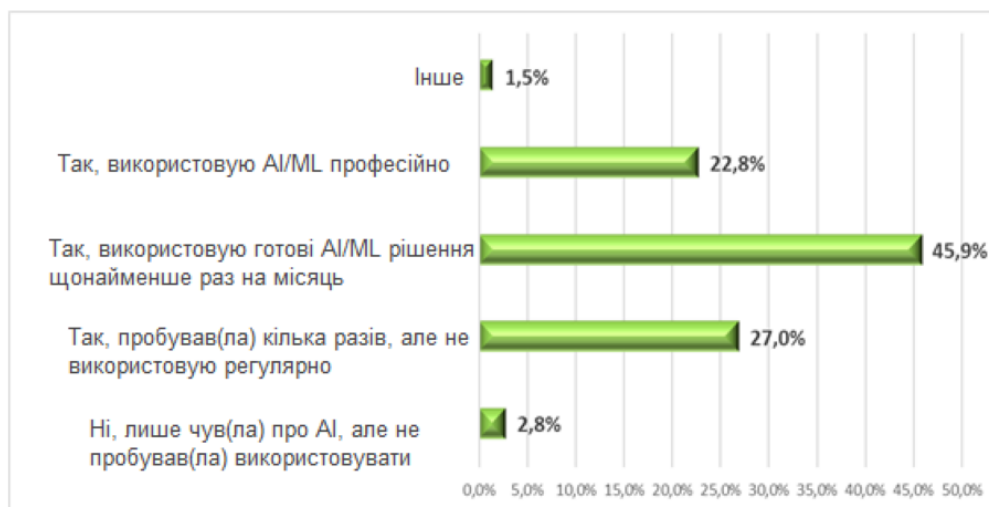


Рис. 3.17. Досвід використання ШІ студентами [90].

На питання «Чи відчуваєте Ви, що частота звернень до штучного інтелекту зростає?» так відповіли – 40,4%, скоріш так – 43,3%. Проте 5,3% студентів зазначили, що не відчувають таких змін (9 студентів з 1 курсу, 4 студента з 2 курсу, з 3 та 4 курсів не відчули таких змін по 1 студенту), 11% студентів скоріш не відчувають зростання звернень до ШІ (1 курс – 10 осіб, 2 курс – 8 осіб, 3 курс – 11 осіб і 4 курс – 2 особи) (рис. 3.18).

Аналіз відповідей на питання «Чи відчуваєте Ви, що **під** час роботи зі ШІ у Вас виникають нові питання?» майже 70,8% студентів зазначили, що під час роботи з ШІ вони починають генерувати нові запитання. Запитання, що виникають, стосуються як уточнення самого запиту до ШІ, так і відповідей ШІ – їх деталізації, уточнення та доповнення [90].

Цей аспект роботи з ШІ підтверджує думку авторів, що при роботі зі ШІ відбувається поступове занурення студентів у питання, яке вивчається. Уточнення і деталізація відповідей допомагають студентам зрозуміти суть питання або поглибити свої знання з питань, що вивчаються або досліджуються.

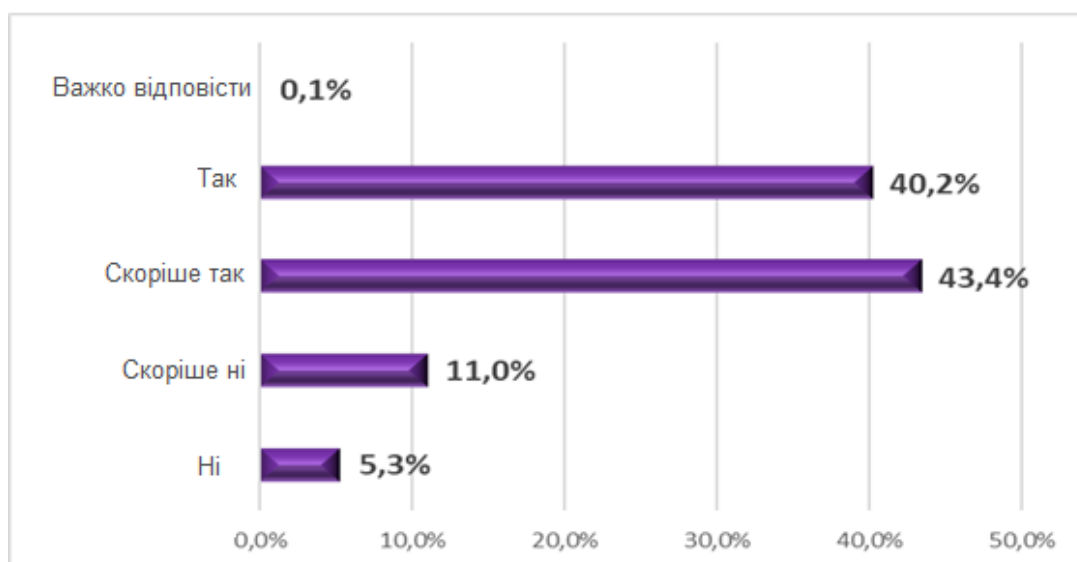


Рис. 3.18. Потреба студентів у зверненнях до ШІ [90].

Аналізуючи відповіді «ні», «скоріш ні» було з'ясовано, що 29,2% студентів задовольняють первинні відповіді ШІ (рис. 3.19).

Використання штучного інтелекту студентами-програмістами в навчанні та виконанні практичних завдань має низку значних переваг:

Покращення продуктивності: автоматизація рутинних завдань – ШІ може автоматизувати багато рутинних і повторюваних завдань, таких як тестування, написання документації або генерування кодів, надаючи

студентам можливість зосередитися на більш творчих і складних аспектах програмування;

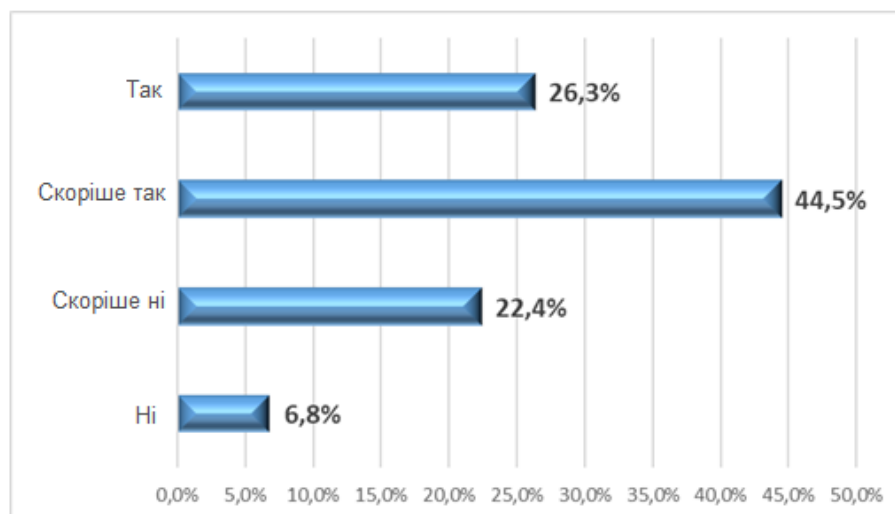


Рис. 3.19. Занурення студентів у роботу з ШІ [90].

Швидкий пошук інформації – ШІ може швидко знаходити необхідні теоретичні відомості або приклади коду, що значно скорочує час, необхідний на дослідження і навчання; *підвищення якості коду* – відбувається шляхом виявлення помилок і недоліків, а інструменти ШІ для статистичного аналізу коду можуть автоматично виявляти помилки, потенційні баги або недоліки в коді, що допомагає студентам писати більш якісний і надійний код; *оптимізація коду* – ШІ може рекомендувати деякі покращення для оптимізації продуктивності коду, що є важливим навиком для програмістів [90].

Індивідуалізація навчання: адаптивне навчання – системи, які використовують ШІ, можуть адаптувати навчальні матеріали та завдання відповідно до рівня знань та темпу навчання студента, що забезпечує більш ефективне і персоналізоване навчання; *інтерактивні помічники* – чат-боти та віртуальні помічники на основі ШІ можуть відповідати на запитання студентів у режимі реального часу, надаючи підтримку і роз'яснення.

Розвиток навичок роботи з новими технологіями: практика з реальними інструментами ШІ – студенти отримують можливість працювати з сучасними інструментами та технологіями ШІ, що є важливим для їхньої майбутньої

кар'єри; *знайомство з передовими методами* – використання ІІІ надає студентам можливість вивчати і застосовувати передові методи та підходи в програмуванні і розробці програмного забезпечення.

Покращення аналітичних і критичних навичок: аналіз великих даних – ІІІ може допомагати студентам аналізувати великі обсяги даних, виявляти закономірності та робити прогнози, що створює умови для розвитку їх аналітичних здібностей; *розв'язання складних завдань* – ІІІ може пропонувати нові підходи до вирішення складних задач, сприяючи розвитку критичного мислення у студентів.

Спрощення процесу тестування: автоматичне генерування тестів – ІІІ може автоматично створювати різні тестові випадки для перевірки коду, що значно полегшує процес тестування і виявлення багів; *автоматичне тестування* – інструменти ІІІ можуть проводити автоматичне тестування коду, що забезпечує швидке і ефективне виявлення та виправлення помилок [90].

Провівши аналіз наведених переваг, логічним постало запитання «Чи відчують студенти потребу звернутися до ІІІ, якщо вони починають розроблення нової програми, генерування тексту чи створення зображення?»

У відповідь на це запитання отримали такі результати: ще не зовсім впевнені в ефективності ІІІ, проте 69,1% студентів все ж звертаються до ІІІ і майже 30% розпочинають роботу самостійно (рис. 3.20).

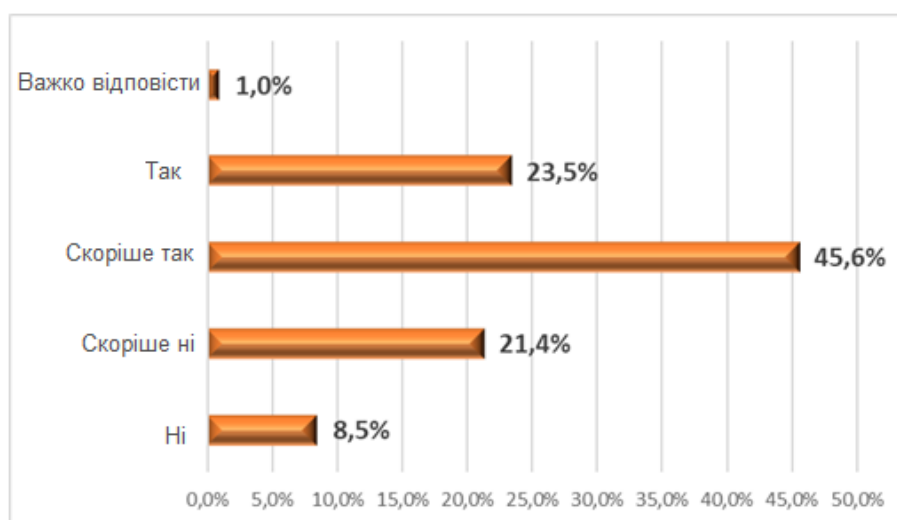


Рис. 3.20. Потреба в ІІІ [90].

Зазначимо, що задачею студентів є створення коду. І саме для виконання такого виду роботи студенти звертаються до ІІІ за потребою (створення первинного коду) – 39,1%, 10% – використовують постійно, один раз на тиждень – 9,6%, використали всього один раз – 11,7%, не використовували – 21%.

Під час деталізації відповідей студентів було виявлено критерії їх звернення до ІІІ: 1) не можуть знайти помилку; 2) створення прикладів, що працюють; 3) для створення структури програми; 4) як джерело інформації; 5) для пошуку рішень тощо (рис. 3.21).

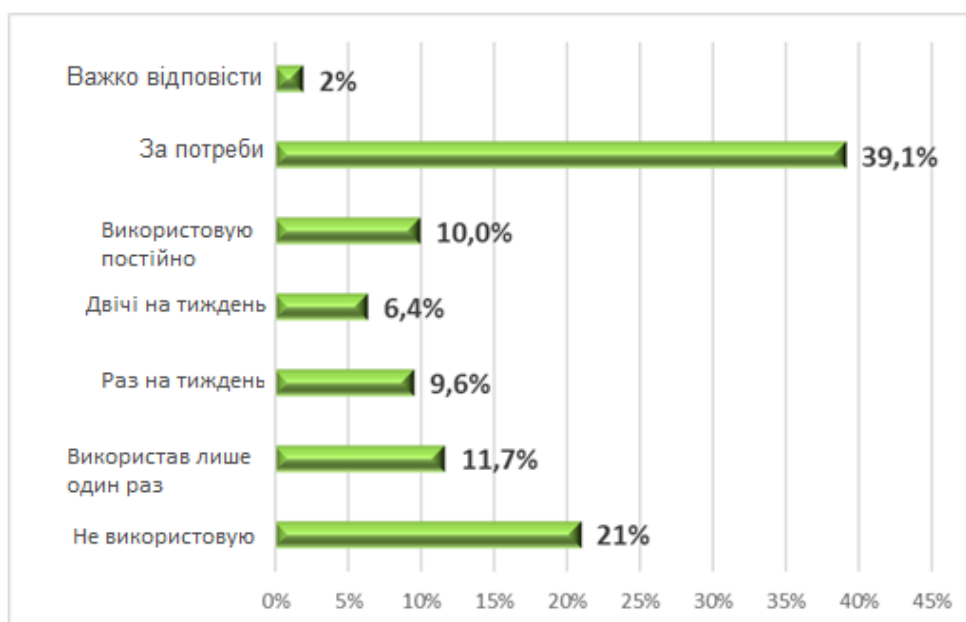


Рис. 3.21. Частота використання ІІІ студентами [90].

Логічним постає запитання про якість коду, згенерованого штучним інтелектом. Як з'ясувалося в процесі дослідження, в первинному варіанті студенти 1 курсу використовували 68,7%, 12,2% здавали комп'ютерні практикуми з допрацьованим кодом, використовували тільки частину коду – 17,8%, 0,7% - не використовували (рис. 3.22).



Рис. 3.22. Використання студентами 1 курсу коду, згенерованого ІІІ [90].

В той час ,як студенти 3-4 курсів використали код тільки 2,5% студентів – тобто, код згенерований ІІІ, студенти здали як виконану роботу. 55,2% – скористалися ІІІ, але доопрацьовували код (вносили правки) самостійно і 12,8% – використали частину коду, згенерованого ІІІ (рис. 3.23) [90].

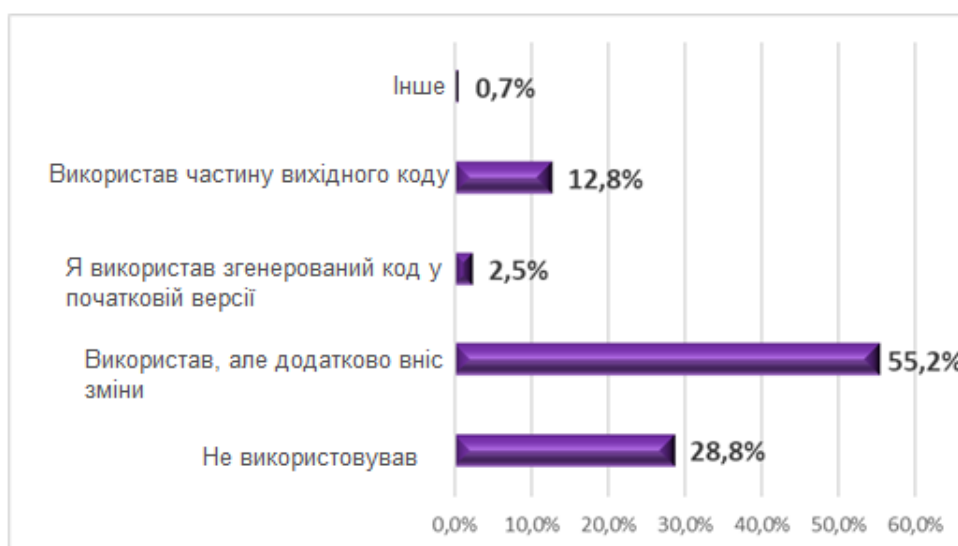


Рис. 3.23. Використання студентами 3-4 курсів коду, згенерованого ІІІ [90].

Крім генерації коду студентами було зазначено де саме вони використовують ІІІ в навчанні: 79,9% – для власних потреб, пов’язаних не з навчанням; для генерації ідей – 63,3%; для пошуку інформації – 59,8%; створення тексту – 58%; для виконання лабораторних – 51,6% і практичних робіт – 47,3%; для аналізу даних – 46,6%; для отримання коду програми – 43,4;

переклад тексту – 43,1%; створення презентацій – 34,5%; перевірка достовірності фактів – 32,4% тощо (рис. 3.24).

Потреби студентів спонукали провести деталізацію моделі ШІ, які використовують студенти: ChatGpt – 81,5%; 40,9% – використовують його постійно для навчання, а 1,8% – не використовують взагалі. Версія 3,5 набула популярності, як безкоштовна, але 16,4% студентів використовують платну версію 4,0.

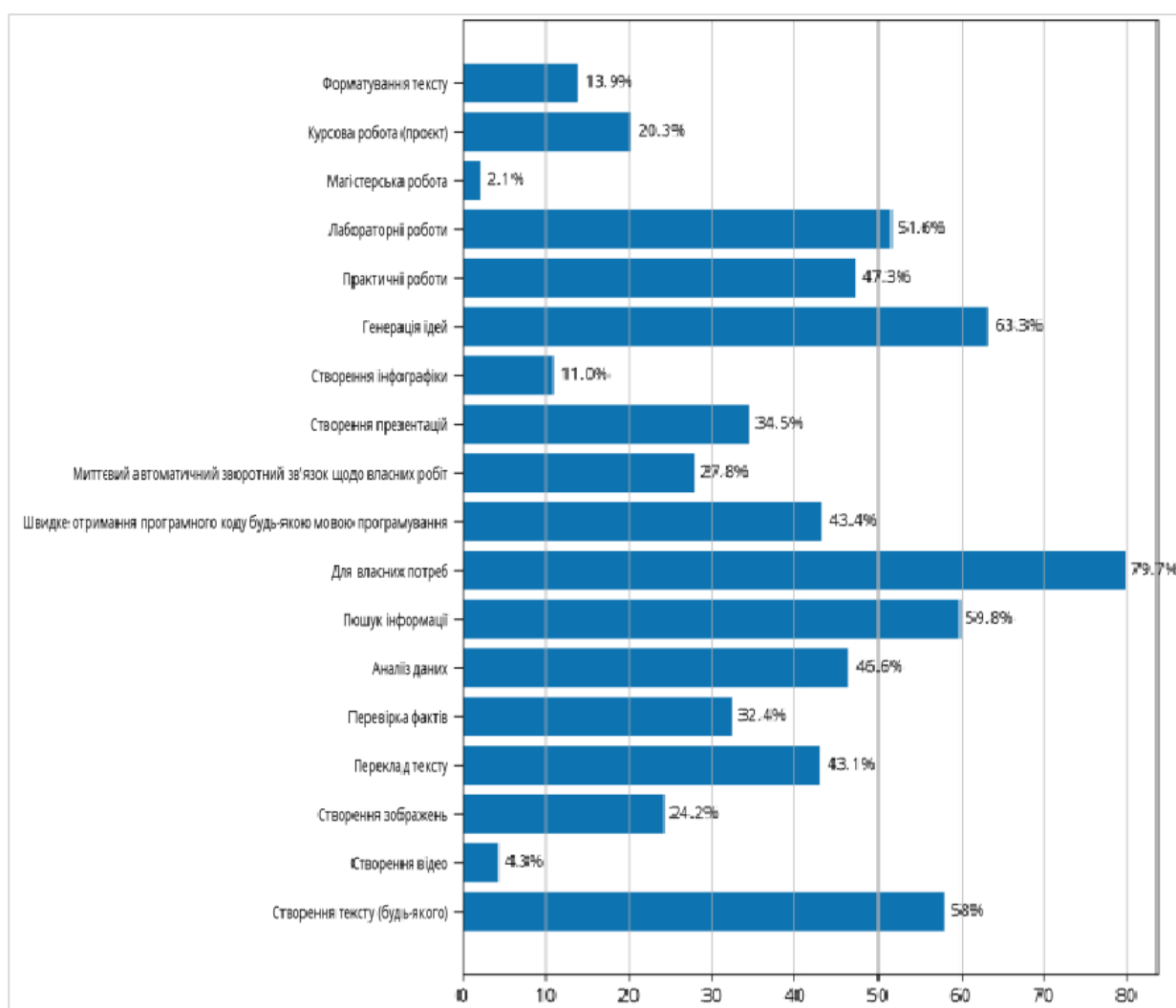


Рис. 3.24. Досвід студентів зі створення матеріалів з використанням ШІ [90].

Модель Gemini ще не набула великої популярності. Її не використовує понад 60,1% студентів, 16% відсотків студентів звертаються до цієї моделі ШІ за потребою (час від часу): один раз на тиждень – 4,6%, два рази на тиждень – 2,8% і використовують постійно – 5,3% [90].

Модель Copilot з'явилася пізніше за попередню і теж не набула популярності серед студентів: 64,8% студентів цю модель не використовують, за потреби до неї звертаються 12,1% студентів, один раз на тиждень – 4,3% і два рази на тиждень – 2,5%. Використовують постійно – 5,3%.

Як показує моніторинг появи і впровадження новітніх технологій (2004-2023 рр.): від появи технології до її використання час скорочується, дизайн та технологічність спрощується і стає доступною для усіх користувачів. Саме тому студенти не скористалися додатковим навчанням щодо використання ШІ і 95,7% студентів змогли самостійно опанувати використання ШІ (рис. 3.25).

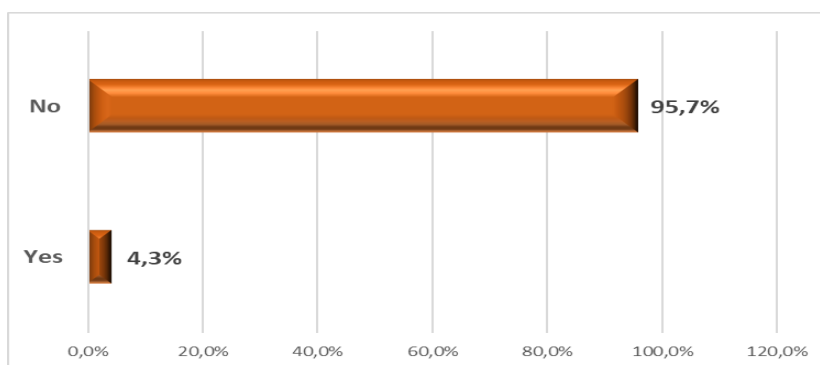


Рис. 3.25. Навчання студентів на додаткових курсах щодо використання ШІ[90].

Важливим аспектом розвитку студента як професіонала є його креативність, творчість та затребуваність на ринку праці. Тому питання зниження креативності студентів, що є одним із недоліків впровадження ШІ в освітній процес, постало як ключове. У процесі аналізу відповідей студентів було з'ясовано, що креативність, на думку студентів, не зменшується: 48,6% студентів відповіли, що скоріш ні і 28,9% упевнено сказали ні (рис. 3.26).

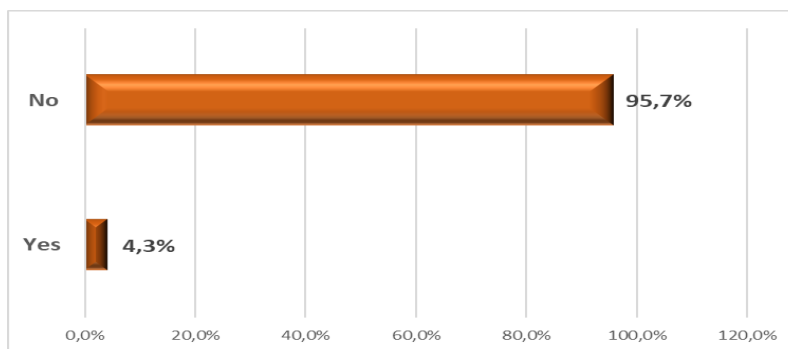


Рис. 3.26. Зниження креативності у студентів у процесі використання ШІ [90].

Проте 22,5% студентів стурбовані, що таке може статися. На це можуть вказувати дані (Рис. 3.27), що у студентів знижується бажання зробити завдання самому понад 39,2%. Проте 58,7% студентів стверджують що цього не відбувається.

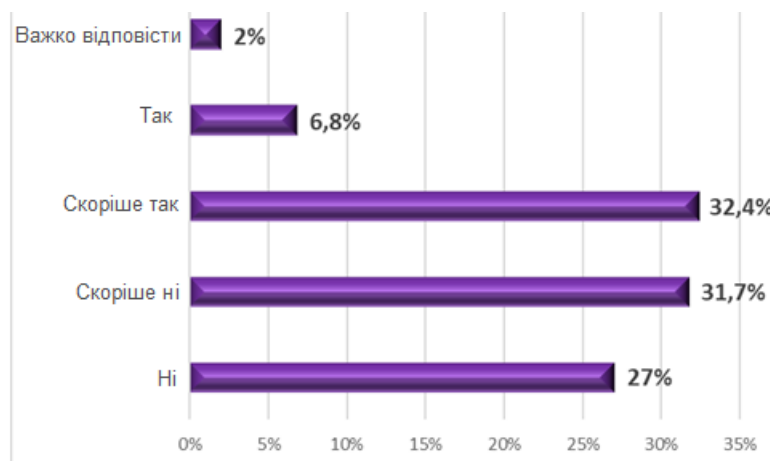


Рис. 3.27. Зниження бажання зробити завдання [90].

У перспективі ІІІ має стати асистентом не тільки викладача, а й студента. Про це стверджує 81,4% студентів, хоча 18,5% студентів не розділяють цю думку (рис. 3.28).

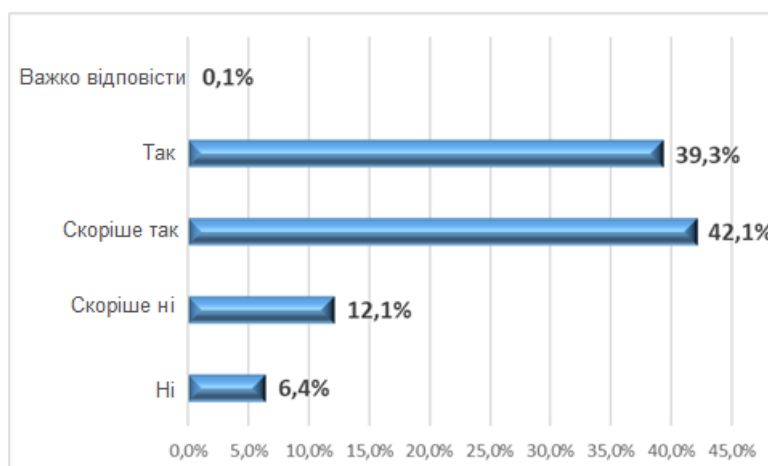


Рис. 3.28. Тенденції щодо використання ІІІ як асистента студентів [90].

Саме тому важливо сьогодні в суспільстві сформувати позитивне ставлення до його використання в системі освіти, окреслити відповідні вимоги, дотримуватися етики.

Не зважаючи на окреслені недоліки, використання штучного інтелекту в навчанні студентів ЗВО мовам програмування є перспективним напрямком, що надає можливість підвищити якість освіти та забезпечити підготовку висококваліфікованих фахівців, конкурентоспроможних на світовому ринку праці.

Результати проведеного дослідження засвідчили наступне:

- студенти першого курсу спеціальності програмування активніше використовують новітні технології, ніж студенти старших курсів, зокрема 4 курсу бакалаврату [90];
- попри наявні проблеми, розвиток технологій та зростаюча потреба у фахівцях з програмування роблять інтеграцію ІІІ в навчальний процес актуальною і необхідною;
- не всі ЗВО мають достатню технічну базу та фінансові можливості для впровадження інноваційних технологій, таких як ІІІ. Це створює розрив у наданні освітніх послуг між технічно забезпеченими і менш технічно забезпеченими навчальними закладами;
- існуючі навчальні програми можуть бути застарілими та не відповідати сучасним вимогам ринку праці. Постійне оновлення змісту курсів відповідно до нових технологій і методик навчання вимагає значних зусиль і часу з боку викладачів;
- використання ІІІ надає можливість адаптувати навчальні матеріали під потреби кожного студента. Проте, це вимагає створення нових методик оцінювання та індивідуального підходу до навчання, що може бути складним у великих групах;
- впровадження нових технологій часто зустрічає опір як з боку викладачів, так і студентів, які звикли до традиційних методів навчання. Зміна менталітету і адаптація до нових умов вимагають часу [90];
- викладачі повинні мати відповідні знання і навички для роботи зі ІІІ та іншими сучасними технологіями. Це потребує регулярного підвищення кваліфікації та професійного розвитку викладацького складу. Використання ІІІ

в освіті піднімає питання конфіденційності даних студентів та їх захисту. Необхідно створити нормативно-правову базу, яка б регулювала ці аспекти;

– викладання мов програмування та впровадження ІІІ вимагають інтеграції знань з різних дисциплін, що може ускладнити навчальний процес і вимагати більш тісної співпраці між різними факультетами та кафедрами[90].

Таким чином, впровадження ІІІ в освіту, особливо у викладання мов програмування, відкриває нові можливості для індивідуалізації та підвищення ефективності навчання. Однак, це також ставить перед ЗВО ряд викликів, які потребують системного підходу та комплексних рішень. Подальше дослідження і вдосконалення ІІІ-технологій у цій сфері сприятиме їх більш ефективному використанню та вирішенню існуючих проблем [90].

В процесі дисертаційного дослідження розроблені методичні підходи до використання ІІІ у навчанні програмування бакалаврів комп'ютерних наук.

Авторський підхід відображає комплексний підхід до навчання, де класична педагогіка поєднується з передовими технологіями штучного інтелекту.

1. *Концептуально-цільовий складник*. Він визначає мету навчання – підготовку фахівця, здатного створювати сучасні веборієнтовані системи. Тут закладаються теоретичні засади (наприклад, компетентнісний підхід згідно з DigComp 3.0), визначаються об'єкт та предмет навчання, а також формулюються кінцеві результати, якими має володіти студент після завершення курсу [90].

2. *Технологічний складник*. Визначає інструментальну систему розробки. Замість локальних налаштувань акцент зміщується на хмароорієнтовані середовища. Він включає використання хмарних середовищ розробки (наприклад, GitHub Codespaces), систем контролю версій (GitHub) та платформ для автоматичного розгортання проєктів (Vercel, Netlify). Це створює єдиний технологічний стандарт для всіх студентів.

Авторський підхід замінює використання традиційних локальних середовищ, а саме: *перехід у використанні від локального до хмаро орієнтованого середовища*. До переваг такого переходу необхідно віднести:

- усунення проблеми «на моєму комп'ютері працює, а на іншому – ні»;
- студент має доступ до робочого середовища з будь-якого пристрою через браузер [90].;
- використання інструментів на кшталт GitHub Codespaces, Replit або Gitpod дозволяє студентам працювати над кодом одночасно прямо в браузері;
- навчання будується навколо репозитаріїв (GitHub/GitLab), у яких весь прогрес студента – це серія фіксованих змін;
- студент бачить результат своєї роботи не в консолі, а за реальним URL-посиланням;
- використання Docker та Cloud-хостингів (Vercel, Netlify) для миттєвої публікації результатів сприяє навчання розгортання складних систем, що є критичним для сучасного CS-бакалавра [90].

3. *Змістовно-процесуальний складник*. Описує безпосереднє наповнення навчальної дисципліни та етапи її опанування. Зміст структурується за логічними блоками (HTML/CSS, JavaScript, React), а процес побудований на послідовному виконанні ітерацій: від створення макета у Figma до програмування бізнес-логіки та фінальної публікації продукту.

Відмічається трансформація навчального матеріалу у динамічний контент, зокрема активно використовуються такі види навчання, як:

- мікронавчання – це короткі відеолекції та інтерактивні вправи на платформах типу FreeCodeCamp або власних LMS (Moodle/Canvas);
- проєктно-орієнтоване навчання під час якого замість абстрактних задач реалізується розроблення реальних Web-сервісів;
- гейміфікація та змагальність – це інтеграція з платформами для алгоритмічного програмування (LeetCode, Codewars);

– проєктний підхід навчання, що будується навколо створення реального продукту (MVP – Minimum Viable Product) [90].

1. *Складник інтелектуальної підтримки (AI-асистенти).*

Фокусується на ролі ІІІ як персонального асистента студента. Інструменти на кшталт GitHub Copilot або ChatGPT використовуються для когнітивної підтримки, а саме: пояснення складних фрагментів коду, допомоги у пошуку помилок (дебагінгу) та генерації початкових структур (промпт-інжиніринг). Це знижує когнітивне навантаження при вивченні складного синтаксису.

Впровадження авторської методики передбачає трансформацію традиційної взаємодії «викладач–студент» у гнучку систему інтелектуальної підтримки, що базується на таких інструментах:

– навчання будується на принципах парного програмування, де роль помічника виконує інтелектуальний асистент. Студент вчиться формулювати чіткі технічні завдання для ІІІ, що розвиває навички декомпозиції задач, а асистент не просто генерує фрагменти коду, а пропонує варіанти написання «чистого коду», супроводжуючи їх коментарями щодо логіки побудови алгоритму [90].;

– ІІІ-асистент виступає персоналізованим посередником між складним теоретичним змістом та когнітивними можливостями студента. Система здатна адаптувати пояснення абстрактних концепцій (наприклад, замикання в JavaScript чи стан у React) через зрозумілі студенту терміни;

– студент отримує миттєвий звіт про помилки відразу після фіксації змін у репозиторії, що прискорює цикл роботи над помилками та підвищує якість фінального продукту (MVP);

– студент отримує миттєву допомогу у виправленні синтаксичних та логічних помилок у режимі 24/7, що підтримує стан потоку та запобігає втраті мотивації при виникненні технічних труднощів;

2. *Інтелектуальний педагогічний менеджмент (ІІІ для викладачів).*
Цей складник спрямований на трансформацію ролі викладача в умовах

цифровізації. Штучний інтелект допомагає автоматизувати перевірку коду, аналізувати успішність групи, генерувати індивідуальні завдання та адаптувати навчальний темп під потреби конкретного студента, дозволяючи викладачу зосередитися на менторстві та творчому супроводі [90].

Важливим аспектом для викладача є автоматизація рутини для вивільнення часу на консультування, зокрема ШІ може підтримувати такі види роботи:

- перевіряти код на плагіат, якість архітектури та безпеку;
- інтелектуальна система може аналізувати динаміку розробки студента та сигналізувати про критичне зниження темпів роботи або виникнення технічних перешкод;
- аналізувати профілі студентів і допомагати формувати збалансовані команди для групових проєктів [90].

6. *Моніторингово-результативний складник*. Відповідає за систему оцінювання та зворотного зв'язку. Він включає критерії перевірки набутого рівня компетентності, оцінку якості створених MVP (мінімально життєздатних продуктів) та аналіз динаміки прогресу студента на кожному етапі розробки.

7. *Правила використання ШІ у процесі програмування* - це регламентуючий складник, що визначає етичні та академічні межі. Сюди входить культура взаємодії з ШІ, а саме: розуміння того, що студент залишається відповідальним за архітектуру та логіку, а ШІ виступає лише інструментом прискорення рутинних операцій. Це вчить критично оцінювати згенерований код і запобігає «сліпому» копіюванню.

Університети підтримують раціональне експериментування з генеративними інструментами ШІ, але за умови врахування важливих аспектів використання таких інструментів, зокрема: інформаційної безпеки, конфіденційності даних, дотримання вимог авторського права та академічної доброчесності. Університети здійснюють підтримку науково-педагогічних працівників та здобувачів вищої освіти у підвищенні грамотності щодо використання ШІ в освітньому процесі [90].

3.3.1 Особливості розроблення комп'ютерних практикумів

У процесі навчання студентів програмування з використанням веборієнтованих технологій важливу роль відіграють комп'ютерні практикуми. Розглянемо на прикладі дисципліни «Веборієнтовані технології. Frontend-розробка» особливості організації та виконання комп'ютерного практикуму.

Для успішного отримання заліку дисципліни «Веборієнтовані технології. Frontend-розробка» кожен студент повинен:

- створити макет власного вебзастосунку в програмі Figma (тему обрати на власний розсуд);
- розробити власний індивідуальний веборієнтований застосунок (сайт), який на кожному комп'ютерному практикумі має бути представлений у вигляді працездатної версії сайту (релізу), доповненої новим функціоналом відповідно до завдань відповідного практикуму;
- створити звітний HTML-документ, який міститиме звіт про виконання комп'ютерних практикумів;
- створити два репозиторії: один для власного індивідуального веборієнтованого застосунку, другий – для звітів комп'ютерних практикумів з дисципліни;
- виконати 6 комп'ютерних практикумів, результати яких студент поетапно розміщує у звітному HTML-документі після виконання кожного практикуму;
- виконати дві модульні контрольні роботи (МКР-1, МКР-2): першу – після проходження Розділу №1, другу – після Розділу №2.

Для прикладу розглянемо комп'ютерний практикумі №1. У межах виконання цього практикуму студент повинен вибрати будь-яку предметну галузь або її частину, здійснити опис предметного середовища майбутнього вебзастосунку, тобто визначити його функціональні можливості. Далі, на основі зазначеного опису предметного середовища, необхідно виконати опис

бізнес-логіки (скорочений опис предметного середовища) та надати його у вигляді сценарію.

Необхідно створити UML-діаграму прецедентів, а також визначити функціональні та нефункціональні вимоги до вебзастосунку. Додатково слід розробити макет власного вебзастосунку у програмі Figma на власний розсуд. Для ознайомлення з можливостями зазначеного інструменту, надається перелік із дев'яти коротких навчальних уроків з Figma, які опрацьовуються самостійно.

Наступним етапом є виконання верстки розробленого макета із використанням семантичної HTML-розмітки. Після цього в середовищі Visual Studio Code необхідно реалізувати програмний код відповідно до завдання першого комп'ютерного практикуму. Зберегти його у вигляді коміта, який у подальшому буде розміщується у попередньо створеному репозиторію GitHub .

Другий репозиторії GitHub.використовується для відображення результатів виконання всіх комп'ютерних практикумів. Структура практикумів побудована таким чином, що кожен наступний етап доповнює попередній, формуючи цілісний проєкт. У результаті послідовного виконання завдань створюється завершена програмна розробка, яка може бути використана як частина професійно портфоліо для представлення на співбесідах.

Кожен комп'ютерний практикум відповідає окремій темі дисципліни. Розглянемо приклад комп'ютерного практикуму

Комп'ютерний практикум №1.

Курс: 3 курс бакалаврату спеціальності «Комп'ютерні науки».

Дисципліна: «Веборієнтовані технології. Frontend-розробка».

Тема: «Проектування інтерфейсу та створення структурної розмітки вебзастосунку» (відповідає Розділу 1 дисципліни).

Зміст завдання:

Мета: опанувати повний цикл ініціації проєкту – від ідеї до первинної публікації в хмарі.

Покроковий алгоритм виконання:

1. *Етап проєктування (UI/UX-дизайн).* Процес починається з концептуалізації бізнес-логіки застосунку. На основі опанованих дев'яти навчальних мікромодулів студент здійснює візуалізацію інтерфейсу в середовищі Figma з урахуванням принципів адаптивності та Mobile First, формуючи прототип майбутнього програмного продукту.

2. *Формалізація вимог (Специфікація).* Для чіткого визначення функціональних можливостей програмного продукту здійснюється побудова UML-діаграми прецедентів (Use Case Diagram). Це дозволяє структурувати взаємодію користувача із системою, формалізувати основні сценарії використання та закласти основу для подальшого проєктування і реалізації вебзастосунку.

3. *Інтелектуальна підтримка.* На етапі переходу від макету до програмної реалізації впроваджується підхід спільної діяльності зі ШІ, зокрема використання інструментів ChatGPT та GitHub Copilot. У межах цього підходу, шляхом застосування методів промпт-інжинірингу, студент формує та уточнює запити для генерації оптимальної семантичної структури HTML5. Такий підхід забезпечує підвищення якості розмітки вебсторінок, сприяє дотриманню сучасних стандартів веброзробки та покращує структурованість і семантичну коректність коду.

4. *Розробка в хмаро орієнтованому середовищі.* Практична реалізація відбувається у хмарному середовищі шляхом створення репозиторію на платформі GitHub із подальшим розгортанням ідентичного робочого середовища в GitHub Codespaces. На цьому етапі відбувається безпосереднє написання коду структурної розмітки.

5. *Контроль версій.* Методика передбачає обов'язкове дотримання культури програмної розробки через фіксацію проміжних результатів. Студент

виконує щонайменше три змістовні операції фіксації змін (коміти), які поетапно відображають процес розробки та еволюцію структури проєкту.

6. *Розгортання та апробація.* Завершальним етапом ітерації є автоматичне розгортання та публікація результатів на платформах Vercel або Netlify. У результаті студент отримує функціональне URL-посилання на поточну ітерацію проєкту, що забезпечує можливість оперативної перевірки працездатності системи в реальних умовах мережі. Автоматизоване розгортання реалізує підхід безперервної інтеграції та доставки (CI/CD), який є невід’ємною складовою сучасних практик розробки ПЗ та відповідає вимогам підготовки фахівців в галузі комп’ютерних наук відповідно до міжнародних.

Очікуваний результат. Працездатний реліз (сайт-заготовка), розміщений у мережі Інтернет, а також посилання на GitHub-репозиторій із зафіксованою історією змін.

Для оцінювання комп’ютерного практикуму на рівні «Створення» (95-100 балів) відповідно до таксономії Блума, що відповідає експертному рівню 7–8 за рамкою DigComp 3.0, розроблено наступні деталізовані критерії, які представлені у таблиці 3.11.

Обґрунтування оцінки за таксономією Блума та DigComp 3.0:

– *Рівень «Створення» (Блум VI)* - студент не лише виконує навчальне завдання, а здійснює синтез знань з UI-дизайну, семантичного кодування та хмарної інфраструктури для створення мінімально життєздатного продукту (MVP).

Таблиця 3.11

Критерії оцінювання виконання практикуму № 1

Критерій оцінювання	Дескриптори експертного рівня (DigComp 3.0, рівні 7–8)	Вимоги до виконання завдання
1. Відповідність макета Figma та HTML-коду	напрям 3.1: Розроблення цифрового контенту. Здатність створювати складні цифрові об'єкти з	Повна візуальна ідентичність коду макету у Figma. Реалізація стратегії Mobile First та коректне використання Flexbox/Grid

Критерій оцінювання	Дескриптори експертного рівня (DigComp 3.0, рівні 7–8)	Вимоги до виконання завдання
	високим рівнем точності та адаптивності.	для адаптивності на всіх типах пристроїв.
2. Валідність семантичної розмітки	напрям 3.4: Програмування. Глибоке розуміння структури даних та стандартів. Здатність виступати аудитором коду.	Використання виключно семантичних тегів HTML5 (header, main, section, article, footer) для забезпечення SEO та доступності (Accessibility). Відсутність помилок у валідаторі W3C.
3. Наявність «цифрового сліду»	напрям 5.2: Визначення технологічних відповідей. Використання професійних систем контролю версій та CI/CD процесів.	Наявність ітеративної історії розробки у GitHub (мінімум 10 змістовних комітів). Чіткі коментарі до кожного коміту, що відображають еволюцію проєкту. Відсутність завантаження всього коду одним архівом.
4. Автоматизація та реліз	напрям 3.4: Програмування. Налаштування життєвого циклу ПЗ: від коду до реалізації.	Наявність налаштованого автоматичного розгортання репозиторію на платформі Vercel або Netlify. Кожен коміт автоматично оновлює працездатний реліз за реальним URL-посиланням.

– *Експертний рівень (DigComp 7–8)* - студент демонструє здатність працювати у професійному хмаро орієнтованому середовищі, де веббраузер виступає єдиною точкою доступу до робочого середовища (через GitHub Codespaces), а результат розробки функціонує в реальних мережових умовах.

Додатковий бал (підвищення рейтингу). Відповідно до Таблиці 2.8, студент може отримати до 20% додаткових балів за умови використання ІІІ (ChatGPT, GitHub Copilot) як інструмента інтелектуальної підтримки для оптимізації коду та проведення самостійної перевірки програмного коду засобами ІІІ (AI Code Review) перед поданням роботи.

Звіт про виконання першого комп'ютерного практикуму представлений на рисунках. 3.29- 3.30.



Рис. 3.29. Головне вікно звіту студентки 3 курсу

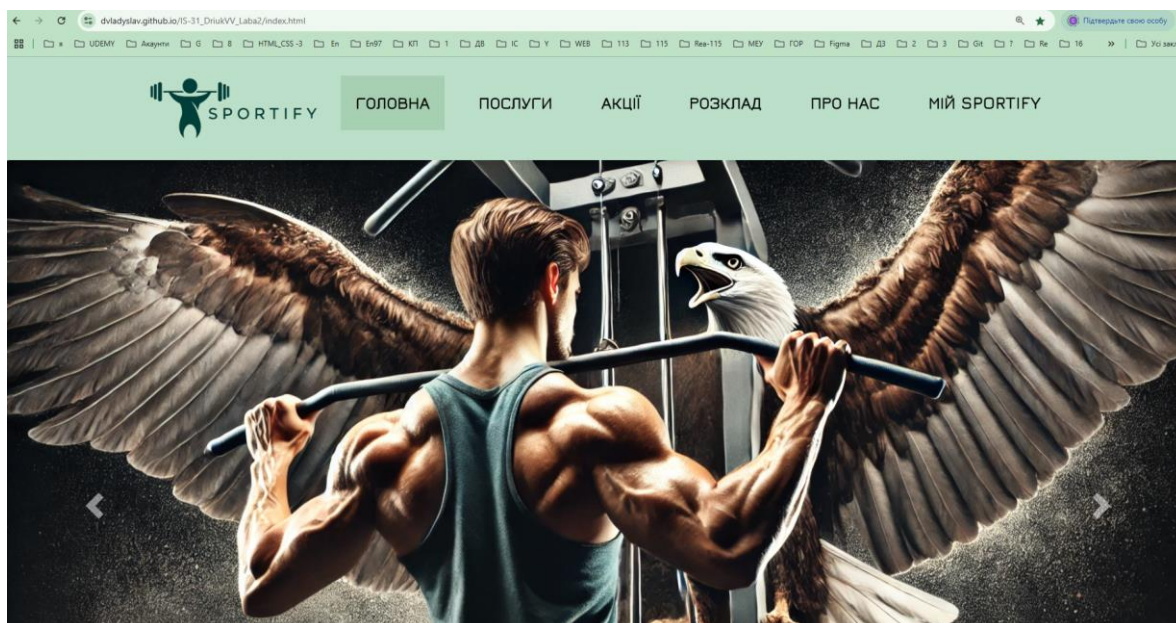


Рис. 3.30. Головне вікно власного вебзастосунку студента 3 курсу

Комп'ютерний практикум має ваговий бал 10.

Критерії оцінювання комп'ютерного практикуму.

10 балів. Під час захисту комп'ютерного практикуму студент демонструє безпомилкову роботу програмного продукту, надає обґрунтовані, повні й чіткі відповіді на поставлені запитання. Програмний код є структурованим, читабельним і відповідає сучасним вимогам до розробки ПЗ. Студент здатний обґрунтувати прийняті проєктні та програмні рішення.

7,5-9,9 бала. Студент демонструє безпомилкову роботу програми, але надає неповні й нечіткі відповіді на поставлені запитання. Виконана робота містить програмний код, який частково не відповідає вимогам структурованості.

6,0-7,4 бала. Написана студентом програма працює, але містить невідповідності архітектурним патернам, рекомендаціям до оформлення коду, використовує неефективні алгоритми. Під час відповіді на деякі питання студент припускається істотних помилок.

0 балів. Програма не написана, не працює, або має грубі порушення в програмному коді. Під час відповіді на питання студент припускається грубих помилок або не може дати відповіді на питання.

Максимальна кількість балів за всі комп'ютерні практикуми складає:

$R_{kp} = \text{ваговий бал} * \text{кількість комп'ютерних практикумів}$

$R_{kp} = 10 \text{ балів} * 6 \text{ комп'ютерних практикумів} = 60 \text{ балів}$, що є об'єктивним показником професійного зростання студента за рівнями таксономії Блума та рамкою DigComp 3.0.

На комп'ютерних практикумах студенти демонструють результати ітеративної розробки, зафіксовані у хмарному середовищі GitHub Codespaces, та працездатність релізів, розгорнутих на платформах Vercel/Netlify. Для допуску до поточного комп'ютерного практикуму необхідно мати звітний документ цього комп'ютерного практикуму, та захищений попередній комп'ютерний практикум.

3.3.2 Самостійна робота, індивідуальний підхід

Успіх підготовки майбутніх бакалаврів комп'ютерних наук залежить від багатьох факторів, одним з яких є самостійна робота студентів. В сучасних наукових дослідженнях самостійна робота розглядається як вища форма навчальної діяльності, що спрямована на ефективне засвоєння досвіду людства, на розвиток та самовдосконалення пізнавальної сфери майбутнього фахівця. [91].

На сьогоднішній день в навчально-педагогічній літературі зустрічається велика кількість означень самостійної роботи. Зокрема, у посібнику про організацію навчального процесу у вищих закладах освіти, Болубаш Я.Я. характеризує таку роботу як основний засіб засвоєння студентом навчального матеріалу в час, вільний від обов'язкових навчальних занять [91], [92]. В “Енциклопедії освіти” приводиться: “самостійна робота студентів - це планова індивідуальна або колективна робота студентів, що виконується за завданням і при методичному керівництві викладача, але без його безпосередньої участі” [93].

За своєю суттю самостійна робота є активною розумовою діяльністю студента, пов'язаною з виконанням навчального завдання. Наявність завдання і цільової установки на його виконання вважають характерними ознаками самостійної роботи [94].

Аналізуючи визначення, запропоновані різними науковцями, можна дійти висновку, що самостійна робота майбутнього бакалавра комп'ютерних наук розглядається, з одного боку, як вид навчальної діяльності, що стимулює активність і пізнавальний інтерес студентів, створює передумови для самоосвіти та подальшого професійного розвитку, а з іншого - як система педагогічних заходів і умов, що забезпечують організацію та керівництво самостійною діяльністю студентів [91].

У статті 50 Закону України «Про вищу освіту» констатується, що освітній процес у ЗВО здійснюється за такими формами: навчальні заняття; самостійна робота; практична підготовка; контрольні заходи [95].

Основними видами навчальних занять у закладах вищої освіти є: лекція; лабораторне, практичне, семінарське, індивідуальне заняття; консультація.

Зміст і види самостійної роботи майбутніх бакалаврів комп'ютерних наук визначаються робочими програмами навчальних дисциплін з програмування (Алгоритмізація та програмування, Об'єктно-орієнтоване програмування та інших дисциплін, пов'язаних з програмуванням), а також завданнями й методичними рекомендаціями викладача.

У навчальному процесі виокремлюють два види самостійної роботи студентів: аудиторна та позааудиторна. Аудиторна самостійна робота виконується під час навчального заняття під безпосереднім керівництвом викладача після отримання відповідного завдання. Позааудиторна самостійна робота виконується студентом за завданням викладача, але без його безпосередньої участі. У межах цього дослідження розглядається позааудиторна самостійна робота, яку можна поділити на такі типи:

- підготовка до аудиторних занять;
- закріплення знань, отриманих під час лекційних занять;
- виконання комп'ютерних практикумів;
- опрацювання окремих тем дисциплін з програмування, винесених на самостійне опрацювання студентами;
- виконання індивідуальних завдань із навчальних дисциплін програмування протягом семестру;
- робота з електронними ресурсами та систематизація отриманих даних;
- підготовка та виконання курсового проєктування;
- використання сучасних інформаційно-комунікаційних технологій в освітньому процесі;

- підготовка до всіх видів контрольних заходів (контрольно-модульних робіт, ректорського контролю);
- участь у наукових і науково-практичних конференціях, конкурсах, олімпіадах тощо;
- робота у студентських наукових гуртках, участь у семінарах тощо;

В умовах розвитку інтелектуальних вебтехнологій самостійна робота студента зазнає суттєвої трансформації, переходячи від епізодичного виконання окремих завдань до моделі неперервного навчання. Традиційний підхід до позааудиторної діяльності як до підготовки статичних навчальних матеріалів поступово змінюється активною професійною діяльністю студента в інтегрованих мароорієнтованих середовищах розробки.

Використання хмарних ICP, зокрема GitHub Codespaces, забезпечує неперервний перехід між аудиторною та позааудиторною діяльністю, зменшуючи потребу в локальному зберіганні даних. При цьому результати діяльності студента фіксуються у вигляді «цифрового сліду» – історії ітеративних фіксацій у репозиторії. Такий підхід дає змогу відстежувати процес розробки програмного продукту та аналізувати індивідуальну траєкторію навчання.

У цьому контексті індивідуалізація навчання реалізується через співпрацю студента з індивідуальними цифровими асистентами. ІІІ-асистент виступає не лише інструментом пошуку інформації, а й адаптивним помічником, який у режимі постійного доступності надає підтримку під час налагодження програмного коду та пояснення складних алгоритмічних понять. Це сприяє формуванню цифрової професійної ідентичності майбутнього бакалавра комп'ютерних наук.

Відповідно до принципів Болонського процесу, організація освітнього процесу у ЗВО передбачає значну частку навчального навантаження, відведеного на самостійну роботу студентів. Наприклад, у Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського» навчальний час, передбачений для самостійної роботи

майбутніх бакалаврів комп'ютерних наук денної форми навчання, визначається робочими навчальними планами та становить, як правило, близько 50% загального обсягу часу, відведеного на вивчення дисципліни. З урахуванням того, що тижнева навчальне навантаження студента становить, 45 годин, обсяг самостійної роботи має складати приблизно 23 години на тиждень [96].

Для успішного виконання самостійної роботи викладачі кафедр ознайомлюють майбутніх бакалаврів комп'ютерних наук із формами та методами організації навчального процесу в університеті, основами наукової організації праці, методикою самостійної роботи, критеріями оцінювання її якості, також визначають мету, засоби, трудомісткість, строки виконання та форми контролю.

У процесі навчання у студентів формуються навички роботи з електронними підручниками, класичними першоджерелами, сучасною науковою літературою та програмними засобами; розвиваються вміння знаходити оптимальні способи розв'язання завдань, виконувати розрахунки, аналізувати результати науково-дослідної роботи. Для надання необхідної допомоги викладачі проводять групові з індивідуальні консультації, здійснюють систематичний контроль за виконанням завдань, рекомендованих для самостійного опрацювання, а також проводять аналіз та оцінювання результатів роботи студентів.

Особливу увагу необхідно приділяти формуванню позитивного ставлення майбутніх бакалаврів комп'ютерних наук до позааудиторної роботи, особливо на першому курсі. На початку вивчення дисципліни студентам слід пояснити основні завдання, винесені на самостійне опрацювання, місце розташування навчального методичного матеріалу та порядок їх використання [91]

На кожному проміжному етапі виконання самостійної роботи необхідно пояснювати її мету, перевіряти рівень розуміння завдань студентами, здійснювати контроль знань та нагадувати про кінцеві строки виконання.

Такий підхід сприяє поступовому формуванню самостійно визначати завдання, планувати діяльність і виконувати поставлені цілі. У процесі самостійної роботи майбутні бакалаврів комп'ютерних наук набувають навичок планування власної діяльності, в по, аналізу, опрацювання та представлення інформації, а також формування отриманих результатів [91].

Для забезпечення високого рівня підготовки майбутніх бакалаврів комп'ютерних наук учні та заочної форми навчання, систематизації навчальних навчальної інформації та постійного підвищення професійного рівня майбутніх програмістів викладачеві необхідно правильно організовувати самостійну роботу студентів. Це передбачає детальне складання плану самостійної роботи з кожної дисципліни, розроблення методичних рекомендацій щодо виконання комп'ютерних практикумів, курсового та дипломного проектування. При цьому студент повинен чітко знати, які теми призначені для самостійного вивчення, які види та форми роботи будуть застосовані, а також визначені форми контролю та строки виконання завдань.

Для організації самостійної роботи майбутнього бакалавра комп'ютерних наук необхідні такі умови: мотивація студента до самостійної роботи; наявність і доступність навчально-методичного забезпечення та довідкового матеріалу; наявність комп'ютерних класів; система регулярного контролю якості виконаної самостійної роботи; консультаційна допомога викладача[91].

Для розміщення навчально-методичного забезпечення для самостійної роботи студентів використовуються сучасні цифрові технології навчання. Зокрема, широко застосовуються системи управління навчанням, такі як Moodle та Campus, ClassRoo, які забезпечують організацію, підтримку й супровід освітнього процесу на основі сучасних інформаційно-комунікаційних технологій. Зазначені системи надають можливість розміщувати навчальні матеріали, організовувати самостійну роботу студентів, здійснювати контроль навчальних досягнень та забезпечувати зворотній зв'язок між викладачем і студентами. Крім того, студенти можуть

завантажувати виконані лабораторні, комп'ютерні практикуми та контрольні роботи для подальшої перевірки, оцінювання і коментування викладачем [91].

Для підготовки до навчальних занять і комп'ютерних практикумів, а також опрацювання окремих тем дисциплін з програмування, винесених на самостійне вивчення, активно використовуються електронні освітні ресурси. Їх застосування дає змогу забезпечити індивідуальний темп і послідовність опрацювання навчального матеріалу, самостійно організовувати навчальну діяльність, здійснювати самоконтроль рівня сформованості знань і вмінь, а також отримувати доступ до віддалених ресурсів, віртуальних подорожей та інших інформаційних матеріалів, розміщених у цифрових бібліотеках і на вебсайтах [97].

Одним із показників ефективності самостійної роботи є регулярність виконання завдань з програмування, яка може оцінюватися за кількістю годин, витрачених на програмування протягом дня, а також за частотою занять протягом тижня (щоденно, кілька разів на тиждень, епізодична тощо). Це дає змогу оцінити рівень залученості студентів до навчальної діяльності та їхню навчальну дисциплінованість.

Для підвищення ефективності самостійної роботи впровадження ІІІ-інструментарію в межах методики сприяє переходу до адаптивної моделі навчання, яка характеризується такими функціональними можливостями.

Замість автоматичної генерації виправленого програмного коду, система здійснює діагностику помилки та формулює систему евристичних запитань. Такий підхід стимулює рефлексію студента та сприяє розвитку навичок самостійного розв'язання проблем.

Використання інструментів ІІІ дає змогу трансформувати складні теоретичні концепції більш наочні форми подання інформації, зокрема у вигляді ментальних карт, схем, візуалізація та аналогів. Такий підхід сприяє кращому розумінню складних структур даних, алгоритмів і принципів

програмування завдяки адаптації пояснень до рівня підготовки та освітніх потреб бакалавра комп'ютерних наук [91].

Крім того, аналіз результатів навчальної діяльності студентів і типових помилок під час виконання програмних завдань створює можливість для формування персоналізованих навчальних рекомендацій і добору додаткових тренувальних вправ. Це дає змогу зосередити увагу на проблемних аспектах навчального матеріалу, забезпечуючи індивідуалізацію освітнього процесу та підвищуючи ефективність засвоєння знань і формування практичних навичок програмування [91].

З метою дослідження зазначеного питання було проведено опитування серед студентів-програмістів 2 курсу факультету інформатики та обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». В опитуванні приймало участь 120 респондентів.

Опитування виявило, що 37% студентів-програмістів приділяють програмуванню всього 1 день на тиждень, 25% – кожен день, 24% - не кожен день – , 9%, – 2 дні на тиждень, 5% - 3 дні на тиждень (рис. 3.31).

Цей факт свідчить про те що студентам-програмістам слід збільшити обсяг часу на самостійне виконання практичних завдань з програмування.

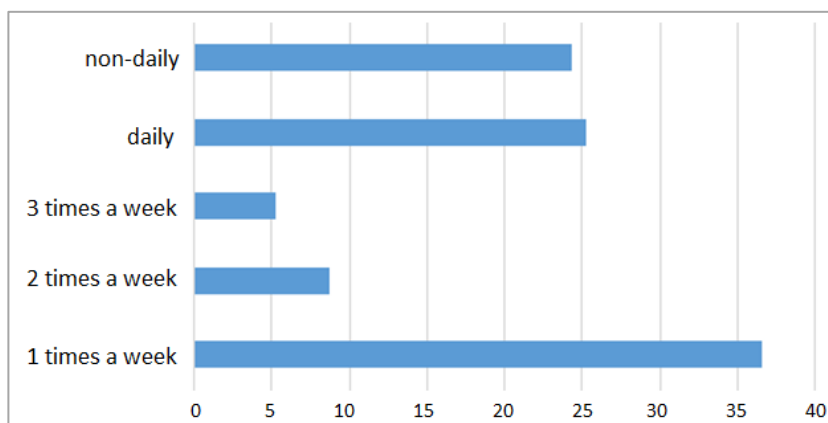


Рис. 3.31. Періодичність заняття з програмуванням [91].

На діаграмі (рис. 3.32) представлено результати опитування студентів щодо кількості часу, який вони щоденно витрачають на виконання завдань із програмування написання програмного коду. Аналіз отриманих даних показав, що 34% майбутніх бакалаврів комп'ютерних наук приділяють практичним заняттям з програмування більше 3-х години на день, 24% – близько 3 годин, 25% – близько 2 годин, 4% – близько 1 години, а 13% – лише 0,5 години. Ці результати свідчать, що 14% відсотків студентів приділяють практичному програмуванню менше однієї години на день [91].

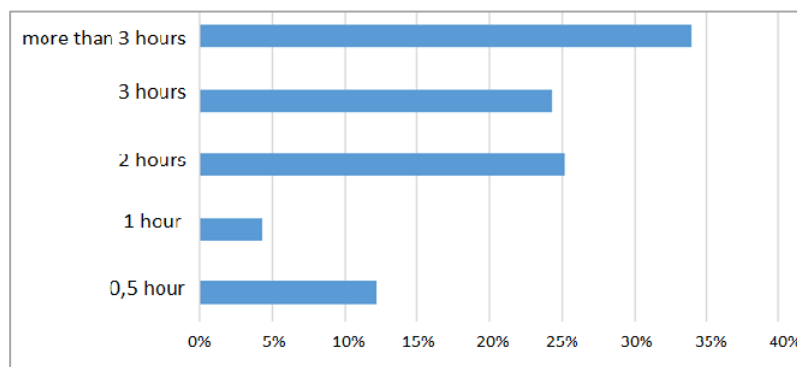


Рис. 3.32. Кількість годин заняття програмуванням на добу [91].

На нашу думку, недостатній обсяг часу, який студенти-програмісти протягом тижня приділяють самостійному виконання практичних завдань із програмування, може бути пов'язаний із необхідністю опрацювання значних обсягів навчального матеріалу з інших дисциплін. Водночас для досягнення високого рівня сформованості програмістських компетентностей необхідно забезпечити більшу регулярність самостійних занять із програмування. Це потребує раціонального планування навчального часу, оптимізації щоденного розкладу та визначення пріоритетності видів навчальної діяльності.

Під час анкетування особливу увагу було приділено використанню студентами в орієнтованих технологій підтримки навчання програмування зокрема автоматизованих систем перевірки програмних завдань та інструментів візуалізації й структурування навчального матеріалу. Серед

досліджувальних засобів розглядались такі платформи, як e-olymp Algotester, NetOI Olympiad.

За результатами опитування студентів другого курсу, з'ясувалося, що переважна більшість майбутніх бакалаврів комп'ютерних наук надає перевагу використанню Інтернет-порталу E-olymp (75%;) Algotester – 5%, NetOI Olympiad – 3%, іншим – 20%. Дані приведені на рисунку 3.33 [91].

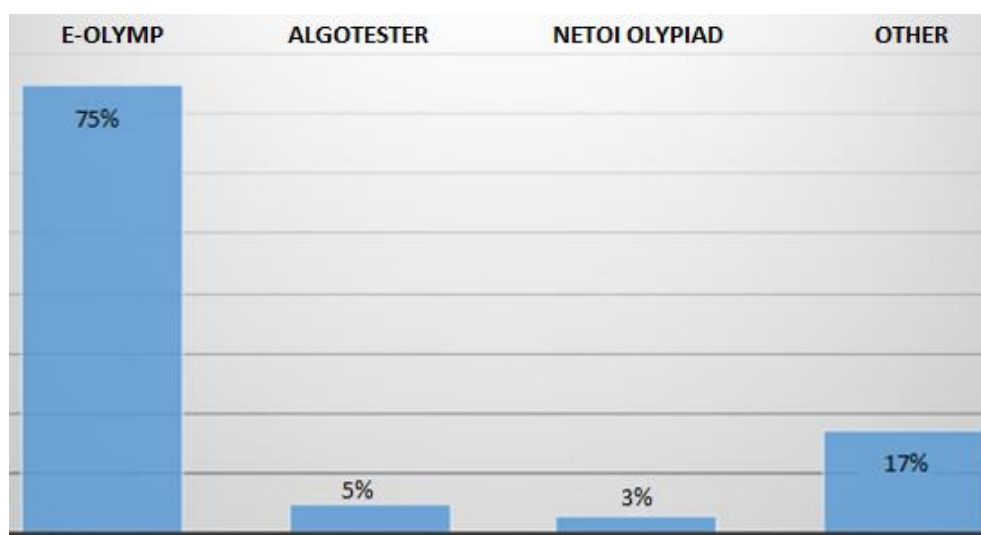


Рис. 3.33. Застосування автоматизованих системах перевірки завдань з програмування [91].

Автоматизована система E-olimp містить понад 7000 задач, наявність загального рейтингу зареєстрованих учасників, можливість перевірки програмних кодів мовами програмування Pascal, C#, C++, Java, Php, Python, Ruby, Haskell, можливість переглянути розв'язки, наявність відомостей про усі спроби розв'язання усіх задач, існування методичного розділу.

Також система E-olimp може використовуватися для проведення поточного контролю знань студентів у поєднанні з регулярним моніторингом та аналізом результатів навчальної діяльності з боку науково-педагогічних працівників. Саме результати контролю дають змогу викладачеві вносити корекцію до оцінки рівня знань. Для об'єктивного оцінювання результатів самостійної роботи студентів-програмістів розробляються відповідні критерії та показники оцінювання. Використання автоматизованих систем перевірки завдань із програмування надає студентам можливість самостійно виконувати

практичні завдання, здійснювати підготовку до занять та перевіряти правильність власних рішень без безпосередньої участі викладача. Крім того такі системи дозволяють порівнювати результати власної навчальної діяльності з результатами інших користувачів платформи, що сприяє підвищенню мотивації до вдосконалення програмістських компетентностей розвитку навичок самооцінювання та формування відповідного ставлення до процесу навчання [91].

Технологія використання інтелект-карт, надає можливість студенту візуалізувати, структурувати та запам'ятовувати великі обсяги навчальної інформації, швидко знаходити потрібні відомості, готуватися до іспитів та семінарів, створювати презентації, планувати власний час. На жаль, ця технологія застосовується лише на 10%, про що свідчать результати опитування майбутніх бакалаврів комп'ютерних наук.

До складу самостійних робіт студентів-програмістів входить підготовка до олімпіад університетського, всеукраїнського та міжнародного рівнів і участь у них. Як правило, частка обдарованої молоді становить від 10 % до 15 %.

Олімпіади з програмування є одним з важливих напрямів роботи з обдарованою молоддю. Під час виконання завдань, студенти-програмісти застосовують набуті знання та практичні навички з програмування, мають можливість прийняти участь у командній роботі, використовувати різноманітні інструменти для розроблення ПЗ.

У процесі участі в олімпіадах студенти не лише вдосконалюють професійні знання, уміння, а й розвивають важливі соціальні компетентності: комунікативні навички, уміння співпрацювати та планувати власну діяльність. Результати олімпіад можуть бути цінним навчальним матеріалом, який можна негайно використати в навчальному процесі [91].

Самоосвіта є важливою складовою самостійної роботи, що сприяє формуванню ІТ-компетентностей студентів. Онлайн-курси даб зможу опановувати сучасні веборієнтовані технології, розвивати теоретичні знання

та практичні навички, а також адаптувати навчання до власного тему (табл. 3.12).

Таблиця 3.12

Назви курсів для самостійного опрацювання

№	Назва курсу	Автор / Університет	Рівень	Тривалість	Доступ безкоштовний	Мова
1	<u>HTML, CSS, and JavaScript for Web Developers</u>	Johns Hopkins University	Початковий	~5 тижнів	Так	GB (субтитри UA)
2	<u>Web Design for Everybody</u>	University of Michigan	Початковий	~3–6 міс. (5 курсів)	Так	GB (є UA субтитри)
3	<u>Programming with JavaScript</u>	Meta (Facebook)	Початковий	~4 тижні	Так	GB
4	<u>Responsive Web Design in Practice</u>	University of London	Середній	~4 тижні	Так	GB
5	<u>Introduction to Front-End Development</u>	Meta (Facebook)	Початковий	~4 тижні	Так	GB

Це особливо актуально у сфері вебпрограмування. Перевагами самоосвіти є, в першу чергу, доступність контенту 24/7, постійне оновлення матеріалу, можливість постійно підвищувати рівень складності, отримувати сертифікати

Для розширення теоретичних знань, практичних вмінь та навичок в процесі самостійної діяльності студент може використовувати

1) Електронні підручники:

- HTML Підручник. w3schools.com.
українською. <https://w3schoolsua.github.io/html/index.html>
- HTML. Мова для створення веб-сторінок. w3schools.com.
Українською. <https://w3schoolsua.github.io/index.html#gsc.tab=0>

- HTML Довідник тегів. <https://w3schoolsua.github.io/tags/index.html>
 - CSS Підручник. w3schools.com.
- українською. <https://w3schoolsua.github.io/css/index.html#gsc.tab=0>
- HTML.Language for creating web pages. W3Schools in English. https://w3schoolsua.github.io/index_en.html#gsc.tab=0
 - HTML Tutorial. https://w3schoolsua.github.io/html/index_en.html#gsc.tab=0
 - Підручники HTML та CSS. https://htmlbook.at.ua/news/tutorial_html/1-0-1
 - Learn to Code. With the world's largest web developer site. <https://www.w3schools.com/>
 - Український веб-довідник. <https://css.in.ua/>
 - Довідник по HTML тегам <https://css.in.ua/html/tags>
 - Довідник HTML тегів <https://html-css.co.ua/dovidnuk-html-tags/>
 - HTML конструювання. <http://htmlbook.in.ua/>
 - <http://htmlbook.in.ua/tegs-html/> 6.<http://htmlbook.in.ua/pr01-html/> 7
- CSS-TRICKS. GUIDES. ARTICLES. <https://css-tricks.com/>

Популярні платформи:

- *Coursera* – повноцінні курси з HTML, CSS, JS, React, часто від провідних університетів.
- *Udemy* – курси зі створення сайтів, фреймворків, back-end.
- *FreeCodeCamp* – безкоштовна інтерактивна платформа з практикою.
- *Codecademy* – інструмент для покрокового навчання програмуванню.
- *CS50* від *Harvard* – фундаментальні знання комп'ютерних наук.

Приклад використання. Студент проходить курс «Responsive Web Design» на FreeCodeCamp, створює власний проєкт у CodePen, презентує його в класі – таким чином формується не лише знання, але й портфоліо.

3.3.3 Діагностика знаннєвого складника компетентності з програмування засобами модульного контролю

У межах запропонованої авторської методики модульний контроль розглядається не лише як засіб сумарного оцінювання, а як інструмент діагностики когнітивного складника компетентності з програмування, що є основою цифрової компетентності студента. Для забезпечення об'єктивності та відповідності парадигмі інтелектуального вебу, модульні контрольні роботи (МКР-1 та МКР-2) реалізуються в LMS Moodle (елемент «Тест»), що дозволяє створювати адаптивні сценарії перевірки та інтегрувати складні фрагменти коду безпосередньо у запитання.

Структура та зміст тестових завдань.

Замість репродуктивної перевірки знань, банк запитань проєктується як діагностична матриця, де завдання розподілені за рівнями таксономії Блума:

- *Рівні 1-2 (Пригадування/Усвідомлення, 20% питань)* - перевірка знання семантики HTML5, стандартів CSS3 та базового синтаксису JavaScript ES6+.
- *Рівні 3-4 (Застосування/Аналіз, 50% питань)* - завдання на прогнозування результату виконання асинхронного коду або ідентифікацію логічних помилок у фрагментах, згенерованих ШІ-асистентами.
- *Рівень 5 (Оцінювання, 30% питань)* - обґрунтування вибору конкретного архітектурного рішення (наприклад, Redux Toolkit vs Context API) або вибір методу деплою для MVP.

Приклад інноваційного тестового завдання (рівень «Аналіз»).

Студенту пропонується фрагмент коду для опрацювання HTTP-запиту (Fetch API), створений ШІ-агентом. Необхідно провести аудит коду, виявити

потенційну вразливість або помилку в асинхронних операціях та обрати варіант рефакторингу, що забезпечує стабільність інтерфейсу.

Рейтингова система та критерії оцінювання.

Загальний ваговий бал модульного контролю становить 40 балів. Кожен тест містить 50 питань з вагою 0,4 бала за правильну відповідь. Критерії оцінювання синхронізовані з рівнями міжнародної рамки DigComp 3.0 представлені в табл. 3.13.

Таблиця 3.13

**Синхронізація критеріїв оцінювання з рівнями міжнародної рамки
DigComp 3.0**

Бали	Рівень за DigComp 3.0	Характеристика досягнень студента
16 – 20	7–8 (Експертний)	Студент виконує завдання у повному обсязі, демонструє здатність виступати експертом коду згенерованого ШІ та пропонувати оптимізовані архітектурні рішення
10 – 15	5–6 (Просунутий)	Студент успішно аналізує логіку складних систем, проте припускається неточностей у питаннях безпеки або оптимізації швидкодії
5 – 9	4 (Середній)	Студент володіє базовою логікою застосування технологій, але потребує підказок при вирішенні нестандартних задач
1-4	< 3 (Базовий)	Студент надав менше 60% правильних відповідей, що свідчить про несформованість цілісної системи знань

Система заохочувальних балів.

Для стимулювання формування цифрової ідентичності студента, традиційні бонуси «за активність» замінено на показники професійної соціалізації розробника (сумарно до 10 балів):

- до 5 балів – активна участь у рецензуванні коду одногрупників у GitHub, виявлення багів та надання конструктивних коментарів;
- до 3 балів – успішне проходження мікромодулів або сертифікацій на зовнішніх платформах (FreeCodeCamp, Coursera, Meta Front-End Developer);

– до 2 балів – реалізація у власному проєкті додаткового функціоналу, що виходить за межі технічного завдання практикуму.

Такий підхід до контрольних заходів дозволяє реалізувати принцип співпраці з ІІІ та забезпечити неперервність оцінювання не лише результату (коду), а й знаннєвої здатності студента до системного аналізу в інтелектуальному веборієнтованому середовищі.

Нижче наведено приклад структури та конкретних завдань для МКР-1 (Розділ «Основи клієнтської розробки та JavaScript»), розроблений з урахуванням рівнів DigComp 3.0.

Приклад структури тестового пакета МКР.

- Загальна кількість питань – 50.
- Вага кожного питання – 0.4 бала.
- Максимальний бал – 20 балів.

Типові приклади запитань за рівнями.

Рівень «Усвідомлення» (II рівень – 20% питань).

Завдання. Студенту пропонується графічна схема DOM-дерева.

Необхідно описати принцип роботи методу fetch API для отримання даних із зовнішнього сервера та пояснити, чому виникає процес reflow у браузері при зміні певних CSS-властивостей.

Мета: Перевірка розуміння клієнт-серверної взаємодії.

Рівень «Аналіз» (IV рівень — 50% питань).

Завдання. Студенту надається фрагмент коду для обробки асинхронного запиту, згенерований ІІІ-асистентом (наприклад, ChatGPT), який містить логічну помилку в обробці станів Promise (pending, fulfilled, rejected).

Дія студента: Студент повинен провести експертизу коду, ідентифікувати помилку, яка призводить до «зависання» інтерфейсу, та обрати варіант рефакторингу, що забезпечує стабільність системи.

Відповідність DigComp 3.0: Напрямок 3.4 (Програмування) – здатність бути експертом коду.

Рівень «Оцінювання» (V рівень – 30% питань).

Завдання. Дано опис технічного завдання для MVP-проєкту (наприклад, To-Do List з авторизацією). Студенту необхідно обґрунтувати вибір методу деплою (Vercel vs GitHub Pages) та вибір технології адаптивної верстки (Flexbox vs Grid) для забезпечення вимоги Mobile First.

Мета: Перевірка здатності приймати професійні рішення та оцінювати ризики безпеки.

Критерії успішності:

- 18.0 – 20.0 балів (експертний рівень 7–8) – студент не лише знає синтаксис, а й здатний критично оцінювати AI-генеровані рішення та пропонувати безпечні архітектурні складники.
- 15.0 – 17.9 бала (просунутий рівень 5–6) – студент успішно дебажить код, але може припускатися неточностей у питаннях оптимізації продуктивності.
- 1 – 11 бала (нижче базового) – свідчить про наявність грубих порушень у логіці програмування або нездатність працювати з документацією.

Заохочувальний бонус. Якщо студент під час семестру брав активну участь у Code Review робіт одногрупників у GitHub, він може отримати до 5 додаткових балів до загального рейтингу дисципліни.

3.4 Види контролю та підходи до оцінювання знань студентів

Випускники технічних закладів вищої освіти ІТ-напрямів підготовки, зокрема майбутні бакалаври комп'ютерних наук, мають високий попит на вітчизняному ІТ-ринку праці. Щороку ЗВО здійснюють підготовку значної кількості бакалаврів ІТ-спеціальностей, однак, за даними досліджень, лише близько 25% випускників можуть успішно працевлаштуватися в ІТ-компаніях [98].. Значна частина молодих фахівців потребують подальшого розвитку фахових компетентностей для повної відповідності вимогам роботодавців. Важливим залишається питання підвищення якості професійної підготовки

майбутніх бакалаврів комп'ютерних наук, рівня їх професійної, зокрема ІК-компетентності, яка б відповідала сучасним потребам ІТ-ринку, світовим вимогам і вимогам роботодавців [99].

На сучасному етапі розвитку вітчизняної вищої освіти актуальною залишається проблема уніфікації інструментів оцінювання результатів навчання та рівня сформованості фахових компетентностей здобувачів освіти. Водночас на міжнародному рівні активно використовуються стандартизовані цифрові сервіси та платформи, інтеграція яких в освітній процес створює умови для залучення українських студентів до глобального цифрового освітнього простору. За таких умов традиційна функція педагогічного контролю поступово трансформується в систему моніторингу безперервного професійного розвитку, що реалізується через використання хмарних сервісів, аналітичних інструментів та аналізу даних про навчальну дисципліну здобувачів освіти в цифровому середовищі [100].

Проте вдосконалення форм та методів, спрямованих на створення цілісної системи безперервної освіти дозволяє виробити єдину систему оцінювання динаміки руху бакалаврів комп'ютерних наук у навчальній діяльності.

Із введенням в освітній процес ЗВО нових технологій з'являються нові нетрадиційні форми і методи оцінювання навчальних досягнень студентів ЗВО. У рамках нашого дослідження розглянемо дослідження науковців відносно питань оцінювання навчальних досягнень студентів, зокрема майбутніх бакалаврів комп'ютерних наук ЗВО.

На думку В.М. Бочарнікової оцінювання, що здійснюється в ході контролю навчальних досягнень студентів, стимулює їх до активної навчально-пізнавальної діяльності, а отже, виступає навчаючим фактором, що зумовлює позитивні результати навчального процесу [101].

Сучасна світова практика підготовки ІТ-фахівців (зокрема атлантична модель) демонструє перехід до використання хмарних середовищ розробки та автоматизованих систем перевірки алгоритмів (Online Judges), інтегрованих безпосередньо у навчальні плани. Це забезпечує прозорість контролю та

дозволяє викладачу зосередитись на архітектурній експертизі, тоді як рутинна перевірка правильності коду виконується автоматично системами типу GitHub Actions.

В українських ВЗО кількість балів за одну дисципліну також нараховується по 100-бальній системі. Прикладом є положення про рейтингову систему оцінювання результатів навчання НТУУ «Київський політехнічний інститут ім. І.Сікорського» [102].

У своїх роботах Л. Огнівчук запропонувала систему оцінювання навчальних досягнень студентів ВЗО на основі компетентнісного підходу. Також, вона дослідила, що за кордоном прийнято виділяти три основні підходи до визначення і введення в практику освіти компетентнісного трактування якості результатів навчання: поведінковий підхід (США), функціональний підхід (Великобританія) і багатовимірний і цілісний підхід (Франція і Німеччина). Ці підходи з'явилися незалежно один від одного спочатку в США, потім у Великобританії і в останню чергу у Франції та Німеччині [100].

У своїх дослідженнях С. Г. Литвинова зазначає, що за допомоги формувального оцінювання можна підвищити ефективність контролю та оцінювання знань студентів. Ми погоджуємося, що формуюче оцінювання застосовується викладачами для отримання даних щодо поточного стану засвоєння знань студентами конкретної теми і визначення найближчих кроків щодо їх покращення [103].

Дієвими показниками дидактичної ефективності формувального оцінювання навчальних досягнень є періодичність, умотивованість навчально-пізнавальної діяльності, індивідуалізація та диференціація.

Особливості формувального оцінювання полягають в тому, що оцінюється робота студента в досягненні цілей навчання, а не його особистість; пропонується чіткий алгоритм визначення оцінки, зрозумілий студенту; увага акцентується на персональному прогресі студента, а не на оцінці. Форми проведення формувального оцінювання студентів пропонуються такі: рефлексивні техніки (сигнали рукою, карточками) для з'ясування і виявленням

складних питань; уточнюючі питання; аналітичні питання; міні-тести; перевірка творчих робіт з метою виявлення помилок тощо. [98]. Серед функцій виділяють: навчальну, стимулюючу, контролюючу.

Враховуючи позитивний досвід викладачів Національного технічного університету «Харківський політехнічний інститут», зокрема В.М. Кухаренка, щодо проведення експерименту із застосування таксономії Блума для оцінювання навчальних досягнень студентів, застосуємо цей підхід для розроблення авторської методики оцінювання навчальних досягнень майбутніх бакалаврів комп'ютерних наук.

Хочеться зауважити, що саме цей підхід покладено в основу контрольно-оцінювального блоку моделі використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук. Оцінювання рівня знань студентів-програмістів здійснюється за допомогою таксономії Блума, яка містить 6 рівнів складності. Кожне практичне завдання буде відповідати своєму рівню з визначеною кількістю балів [104]

При формуванні контрольно-оцінювального блоку цієї моделі, слід пам'ятати, що роботодавець ставить високі вимоги до працівників, а рівень знань, умінь і навиків профільних випускників не відповідає цим вимогам, оскільки носить здебільшого теоретичний характер. Це, у свою чергу, вимагає постійної корекції навчальних планів і навчальних дисциплін, що вивчаються у вищих закладах освіти, регулярної перепідготовки кадрів. [104]

При оцінюванні рівня знань, велика увага приділяється контролю за результатами самостійної роботи майбутнього бакалавра комп'ютерних наук, як невід'ємної складової частини навчально-виховного процесу, що має на меті забезпечити зворотній зв'язок «студент-викладач» і виявити на основі цього правильність її організації. балів [98].

Формування завдань здійснюється у відповідності до сфер пізнавальних (Cognitive Domain), емоційних (Affective Domain) та рухових (Psychomotor Domain) цілей. Пізнавальні цілі охоплюють усе, що пов'язано з набуттям знань і розвитком розумових навичок. Емоційні цілі містять у собі всі завдання,

пов'язані з формуванням цінностей, відношень, розвитком емоційного самоконтролю студентів. До рухових цілей належить розвиток рухових навичок, фізичної витривалості [104].

При складанні практичних завдань особлива увага приділяється сфері пізнавальних цілей, яка поділяється на наступні шість рівнів:

1. Пригадування (Knowledge Level) - нижчий рівень.
2. Усвідомлення (Comprehension Level).
3. Застосування (Application Level).
4. Аналізу (Analysis Level).
5. Оцінювання (Synthesis Level).
6. Створення (Evaluation Level) - вищий рівень.

Ця класифікація є класифікацією мислення, організованого за рівнями складності й дає викладачам і студентам можливість вчитися та діяти в інформаційно-освітньому просторі, забезпечує просту структуру для багатьох видів питань [104].

У системі оцінювання ECTS бальна система також має шість основних рівні (табл. 3.14).

Таблиця 3.14

Бальна система оцінювання навчальних досягнень студентів

Рівень за таксономією Блума	Кількість балів	Рівень за DigComp 3.0
VI. Створення	95-100 / A	7–8 (Експертний)
V. Оцінювання	85-95 / B	5–6 (Просунутий)
III–IV. Застосування / Аналіз	75-85 / C	4 (Середній)
I–II. Пригадування / Усвідомлення	70-74 / D	1–3 (Базовий)

У процесі вивчення дисципліни застосовуються наступні види контролю: попередній (вхідний) контроль; поточний контроль; проміжний (тематичний) контроль; семестровий (підсумковий) контроль; самоконтроль. Підсумковий семестровий контроль у межах розробленої методики базується на публічній презентації GitHub-портфоліо, яке містить не лише вихідний код, а й розгорнуті релізи проєктів на платформах Vercel або Netlify. Такий підхід дозволяє

об'єктивно оцінити здатність бакалавра керувати повним життєвим циклом програмного продукту (SDLC) у співпраці з ШІ як експерта та промпт-інженера.

Кожен вид контролю має свою мету та форми реалізації, які представлені у таблиці 3.15.

Таблиця 3.15

Опис видів контролю

Вид контролю	Мета	Основні форми
Попередній (вхідний)	Визначити початковий рівень знань, умінь та навичок	Тести, опитування, діагностичні роботи
Поточний	Перевірити засвоєння матеріалу на кожній темі	Усне та письмове опитування, мінітести, практичні завдання
Проміжний (тематичний, модульний)	Оцінити результати вивчення теми чи модуля	модульна контрольна, захист лабораторних
Семестровий (підсумковий)	Визначити рівень досягнення програмних результатів	Іспит, залік, підсумковий проєкт, курсова робота
Самоконтроль	Розвинути навички самооцінювання	Самостійне тестування, рефлексія, взаємоперевірка

Поточний контроль.

Рейтинг студента з дисципліни складається з балів, які він отримує за:

- виконання та захист 6 лабораторних робіт;
- виконання модульної контрольної роботи (МКР);
- проміжні тести для перевірки засвоєння вивченого матеріалу;
- заохочувальні бали;
- штрафні бали (віднімаються від загальної суми оцінювання комп'ютерного практикуму);

Семестровий контроль – залік.

Запропонована система контролю трансформується з формального адміністративного підрахунку балів у цілісну методику діагностики професійного зростання бакалавра комп'ютерних наук. Акцент у методиці переноситься з перевірки статичного кінцевого файлу на безперервний аналіз «цифрового сліду» студента у хмарному середовищі розробки, що дозволяє об'єктивно оцінити ітеративності розробки та академічної доброчесності студента.

Синхронізація рівнів інтелектуальної діяльності за таксономією Блума з експертними рівнями 7–8 рамки DigComp 3.0 забезпечує відповідність навчальних досягнень реальним вимогам ІТ-індустрії періоду розвитку інтелектуального вебу, де фінальним показником компетентності стає публічне портфоліо.

Висновки до розділу 3

Обґрунтування методики використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук розкриває практичний аспект і переваги та слугує підґрунтям для системного впровадження в освітню практику викладачів.

Визначення методики як інтегрованої дидактичної системи, що базується на принципах неперервності та інтелектуальної співпраці, дозволяє вийти за межі простого вивчення синтаксису мов і забезпечує досягнення високих рівнів цифрової компетентності за рамкою DigComp 3.0.

Перехід від локальних налаштувань до хмаро орієнтованих середовищ розробки (GitHub Codespaces, Replit) забезпечує ідентичність навчальних умов для всіх студентів, вирішує проблему «на моєму комп'ютері не працює» та реалізує принцип неперервності навчання, що дозволяє студенту миттєво продовжувати роботу з будь-якого пристрою.

Впровадження технології використання ІІІ як інтелектуального помічника дає змогу студенту отримати інтелектуальну підтримку в режимі

24/7, що суттєво прискорює дебагінг коду та дозволяє зосередитися на високорівневій архітектурі та логіці алгоритмів, а не на механічному написанні синтаксичних конструкцій.

Використання авторської пропорції змішаного навчання (30% традиційного, 50% дистанційного та 20% проєктного) сприяє швидкій адаптації студентів до вимог роботодавців, оскільки навчальний процес моделює реальний життєвий цикл розробки програмного забезпечення (SDLC).

Аналіз історії фіксацій змін у GitHub замість перевірки статичного фінального файлу дає викладачу інструмент для підтвердження академічної доброчесності та дозволяє оцінити ітеративність розробки, що унеможливорює плагіат готового коду одним блоком.

Орієнтація на створення мінімально життєздатного продукту (MVP) з автоматичним розгортанням на платформах Vercel чи Netlify забезпечує студенту миттєвий практичний результат у вигляді діючого вебзастосунку та готове публічне портфоліо, що є ключовою перевагою при працевлаштуванні.

Синхронізація практичних завдань із рівнями знаннєвої діяльності за Блумом надає студенту можливість самостійно обирати складність завдань, що стимулює внутрішню мотивацію до переходу від репродуктивного відтворення до рівня «створення» інтелектуальних продуктів.

Для підтвердження ефективності розробленої методики та обґрунтування гіпотези дослідження щодо формування компетентності з програмування майбутніх бакалаврів комп'ютерних наук було організовано та проведено педагогічний експеримент, результати якого викладено в наступному розділі.

Список використаних джерел до Розділу 3

1. Гончаренко С. У. та ін. Методика навчання і наукових досліджень у вищій школі. URL: <https://i.twirpx.link/file/194128/> (дата звернення 27.03.26)
2. Proskura S. L., Lytvynova S. H. The approaches to Web-based education of computer science bachelors in higher education institutions. Cloud technologies

in education. CTE-2019. *CEUR Workshop Proceedings*, 2020. 2643. P.609-625.
URL: <https://ceur-ws.org/Vol-2643/paper36.pdf> (date of access: 27.03.2026).

3. Проскура С. Л., Литвинова С. Г., Кронда О. П. Засоби організації дистанційного навчання в період карантину 2020 року в закладах вищої освіти України. *Екстрене дистанційне навчання в Україні: монографія/ за ред. В. М. Кухаренка, В. В. Бондаренка. Харків : КП "Міська друкарня", 2020. С. 299–313.*

URL: https://duan.edu.ua/images/News/UA/Departments/Management/2020/monograph_ekstr_dyst_navch.pdf (дата звернення: 27.03.26).

4. Proskura S. L., Lytvynova S. H. The use of WEB-oriented technologies in the process of WEB-programming teaching for technical universities students. *ICTERI 2021: Proceedings of the 17th International Conference on ICT in Education, Research and Industrial Applications*, 2021. P. 129–141.
DOI: <https://doi.org/10.31812/educdim.4724> (date of access: 27.03.2026).

5. Проскура С. Л., Литвинова С. Г. Підготовка фахівців з інформаційних технологій у закладах вищої освіти: стан, проблеми і перспективи. *Інформаційні технології в освіті*, 2018. № 35. С. 72–88.
DOI: <https://doi.org/10.14308/ite000668> (дата звернення: 27.03.26).

6. Наказ про затвердження стандарту вищої освіти за спеціальністю 122 «Комп'ютерні науки» для першого (бакалаврського) рівня вищої освіти.
URL: <https://mon.gov.ua/static-objects/mon/sites/1/vishcha-osvita/zatverdzeni%20standarty/2019/07/12/122-kompyut.nauk.bakalavr-1.pdf>
(дата звернення: 27.03.26).

7. Проскура С. Л. Застосування інтелект-карт для підвищення якості та ефективності навчання студентів курсу програмування вищих навчальних закладів. *Актуальні питання природничо-математичної освіти*, 2017. С. 220- 228 URL: http://fizmatsspu.sumy.ua/Konferencii/sbor/appmo/appmo_v7-8_2016.pdf#page=220 (дата звернення: 27.03.26).

8. Novak J. D., Cañas A. J. The Theory Underlying Concept Maps and How to Construct and Use Them. *Technical Report IHMC*, 2006.

URL: https://cmap.ihmc.us/docs/theory-of-concept-maps.php?utm_source (date of access: 27.03.2026).

9. How To Make A Mind Map. URL: <https://www.mindmapping.com/> (date of access: 27.03.2026).

10. Бирка М. Теоретико-методичні основи використання інтелектуальних технологій у професійній діяльності вчителів природничо-математичних дисциплін. *Нова педагогічна думка*, 2013. № 3. С. 3–6. URL: http://nbuv.gov.ua/UJRN/Npd_2013_3_2 (дата звернення 27.03.26)

11. Терещенко Н. В. Інтелект-карти – сучасні інноваційні соціальні технології навчання в системі освіти. *Вчені записки Київського національного економічного університету імені Вадима Гетьмана*, 2012. Вип. 14, ч. 1. С. 139–145. URL: https://ir.kneu.edu.ua/items/8562bc95-25eb-4b32-a92b-35d77d596b5f?utm_source (дата звернення 27.03.26)

12. Тулашвілі Ю. Й., Олексів Н. А. Інтенсифікація навчальної діяльності інженерів-педагогів комп'ютерного профілю за допомогою інтелект-карт. *Педагогічний часопис Волині*. 2016. № 1. С. 46–51. URL: https://irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?C21COM=2&I21DBN=UJRN&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF%2Fpchv_2016_1_11.pdf&P21DBN=UJRN (дата звернення 27.03.26)

13. Машкіна В. В. Використання ментальних карт як інноваційних засобів викладання географії. *Проблеми безперервної географічної освіти і картографії*, 2012. №16. URL: <http://periodicals.karazin.ua/pbgok/article/view/4115> (дата звернення 03.04.2026)

14. Загальна характеристика організаційних форм навчання. *Studies.in.ua*. URL: <https://studies.in.ua/ekzamen-pedagogika/1420-zagalna-harakteristika-organizacynih-form-navchannya.html> (дата звернення: 27.03.2026).

15. The Pedagogic Value of Live Coding. *Teaching College*, University of Manchester. 2015. URL: <https://www.teachingcollege.fse.manchester.ac.uk/the-pedagogic-value-of-live-coding/> (date of access: 27.03.2026).

16.About CodePen. URL: <https://codepen.io/about> (date of access: 27.04.2026).

17.GitHub Codespaces. URL: <https://www.geeksforgeeks.org/git/github-codespaces/> (date of access: 27.04.2026).

18.Shihab M. I. H., Hundhausen C. D., Tariq A., Haque S., Qiao Y., Mulanda B. W. The Effects of GitHub Copilot on Computing Students' Programming Effectiveness, Efficiency, and Processes in Brownfield Coding Tasks. *arXiv*, 2025. P. DOI: <https://doi.org/10.48550/arXiv.2506.10051> (date of access: 27.03.2026).

19.GitHub Copilot documentation. *GitHub Docs*. URL: <https://docs.github.com/en/copilot> (date of access: 27.04.2026).

20.Sánchez-Vera F. Developing Effective Educational Chatbots with GPT: Insights from a Pilot Study in a University Subject. *Trends in Higher Education*, 2024. Vol. 3(1). P. 155–168. DOI: <https://doi.org/10.3390/higheredu3010009> (date of access: 27.04.2026).

21.Casheekar A., Lahiri A., Rath K. et al. A contemporary review on chatbots, AI-powered virtual conversational agents, ChatGPT: Applications, open challenges and future research directions. *Computer Science Review*, 2024. Vol. 52. Art. 100632. DOI: <https://doi.org/10.1016/j.cosrev.2024.100632> (date of access: 27.04.2026).

22.Sharma A., Karmakar S., Kancharla G. P., Bichhawat A. On the Prevalence and Usage of Commit Signing on GitHub: A Longitudinal and Cross-Domain Study. *arXiv*, 2025. DOI: <https://doi.org/10.48550/arXiv.2504.19215> (date of access: 27.04.2026).

23.Oliveira E., Fu M., Thongtanunam P., López-Pernas S., Saqr M. AI-Assisted Code Review as a Scaffold for Code Quality and Self-Regulated Learning: An Experience Report. *arXiv*, 2026. URL: <https://arxiv.org/abs/2604.23251> (date of access: 27.04.2026).

24. What is automated code review? Tools and best practices. *Wiz*, 2025. URL: <https://www.wiz.io/academy/automated-code-review> (date of access: 27.04.2026).

25. Jain, N., Jain, A. Software Development Life Cycle: A Detailed Study. *International Journal of Advanced Research in Computer Science* URL, 2017. Vol 2 No 3 URL: <https://ijarcs.info/index.php/Ijarcs/article/view/512> (date of access: 27.04.2026).

26. Hata H. et al. GitHub Discussions: An Exploratory Study of Early Adoption. *arXiv*. 2021. URL: <https://arxiv.org/abs/2102.05230> (date of access: 27.04.2026).

27. GitHub Docs. About GitHub Discussions. URL: <https://docs.github.com/en/discussions/collaborating-with-your-community-using-discussions/about-discussions> (date of access: 27.04.2026).

28. Knight. S. Mozilla's Hubs is a browser-based VR chatroom for every headset. *TechSpot*, 2018. URL: <https://www.techspot.com/news/74342-mozilla-hubs-browser-based-vr-chatroom-every-headset.html> (date of access: 27.04.2026).

29. CodeSandbox built for scale. URL: <https://codesandbox.io/> (date of access: 27.04.2026).

30. Савченкова М. В. Формування soft skills як основа професійної мобільності фахівців у сучасному освітньому процесі. *Меньальне здоров'я*, 2025. Т. №1. DOI: <https://doi.org/10.32782/3041-2005/2025-1.5> (дата звернення: 26.04.2026).

31. Nadiia Samoliuk. Soft skills and hard skills in the labor market: essence and components. *Bulletin of NUWEE*. URL, 2021. Vol. 4. No. 96. DOI: <https://doi.org/10.31713/ve4202122> (date of access: 27.04.2026).

32. What is Trello used for? Our fave project management software explained. URL: <https://www.atlassian.com/blog/trello/what-is-trello-used-for> (date of access: 27.04.2026).

33. What is a minimum viable product (MVP)? How to get started.. URL: <https://www.atlassian.com/agile/product-management/minimum-viable-product> (date of access: 27.04.2026).
34. Official Website Coursera. URL: <https://www.coursera.org/> (date of access: 27.03.2026).
35. Official Website Udemy. URL: <https://www.udemy.com> (date of access: 27.03.2026).
36. Official Website Prometheus. URL: <https://prometheus.org.ua> (date of access: 27.03.2026).
37. FreeCodeCamp Curriculum. URL: <https://www.freecodecamp.org/learn> (date of access: 27.04.2026).
38. Harvard University. CS50's Introduction to Computer Science. edX. URL: <https://www.edx.org/cs50> (date of access: 27.04.2026).
39. Redux Toolkit Documentation. URL: <https://redux-toolkit.js.org/> (date of access: 27.04.2026).
40. What is CI/CD?. URL: <https://github.com/resources/articles/ci-cd>. (date of access: 27.04.2026).
41. GitHub Actions Documentation. URL: <https://docs.github.com/en/actions> (date of access: 27.04.2026).
42. Проскура С. Л. Модель формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*, 2019. № 3 (21). С. 104–112. URL: <https://fmo-journal.fizmatsspu.sumy.ua/publ/4-1-0-522> (дата звернення 27.03.26)
43. Proskura S. L., Lytvynova S. H., Kronda O. P. Students academic achievement assessment in higher education institutions. *CTERI 2020: Integration, Harmonization and Knowledge Transfer. Volume II: Workshops*. Kharkiv, Ukraine, 2020. С. 734 – 745. URL: <http://ceur-ws.org/Vol-2732/> (date of access: 27.04.2026).
44. Ткачук Г В. Зарубіжний досвід реалізації змішаного навчання. *Фізико математична освіта*, 2018. Випуск. 1(15). С. 98 – 102.

URL: <https://cyberleninka.ru/article/v/zarubizhniy-dosvid-realizatsiyi-zmishanogo-navchannya> (дата звернення 27.03.26)

45. Wong L., Looi C. The conceptual niche of seamless learning: an invitation to dialogue. In: *Seamless Learning: Perspectives, Challenges and Opportunities*. Springer, 2019. P. 3–27. DOI: https://doi.org/10.1007/978-981-13-3071-1_1

46. Проскура С. Л. Педагогічні умови використання WEB-орієнтованих технологій у підготовці бакалаврів з інформаційних технологій та систем. *Інформаційні технології і засоби навчання*, 2021. Т. 84, № 4. С. 130–132.

URL: https://lib.iitta.gov.ua/id/eprint/740554/1/Збірник_тез_звітної_2024_v2.pdf (дата звернення 27.03.26).

47. Visual Studio Code. URL: <https://code.visualstudio.com> (date of access: 27.04.2026).

48. WebStorm. The JavaScript and TypeScript IDE. URL: <https://www.jetbrains.com.cn/en-us/webstorm/> (date of access: 27.03.2026).

49. Sublime Text. URL: <https://www.sublimetext.com> (date of access: 27.03.2026).

50. Visual Studio Community. URL: <https://biblprog.org.ua/ua/visual-studio-community/> (date of access: 27.03.2026)

51. Eclipse. URL: <https://www.eclipse.org/downloads/> (date of access: 27.03.2026)

52. NetBeans/
URL: <https://netbeans.apache.org/front/main/download/index.html> (date of access: 27.03.2026)

53. Node.js. URL: <https://nodejs.org/en/download> (date of access: 27.03.2026)

54. Zoom. URL: <https://zoom.us/download> (date of access: 27.03.2026)

55. Google Meet. URL: <https://meet.google.com> (date of access: 27.03.2026).

56. My Own Conference. URL: <https://myownconference.com/> (date of access: 27.03.2026)
57. Open Meeting. URL: <https://openmeetings.apache.org> (date of access: 27.03.2026)
58. Sign in to Microsoft Teams. URL: <https://www.microsoft.com/uk-ua/microsoft-teams/log-in>
59. 12 найкращих платформ для віртуальних конференцій у 2022 році. URL: <https://staff-capital.com/12-найкращих-платформ-для-віртуальних-к/> (date of access: 27.03.2026)
60. Bounegru L. The platformisation of software development: Connective coding and platform vernaculars on GitHub. *New Media & Society*, 2024. Vol. 30, Issue 6. URL: <https://journals.sagepub.com/doi/abs/10.1177/13548565231205867> (date of access: 27.03.2026).
61. Official Website CyberBionic Systematics. URL: <https://edu.cbsystematics.com/en> (date of access: 27.03.2026).
62. Gamage S. H. P. W., Ayres J. R., Behrend M. B. A systematic review on trends in using Moodle for teaching and learning. *International Journal of STEM Education*, 2022. Vol. 9. N. 9. URL: https://link.springer.com/article/10.1186/s40594-021-00323-x?utm_source (date of access: 27.03.2026).
63. Auliasari M. M., Pratama A. D. Efektivitas e-learning pada pendidikan tinggi dengan menggunakan LMS Moodle dan Google Classroom. *Jurnal Inovasi Akademik*, 2023. Vol 2. No 1. URL: https://journal.ugm.ac.id/v3/jinovak/article/view/11719?utm_source (date of access: 27.03.2026).
64. Official Website Replit. URL: <https://replit.com/> (date of access: 27.03.2026).
65. Official Website JSFiddle. URL: <https://jsfiddle.net/> (date of access: 27.03.2026).

66. Інформатика: великий довідник учня. Харків : Видавництво «Школа», 2019. 640 с.

URL: https://subjectum.eu/dovidnik/inform/11.html?utm_source

67. Proskura S. L., Lytvynova S. H. Organization of independent studying of future bachelors in computer science within higher education institutions of Ukraine. *ICTERI 2018 (3L-Person 2018)*, 2018. P. 348–358. URL: http://ceur-ws.org/Vol-2104/paper_160.pdf (date of access: 27.03.2026).

68. Harish, Rajora. Elements of Modern Web Design That You Need to Know, 2021. URL: <https://dev.to/harishrajora12/20-elements-of-modern-web-design-that-you-need-to-know-3mh7> (date of access: 27.03.2026).

69. What Is An Integrated Development Environment (IDE)?. URL: <https://www.veracode.com/security/integrated-development-environment> (2021) (date of access: 27.03.2026).

70. General Assembly adopts landmark resolution on artificial intelligence. URL: <https://news.un.org/en/story/2024/03/1147831> (date of access: 27.03.2026).

71. United Nations Ai Resolution: A Significant Global Policy Effort To Harness The Technology For Sustainable Development. Available at: URL: <https://executive.graduateinstitute.ch/communications/news/united-nations-ai-resolution-significant-global-policy-effort-harness> (date of access: 27.03.2026).

72. Harris K. Statement by the Vice President on the UN General Assembly Resolution on Artificial Intelligence/ *The American Presidency Project*, 21.03.2024. URL: <https://www.presidency.ucsb.edu/documents/statement-the-vice-president-the-un-general-assembly-resolution-artificial-intelligence> (дата звернення: 27.03.2026).

73. Концепція розвитку штучного інтелекту в Україні. URL: <https://zakon.rada.gov.ua/laws/show/1556-2020-%D1%80#Text> (дата звернення 27.03.26)

74. Політика використання штучного інтелекту для академічної діяльності в КПІ ім. Ігоря Сікорського, наказ від 29.12.2023 р.

№ НОН/393/2023. URL: <https://osvita.kpi.ua/node/1225> (дата звернення 27.03.26)

75. Ahmad T. Scenario based approach to re-imagining future of higher education which prepares students for the future of work. *Higher Education, Skills and Work-Based Learning*, 2020. Vol. 10, № 1. P. 217–238. URL: <https://doi.org/10.1108/HESWBL-12-2018-0136> (date of access: 27.03.2026).

76. Chen L., Chen P., Lin Z. Artificial Intelligence in Education. IEEE Access, 2020. Vol. 8. P. 75264–75278. DOI: <https://doi.org/10.1109/ACCESS.2020.2988510> (date of access: 27.03.2026).

77. Yuensook T., Jantakoon T., Limpinan P. AI-driven adaptive learning systems in higher education: a systematic review. *Journal of Education and Learning*, 2025. DOI: <https://doi.org/10.5539/jel.v15n2p117> (date of access: 27.03.2026).

78. Luan H., Geczy P., Lai H., Gobert J., Yang S. J. H., Ogata H., Baltes J., Guerra R., Li P., Tsai C. Challenges and Future Directions of Big Data and Artificial Intelligence in Education. *Frontiers in Psychology*, 2020. Vol. 11. Article 580820. URL: <https://doi.org/10.3389/fpsyg.2020.580820> (date of access: 27.03.2026).

79. Nguyen A., Kremantzis M., Essien A., Petrounias I., Hosseini S. Editorial: Enhancing Student Engagement Through Artificial Intelligence (AI): Understanding the Basics, Opportunities, and Challenges. *JUTLP*, 2024. Vol. 21, № 06. URL: <https://doi.org/10.53761/caraaq92> (date of access: 27.03.2026).

80. Chatterjee S., Bhattacharjee K. K. Adoption of artificial intelligence in higher education: a quantitative analysis using structural equation. *Educ Inf Technol*, 2020. Vol. 25. C. 3443–3463. DOI: <https://doi.org/10.1007/s10639-020-10159-7> (date of access: 27.03.2026).

81. Ting W. та ін. Вплив генеративного AI на студентів. *Applied Sciences*, 2023. Vol. 13, Issue 11, Art. 6716. DOI: <https://doi.org/10.3390/app13116716> (date of access: 27.03.2026).

82. Ouyang F., Jiao P. Три парадигми AI в освіті. *Computers and Education: Artificial Intelligence*, 2021. Vol. 2. DOI: <https://doi.org/10.1016/j.caeai.2021.100020> (date of access: 27.03.2026).
83. Chassignol M., Khoroshavin A., Klimova A., Bilyatdinova A. Тенденції розвитку штучного інтелекту в освіті. *Procedia Computer Science*. 2018, Vol. 136. P. 16–24. URL: <https://www.sciencedirect.com/science/article/pii/S1877050918315382> (date of access: 27.03.2026).
84. Hooda M., Rana C., Dahiya O., Rizwan A., Hossain M. S. Artificial Intelligence for Assessment and Feedback to Enhance Student Success in Higher Education *Mathematical Problems in Engineering*, 2022. Vol. 2022. Article ID 5215722. P. 1–19. DOI: <https://doi.org/10.1155/2022/5215722> (date of access: 27.03.2026).
85. Keleş P. U., Aydın S. University Students' Perceptions About Artificial Intelligence. *Shanlax International Journal of Education*, 2021. Vol. 9, Special Issue 1. P. 212–220. DOI: <https://doi.org/10.34293/education.v9iS1-May.4014> (date of access: 27.03.2026).
86. Годунова А. В., Толочко С. В. Сприйняття AI підлітками / А. В. Годунова, С. В. Толочко // *Суспільство та національні інтереси*, 2024. № 1. С. 216–227. DOI: [https://doi.org/10.52058/3041-1572-2024-1\(1\)-216-227](https://doi.org/10.52058/3041-1572-2024-1(1)-216-227)
87. Kong S.-Ch., Cheung W. M., Zhang G. Оцінювання курсу AI-грамотності. *Computers and Education: Artificial Intelligence*, 2021. Vol. 2. DOI: <https://doi.org/10.1016/j.caeai.2021.100026> (date of access: 27.03.2026)..
88. Cheng L. та ін. Валідація інструменту оцінювання уявлень студентів про AI. *Journal of Interactive Learning Research*, 2023. Vol. 34, Issue 2. P. 275–311. URL: <https://www.learntechlib.org/primary/p/222246/> (date of access: 27.03.2026).
89. Algerafi M., Zhou Y., Alfadda H., Wijaya T. Розуміння чинників впровадження AI-роботів студентами. *IEEE Acces*, 2023. Vol. 11.

URL: <https://ieeexplore.ieee.org/abstract/document/10247529> (дата звернення: 27.03.2026) (date of access: 27.03.2026).

90. Литвинова С. Г., Рашевська Н. В., Проскура С. Л. Використання штучного інтелекту у навчанні студентів мов програмування. *Матеріали IX Міжнародного воркшопу з професійної перепідготовки та навчання впродовж життя з використанням ІКТ: особистісно орієнтований підхід* (3L-Person 2024), 2024. CEUR Workshop Proceedings, 2024. Т. 3781. С. 10–29. URL: <https://ceur-ws.org/Vol-3781/paper01.pdf>

91. Lomakina L. V Students' independent study as a component in the professional training of future specialists. *Scientific Notes. Series: Pedagogical Sciences*, 2021.P.213-217 DOI: <https://doi.org/10.36550/2415-7988-2021-1-194-213-217> (дата звернення: 27.03.2026).

92. Болюбаш Я. Я. Організація навчального процесу у вищих навчальних закладах : навчальний посібник для студентів закладів вищої освіти. Київ, 1997. 64 с.

93. Зінковський Ю. Ф. Самостійна робота студентів. *Енциклопедія освіти*. Акад. пед. наук України; гол. ред. В. Г. Кремень, Київ, 2008. С. 804.

94. Бирта Г. О., Бургу К. К. Методологія та організація наукових досліджень : навчальний посібник. Київ : Центр учбової літератури, 2014. 142 с

95. Закон України «Про освіту» : Закон України від 05.09.2017 № 2145-VIII. URL: <http://zakon5.rada.gov.ua/laws/show/2145-19> (дата звернення: 27.03.2026).

96. Гришко Л. В. Методична система навчання основ програмування майбутніх інженерів-програмістів : дис. ... канд. пед. наук. Київ : Національний педагогічний університет імені М. П. Драгоманова, 2009. 276 с. URL: <https://enpuir.udu.edu.ua/entities/publication/6ae8bdf1-d26f-46b0-8373-0a0d8dc0254e> (дата звернення: 27.03.2026).

97. Литвинова С. Г. Організаційно-педагогічні проблеми впровадження інформаційно-комунікаційних технологій у загальноосвітніх навчальних

- зкладах. *Інформаційні технології і засоби навчання*, 2009. № 6(14)
URL: <https://journal.iitta.gov.ua/index.php/itlt/article/view/210>
98. Проскура С. Л., Литвинова С. Г., Кронда О. П. Оцінювання рівня знань бакалаврів комп'ютерних наук у закладах вищої освіти. *Сучасні тенденції та фактори розвитку педагогічних та психологічних наук в Україні та країнах ЄС* : матеріали міжнародної науково-практичної конференції (Люблін, Польща), 2020. С. 250 – 254.
URL: <http://www.baltijapublishing.lv/omp/index.php/bp/catalog/download/68/1534/3505-1> (дата звернення: 27.03.26).
99. Проскура С. Л., Литвинова С. Г. Формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*. 2019. № 2 (20). DOI: <https://doi.org/10.31110/2413-1571-2019-020-2-022>
URL: https://fmo-journal.fizmatsspu.sumy.ua/journals/2019-v2-20/2019_2-20-Proskura_Lytvynova_FMO.pdf (дата звернення: 27.03.2026).
100. Огнівчук Л. М. Оцінювання навчальних досягнень студентів вищих навчальних закладів на основі компетентнісного підходу. *Освітологічний дискурс*, 2014. № 3 (7). С. 154–166.
URL: https://od.kubg.edu.ua/index.php/journal/article/view/128?utm_source (дата звернення: 27.03.2026).
101. Бочарнікова В. М. Стимулююча функція контролю знань, умінь і навичок студентів вищої школи : автореф. дис. ... канд. пед. наук : 13.00.01. Київ, 1999. 23 с. (дата звернення: 27.03.2026).
102. Положення про систему оцінювання результатів навчання в КПІ ім. Ігоря Сікорського: *КПІ ім. Ігоря Сікорського*. URL: <https://osvita.kpi.ua/node/37> (дата звернення: 27.03.2026).
103. Литвинова С. Г. Модель використання системи комп'ютерного моделювання для формування компетентностей учнів з природничо-математичних. *Фізико-математична освіта*, 2019. Т. 1(19). С.108–115
URL: <https://repository.sspu.edu.ua/items/4d7f10c8-e2be-4ea8-8aad-c255b14fb32e> (дата звернення: 27.03.2026).

104. Кухаренко В. М., Березенська С. М., Бугайчук К. Л., Олійник Т. О., Рибалко О. В. Теорія і практика змішаного навчання: монографія. Харків: НТУ «ХПІ», 2016. 284 с. URL: <http://repository.kpi.kharkov.ua/handle/KhPI-Press/23536> (дата звернення: 27.03.2026).

У розділі були використані праці автора: [2], [3], [4], [5]. [7], [42], [43], [46]. [67], [98], [99].

РОЗДІЛ 4. АНАЛІЗ ТА ІНТЕРПРЕТАЦІЯ РЕЗУЛЬТАТІВ ЕКСПЕРИМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ

З метою експериментальної перевірки гіпотези дослідження щодо ефективності методики використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук був проведений педагогічний експеримент.

Педагогічний експеримент – це науково обґрунтована і добре продумана система організації педагогічного процесу, направлена на відкриття нового педагогічного знання, перевірки і обґрунтування заздалегідь розроблених наукових припущень, гіпотез [1].

Як зазначає вчений С. Гончаренко [1], експеримент – «це комплексний метод дослідження, який забезпечує науково-об'єктивну і доказову перевірку правильності обґрунтованої на початку дослідження гіпотези. Він дає можливість глибше, ніж інші методи, перевірити ефективність тих чи інших інновацій в галузі навчання, порівняти значущість різних факторів у структурі педагогічного процесу й обрати найкраще (оптимальне) для відповідних ситуацій їх поєднання, виявити необхідні умови реалізації певних педагогічних завдань. Експеримент дає можливість виявити стійкі, необхідні, істотні зв'язки між повторюваними явищами, тобто вивчати закономірності, характерні для педагогічного процесу. Експеримент проводиться в тому випадку, якщо немає можливості довести те чи інше твердження іншим способом і, звичайно, тоді, коли існують сумніви, вибір, альтернатива. Експеримент вимагає від дослідника високої методологічної культури, старанного опрацювання його програми і надійного критеріального апарату, який дає можливість фіксувати ефективність освітнього процесу» [1].

Педагогічний експеримент – науково поставлений дослід у галузі навчальної чи виховної роботи, спостереження досліджуваного педагогічного явища в спеціально створених і контрольованих дослідником умовах. При цьому встановлюється залежність між тим чи іншим впливом або умовою

навчання й виховання та його результатом. Одержання точних і вірогідних результатів педагогічного експерименту залежить, головним чином, від теоретичної позиції дослідника, вираженої в робочій гіпотезі і методиці дослідження. Експеримент відрізняється від спостереження активним втручанням у ситуацію дослідника, який здійснює планомірне маніпулювання однією або кількома змінними (факторами) і реєстрацію супутніх змін у поведінці об'єкта, що вивчається [1], [2].

У педагогічних дослідженнях експеримент є одним із основних методів перевірки ефективності нових методик технологій і засобів навчання. Залежно від умов проведення та мети дослідження педагогічний експерименти поділяються на різні види. Зокрема, за умовами проведення експерименти поділяються на природні та лабораторні.

Природний експеримент проводиться у звичайних умовах освітнього процесу без суттєвого порушення навчальної діяльності учасників. Лабораторний експеримент здійснюється у спеціально створених умовах, що дає змогу більш точно контролювати фактори педагогічного впливу [3].

— За ознаками мети педагогічний експерименти поділяються на: констатувальні (діагностичні, контрольні) — спрямовані на визначення початкового стану досліджуваного явища;

— пошукові — використовуються для виявлення нових факторів умов або шляхів розв'язання педагогічної проблеми;

За класифікацією експериментів, яку наводить С. Гончаренко, педагогічні експерименти поділяють за умовами проведення на природні й лабораторні, а за ознаками мети — на констатувальні (діагностичні, контрольні) та формувальні (перетворювальні) [1], [2],[4].

Педагогічний експеримент включав два етапи:

- констатувальний;
- формувальний;

4.1. Організація та хід констатувального етапу педагогічного експерименту

Педагогічний експеримент проводився у 2 етапи (констатувальний і формувальний) з метою перевірки ефективності авторської методики використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук.

Експериментальна робота здійснювалась продовж 8 років на базах кафедри інформаційних систем та технологій факультету інформатики та обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», кафедри інформаційних технологій Європейської школи бізнесу Міжнародного європейського університету, факультету інформаційних технологій та математики Київського університету імені Бориса Грінченка, кафедри програмних засобів факультету комп'ютерних наук і технологій Національного університету «Запорізька політехніка», кафедри комп'ютерних наук математичного факультету Запорізького національного університету.

Констатувальний етап експерименту охопив понад 300 студентів технічних спеціальностей («Комп'ютерні науки», «Інженерія програмного забезпечення», «Комп'ютерна інженерія», «Інформаційні системи та технології»), що забезпечує репрезентативність отриманих результатів.

На цьому етапі експерименту було встановлено, що сучасний стан підготовки майбутніх фахівців характеризується певним розривом між рівнем доуніверситетської освіти та вимогами ІТ-ринку. Зокрема, з'ясувалося, що 36,2% студентів взагалі не вивчали жодної мови програмування до вступу в університет, а ще 36,2% володіли лише застарілою мовою Pascal [5].

У межах університетської програми пріоритетними залишаються мови C++ (69,5%) та C# (64,8%), які вивчаються з першого курсу, тоді як веборієнтована мова JavaScript (50,5%) здебільшого включається до освітньої підготовки студентів на старших курсах. Проте аналіз професійної діяльності

студентів 4-го курсу свідчить про домінування вебтехнологій: лідером у реальних проєктах є JavaScript (51,5%), а мова TypeScript (29,7%) демонструє стрімку динаміку зростання популярності. Найбільш затребуваним середовищем розробки (IDE) як у навчанні (61,5%), так і в практичній роботі (45,1%–61,5%) визнано Microsoft Visual Code [5].

Незважаючи на активну цифровізацію, рівень застосування спеціалізованих вебінструментів залишається низьким. Лише 11,8% студентів використовують веборієнтовані інтелект-карти для структурування знань, а близько 89% респондентів не застосовують хмарні компілятори у самостійній роботі. Крім того, 78% студентів не використовують жодних автоматизованих систем перевірки завдань, хоча серед наявних рішень найбільш популярним є портал E-olymp.

Результати опитування також показують, що онлайн-платформи, такі як Coursera та Prometheus найбільше обираються студентами, щоб повністю опанувати навчальні матеріали та програми, у той час, як інші, популярні за кордоном ресурси, такі як CS50 (Гарвард) чи Canvas (Instructure), ще не набули достатнього розповсюдження серед українських студентів [6].

Окрему увагу на констатувальному етапі було приділено використанню штучного інтелекту (ШІ). Встановлено, що понад 45,9% студентів звертаються до готових рішень на основі ШІ принаймні раз на місяць, а 22,8% застосовують їх на професійному рівні. Абсолютним лідером серед моделей є ChatGPT (81,5%). Основними цілями використання ШІ є генерація ідей (63,3%), пошук інформації (59,8%), створення текстів (58%) та безпосереднє виконання лабораторних робіт (51,6%). Дослідження якості роботи з ШІ показало високий рівень критичного мислення студентів: лише 2,5% використовують згенерований код у первинному вигляді, тоді як 55,2% самостійно вносять правки та доопрацьовують запропоновані нейромережею рішення [7].

Готовність до інновацій має вікову кореляцію: найбільш вмотивованими до використання ШІ є студенти 1-го курсу, у той час як старшокурсники

ставляться до цих інструментів з певною засторогою через недосконалість існуючих моделей.

Отримані дані дають підстави стверджувати, що веборієнтовані технології та інструменти ІІІ мають стати невід'ємною частиною освітнього простору, тому потребують системної інтеграції в освітній процес через оновлення методик та перехід до комплексних моделей навчання, що поєднують традиційні (30%), дистанційні (50%) та проєктні (20%) підходи.

Важливим етапом *констатувального експерименту* стало проєктування та розроблення вебзастосунку інформаційної системи «E-student», покликаного оптимізувати процес навчання й контролю знань студентів. Архітектура системи передбачає дворівневий доступ (режими «Студент» та «Викладач-адміністратор»), що забезпечує автоматизоване оцінювання навчальних досягнень із програмування після виконання практичних робіт, модульних контролів та опрацювання теоретичного матеріалу.

Інформаційна системи «E-student» розроблена на основі таксономії Блума, що дозволяє реалізувати диференційоване оцінювання сформованості знань, умінь і навичок майбутніх бакалаврів комп'ютерних наук за шістьма рівнями: знання, розуміння, застосування, аналіз, оцінювання та створення. Такий підхід, реалізований за допомогою веборієнтованих технологій, забезпечує системну узгодженість між цілями навчання, змістом освітніх завдань та критеріями оцінювання результатів діяльності студентів.

Функціональне наповнення ІС «E-student» ґрунтується на структурованій базі навчальних завдань, що охоплює шість рівнів складності, ієрархія яких сформована на основі класичної таксономії Блума. Такий підхід дає змогу здійснювати поетапну діагностику навчальних досягнень студентів. Оцінювання результатів виконання завдань в інформаційній системі відбувається відповідно до авторської методики, детальний опис та обґрунтування якої наведено у розділах 2 та 3 цієї роботи. Зазначений інструментарій дозволяє забезпечити об'єктивність вимірювання навчальних досягнень та автоматизувати процес моніторингу успішності студентів.

Опис інформаційної системи «E-student» представлено в Додатку Н.

Доступ до інформаційної системи здійснюється за посиланням <https://e-kpi-hub.pages.dev/admin> з командного рядка любого браузера. Сторінка авторизації користувача та домашня сторінка інформаційної системи «E-student» представлена на рисунках. 4.1-4.2.

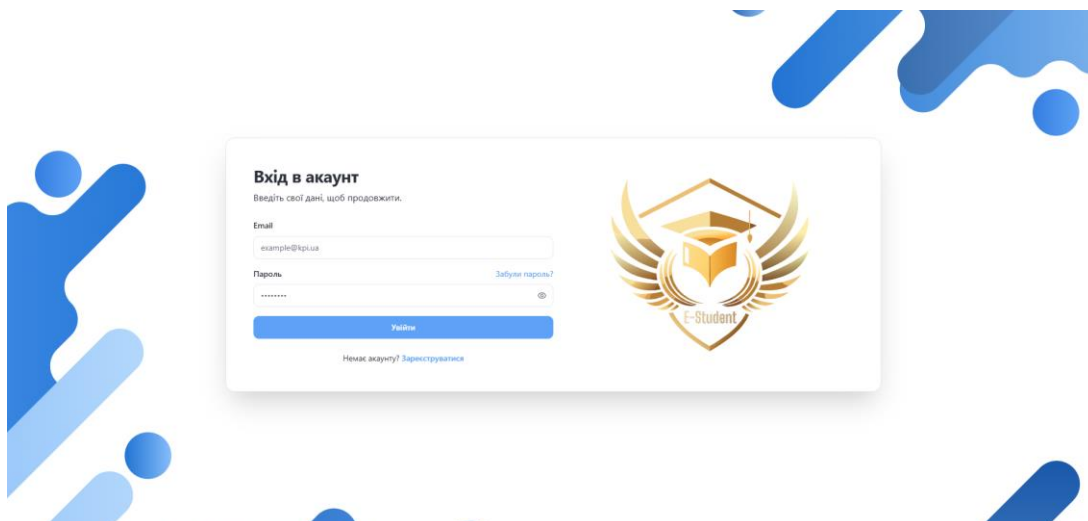


Рис.4.1 Вікно авторизації (реєстрації) програми «E-student»

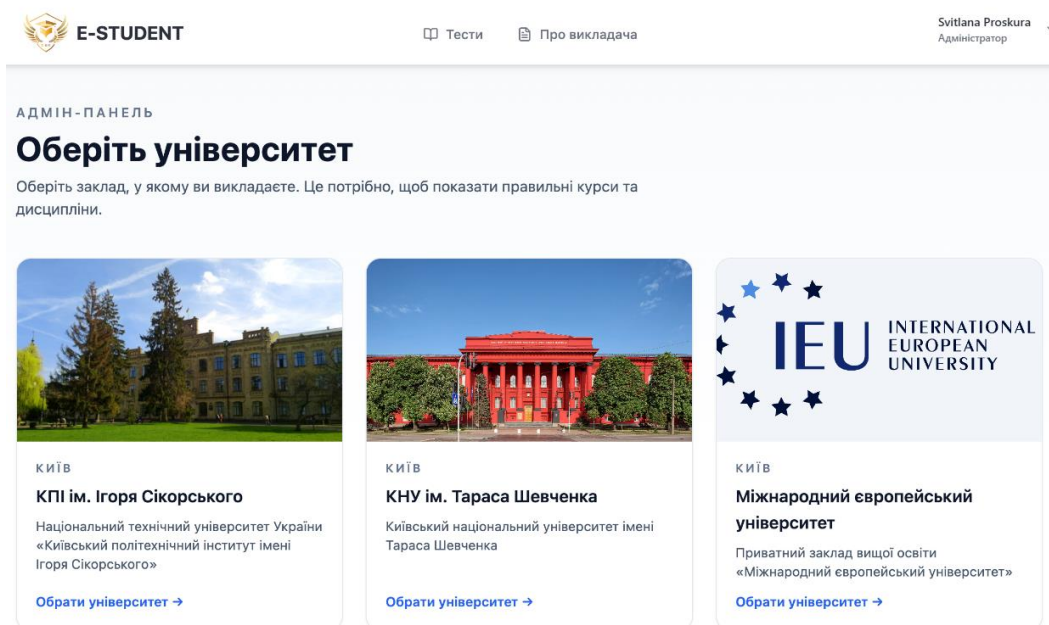


Рис.4.2. Домашня сторінка інформаційної системи «E-student»

Оцінювання навчальних досягнень студентів в інформаційній системі «E-student» здійснювалося згідно з розробленою авторською методикою обґрунтованою в розділах 2 та 3.

4.2. Аналіз результатів формувального етапу педагогічного експерименту

Отримані на констатувальному етапі результати, що засвідчили необхідність коригування навчального процесу, стали підґрунтям для переходу до формувального етапу експериментальної роботи.

Метою формувального етапу експерименту стала перевірка ефективності авторської методики використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук.

Для проведення експерименту було сформовано дві групи студентів:

- контрольна (КГ) (навчання студентів здійснювалося без використання авторської методики);
- експериментальна група (навчання студентів здійснювалося з використанням авторської методики).

У контрольній групі навчання велося за традиційною моделлю (локальні IDE, відсутність системного використання ІІІ, оцінювання кінцевих файлів), на відміну від ЕГ, де застосовувався Cloud-native підхід та авторська система навчання і оцінювання за таксономією Блума.

Зазначимо, що існують певні обмеження щодо реалізації авторської методики – необхідні умови, зокрема методика потребує стабільного інтернет-зв'язку та готовності викладачів до підтримки студентів (іноді в режимі 24/7).

Загальна кількість учасників експерименту становила – 112 осіб, із яких:

- експериментальна група (ЕГ) 56 студентів;
- контрольна група (КГ) – 56 студентів.

Контрольна та експериментальна групи формувалися за принципом максимальної однорідності щодо:

- вікових характеристик;
- умов освітнього процесу;
- змісту освітньої програми.

Для перевірки ефективності авторської методики було розроблено гіпотези.

Загальна гіпотеза дослідження: впровадження авторської методики навчання на основі веборієнтованих технологій сприятиме підвищенню рівня навчальних досягнень студентів із програмування.

Часткові гіпотези :

H_0 . Частка студентів, які підвищили свій рівень навчальних досягнень у процесі навчання програмування з використанням веборієнтованих технологій не є статистично значущою в експериментальній групі, ніж у контрольній групі.

H_1 . Частка студентів, які підвищили свій рівень навчальних досягнень у процесі навчання програмування з використанням веборієнтованих технологій є статистично значущою в експериментальній групі, ніж у контрольній.

На *формуальному етапі педагогічного експерименту* для перевірки статистичної значущості відмінностей між показниками контрольної та експериментальної груп було застосовано t-критерій Стюдента для незалежних вибірок (у модифікації Велча), що дозволяє коректно порівнювати середні значення за умови нерівності дисперсій.

Нерівність дисперсій означає наскільки сильно результати «розкидані» навколо середнього значення. Тобто нерівність дисперсій свідчить про те, що в одній групі результати дуже схожі між собою, а в іншій – дуже різнобічні.

З огляду на можливі відхилення від нормального розподілу та наявність граничних значень результатів, додатково використано непараметричний U-критерій Манна–Вітні. Застосування двох статистичних підходів дало змогу підвищити надійність та валідність отриманих висновків.

Узгоджені результати параметричного та непараметричного аналізу свідчать про наявність статистично значущих відмінностей між групами та підтверджують ефективність експериментального педагогічного впливу.

Розглянемо порівняльну характеристику результатів (табл. 4.1).

Таблиця 4.1

Основні статистичні показники

Основні описові статистичні показники результатів дослідження

Показник	Контрольна група (КГ, n = 56)	Експериментальна група (ЕГ, n = 56)
Середнє значення (M)	66.83	81.93
Медіана (Me)	66	90
Мода (Mo)	60	100
Мінімум (Min)	32	50
Максимум (Max)	100	100
Розмах (Max–Min)	68	50

Аналіз результатів навчання студентів засвідчив, що експериментальна група демонструє суттєво вищі показники порівняно з контрольною. Середнє значення результатів у експериментальній групі становить 81,93 б., тоді як у контрольній групі – 66,83 б., що відповідає різниці приблизно 15 балів (рис. 4.3-4.4). Це свідчить про позитивний вплив упровадження авторської методики.

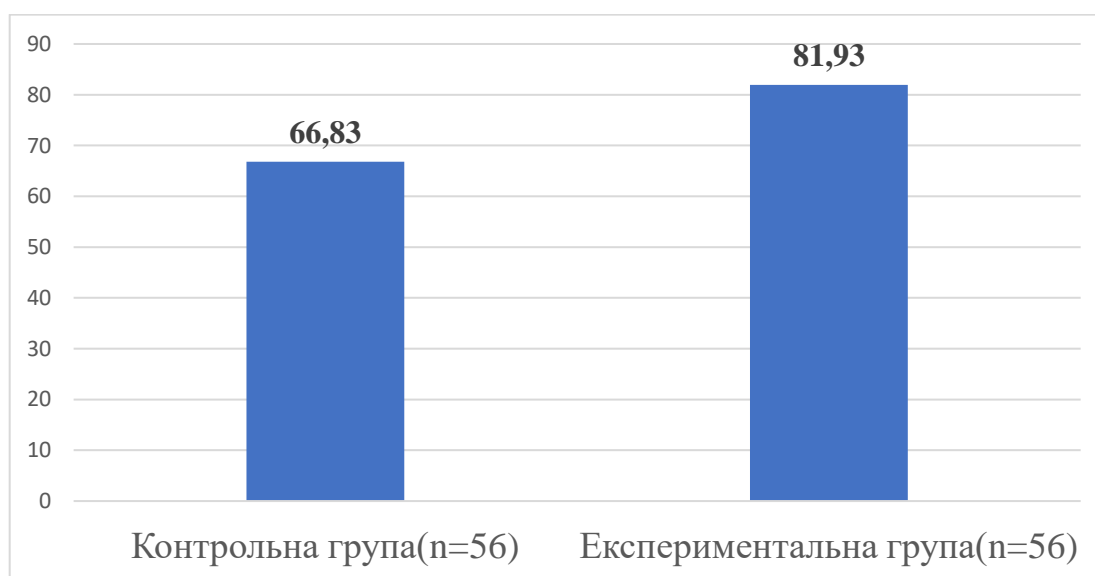


Рис. 4.3. Середнє значення результатів навчальних досягнень студентів в КГ і ЕГ на формувальному етапі

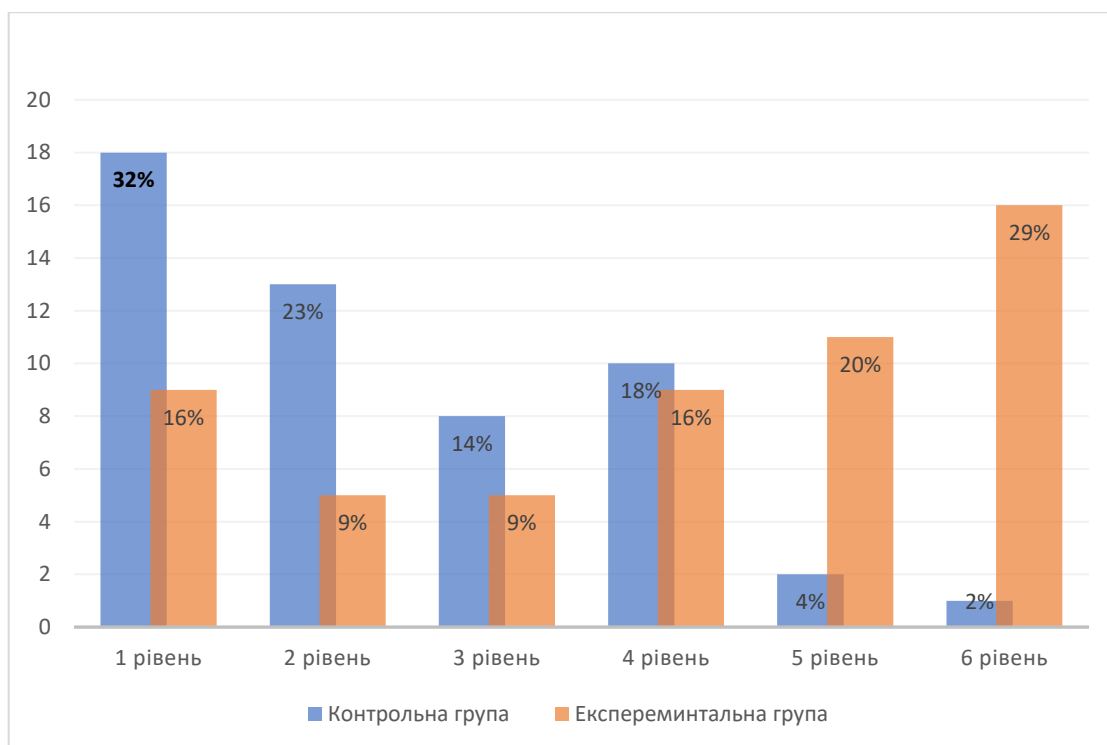


Рис. 4.4 Відносна кількість студентів в КГ і ЕГ, що досягли певного рівня за таксономією Блума на формувальному етапі.

Результати експерименту у контрольній групі, де навчання відбувалося за традиційною моделлю, зафіксовано значну кількість здобувачів на початкових рівнях таксономії: 1-й рівень («знання») охоплює 32% респондентів, а 2-й рівень («розуміння») – 23%. Сумарно понад 55% студентів контрольної групи зосереджені на репродуктивному рівні пізнавальної діяльності. Натомість в експериментальній групі спостерігається суттєвий зсув у бік вищих навчальних досягнень. Застосування диференційованих завдань сприяло активізації аналітичних та творчих процесів у студентів. Найбільш показовим є розподіл показників на 5-му та 6-му рівнях таксономії. Якщо в контрольній групі частка здобувачів, що досягли 6-го рівня («створення»), становить лише 2%, то в експериментальній групі цей показник зріс до 29%. Водночас на 5-му рівні («синтез/оцінювання») частка студентів експериментальної групи (20%) значно перевищує відповідний показник контрольної групи (4%). Отже, емпіричні дані засвідчують ефективність впровадження авторської методики навчання та використання системи завдань за Блумом, що дозволило трансформувати

навчальну діяльність студентів експериментальної групи з репродуктивної на продуктивно-творчу.

У контрольній групі більшість результатів зосереджена в інтервалі 60–70 б., що підтверджується як медіанним значенням (66 балів), так і модою, яка дорівнює 60 б. Такий розподіл характеризує переважно середній рівень навчальних досягнень. Водночас у контрольній групі спостерігається значний розкид результатів, зокрема наявні низькі показники (<60 б.), що свідчить про неоднорідність рівня підготовки студентів (рис. 4.5).

Разом із тим, наявність поодиноких випадків низьких результатів в експериментальній групі (наприклад, один здобувач, що отримав менш як 60 балів) свідчить про індивідуальні особливості темпу засвоєння матеріалу. Це підтверджує необхідність подальшої індивідуалізації навчання та вказує на те, що навіть за ефективної методики чинник індивідуальної освітньої траєкторії та особистої мотивації здобувача залишається визначальним.

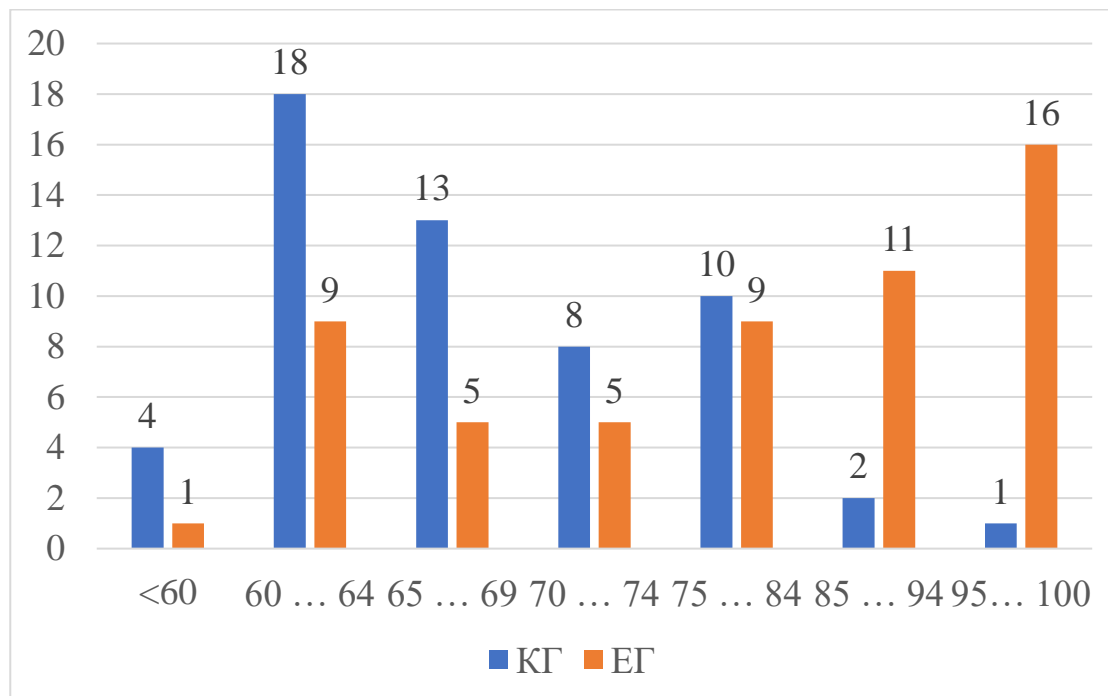


Рис.4.5 Кількість студентів в ЕГ та КГ за рівнями навчальних досягнень.

Для того, щоб перевірити, чи є різниця між групами статистично значущою, а не випадковою, ми використаємо t-критерій Стюдента для

незалежних вибірок. Наше завдання полягало у встановленні статистично значущих відмінностей між результатами навчання двох вибірок студентів: контрольної групи (КГ) та експериментальної групи (ЕГ), які не перетинаються за складом.

Відомо, що t-критерій Стюдента для незалежних вибірок (у модифікації Велча) як статистичний метод використовується для перевірки, чи є статистично значущою різниця між середніми значеннями двох незалежних груп [8].

Його застосовують тоді коли:

- порівнюються дві незалежні вибірки (наприклад, контрольна та експериментальна групи);
- дані мають приблизно нормальний розподіл;
- дисперсії груп можуть бути нерівними.

На відміну від класичного t-критерію Стюдента, модифікація Велча не вимагає рівності дисперсій, тому вважається більш надійною та точною для педагогічних психологічних і соціальних досліджень.

Суть методу полягає у визначенні, чи могла різниця між середніми значеннями навчальних досягнень студентів виникнути випадково, чи вона є результатом впливу певного фактора (наприклад, авторської методики навчання).

Формула критерію Велча:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

де:

\bar{x}_1, \bar{x}_2 – середні значення груп;

s_1^2, s_2^2 – дисперсії;

n_1, n_2 – обсяги вибірок.

Розглянемо алгоритм розрахунку t-критерію Стюдента для незалежних вибірок для нашого педагогічного експерименту.

1. Вихідні статистичні показники.

Контрольна група (КГ).

- Кількість: $n_1 = 56$.
- Середнє значення: $\bar{x}_1 = 66,83$.
- Стандартне відхилення: $s_1 = 10,63$.

Експериментальна група (ЕГ).

- Кількість: $n_2 = 56$.
- Середнє значення: $\bar{x}_2 = 81,93$.
- Стандартне відхилення: $s_2 = 15,34$.

2. Формулювання гіпотез.

- H_0 : статистично значущих відмінностей між контрольною (КГ) та експериментальною (ЕГ) групами немає.
- H_1 : між контрольною (КГ) та експериментальною (ЕГ) групами існують статистично значущі відмінності.

3. Формула t-критерію Стьюдента (Welch):

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

4. Підстановка числових значень.

- Квадрати стандартних відхилень:

$$s_1^2 = 10,63^2 = 112,99$$

$$s_2^2 = 15,34^2 = 235,29$$

Частини формули:

$$\frac{112,99}{56} = 2,02$$

$$\frac{235,29}{56} = 4,20$$

Знаменник:

$$\sqrt{2,02 + 4,20} = \sqrt{6,22} = 2,49$$

Чисельник:

$$66,83 - 81,93 = -15,10$$

Значення t :

$$t = \frac{-15,10}{2,49} = -5,72$$

Беремо модуль:

$$|t| = 5,72$$

5. Ступені вільності (Welch):

$$df \approx 95$$

6. Перевірка значущості.

Критичні значення:

$$t_{кр}(p = 0.05) \approx 1.99$$

$$t_{кр}(p = 0.01) \approx 2.63$$

Порівняння: $5,72 > 2,63$

7. Висновок. Нульову гіпотезу H_0 відхилено. Різниця між групами статистично значуща на рівні $p < 0.01$. Експериментальна група має суттєво вищі результати навчальних досягнень студентів.

Отже, за результатами t -критерію Стьюдента для незалежних вибірок ($t = 5.72$; $p < 0.01$) встановлено статистично значущу різницю між показниками контрольної групи ($M = 66.83$; $SD = 10.63$) та експериментальної групи ($M = 81.93$; $SD = 15.34$), що підтверджує ефективність експериментального педагогічного впливу.

Перевіримо дані експерименту за U -критерієм Манна-Вітні (непараметричний аналог t -критерію). U -критерій Манна-Вітні застосовується, якщо розподіл може бути ненормальним; шкала – порядкова або інтервальна; вибірки незалежні [9].

У нашому випадку КГ та ЕГ незалежні, відмічається асиметрія у кількості максимальних значень (95-100 б.). Тому використання Манна-Вітні методологічно виправдане.

1. Вихідні параметри:

КГ: $n_1 = 56$.

ЕГ: $n_2 = 56$.

Загальна кількість спостережень: $N = 112$.

2. Алгоритм обчислення - усі 112 значень об'єднано в один ряд і впорядковано за зростанням – від найменшого балу до найбільшого. Кожному значенню в загальному ряду було присвоєно відповідний ранг (порядковий номер). У випадках наявності однакових значень (повторів) їм присвоювався середній арифметичний ранг. Після завершення ранжування було окремо розраховано суми рангів для КГ та ЕГ.

3. Визначення емпіричного значення U_{emp} . Статистичний розрахунок проводився за формулою:

$$U = n_1 n_2 + \frac{n_1(n_1 + 1)}{2} - R_1$$

де за кінцевий результат приймалося менше з двох отриманих значень U .

4. Результат обчислення.

Для даних отримано: $U = 684$

Очікуване середнє:

$$U_{mean} = \frac{n_1 n_2}{2} = 1568$$

Стандартне відхилення:

$$\sigma_U = \sqrt{\frac{n_1 n_2 (n_1 + n_2 + 1)}{12}} \approx 171.6$$

5. Z-перетворення

$$Z = \frac{U - U_{mean}}{\sigma_U} = \frac{684 - 1568}{171.6} \approx -5.15$$

6. Рівень значущості: $|Z| = 5.15 > 2.58$. $p < 0.001$

Висновок. За результатами U-критерію Манна–Вітні ($U = 684$; $Z = -5.15$; $p < 0.001$) встановлено статистично значущу різницю між показниками контрольної та експериментальної груп, що підтверджує стійкість отриманих результатів незалежно від припущень щодо нормальності розподілу.

Для підтвердження достовірності отриманих результатів було використано кутове перетворення Фішера. Для визначення кутового перетворення Фішера обрано оптимальний критерій – частка студентів із високим рівнем навчальних досягнень (≥ 80 балів).

1. Формування 2×2 таблиці (основа ϕ^* -критерію) (табл. 4.2).

Таблиця 4.2

Основа ϕ^* -критерію

Група	≥ 80 балів	< 80 балів	Разом
Контрольна (КГ)	9	47	56
Експериментальна (ЕГ)	33	23	56

На основі даних у контрольній групі частка студентів з результатом ≥ 80 є відносно невеликою; в експериментальній групі спостерігається істотно більша кількість таких студентів (зсув у верхню частину шкали, підтверджений медіаною = 90 і модою = 100).

2. Формула кутового перетворення Фішера.

Обчислюється кутова величина:

$$\phi = 2 \cdot \arcsin \sqrt{p}$$

Де p – частка студентів з ознакою (≥ 80 балів).

Критерій:

$$\phi^* = |\phi_1 - \phi_2|$$

3. Критичні значення

$$\phi_{кр}^* = 1.64 \rightarrow p < 0.05$$

$$\phi_{кр}^* = 2.31 \rightarrow p < 0.01$$

4. У контрольній групі кількість студентів, які мають бали ≥ 80 балів складає 9 осіб. В експериментальній групі – 33 особи. Знайшли різницю між кутами $1,8446 - 0,8566 = 0,988$. За розрахунками $\phi^* = 4.97$.

4. Результат перевірки. Розрахунок ϕ^* показує, що:

$$\phi^* > 2.31$$

Отже, різниця між частками студентів з високим рівнем навчальних досягнень у контрольній та експериментальній групах є статистично значущою на рівні $p < 0.01$. (рис. 4.6).

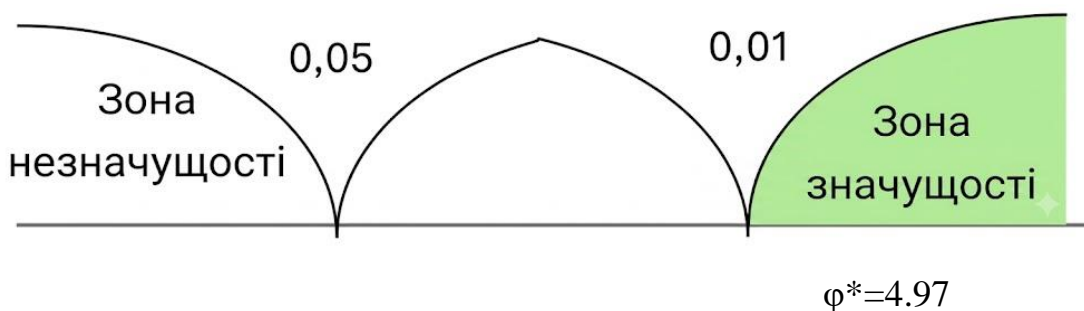


Рис. 4.6. Вісь значущості навчальних досягнень студентів

Отже, кутове перетворення Фішера підтверджує статистично значущу різницю між групами ($\varphi^* > 2.31$; $p < 0.01$), що зростання кількості студентів з високими результатами не є випадковим, авторська методика системно впливає на структуру результатів, а не лише на середні значення. Ефект проявляється на рівні навчальних досягнень всієї групи, що є ключовим для педагогічних досліджень.

Таким чином, експериментальний вплив авторської методики сприяв істотному зростанню частки студентів з високими навчальними результатами, що узгоджується з результатами параметричного та непараметричного статистичного аналізу.

Узгодженість результатів усіх трьох методів аналізу дозволяє стверджувати, що отримана різниця в показниках є статистично надійною та достовірною. Експериментальний вплив не лише підвищив середній рівень знань, а й зробив групу більш однорідною за високими показниками успішності.

Висновки до розділу 4

На основі результатів педагогічного експерименту, проведеного на базі провідних ЗВО України (НТУУ «КПІ імені Ігоря Сікорського», МЕУ, КУ імені Бориса Грінченка, Запорізького національного університету, НУ «Запорізька політехніка» та ін.), можна зазначити про таке:

- Застосування трьох взаємодоповнюючих методів аналізу (параметричного t-критерію Велча, непараметричного U-критерію Манна–Вітні та частотного перетворення Фішера) дозволило встановити, що позитивні зміни в Е-групі є статистично значущими та достовірними. Це дає наукове підґрунтя стверджувати, що ефективність методики не залежить від особливостей розподілу даних і повністю виключає фактор випадковості у досягненні високих результатів.

- Упровадження методики забезпечило системне підвищення середнього рівня навчальних досягнень студентів на 15 балів (з 66,83 у контрольній до 81,93 в експериментальній групі). Це свідчить про те, що використання веборієнтованих технологій та ШІ дає змогу студентам швидше опановувати складний матеріал, трансформуючи теоретичні знання у практичну здатність створювати програмні продукти.

- Зміщення показника моди до 100 балів та медіани до 90 балів в Е-групі доводить, що авторська методика ефективно стимулює перехід студентів від репродуктивного відтворення до рівнів «Створення» та «Оцінювання» за таксономією Блума. Це дає студентам можливість досягати високих рівнів цифрової компетентності за рамкою DigComp 3.0, що є критичною вимогою сучасного ІТ-ринку.

- Експериментально підтверджено зростання мінімального балу в групі з 32 до 50, що вказує на здатність методики мінімізувати кількість низьких результатів та робити групу більш однорідною. Це дає викладачу інструмент для ефективного навчання студентів із різним початковим рівнем підготовки, забезпечуючи стабільний якісний результат для всієї вибірки.

– Успішне підтвердження гіпотези дослідження за допомогою кутового перетворення Фішера доводить, що методика системно змінює структуру результатів, суттєво збільшуючи частку студентів із високим рівнем підготовки. Це дає змогу ЗВО готувати конкурентоспроможних фахівців, чия компетентність підтверджена не лише балами, а й реальним «цифровим слідом» у хмарному середовищі розробки.

Список використаних джерел до Розділу 4

1. Гончаренко С. У. Український педагогічний словник. С. Гончаренко. Київ: Либідь, 1997. 366 с. URL: <https://surl.lu/sgqdmn> (дата звернення: 21.04.2026).
2. Гончаренко С. У.: Методологічні характеристики Педагогічні дослідження молодим науковцям. Київ–Вінниця : ДОВ «Вінниця», 2008. 278 с.: Праці автора у цифровій бібліотеці НАПН України URL: https://lib.iitta.gov.ua/id/eprint/711013/?utm_source (дата звернення: 21.05.2026).
3. Підласий І. П. Природний і лабораторний педагогічний експеримент. *Педагогіка* Матеріал про природний і лабораторний експеримен. URL: https://iua.archive.cx.ua/articles/prirodnij-i-laboratornij-pedagogichnij-eksperiment.php?utm_source (дата звернення: 21.05.2026).
4. Хриков Є. М., Адаменко О. В., Савченко С. В. Методологічні засади педагогічного дослідження : монографія. Луганськ : Луганський національний університет імені Тараса Шевченка, 2013. 244 с. URL: https://dspace.luguniv.edu.ua/xmlui/handle/123456789/2763?utm_source (дата звернення: 21.05.2026).
5. Proskura S. L., Lytvynova S. H., Kronda O. P. The use of WEB-oriented technologies in the process of WEB-programming teaching for technical universities students. *ICTERI 2021: Proceedings of the 17th International Conference on ICT in Education, Research and Industrial Applications*, 2021. P. 129–141. DOI: <https://doi.org/10.31812/educdim.4724> (дата звернення: 27.03.26).

6. Проскура С. Л., Литвинова С. Г., Кронда О. П. Засоби організації дистанційного навчання в період карантину 2020 року в закладах вищої освіти України Екстрене дистанційне навчання в Україні : кол. монографія. В. М. Кухаренко, В. О. Бондаренко ; за ред.: В. М. Кухаренка, В. В. Бондаренка ; Нац. техн. ун-т "Харків. політехн. ін-т". – Харків : КП "Міська друкарня", 2020. – 409 с. URL: <https://repository.kpi.kharkov.ua/items/64746eb5-a843-4089-9931-aa413f13435e> (дата звернення: 27.03.26).
7. _Lytvynova S. H., Rashevskaya N. V., Proskura S. L. The use of artificial intelligence in teaching students programming languages *Proceedings of the IX International Workshop on Professional Retraining and Life-Long Learning using ICT: Person-oriented Approach (3L-Person 2024), co-located with ICTERI 2024. CEUR Workshop Proceedings*, 2024. Vol. 3781. P. 10–29. URL: <https://ceur-ws.org/Vol-3781/paper01.pdf> (дата звернення: 27.03.26).
8. Delacre M., Lakens D., Leys C. Why Psychologists Should by Default Use Welch's t-test Instead of Student's t-test *International Review of Social Psychology*, 2017. Vol. 30(1). P. 92–101. DOI: <https://doi.org/10.5334/irsp.82> URL: https://rips-irsp.com/articles/10.5334/irsp.82?report=reader&utm_source
9. Nachar N. The Mann-Whitney U: A Test for Assessing Whether Two Independent Samples Come from the Same Distribution, 2008 *Tutorials in Quantitative Methods for Psychology*, 4(1), 13–20. DOI: _
10. Berr K. J., Mielke P. W. A Monte Carlo Investigation of the Fisher Z Transformation for Normal and Nonnormal Distributions. *Psychological Reports*, 2000 URL: https://journals.sagepub.com/doi/10.2466/pr0.2000.87.3f.1101?utm_source

ВИСНОВКИ

На основі проведеного комплексного дослідження теоретичних засад, моделювання та експериментальної перевірки методики використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук, сформульовано такі загальні висновки:

1. Установлено, що в умовах системної цифрової трансформації компетентність з програмування трансформується з набору технічних навичок у цифрову компетентність, яка базується на здатності до алгоритмічного мислення в межах інтелектуалізованих цифрових систем. Проведений порівняльний аналіз атлантичної, континентальної та східноазійської моделей підготовки фахівців, у поєднанні з імплементацією вимог рамки DigComp 3.0, засвідчив зміну освітньої парадигми. Сучасний ІТ-спеціаліст функціонує в умовах хмаро орієнтованого середовища, де веббраузер трансформувався з пасивного інструменту на універсальне інтегроване робоче середовище. Це вимагає від закладів вищої освіти переходу від вивчення локально встановленого програмного забезпечення до формування компетентностей роботи з розподіленими хмарними сервісами та інтелектуальними платформами.

2. Розроблено та впроваджено авторські моделі веборієнтованого навчання і використання веборієнтованих технологій у підготовці бакалаврів комп'ютерних наук. Модель веборієнтованого навчання базується на оптимальному розподілі видів діяльності: 30% – традиційне навчання, 50% – дистанційне (Cloud-native) та 20% – проєктне. Це дозволило реалізувати принцип неперервності освіти, усунувши технічні перешкоди, пов'язані з локальним налаштуванням програмного забезпечення, та забезпечивши ідентичність навчальних умов для всіх студентів, незалежно від їхнього місця перебування чи технічних характеристик пристроїв. Модель використання веборієнтованих технологій розкриває різні аспекти застосування вебсервісів, хмароорієнтованих середовищ та ІІІ для забезпечення доступності й гнучкості

навчання, створюючи єдиний інформаційно-освітній простір, де студенти можуть: взаємодіяти з навчальним контентом незалежно від часу та місця; масштабувати ресурси для спільної роботи над проєктами в реальному часі; забезпечувати надійне зберігання освітніх даних, що є критично важливим для формування цифрового портфоліо студента; отримувати доступ до адаптивних інструментів аналізу навчальних досягнень, автоматизації зворотного зв'язку та підтримки індивідуальних освітніх траєкторій, що дозволяє виявляти прогалини в знаннях та коригувати процес навчання. Науково обґрунтовано перехід від епізодичного використання ІІІ до режиму ІІІ як інтелектуальний помічник. Впровадження ІІІ-асистентів (GitHub Copilot, ChatGPT) для дебагінгу, пояснення логіки коду та автоматизованого Code Review дозволило оптимізувати когнітивне навантаження студентів, спрямовуючи їхню увагу на архітектурні рішення та логіку алгоритмів, а не на механічне написання синтаксису.

3. Запропоновано авторську методику використання веборієнтованих технологій у навчанні програмування бакалаврів комп'ютерних наук, що інтегрує форми, методи та засоби навчання шляхом синхронізації рівнів таксономії Блума з експертними рівнями (1–8) міжнародної рамки DigComp 3.0. Ключовою інновацією методики є використання історії фіксацій змін (commits) у системі GitHub як основного об'єкта оцінювання. Такий підхід забезпечує об'єктивність контролю, підтверджує академічну доброчесність здобувачів та дозволяє кількісно й якісно оцінити ітеративність розробки програмного продукту.

4. Розроблено авторську методику формувального оцінювання навчальних досягнень студентів, що базується на інтеграції Таксономії Блума з інноваційною накопичувальною системою (механізм додавання/віднімання балів). Методику реалізовано у вебзастосунку інформаційній системі «E-student», що дозволило забезпечити прозорість контролю та автоматизувати процес моніторингу прогресу здобувачів та формувати індивідуальну траєкторію студента щодо розвитку компетентності з програмування.

5. Експериментально підтверджено статистичну значущість результатів. Педагогічний експеримент, проведений у провідних ЗВО України (НТУУ «КПІ ім. Ігоря Сікорського», МЕУ, КУ ім. Б. Грінченка), довів, що впровадження методики забезпечує зростання середнього рівня успішності на 15 балів (з 66,83 до 81,93). Узгодженість результатів трьох методів перевірки (t-критерій Велча, U-критерій Манна-Вітні, ϕ^* -критерій Фішера) підтвердила, що зміни є достовірними, надійними та системними, а експериментальна група стала більш однорідною за високими показниками успішності (мода 100, медіана 90). Таким чином, розроблена методика є цілісним дидактичним інструментарієм, який дозволяє закладу вищої освіти готувати конкурентоспроможних бакалаврів комп'ютерних наук, готових до професійної діяльності в умовах інтелектуального вебу та активного розвитку штучного інтелекту.

У розділі були використані праці автора: [4], [6], [7].

ДОДАТОК А

Таблиця А.1

Типи веборієнтованих технологій

Технології	Типи	Приклади	Опис	Область застосування	Спосіб поширення	Рівень складності
Фронтенд технології	Мови програмування	HTML, CSS, JavaScript	Використовуються для створення інтерфейсу користувача.	Веброзробка	Відкритий код	Початковий/Середній
	Фреймворки та бібліотеки	React, Angular, Vue.js	Інструменти для спрощення розробки інтерфейсів.	Веброзробка	Відкритий код	Середній/Високий
	Препроцесори стилів	Sass, LESS	Інструменти для поліпшення та організації CSS.	Веб-розробка	Відкритий код	Середній
	Інструменти для побудови	Webpack, Gulp, Parcel	Використовуються для збирання та оптимізації коду.	Веб-розробка	Відкритий код	Середній
	Мобільні фреймворки	React Native, Ionic	Інструменти для створення мобільних додатків.	Мобільна розробка	Відкритий код	Середній/Високий
Бекенд технології	Мови програмування	Python, Ruby, PHP, Java, C#, Node.js	Використовуються для створення серверної логіки.	Серверна розробка	Відкритий/Закритий код	Початковий/Високий
	Фреймворки	Django, Ruby on Rails, Laravel, Spring Boot	Спрощують розробку серверної частини додатків.	Веб-розробка	Відкритий код	Середній/Високий

Технології	Типи	Приклади	Опис	Область застосування	Спосіб поширення	Рівень складності
	Веб-сервери	Apache, Nginx, Microsoft IIS	Використовуються для обслуговування веб-запитів.	Веб-розробка	Відкритий /Закритий код	Середній
Бази даних	Реляційні	MySQL, PostgreSQL, Oracle, SQL Server	Використовуються для зберігання структурованих даних.	Зберігання даних	Відкритий/Закритий код	Середній/Високий
	Нереляційні (NoSQL)	MongoDB, Cassandra, Redis, CouchDB	Використовуються для зберігання неструктурованих даних або даних, що швидко змінюються.	Зберігання даних	Відкритий/Закритий код	Середній/Високий
API технології	REST	JSON, XML	Архітектурний стиль для створення веб-сервісів.	Веб-розробка, інтеграції	Відкритий код	Середній
	GraphQL	-	Мова запитів для API, яка дозволяє клієнтам запитувати саме ті дані, які потрібні.	Веб-розробка, інтеграції	Відкритий код	Середній
Семантичний веб	RDF, OWL, SPARQL	DBpedia, Wikidata, Google Knowledge Graph	Семантична мережа — це підхід до організації даних, де інформація структурується за допомогою метаданих, що описують зв'язки між сутностями.	Пошукові системи, освітні бази знань	Через відкриті онтології, API та SPARQL-ендпоїнти	Високий (потребує знань онтологій і запитів)

Технології	Типи	Приклад и	Опис	Область застосування	Спосіб поширення	Рівень складності
Інтелектуальні системи - AI / ML (Штучний інтелект /	AI, ML, Deep Learning, NLP	ChatGPT, Google Bard, Siri, Grammarly, Replika	Системи, що здатні навчатися на основі даних, розпізнавати шаблони, адаптуватися та генерувати відповіді або дії на основі попереднього досвіду.	Освіта, медіа, підтримка користувачів	Хмарні сервіси, API, інтеграція в платформи	Високий (необхідні знання алгоритмів)
Аналітика даних Big Data	Hadoop, Spark, NoSQL, Data Lakes	Amazon Redshift, Google BigQuery, Cloudera	Обробка великих обсягів структурованих і неструктурованих даних для виявлення трендів, прогнозів та прийняття рішень.	Освітня аналітика, наука, дослідження	Платформи хмарних обчислень, SaaS-рішення	Високий (аналітика, програмування)
Розподілені технології (Blockchain)	Public / Private Blockchain, Smart Contracts	Ethereum, Hyperledger, Solana, OpenCerts	Технологія розподіленого реєстру для захищеного зберігання і підтвердження транзакцій, записів, сертифікатів без потреби в централізованих серверах	Електронні сертифікати, фінанси, освіта	Децентралізовані мережі, API, смарт-контракти	Високий (криптографія, інфраструктура)

Технології	Типи	Приклади	Опис	Область застосування	Спосіб поширення	Рівень складності
Формати даних	Schema.org , JSON-LD (JavaScript Object Notation for Linked Data)	Контент сайтів Google, Coursera, Wikipedia	Формат даних, що дозволяє структурувати інформацію у вигляді зв'язків для полегшення індексації та семантичної обробки веб- сторінок.	SEO, онлайн-курси, семантичний пошук	Вбудовується в HTML як скрипт JSON-LD	Середній (знання структури JSON та мета- даних)
Семантичні моделі Knowledge Graphs (Графи знань)	Domain- specific, Global graphs	Google Knowledge Graph, Microsoft Satori, Wikidata	Мережа пов'язаних об'єктів та понять, яка дозволяє комп'ютерам розуміти зміст і контекст інформації як для пошуку, так і для AI- аналітики	Пошукові системи, e- learning, дослідницькі бази	API, SPARQL, інтеграція через RDF/OWL	Високий (логіка, онтології, семантика)
Цифрові пристрої IoT (Інтернет речей)	Smart Devices, Sensors, Wearables	Розумні дошки, фітнес- трекери, датчики температури, освітлення, IoT Arduino, Raspberry Pi	Мережа фізичних пристроїв, які обмінюються даними через інтернет. Дає змогу автоматизувати освітній простір, проводити збір даних у реальному часі	Освіта, медицина, логістика, смарт- класи	Підключення через Wi-Fi/Bluetooth, мобільні застосунки, хмарні платформи	Середній/Високий (залежать від інфраструктури)

Технології	Типи	Приклади	Опис	Область застосування	Спосіб поширення	Рівень складності
Інтелектуальні агенти Віртуальні асистенти	Голосові, текстові, контекстні	Siri, Alexa, Google Assistant, ChatGPT	Програмні агенти, які розуміють мову користувача, відповідають на запити, планують завдання, допомагають в навчанні та повсякденному житті	Освіта, побут, техпідтримка, бізнес	Інтеграція у смартфон, браузери, сайти	Середній / Високий (AI, NLP, інтеграція)
Децентралізовані системи DApps (децентралізовані)	Фінансові, навчальні, соціальні	OpenSea, Steemit, Brave, Audius	Програми, які працюють на блокчейні без централізованого контролю. Дають змогу зберігати права власності, сертифікати, контент без цензури	Освіта, творчість, фінанси, ігри	Через блокчейн-платформи (Ethereum, Polygon), токенизація	Високий (блокчейн, смарт-контракти)

Таблиця А.2

Класифікація веборієнтованих технологій

1 За поколіннями розвитку	
Покоління	Основні характеристики/Технології
Web 1.0	Статичний контент, лише читання / HTML, CSS, CGI
Web 2.0	Інтерактивність, спільна робота/ JavaScript, AJAX, CMS
Web 3.0	Семантика, персоналізація,/ AI RDF, OWL, SPARQL, ML
Web 4.0	Інтелектуальна взаємодія/ IoT, емоційний AIAI, IoT, WebXR, blockchain

2 За призначенням / функціональністю	
<i>Категорія</i>	<i>Приклади технологій</i>
Інтерфейс користувача (UI)	HTML5, CSS3, JavaScript, React, Vue.js
Логіка застосунку (бекенд)	Node.js, Django, Ruby on Rails, Laravel
Передача даних / API	REST, GraphQL, WebSockets
Бази даних	MySQL, MongoDB, Firebase, PostgreSQL
Хмарні платформи	AWS, Google Cloud, Azure
Штучний інтелект / ML	OpenAI API, TensorFlow.js, Dialogflow
Інтернет речей (IoT)	MQTT, ThingSpeak, Blynk, Arduino Cloud
Блокчейн / Web3	Ethereum, Solidity, Web3.js, IPFS
AR/VR/WebXR	A-Frame, Three.js, Babylon.js
Контроль доступу / безпека	OAuth, HTTPS, JWT, Captcha
За сферою застосування	
3 Сфера застосування	
<i>Сфера застосування</i>	<i>Типові технології</i>
Освіта	Moodle, Google Classroom, CoSpaces EDU
Медицина	IoT-моніторинг, AI-діагностика, VR-симуляції
Бізнес / eCommerce	Shopify, Stripe, CRM-системи
Аналітика / Big Data	Power BI, Tableau, Hadoop
Розваги / медіа	Spotify API, Netflix AI, WebGL
Метавсесвіт / ігри	Unity WebGL, WebXR, Unreal Engine
4 За рівнем інтерактивності	
Тип	<i>Характеристика</i>
Статичні	Показують фіксований контент, без взаємодії
Динамічні	Користувач взаємодіє, змінює вміст (форми, коментарі)
Інтелектуальні	Реагують на поведінку користувача, персоналізують контент (на основі AI)
Імерсивні	Підтримка VR/AR, глибока залученість користувача

Таблиця А.3

Веборієнтованих технологій для навчання програмування

Група веборієнтованих технологій	Інструменти
Навчальні платформи (LMS)	Moodle, Google Classroom, Canvas, Microsoft Teams
Платформи для навчання програмування	HackerRank, LeetCode, Codeforces, E-Olymp
Технології веброзробки	HTML5, CSS3, JavaScript, React, Angular, Django
Системи контролю версій	GitHub, GitLab, Bitbucket
Хмарні сервіси та IDE	AWS, Google Cloud, Replit, VS Code
Бази даних та API	Firebase, MongoDB, REST API, GraphQL
DevOps та CI/CD	Docker, GitLab CI/CD, Netlify, Vercel
Засоби комунікації	Microsoft Teams, Zoom, Discord, Telegram, YouTube
Інструменти автоматизованого тестування та перевірки коду	CodeRunner

ДОДАТОК Б

Дослідження понять «компетентність», «професійна компетентність», «професійна компетентність бакалавра комп'ютерних наук», «компетентність з програмування»

Інтеграція України у світовий освітній простір вимагає постійного вдосконалення національної системи освіти, пошуку ефективних шляхів підвищення якості освітніх послуг, модернізації змісту освіти й організації освітнього процесу відповідно до світових тенденцій і вимог ринку праці [118], [24].

Тому сучасні тенденції розвитку професійної підготовки майбутніх бакалаврів з ІТ-спеціальностей, зокрема бакалаврів комп'ютерних наук, містять: спеціалізацію освіти, фундаменталізацію її змісту, багаторівневність, формування професійних компетентностей, гнучкість навчальних планів, посилення практичної спрямованості, впровадження інноваційних технологій навчання, залучення інтерактивних методів навчання, індивідуалізацію навчання, підвищення ролі науково-педагогічних працівників з високим науково-педагогічним потенціалом, здатних вибудовувати міждисциплінарні навчальні курси і здійснювати продуктивну професійну комунікацію у сфері ІТ-освіти [24], [119].

У рамках цього дисертаційного дослідження розглянемо та уточнимо такі основні поняття, як «компетентність», «професійна компетентність», «професійна компетентність бакалавра комп'ютерних наук», «компетентність з програмування» [24].

Термін «компетентність» західного походження. У «Новому тлумачному словнику української мови» означено так: «1) який має достатні знання в якій-небудь галузі; який з чим-небудь добре обізнаний; тямущий; який ґрунтується на знанні; кваліфікований; 2) який має певні повноваження; повноправний, повновладний» [24]. [120].

Європейські країни сьогодні розпочали ґрунтовну дискусію навколо того, як озброїти людину, зокрема студента-програміста, необхідними вміннями та

знаннями для забезпечення її гармонійної взаємодії з технологічним суспільством, що швидко розвивається. Саме тому важливим є усвідомлення в суспільстві поняття компетентності, що базується на знаннях. Важливо розуміти, яких саме компетентностей необхідно навчати і як, а також, що саме має бути результатом навчання [24], [121].

У міжнародному документі Європейська рамка електронної компетентності (European e-Competence Framework (e-CF)) наголошується, що «компетентність – це продемонстрована здатність застосовувати знання, навички та ставлення для досягнення помітних результатів». Це цілісне поняття, безпосередньо пов'язане з діяльністю на робочому місці та охоплює складну людську поведінку, виражену як вбудовані ставлення [122]. Зміст поняття «компетентність» в е-документі European e-Competence Framework відобразимо на рис. 1.4. [24].

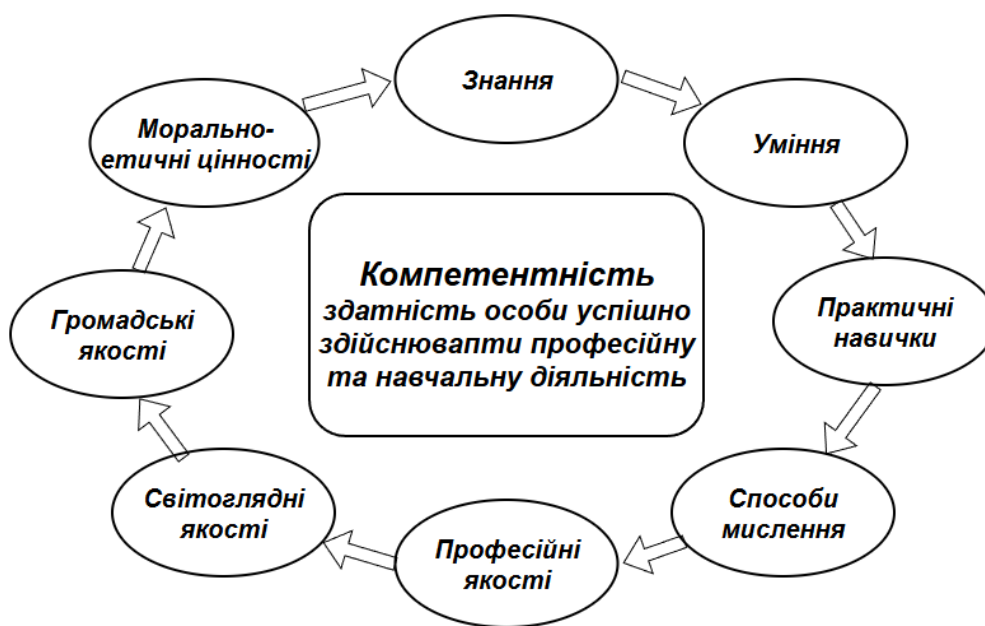


Рис. Б.1. Зміст поняття «компетентність» в е-документі European e-Competence

Поняття «компетентність» досліджували такі вчені, як: В. Биков [8], Т. Ковалюк [123], Р. Захаров [124], С. Вітелло (S. Vitello) [125], Д. Греторекс (J. Greateorex) [125], Лаура Герш Салганік (Laura Hersh Salganik) [126], Домінік С. Рихен (Dominique Simone Rychen) [126], У. Мозер (U. Moser) [126], Джон В. Констант (John W. Konstant) [126], О. Овчарук [127],

С. Кравченко [128], О. Пошетун [129], С. Литвинова [24] та багато інших науковців.

У своїх дослідженнях В. Биков визначає, що компетентність — це інтегрована характеристика особистості, що відображає її здатність ефективно діяти в інформаційно-освітньому середовищі на основі знань, умінь, досвіду та цінностей [8]. Науковець Т. Ковалюк класифікує компетентності як компетентності з: математичної підготовки, програмування, технологій та процесів розроблення програмного забезпечення, опрацювання даних, технічної підготовки, системотехнічної підготовки, загальної виробничої культури [24], [123].

У своєму дисертаційному дослідженні Р. Захаров [124] досліджує необхідність формування компетентностей здобувачів освіти спеціальності F3 «Комп'ютерні науки» в контексті сучасних вимог ринку праці та державних стандартів, з використанням сучасних методів та інформаційних технологій, які задіяні для удосконалення рівня підготовки здобувачів вищої освіти та використовують модель синергії, яка втілює активну співпрацю між здобувачами вищої освіти, державою, роботодавцями та базується на принципах гармонізації та партнерства засобами інформаційних технологій, штучної нейронної мережі таких технологій як інформаційної бази дослідження [124]. Зазначає, що компетентність — це результат підготовки, що формується через взаємодію освіти, технологій і вимог ринку праці.

Дослідники центрального дослідницького відділу С. Вітелло (S. Vitello) та Д. Греторекс (J. Grotorex) видавництва Cambridge University Press & Assessment (Кембридж, Велика Британія) розглядають компетентність як здатність інтегрувати та застосовувати контекстуально відповідні знання, навички та психосоціальні фактори (наприклад, переконання, ставлення, цінності та мотивацію) для послідовного успішного виконання завдань у визначеній сфері [125].

Група експертів з різних галузей, таких як Лаура Герш Салганік (Laura Hersh Salganik) [126], Домінік С. Рихен (Dominique Simone Rychen), У. Мозер

(U. Moser) [126], Джон В. Констант (John W. Konstant) [126] визначають поняття компетентності як здатність успішно задовольняти індивідуальні та соціальні потреби, діяти й виконувати поставлені завдання і зазначають, що кожна компетентність побудована на поєднанні взаємовідповідних пізнавальних ставлень і практичних навичок, цінностей, емоцій, поведінкових компонентів, знань і вмінь, усього того, що можна мобілізувати для активної дії [126].

Науковець О. Овчарук зазначає, що компетентність – це інтегрована характеристика якості особистості, результативний блок, сформований через досвід, знання, вміння, ставлення, поведінкові реакції. Компетентність ґрунтується на комбінації взаємовідповідних пізнавальних відношень та практичних навичок, цінностей, емоцій, поведінкових компонентів, знань та вмінь, усього того, що можна мобілізувати для активної дії [127].

С. Кравченко констатує, що компетентність — це здатність застосовувати набуті знання, вміння, навички у практичній діяльності; особистісна характеристика фахівця, який володіє низкою відповідних компетенцій. Своєю чергою, під компетенцію розуміє сукупність систематизованих знань, умінь і навичок, що є необхідними для якісної діяльності особистості [128].

О. Пометун розглядає компетентність як складну інтегровану характеристику особистості, під якою розуміється сукупність знань, умінь, навичок, а також досвіду, що разом дає змогу ефективно провадити діяльність або виконувати певні функції, забезпечуючи розв'язання проблем і досягнення певних стандартів у галузі професії або виду діяльності [129].

У своїх наукових дослідженнях С. Литвинова доводить, що головною метою нової української освіти є формування компетентного випускника, який буде здатним застосувати отримані знання у житті, насиченому цифровими засобами комунікації, управління, отримання освіти, ведення бізнесу [26].

Важливою вимогою до професійного профілю бакалавра є відповідність восьмирівневій системі володіння компетентностями за рамкою DigComp 3.0. Відповідно, випускник ЗВО повинен досягти 7–8 (експертного) рівня у сфері

створення цифрового контенту, що передбачає не лише написання коду, а й проектування архітектури хмарних застосунків, налаштування CI/CD процесів (GitHub Actions) та проведення групової перевірки коду (Code Review) у розподілених командах.

Перед розглядом проблем формування компетентностей майбутніх бакалаврів комп'ютерних наук, схарактеризуємо визначення поняття «професійна компетентність бакалавра комп'ютерних наук» [24].

У міжнародному проєкті «Узгодження освітніх структур у Європі» («Tuning Educational Structures in Europe»), професійні компетентності випускників ЗВО розглядаються як два різних види компетентностей: загальні компетентності і фахові (професійні) компетентності. Загальні компетентності – не мають прив'язки до конкретної предметної області, мають, так званий, універсальний характер. Загальні компетентності набуваються студентом у процесі виконання певної освітньої програми. Фахові компетентності формуються залежно від предметної галузі, зокрема комп'ютерних наук, й визначають профіль освітньо-професійної програми та кваліфікацію випускника [24], [130].

Аспекти формування професійних(фахових) компетентностей майбутніх бакалаврів ІТ-спеціальностей, зокрема бакалаврів комп'ютерних наук, висвітлені у роботах К. Зубик Л [131] Т. Морозової [132], В. Круглика [44] О. Кривоноса [136], С. Литвинова [24], В. Круглика [44], К. Стрюка [49] та інших науковців.

Тенденції розвитку професійної підготовки майбутніх бакалаврів з ІТ-спеціальностей, зокрема бакалаврів комп'ютерних наук, містять: спеціалізацію освіти, фундаменталізацію її змісту, багаторівневність, формування професійних компетентностей, гнучкість навчальних планів, посилення практичної спрямованості, впровадження інноваційних технологій навчання, залучення інтерактивних методів навчання, індивідуалізацію навчання, підвищення ролі науково-педагогічних працівників з високим науково-педагогічним

потенціалом, здатних вибудовувати міждисциплінарні навчальні курси і здійснювати продуктивну професійну комунікацію у сфері ІТ-освіти [131].

Аналізуючи вищу освіту в Україні, автор Т. Морозова [132] розглядає формування фахових компетентностей майбутніх бакалаврів комп'ютерних наук, як складний багатофункціональний процес, спрямований на оволодіння стійкими, інтегрованими, фаховими знаннями, вміннями застосувати їх у нових, нестандартних ситуаціях з метою забезпечення розвитку особистісних якостей і властивостей, що визначаються здатністю до продуктивної професійної діяльності. Усі чинники спрямовані на реалізацію власного творчого потенціалу [132].

На основі аналізу джерел різних авторів, щодо формування професійної компетентності ІТ-фахівців, В. Круглик [44] констатує, що умовами розвитку професійної компетентності студентів є організаційно-управлінські (навчальний план, графіки семестрів, розклади занять, встановлені критерії рівнів компетентності, матеріально-технічне забезпечення навчального процесу), навчально-методичні (відбір змісту занять, інтеграція різних курсів, виокремлення провідних ідей), технологічні (контрольні оцінювання, організація активних форм навчання, визначення вмінь, які входять до компетентності, застосування інновацій навчання), психолого-педагогічні (діагностика розвитку студентів, розробка системи стимулювання мотивації навчання, визначення критеріїв компетентності, рефлексивно-оцінюючий етап кожного заняття [24], [44].

Відповідно, до освітнього стандарту спеціальності F3 «Комп'ютерні науки» можемо зазначити, що фахова компетентність бакалаврів комп'ютерних наук – це здатність «проводити теоретичні та експериментальні дослідження в галузі комп'ютерних наук; застосовувати математичні методи й алгоритмічні принципи в моделюванні, проєктуванні, розробці та супроводі інформаційних технологій; здійснювати розробку, впровадження і супровід інтелектуальних систем аналізу й обробки даних організаційних, технічних, природничих і соціально-економічних систем [24], [133].

У своїх дослідженнях, В. Круглик, знання, вміння, навички, здатності та способи діяльності згрупував у 7 складових професійної компетентності майбутніх інженерів-програмістів: фахову (з програмування, цифрова, математична, інженерна) і загальну (комунікативна, особистісно-професійна й управлінська) компетентності [24], [44].

Він наголошує що: компетентність з програмування це здатність до засвоєння та використання алгоритмічного стилю мислення та написання програм на різних мовах програмування; математична компетентність — це здатність застосовувати ґрунтовні знання з математики для вирішення професійних завдань з програмування; інженерна компетентність це здатність якісно створювати ПЗ на інженерній основі, адаптуючи його до колективної розробки ПЗ з використанням методів управління проєктами; комунікативна компетентність це здатність спілкуватися та комунікувати з колегами у сфері професійної діяльності розробки ПЗ; управлінська компетентність це здатність інтегрувати управлінські та організаторські вміння в процесі розробки ПЗ; особистісно-професійна компетентність це здатність виявляти внутрішній потенціал особистості, щодо професійної діяльності [24], [44].

ДОДАТОК В

Дослідження поняття «веборієнтоване навчання»

Вимоги до результатів навчання студентів в університетах орієнтують викладача на використання інструментарію, що відповідає сучасному рівню розвитку технологій подання, опрацювання і передачі інформації, а також потребам студентів нового тисячоліття. Очевидно, що інструментальну основу освітнього процесу в ЗВО мають становити засоби і сервіси ІКТ, доповненої та віртуальної реальності, ХО та мережні технології. Викладач має сприяти формуванню у студентів мотивації до пізнання навколишнього світу за допомогою інтернету та сучасних високотехнологічних засобів, які забезпечують повсюдний доступ до освітнього контенту й новітніх ІК-систем навчання, перетворюючи таку діяльність на природну потребу [84].

Зокрема, очікуваними результатами прийняття Закону України «Про освіту» [85], [86], та запровадження нових освітніх стандартів створення системи освіти нового покоління, яка забезпечуватиме умови для здобуття освіти всіма категоріями населення України та бути ефективною системою забезпечення всебічного розвитку людини, а також функціонуватиме як ефективний механізм всебічного розвитку особистості. Така система має сприяти істотному зростанню інтелектуального, культурного та духовно-морального потенціалу суспільства і особистості [9], [11].

Тому, ми сьогодні активно обговорюємо нові підходи до організації веборієнтованого навчання бакалаврів в українських університетах, зокрема для підготовки бакалаврів комп'ютерних наук, що потребує обґрунтування таких положень, як мотиваційне, технологічне, процесуальне забезпечення освітнього процесу як цілісної педагогічної системи, що враховує індивідуальні інтереси, здібності і схильності та новітні стратегії навчання студентів XXI ст [1].

У наукових працях В. Бикова [87], [88], О. Спіріна [89], С. Литвинової [90], [91], С. Семерікова [92], А. Стрюка [92], Г. Ткачука [93], В. Кухаренко [94], [95], С. Триуса [96], К. Лісецького [97] та інших висвітлено

різні аспекти веборієнтованого навчання бакалаврів закладів вищої освіти, що розкривають питання теоретичного, практичного, організаційного, загальнодидактичного, науково-методичного та психологічного супроводу освітнього процесу.

Науковцем В. Биковим [87] було проаналізовано модель організаційних систем відкритої освіти, застосування яких є можливим за умов використання веборієнтованих технологій, а також досліджено дистанційне навчання як складову веборієнтованого навчання. Зокрема, академік В. Биков, наголошує, що дистанційне навчання – це навчання, яке «принципово базується на використанні інформаційних і комунікаційних технологій» де «активні учасники навчального процесу здійснюють переважно індивідуалізовану навчальну взаємодію як асинхронно, так і синхронно в часі» [84]. [88].

У своїх роботах О. Спірін [89] та Т. Вакалюк [89] розглядають основні можливості використання різних видів веборієнтованих технологій, зокрема їх аналіз при навчанні програмування. Науковець С. Литвинова констатує, що хмаро орієнтоване навчальне середовище є складовою веборієнтованого навчання і наголошує, що синтетичне освітнє середовище має бути веборієнтованим для дати можливість студентам задіяти інноваційні засоби навчання [84], [90], [91].

Дослідники С. Семеріков та А. Стрюк пропонують авторську організаційну модель комбінованого навчання у закладах вищої освіти, що передбачає використання системи управління навчанням та відображає поточний стан розвитку теорії та методики використання ІКТ в освіті [92], [84].

Використання веборієнтованих технологій змінюють принципи та підходи до навчання програмування майбутніх бакалаврів комп'ютерних наук у ЗВО. Відповідно виникає необхідність переосмислення та реструктуризації освітнього процесу, оновлення змісту освітньої програми, форм, методів та засобів викладання. Особливо швидко та суттєво відбуваються зміни в організації навчання та використанні веборієнтованих та хмаро орієнтованих технологій як інструментів навчання. Такому стрімкому впровадженню сприяв

розвиток інформаційно-комунікаційних технологій, технічне (комп'ютерне) оснащення студентів, їх високий рівень комп'ютерної грамотності і повсюдний доступ до мережі Інтернет [84].

Вагомий внесок у поширення новітніх ІК-технологій у вищій освіті зроблено в межах низки міжнародних проєктів. Зокрема новітнім стратегіям навчання присвячено міжнародний проєкт Erasmus + «Curriculum for Blended Learning», у якому беруть участь Латвія, Австрія та Великобританія. Цей проєкт має на меті підвищити рівень національного та європейського розуміння веборієнтованого навчання, здійснити цілеспрямовану інтеграцію ІКТ в освітній процес, об'єднати інноваційні практики та системи, а також упровадити науково обґрунтовані стратегії інтеграції ІКТ [84], [93].

Українські експерти, спираючись на результати власних досліджень і досвід зарубіжних колег, розробили і продовжують розробляти підходи до організації веборієнтованого навчання, методологію й систему практичних рекомендацій, які можуть бути застосовані у ЗВО України [84].

Так, В. Кухаренко [94] зазначає, що руйнівні підходи до організації навчання бакалаврів дають новий імпульс розвитку освіти, є більш ефективними, доступними, індивідуалізованими, та з часом будуть превалювати над традиційними підходами. Він також констатує, що веборієнтоване навчання «як руйнівна технологія не може з'явитися сама по собі». Необхідно, щоб сформувалися передумови для її виникнення, а також додаткові зусилля, як з боку викладачів, так і студентів під час її впровадження та розвитку [94].

Аналізуючи роботи різних авторів, В. Кухаренко [95] разом з науковцями С. Березенською [95], К. Бугайчук [95] та іншими, уточнюють, що веборієнтоване навчання це:

- поєднання елементів традиційного навчання й онлайн курсів;
- комбінація педагогічних теорій і технологій;
- навчальна методологія, викладання та підхід, який поєднує в собі

традиційні методи в класі з комп'ютерною діяльністю для навчання;

- результат інтегрування онлайн курсів з традиційними навчанням;
- комбінація різних технологій в єдиний інтегрований навчальний підхід;
- навчальна програма, що містить суміш очного та електронного навчання, спектр форматів і медіа [95].

У своїй роботі Ю. Триус [96] трактує веборієнтоване навчання як «цілеспрямований процес здобування знань, набуття умінь і навичок, засвоєння способів пізнавальної діяльності суб'єктом навчання та розвитку його творчих здібностей на основі комплексного і систематичного використання традиційних, інноваційних педагогічних технологій та інформаційно-комунікаційних технологій навчання за принципами взаємного доповнення з метою підвищення якості освіти» [84], [96].

Веборієнтоване навчання – це освітня концепція, що комбінує традиційне навчання з дистанційними та онлайн методами, яка дозволяє студентам контролювати час, місце, траєкторію і темп процесу навчання, зазначає К А Лісецький [84], [97].

На нашу думку, під веборієнтованим навчанням слід розуміти систему форм, методів і веборієнтованих засобів, спрямованих на формування фахових компетентностей з програмування майбутніх бакалаврів комп'ютерних наук. Слід зауважити, що найбільш ефективною формою навчання є така технологія, як змішане навчання (blended learning), що слугує основою для впровадження інноваційних технологій, зокрема веборієнтованого навчання у закладах вищої освіти.

Науковці інституту Клейтона Крістенсена (The Clayton Christensen Institute, США) [98] наголошують, що змішане навчання передбачає використання мережі Інтернет для забезпечення більш персоналізованого навчального досвіду та надання здобувачам освіти контролю над часом, місцем і темпом навчання. У своєму визначенні змішаного навчання, як складової веборієнтованого навчання, наголошують, що «змішане навчання передбачає використання мережі Інтернет, щоб дати кожному студенту більш

персоналізований навчальний досвід, і разом з тим збільшити контроль над студентами за часом, місцем або темпом навчання“ [98], а також зазначають, що веборієнтоване навчання має високий потенціал. І справа не тільки в підходах до організації, відбору технологій і способах навчання, основна ідея – персоналізація навчання [84].

Науковці інституту Клейтона Крістенсена (The Clayton Christensen Institute, США) [98] наголошують, що змішане навчання передбачає використання мережі Інтернет для забезпечення більш персоналізованого навчального досвіду. У визначенні змішаного навчання, як складової веборієнтованого навчання зазначено, що «змішане навчання передбачає використання мережі Інтернет, щоб забезпечити кожному студенту більш персоналізований навчальний досвід, і водночас розширити його можливості контролювати час, місце та темп навчання“ [98]. Також дослідники підкреслюють високий потенціал веборієнтованого навчання. При цьому ключове значення мають не лише підходи до організації навчання, добір технологій і методів навчання, а насамперед персоналізація освітнього процесу[84].

На думку іноземних фахівців, персоналізація навчання передбачає такі основні тенденції: формування глибокого студентоорієнтованого освітнього досвіду; активніше використання електронних засобів навчання; розвиток навичок мислення вищого порядку; забезпечення взаємної користі для викладачів і студентів; упровадження змішаного та інтерактивного підходів, а також продуктивної гейміфікації тощо [84], [98].

Різні визначення веборієнтованого навчання можна знайти в роботах вчених, які проводять свої дослідження в галузі педагогічної методології. У своїй доповіді про можливі наслідки впровадження технології та підходів до організації веборієнтованого навчання Консорціум Слоан визначив засоби гібридної або змішаної освіти як ті, які «інтегрують онлайн навчання з традиційними заняттями в аудиторії у рамках спланованого, педагогічно ефективного методу». Вони розглядають підхід до організації веборієнтованого

навчання як освітній підхід, який «використовує онлайн технології, щоб не просто доповнювати, а трансформувати і покращувати процес навчання» [84]. [99].

Здобувачі вищої освіти відрізняються індивідуальними особливостями сприйняття навчальної інформації, темпами засвоєння знань та освітніми потребами, тому універсального способу ефективного навчання не існує. У зв'язку з цим веборієнтоване навчання, яке ґрунтується на принципах змішаного навчання, розглядається як ефективний підхід до організації освітнього процесу. Воно забезпечує можливості індивідуалізації навчання, поєднання самостійної роботи студентів із педагогічною підтримкою викладача, а також формування гнучких індивідуальних траєкторій і темпів опанування навчального матеріалу [84].

Розглянемо основні підходи до організації веборієнтованого навчання студентів, зокрема бакалаврів комп'ютерних наук, що можуть бути реалізованими в технічних ЗВО.

Підхід до організації веборієнтованого навчання за сценарієм A La Carte передбачає перегляд і вивчення саме онлайн-запису курсу викладача та спрямований на супровід інших видів навчання. Студентам надається можливість навчатися, переглядаючи онлайн-записи лекцій викладача, що надає їм більшої гнучкості у питаннях присутності в ЗВО. [100]. Навчання за сценарієм La Carte доцільно застосовувати у випадках, коли ЗВО не можуть повною мірою забезпечити необхідні умови для навчання студентів з особливими освітніми потребами. Цей формат є одним із поширених підходів до організації веборієнтованого навчання у світовій освітній практиці [84], [100].

У дослідженні зазначено, що підхід до організації веборієнтованого навчання за сценарієм La Carte може реалізовуватися із використанням відомих онлайн-платформ, зокрема Coursera (США) та Prometheus (Україна), які пропонують значну кількість безкоштовних онлайн-курсів з різних галузей знань, у тому числі з програмування [84].

Проект Prometheus є громадською платформою масових відкритих онлайн-курсів в Україні. Мета проекту є надання доступу до онлайн-курсів всім охочим. На зазначеній платформі представлено курси не лише викладачів провідних українських ЗВО, а й професорів з закордонних університетів. Наприклад, із 2016 року курс «CS50: «Основи програмування», створений на основі лекцій Гарвардського університету [101], активно використовується в ЗВО України у форматі змішаного навчання [84].

Студент має доступ до відеолекцій, записаних прямо в аудиторії у форматі живого спілкування, конспектів, завдань, додаткових відеоматеріалів та семінарів від провідних фахівців у галузі комп'ютерних наук. Зауважимо, що ще у 2015 році Сльський університет відмовився від власного вступного курсу програмування для першокурсників на користь використання «CS50», і ця практика триває донині [84].

Підхід до організації веборієнтованого навчання за сценарієм ротації станцій застосовується для переміщення студентів в межах однієї аудиторії, або у групи аудиторій, одна з яких пов'язана з онлайн навчанням. Організація навчання за цим підходом полягає в тому, що частина занять відбувається в звичайних аудиторіях (фронтальна робота викладача зі студентами), а після чого студенти переходять до комп'ютерного класу, де індивідуально працюють за комп'ютерами або планшетами для закріплення знань [84].

У підході до організації веборієнтованого навчання за сценарієм ротації лабораторій багато спільного з підходом за сценарієм ротації станцій. Різниця в тому, що під час ротації станцій студенти переміщуються в межах однієї аудиторії, тоді як у ротації лабораторій студенти переходять до навчальної лабораторії для здійснення онлайн навчанням [100].

Підхід до організації веборієнтованого навчання за сценарієм індивідуальної ротації надає можливість студенту навчатись за індивідуальним графіком та індивідуальною програмою навчання. Студенти

переміщуються відповідно до індивідуального графіка, розробленого викладачем [100].

У підході до організації веборієнтованого навчання за сценарієм «перевернутого навчання» передбачається, що студенти за допомогою різноманітних гаджетів прослуховують і переглядають відеоуроки, самостійно опрацьовують додаткові джерела у позаурочний час, зокрема курси на платформі Prometheus(<https://courses.prometheus.org>)Після цього в аудиторії відбувається спільне обговорення нових понять та ідей, а викладач сприяє застосуванню отриманих знань на практиці. Такий підхід стимулює взаємне навчання студентів [104]. Індивідуальні вправи, практичні та самостійні роботи також виконуються студентами онлайн. Іншими словами, цей підхід «змінює місцями» аудиторну та домашню роботи: пасивне навчання (читання та перегляд відеолекцій) студенти здійснюють удома, тоді як активне навчання відбувається в аудиторії під час обговорення матеріалу на глибшому рівні [84]. [102].

Науковиця С. Литвинова [102] у своїх дослідженнях доводить загальні переваги організації веборієнтованого навчання за сценарієм «перевернутого навчання», зокрема: створюються умови для активного навчання, реалізується диференційний підхід, використовуються новітні технології різні гаджети, освітній процес організовується з урахуванням потреб кожного студента; створюються умови для командної роботи; розвиваються лідерські якості студентів рамках навчальних дисциплін, навчання носить характер персоналізованого, відбувається активна взаємодія викладача і студента, створюються умови доступності до навчальних матеріалів; створюються умови для діагностики якості знань за допомогою комп'ютерних технологій; батьки мають можливість брати участь в освітньому процесі студента [102].

Аналізуючи різні аспекти використання веборієнтованого навчання бакалаврів комп'ютерних наук в ЗВО України, зупинимось на підході, запропонованому науковцем В. М. Кухаренком. Цей підхід є методичною системою, що базується на технології змішаного навчання та забезпечує цілісне

уявлення про внутрішню структуру зміст, взаємозв'язки елементів процесу навчання технічних дисциплін і поєднує 30% технологій традиційного навчання та 70% технологій дистанційного навчання [95]. Спираючись на результати власних досліджень і праці зарубіжних учених автори пропонують поєднання 30% технологій традиційного навчання та 70% технологій дистанційного навчання [84].

При проектуванні цього підходу до організації веборієнтованого навчання, було застосовано класичні дидактичні принципи: свідомості, наочності, систематичності, міцності, доступності, науковості навчання, а також зв'язку теорії з практикою [95]. Також враховано принципи, що базуються на використанні MOOC (Massive Open Online Course), зокрема принципи педагогіки співробітництва та соціального навчання [84].

Хочеться відмітити, що при використанні веборієнтованого навчання застосовується методика, що включає: форми (індивідуальна, групова, колективна), методи (евристична бесіда, мозковий штурм, дискусія, ситуаційний аналіз, метод проєктів, навчальний квест тощо), засоби (віртуальні тренажери, електронні підручники тощо).

Ще одним з підходів веборієнтованого навчання є використання системи Moodle (Modular Object-Oriented Dynamic Learning Environment). Викладач у середовищі Moodle розміщує навчально-методичне забезпечення дисципліни у різних форматах: текстовому, графічному, анімаційному, гіпертекстовому та ін. [84].

Для підвищення мотивації студентів до вивчення навчального матеріалу, також розміщує відеоуроки. Студент самостійно опрацьовує навчальний матеріал, виконує необхідні навчальні завдання, складає іспити та заліки у вигляді тестування, анкетування, залучається до форуму, e-mail. Це дає можливість студентам спілкуватись як з одногрупниками так і з викладачем і задавати питання не чекаючи лекції. Але, як зауважує Г. Чередніченко, «викладачеві потрібно чітко організовувати навчальний процес, стимулювати самоконтроль і розвивати різні способи продуктивної праці зі студентами,

формувати стійку мотивацію до навчально-пізнавальної діяльності, яка повинна підтримуватися на всьому протязі процесу навчання» [84], [103].

Додамо, що методика веборієнтованого навчання буде ефективно працювати з урахуванням таких складових [95]:

- високоякісного динамічного контенту (персоналізація навчання студента, використанням адаптивних технологій, узгодження з національними стандартами);
- аналітичних можливостей систем управління навчанням (LMS);
- автоматизованого робочого місця викладача;
- мотивація студентів.
- врахування принципів, що базуються на використанні MOOC (Massive Open Online Course), а саме принципів педагогіки співробітництва та соціального навчання [84]. [95].

Важливим фактором у підходах до веборієнтованого навчання бакалаврів комп'ютерних наук є забезпечення доступу студентів-програмістів до інноваційних ресурсів, які здійснюються на засадах веборієнтованих технологій, як потужного інструменту повсюдного доступу до інформаційних і програмних систем [84].

Слід зауважити, що одним із важливих моментів у веборієнтованому навчанні бакалаврів комп'ютерних наук є самостійна робота, для організації якої необхідні такі умови: мотивація студента до самостійної роботи; наявність і доступність навчально-методичного забезпечення та довідкового матеріалу; наявність комп'ютерних класів; система регулярного контролю якості виконаної [84].

Різні визначення веборієнтованого навчання можна знайти в роботах вчених, які проводять свої дослідження в галузі педагогічної методології. У своїй доповіді про можливі наслідки впровадження технології та підходів до організації веборієнтованого навчання Консорціум Слоан визначив засоби гібридної або змішаної освіти як ті, які «інтегрують онлайн навчання з традиційними заняттями в аудиторії у рамках спланованого, педагогічно

ефективного методу». Вони розглядають підхід до організації веборієнтованого навчання як освітній підхід, який «використовує онлайн технології, щоб не просто доповнювати, а трансформувати і покращувати процес навчання» [98].

Зараз українські експерти знаходяться на початку аналізу використання технологій веборієнтованого навчання. На основі досвіду і робіт закордонних колег вони розробляють підходи до організації веборієнтованого навчання, методологію і систему практичних рекомендацій, які можуть бути застосовані у вищих навчальних закладах України. Так В. Кухаренко зазначає, що руйнівні підходи до організації навчання бакалаврів дають новий імпульс розвитку освіти, є більш ефективними, доступними, індивідуалізованими, та з часом будуть превалювати над традиційними підходами. Він також констатує, що веборієнтоване навчання «як руйнівна технологія не може з'явитися сама по собі». Необхідно, щоб сформувалися передумови для її виникнення, а також додаткові зусилля, як з боку викладачів, так і студентів під час її впровадження та розвитку [95]

Ю. Триус трактує веборієнтоване навчання як цілеспрямований процес здобування знань, набуття умінь і навичок, з метою підвищення якості освіти, і трактує його як комплексне використання традиційних та інноваційних технологій на принципах взаємодоповнення [96].

Веборієнтоване навчання - це освітня концепція, що комбінує традиційне навчання з дистанційними та онлайн методами, яка дозволяє студентам контролювати час, місце, траєкторію і темп процесу навчання, зазначає К. Лісецький [97]. Він констатує, що головна ідея змішаного навчання, зокрема веборієнтованого навчання, є інноваційною моделлю організації освітнього процесу, яка поєднує традиційні та цифрові підходи, змінює ролі учасників навчання, підвищує ефективність і гнучкість вищої освіти, що дозволяє контролювати час, місце, траєкторію і темп навчання.

У своїх дослідженнях В. Биков розглядає дистанційне навчання, як систему, що базується на індивідуалізованій взаємодії через електронні транспортні мережі і яка є складовою веборієнтованого навчання [87].

Нижче наведено результати порівняльного аналізу дефініцій веборієнтованого навчання, представлених у вітчизняній науковій літературі та використаних у дисертації (табл. В.1).

Таблиця В.1

Порівняльний аналіз дефініцій веборієнтованого навчання

Автор / Джерело	Ключове визначення	Акцент у методиці
В. Ю. Биков	Система, що базується на індивідуалізованій взаємодії через електронні транспортні мережі	Автономія та гнучкість темпів навчання
Ю. В. Триус	Комплексне використання традиційних та інноваційних технологій на принципах взаємодоповнення	Гібридний формат (30% традиційне, 70% інноваційне)
К. А. Лісецький	Концепція, що дозволяє контролювати час, місце, траєкторію і темп навчання	Персоналізація освітньої траєкторії
Авторське визначення	Інтегрована дидактична система, що реалізується на принципах неперервності та інтелектуальної співпраці	Формування цифрової ідентичності студента

Зокрема, академік В. Ю. Биков, наголошує, що дистанційне навчання це навчання, яке “принципово базується на використанні інформаційних і комунікаційних технологій” де “активні учасники навчального процесу здійснюють переважно індивідуалізовану навчальну взаємодію як асинхронно, так і синхронно в часі” [88].

Студенти всі різні, сприймають інформацію по різному, в різному темпі. Немає «правильного» способу засвоєння знань. Саме веборієнтоване навчання, основою якого є змішане навчання, направлене на те, щоб допомагати студенту,

який матиме змогу отримувати знання як самостійно, так і з викладачем, скласти гнучкий індивідуальний графік навчання тощо.

Веборієнтовані технології навчання - це сукупність методів, інструментів та платформ, що використовуються для розробки, реалізації та підтримки освітнього процесу з використанням Інтернету [4]. Ці технології забезпечують інтерактивність, доступність та мультимедійність навчання, дозволяючи студентам та викладачам ефективно взаємодіяти незалежно від їхнього місцезнаходження.

У контексті трансформації форм підготовки бакалаврів комп'ютерних наук, під методикою навчання програмування з використанням веборієнтованих технологій ми розуміємо інтегровану дидактичну систему, що базується на принципах неперервності та інтелектуальної співпраці, спрямовану на формування цифрової професійної ідентичності бакалавра через організацію діяльності у хмаро орієнтованому середовищі розробки.

Оцінювання ефективності використання веборієнтованих технологій у навчанні бакалаврів комп'ютерних наук є важливим для забезпечення якості освіти та постійного вдосконалення навчального процесу.

Ефективність можна оцінювати за кількома критеріями:

- успішність студентів (академічні результати),
- задоволеність студентів та викладачів,
- технічна підтримка,
- інтеграція технологій у освітній процес.

Для реалізації методики при викладанні дисципліни «WEB-орієнтовані технології. Frontend-розробки» (3 курс, бакалавр) обрано наступні форми організації освітнього процесу: лекції (традиційні чи інтерактивні для обговорення тем дисциплін та опрацювання теоретичного матеріалу, ведення дискусій та обговорень по темах дисциплін); комп'ютерні практикуми (виконання завдань в індивідуальній та проєктній роботах, моделювання професійних ситуацій); самостійна робота (самостійне опрацювання матеріалів та пошуку практичного впровадження отриманих

знань та умінь); модульна контрольна робота (тестування, опитування для контролю якості отриманих знань та умінь); проведення консультацій (індивідуальних і групових). Командний, індивідуальний проєкт, мозковий штурм (підрозділ 2.1).

ДОДАТОК Г

Реалізація підсумкового оцінювання навчальних досягнень студентів з програмування

Відповідно до запропонованої методики оцінювання, результати виконання комп'ютерного практикуму визначається за такими критеріями: виконання завдання, захист роботи, якість виконання практикуму, якість звіту та дотримання встановлених термінів виконання [109].

З метою забезпечення об'єктивності формульовального оцінювання пропонуються таблиці відсоткового коригування підсумкової оцінки за виконання комп'ютерного практикуму (табл. Г.1 – Г.5), які можуть бути використані викладачами під час оцінювання комп'ютерних практикумів з програмування.

Таблиця Г.1

Таблиця зниження відсотків за виконання комп'ютерного практикуму

№	Вид помилки	% зниження
1	Синтаксичні помилки (не виконується компіляція програми)	100
2	Некоректний алгоритм реалізації задачі	25
3	Результат не відповідає умові завдання	10
4	Неінформативний вивід та ввід даних	10
5	Помилки у введеннях даних	10
6	Логічні помилки, що приводять до виключних ситуацій(комп'ютер «завісає»)	45
Разом		100

Таблиця Г.2

Таблиця зниження відсотків за захист комп'ютерного практикуму

№	Вид помилки	% зниження
1	Студент не може пояснити програмний код комп'ютерного практикуму	65
2	Студент не відповідає на контрольні питання за темою комп'ютерного практикуму	10
3	Студент не може внести програмний код у програму для виправлення логічних помилок	15
4	Студент не знає теоретичного матеріалу з теми комп'ютерного практикуму	10
Разом		100

Таблиця Г.3

Таблиця зниження відсотків за звіт з комп'ютерного практикуму

№	Вид помилки	% зниження
1	Недотримання встановлених вимог до структури звіту	10
2	Відсутність обов'язкових структурних елементів сторінки	15
3	Неповний опис мети або завдань роботи	5
4	Відсутність опису використаних технологій (HTML, CSS, JavaScript)	35
5	Неповний або поверхневий опис етапів реалізації завдання.	10
6	Відсутнє посилання на репозиторій власного застосунку	5
7	<i>Відсутнє посилання на репозиторій звітнього HTML-документа</i>	5
8	Висновки не відповідають поставленій меті та завданням	5
9	Висновки відсутні	10
Разом		100

Таблиця Г.4

**Таблиця зниження відсотків за термін затримки виконання
комп'ютерного практикуму**

	Термін затримки	% зниження
1	Два тижня	10
2	Від двох тижнів до чотирьох тижнів (місяць)	20
3	Від чотирьох тижнів до шести тижнів	35
4	Від шести тижнів до восьми тижнів (два місяці)	50
5	Від восьми тижнів до 12 тижнів	70
7	Більше, ніж 12 тижнів	80

Примітка: Показники зниження балів за синтаксичні помилки (100%) та некоректний алгоритм (25%) визначаються автоматизовано за допомогою систем типу E-olymp або веборієнтованих скриптів тестування коду, що забезпечує об'єктивність та миттєвий зворотний зв'язок.

Таблиця Г.5

Таблиця підвищення відсотків за виконання комп'ютерного практикуму

№	Вид нестандартного підходу	% підвищення
1	Нетрадиційний алгоритм розв'язання задачі.	30
2	Використання програмних конструкцій, які оптимізують програмний код за критеріями швидкодії або обсягу оперативної пам'яті	20
3	Використання зручного інтерфейсу користувача. (меню, вікна діалогу)	10
4	Вивчення додаткової літератури, що виходить за межі учбової програми (евристичні алгоритми комбінаторики, програмування для Windows тощо).	20
5	Ітеративність розробки, зафіксована через історію комітів у GitHub, та наявність якісного Code Review	20
Разом		100

Важливим складником підсумкового оцінювання є аналіз «цифрового сліду» студента в хмарному середовищі розробки (GitHub/GitLab). На відміну від одноразової перевірки фінального файлу, викладач оцінює ітеративність розробки через історію записів змін у Git. Це дозволяє перевірити академічну доброчесність (чи був код написаний поступово, чи завантажений готовим блоком) та сформувати об'єктивну оцінку навичок командної роботи за стандартами DigComp 3.0.

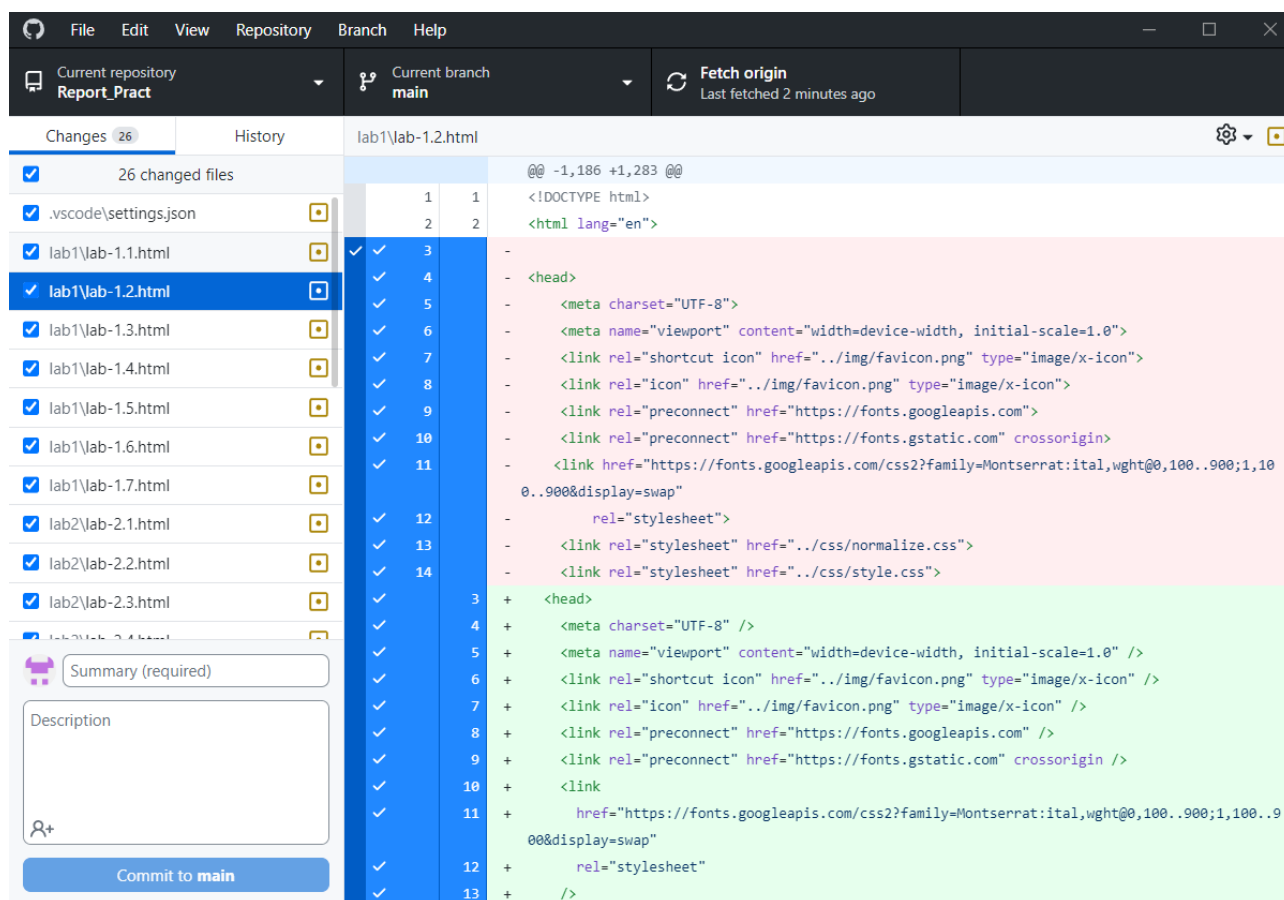


Рис.Г.1. Аналіз «цифрового сліду» студента в хмарному середовищі розробки (GitHub).

Розглянемо приклад формування оцінки за комп'ютерний практикум студента. Проценти зниження у кожній складовій представлені у комп'ютерного практикуму табл. Г.3. Усі отримані відсотки зниження сумуються.

Таблиця Г.3

Нарахування балів зниження/підвищення на кожному рівні таксономії**Блума**

Процедури	Заплановані бали	% зниження	% підвищення	Підсумкові бали
1.Виконання комп'ютерного практикуму		25		
2.Захист комп'ютерного практикуму		0		
3.Звіт з комп'ютерного практикуму		10		
4.Якість комп'ютерного практикуму			0	
5. Термін затримки виконання практичного завдання		0		
Всього	100	35%	0	65

Іншими словами, наприклад, студент виконує комп'ютерний практикум, максимальний бал якого дорівнює – 100. Припустимо, що студент у підрозділі «Виконання комп'ютерного практикуму» отримує 25% зниження за некоректний алгоритм реалізації задачі і у підрозділі «Захист комп'ютерного практикуму» — 10% за неспроможність відповісти на контрольні питання за темою комп'ютерного практикуму. Відсотки зниження сумуються. Реальні бали зменшуються в залежності від сумарних відсотків зниження. Студент може і покращити свої результати. Наприклад, якщо у підрозділі «Якість комп'ютерного практикуму» студент отримав відсотки підвищення – підсумковий бал збільшиться.

ДОДАТОК Д

Практичне застосування авторської методики

Практичне застосування авторської методики продемонструємо на прикладі вибіркової дисципліни «Веборієнтовані технології. Frontend-розробка». Навчальний контент розроблено відповідно до програми курсу, що викладається в Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського». До структури обраної навчальної дисципліни входять три розділи :

Розділ 1. Основи клієнтських розробок. HTML-документ. Стильове оформлення елементів HTML-документу. Адаптивна верстка.

Розділ 2. Програмне використання JavaScript у HTML-документах.

Розділ 3. Створення інтерфейсів вебзастосувань з використанням REACT.

Розглянемо структуру дисципліни «Веборієнтовані технології. Frontend-розробка» більш детально.

Розділ 1. Основи клієнтських розробок. HTML-документ. Стильове оформлення елементів HTML-документу. Адаптивна верстка.

Мета розділу: Ознайомити студентів з основами веброзробки на клієнтському рівні, навчити створювати структуру HTML-документу, оформлювати його з використанням CSS та формувати адаптивні, зручні до сприйняття макети для різних типів пристроїв.

Завдання:

1. Вивчити структуру HTML-документу та основні теги.
2. Освоїти принципи стилізації за допомогою CSS.
3. Розібрати верстку сторінок за макетом (Flexbox).
4. Засвоїти техніки адаптивного дизайну.
5. Ознайомитися з декоративними елементами та ефектами.
6. Навчитися керувати розміщенням елементів і анімацією на сторінці.

Зміст тем розділу:

- Тема 1.1: Основи Інтернет-технологій, види сайтів, структура HTML-документа, вивчення базових тегів і валидація коду.
- Тема 1.2: CSS-стилізація: селектори, псевдокласи, оформлення тексту, каскадування, змінні.
- Тема 1.3: Верстка за макетом: типи макетів, блокова модель, Flexbox, структурні псевдокласи.
- Тема 1.4: Декоративні ефекти: фони, object-fit/position, SVG, оформлення фігур.
- Тема 1.5: Позиціонування елементів, z-index, overflow, CSS-переходи й анімація.
- Тема 1.6: Адаптивна верстка: медіа-запити, viewport, типи верстки, стратегія Mobile First, адаптивна графіка.

Розділ 2. Програмне використання JavaScript у HTML-документах

Мета розділу: Навчити студентів програмно керувати HTML-документами за допомогою мови JavaScript, опанувати основи логіки, структури даних, DOM-маніпуляцій і роботі з асинхронними процесами.

Завдання:

1. Засвоїти базовий синтаксис JavaScript, роботу зі змінними, константами, типами даних.
2. Навчитися використовувати умовні оператори, цикли, функції та масиви.
3. Ознайомитися з об'єктами та їх методами, перебиранням, деструктуризацією.
4. Використовувати функції зворотного виклику, стрілочні функції та ланцюжки методів.
5. Вивчити DOM, події, делегування подій, throttle і debounce.
6. Зрозуміти принципи модульності коду, структуру проєктів, а також навчитися працювати з npm та Webpack.
7. Освоїти асинхронне програмування: роботу з промісами, HTTP-запитами, REST API та AJAX-технологіями.

Зміст тем розділу:

- Тема 2.1: Змінні, константи, типи даних, розгалуження та цикли.
- Тема 2.2: Масиви, методи, псевдомасив arguments, функції.
- Тема 2.3: Об'єкти, методи, масиви об'єктів, spread/rest, деструктуризація.
- Тема 2.4: Callback-функції, forEach, стрілочні функції, ланцюжки методів.
- Тема 2.5: Прототипи, класи, DOM, події, делегування, throttle, debounce.
- Тема 2.6: Модульність, Node.js, npm, Webpack, JSON, LocalStorage.
- Тема 2.7: Асинхронність, проміси, методи класу Promise.
- Тема 2.8: HTTP-запити, REST API, AJAX, CRUD, пагінація.

Розділ 3. Створення інтерфейсів WEB-застосувань з використанням React.

Мета розділу: Навчити студентів розробляти сучасні динамічні інтерфейси користувача за допомогою бібліотеки React, використовуючи компонентну структуру, хуки, маршрутизацію та управління станом через Redux.

Завдання:

1. Освоїти основи React: компоненти, JSX, роботу з подіями та формами.
2. Зрозуміти концепцію життєвого циклу компонентів, хуків, рефів.
3. Навчитися реалізовувати маршрутизацію в додатках.
4. Використовувати Redux Toolkit для керування глобальним станом.
5. Освоїти асинхронну роботу з сервером, оптимізацію роботи з даними.

Зміст тем розділу:

- Тема 3.1: React, JSX, компоненти, DOM, події, форми, стилізація.
- Тема 3.2: Життєвий цикл, хуки, HTTP-запити, контекст, рефи.
- Тема 3.3: Маршрутизація: вкладені, індексні маршрути, програмна навігація.

– Тема 3.4: Redux, Redux Toolkit, createAsyncThunk, оптимізація селекторів.

Слід зауважити, що зміст вибіркової навчальної дисципліни буде систематично змінюватись відповідно до змін в ІТ-індустрії, які в свою чергу будуть враховуватись у стандарті вищої освіти спеціальності F3 [6].

Завдання дисципліни «Веборієнтовані технології. Frontend-розробка» за розробленою методикою полягає в удосконаленні навчання програмування клієнської частини сучасних веборієнтованих застосунків, що сприяє підготовці майбутніх високваліфікованих фахівців у галузі комп'ютерних наук.

Реалізація змісту дисципліни «Веборієнтовані технології. Frontend-розробка» базується на інтегрованому підході, що поєднує розвиток цифрових компетентностей згідно з європейською рамкою DigComp 3.0 та поетапне когнітивне зростання студента за таксономією Блума.

Авторська методика побудована від розробки базових структур до створення складних інтелектуальних продуктів. Кожен розділ відповідає певному когнітивному рівню (за Блумом):

Рівень 1: *«Знання та Розуміння»* (Розділ 1): Студенти опановують термінологію (теги, селектори, каскадування) та принципи побудови адаптивних макетів. На цьому етапі формується фундамент для розуміння семантики вебінтерфейсів.

Рівень 2: *«Застосування та Аналіз»* (Розділ 2): Через вивчення JavaScript фокус зміщується на алгоритмізацію. Студенти не просто пишуть код, а аналізують DOM-дерево, деструктурують дані та оптимізують асинхронні запити (AJAX, CRUD).

Рівень 3: *«Синтез та Оцінювання»* (Розділ 3): Робота з бібліотекою React вимагає від студента здатності синтезувати окремі компоненти в єдину екосистему застосунку. Оцінювання проявляється через вибір оптимального стану (State Management) та архітектурних рішень у Redux Toolkit. Ми виділяємо конкретні дескриптори DigComp 3.0, які безпосередньо корелюють із завданнями дисципліни.

Напрям 3: *Створення цифрового контенту (Core Area)* – це центральна частина методики, що охоплює наступні компетенції:

3.1 *Розроблення цифрового контенту* полягає в створенні структурованих HTML-документів та CSS-стилізації (Розділ 1). Студент переходить від статичного споживання до активного моделювання вебінтерфейсів.

3.2 *Інтегрування та переробка цифрового контенту* полягає в напрацюванні навичок роботи з SVG, медіа-запитами та сторонніми бібліотеками (Розділ 1.4, 3.1).

3.3 *Програмування* – це поглиблене вивчення JavaScript та React (Розділи 2 та 3). Це відповідає вищим рівням володіння (Advanced/Specialized), де студент створює інструкції для комп'ютерних систем для розв'язання складних завдань. Особлива увага приділяється здатності студента виступати аудитором AI-генерованого коду, перевіряючи його на відповідність стандартам безпеки (XSS, SQL-ін'єкції) та етичність використання інтелектуальних агентів.

Напрям 4: Безпека.

4.1 *Захист пристроїв та контенту* передбачає опанування методів безпечної передачі даних через HTTP-запити та розуміння принципів валідації форм на клієнтській стороні з метою запобігання вразливостям.

Напрям 5: Розв'язання проблем.

5.2 *Визначення потреб та технологічних відповідей* – формує здатність студента обирати між методами Flexbox чи Grid (Розділ 1) або між звичайним State та Redux (Розділ 3), виходячи з технічних вимог проєкту.

5.3 *Креативне використання цифрових технологій* полягає у створенні анімацій, переходів та інтерактивних компонентів, що виходить за межі копіювання макета.

Реалізація авторської методики спрямована на формування цілісного комплексу знань та вмінь, що відповідають актуальним вимогам ІТ-індустрії. Структура дисципліни, що охоплює три ключові етапи – від базової верстки

до створення складних реактивних інтерфейсів, зумовлює вибір наступних програмних результатів навчання:

- Опанування мови JavaScript (теми 2.1–2.8) безпосередньо спрямоване на розвиток здатності проєктувати, розробляти та аналізувати алгоритми розв’язання логічних задач. Студенти не просто пишуть програмний код, а й оцінюють його ефективність через використання функцій зворотного виклику, асинхронних запитів та маніпуляцій з DOM-структурою, що є фундаментом алгоритмічної підготовки в галузі комп’ютерних наук.

- Вивчення бібліотеки React (теми 3.1–3.4) дозволяє студенту свідомо вибирати парадигму програмування (зокрема компонентно-орієнтований та декларативний підходи) з позицій зручності та якості реалізації інтерфейсів. Це забезпечує формування вміння розробляти програмні моделі складних предметних середовищ, де інтерфейс є динамічною проєкцією стану даних.

Наскрізним результатом навчання є здатність використовувати інструментальні засоби розробки клієнт-серверних застосунків.

У Розділі 1 це реалізується через засвоєння стандартів адаптивної верстки та валідації коду.

У Розділі 2 – через роботу з HTTP-запитами, REST API та AJAX.

У Розділі 3 – через інтеграцію з хмарними сервісами та використання сучасних стеків розробки (Redux Toolkit, Webpack).

Застосування авторської методики передбачає ознайомлення студентів із повним циклом створення Frontend-проєкту: від аналізу макета (Розділ 1) до збірки та оптимізації готового продукту за допомогою Node.js та NPM (Розділ 2.6). Це забезпечує володіння навичками управління життєвим циклом ПЗ та здатність застосовувати робочі стандарти оцінки якості (Mobile First, кросбраузерність, продуктивність).

Оскільки сфера Frontend-технологій характеризується надзвичайно високою динамікою оновлень, критично важливим є програмні результати навчання (ПРН) щодо розуміння іноземної мови на рівні, достатньому для обробки фахових інформаційних джерел. Зміст усіх трьох розділів передбачає

роботу з оригінальною технічною документацією (MDN, React Documentation), що є обов'язковою умовою підготовки висококваліфікованого спеціаліста.

Таким чином, відбір зазначених результатів навчання дозволяє уникнути надмірної теоретизації та зосередити увагу на професійно-орієнтованих компетентностях, що безпосередньо корелюють із завданнями дисципліни.

ДОДАТОК Е

Приклад роботи у середовищі JSFiddle.net

Розглянемо простий приклад роботи у середовищі JSFiddle.net , у якому при натисканні кнопки *Click me*, виведе повідомлення *Hello from JSFiddle!*". Для реалізації цього прикладу необхідно виконати такі дії:

3. Зайти на <https://jsfiddle.net>
4. Ввести код у відповідні поля:
 - HTML —поле розмітки;
 - CSS —поле стилізації;
 - JavaScript — поле реалізації логіки та інтерактивної

поведінки вебзастосунку.

HTML:

```
<h2>Click the button!</h2>
<button onclick="sayHello()">Click me</button>
<p id="output"></p>
```

JS:

```
function sayHello() {
  document.getElementById("output").innerText = "Hello from JSFiddle!";
}
```

CSS:

```
button {
  padding: 10px 20px;
  font-size: 18px;
  background-color: #00bfff;
  color: white;
  border: none;
  border-radius: 5px;
}
```

5. Натиснути кнопку "Run" або використати автооновлення (*Auto-update preview*)

6. У вікні *Result* з'являється результат (див. рис.3.15)

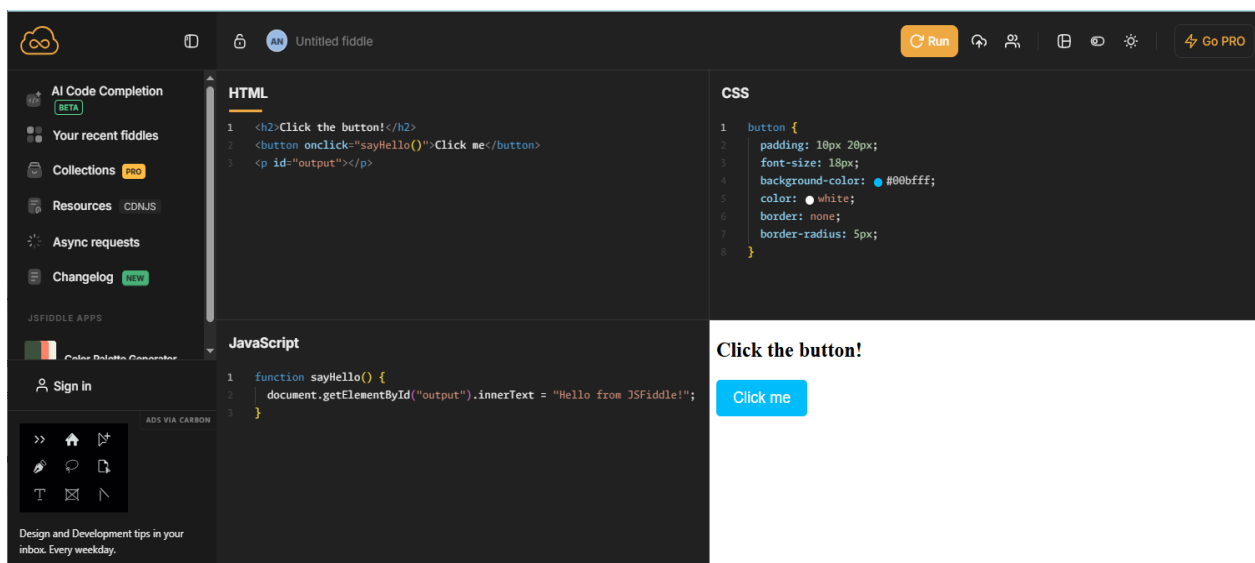


Рис.Е.1. Головне вікно JSFiddle

Переваги JSFiddle:

- *Миттєвий старт*: не потрібна реєстрація, достатньо відкрити та написати програмний код.

- *Просте поширення*: кожен фідл має унікальне посилання.

- *Live preview*: можливість миттєво бачити результат.

- *Гнучкі налаштування*: підтримка багатьох фреймворків.

- *Підходить для навчання*: студентам, менторам, інструкторам

Недоліки JSFiddle:

- *Онлайн-компілятор тільки для Frontend-частини*: немає підтримки Backend (Node.js, PHP тощо).

- *Обмежене спільне редагування*: немає повноцінного “multiplayer mode” як у Replit.

- *Не підходить для великих проєктів*: це повноцінне середовище.

- *Збереження без акаунта тимчасове*: якщо не залогінитися, можна втратити код.

CodePen [16]— це онлайн-редактор для розробки HTML, CSS, JavaScript-коду, який надає можливість:

- писати та тестувати Frontend-код у реальному часі;
- ділитися власними проєктами;
- переглядати приклади інших студентів, зокрема демо-проєкти, анімації, UI компоненти;
- створювати колекції, персональні профілі та шаблони,
- навчатися або надихатися від інших розробників.

Орієнтація: виключно фронтенд.

Немає backend-серверів, але можна підключати API, CDN, Firebase тощо.

У CodePen можна створити “Pen” – міні-проєкт, який складається з:

- HTML: Структура елементів (тіло сторінки);
- CSS: Оформлення елементів;
- JavaScript: Поведінка, логіка, інтерактивність;
- Output (Preview): Живий результат (рендеринг у реальному часі).

Розглянемо приклад практичного завдання у середовищі онлайн-компілятора CodePen. Зайти у середовище компілятора, попередньо зареєструвавшись. Головне вікно містить у верхній частині три розділи : HTML, CSS і JS(JavaScript) для введення відповідного програмного коду і у нижній частині буде надаватись результат виконання програмного коду

Розглянемо приклад практичного завдання у якому потрібно *створити форму зворотного зв'язку з використанням HTML, CSS і JavaScript, використовуючи онлайн компілятор CodePen.*

У кожен розділ вводимо відповідний програмний код, а саме:

- HTML (вкладка HTML)


```
<div class="contact-form">
  <h2>Форма зворотного зв'язку</h2>
  <form id="feedbackForm">
    <label for="name">Ім'я:</label>
    <input type="text" id="name" required />
```

```
<label for="email">Email:</label>
```

```
<input type="email" id="email" required />
```

```
<label for="message">Повідомлення:</label>
```

```
<textarea id="message" rows="4" required></textarea>
```

```
<button type="submit">Надіслати</button>
```

```
</form>
```

```
<p id="confirmation" style="display: none;"> ☒ Повідомлення  
надіслано!</p>
```

```
</div>
```

— CSS (вкладка CSS)

```
body {
```

```
    font-family: Arial, sans-serif;
```

```
    background: #f4f4f4;
```

```
    padding: 30px;
```

```
}
```

```
.contact-form {
```

```
    background: #fff;
```

```
    padding: 20px 25px;
```

```
    border-radius: 10px;
```

```
    max-width: 400px;
```

```
    margin: auto;
```

```
    box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);
```

```
}
```

```
.contact-form h2 {
```

```
    text-align: center;
```

```
    color: #333;
```

```
}
```

```
.contact-form label {  
  display: block;  
  margin-top: 15px;  
}
```

```
.contact-form input,  
.contact-form textarea {  
  width: 100%;  
  padding: 8px;  
  margin-top: 5px;  
  border: 1px solid #ccc;  
  border-radius: 5px;  
}
```

```
.contact-form button {  
  margin-top: 20px;  
  width: 100%;  
  padding: 10px;  
  background-color: #4CAF50;  
  color: white;  
  border: none;  
  border-radius: 5px;  
  cursor: pointer;  
}
```

```
.contact-form button:hover {  
  background-color: #45a049;  
}
```

— JavaScript (вкладка JS)

```
document.getElementById('feedbackForm').addEventListener('submit',
function(e) {
    e.preventDefault();

    const name = document.getElementById('name').value;
    const email = document.getElementById('email').value;
    const message = document.getElementById('message').value;
    if (name && email && message) {
        document.getElementById('confirmation').style.display = 'block';
        this.reset();
    }
});
```

У нижній частині буде виведено результат виконання цих трьох розділів

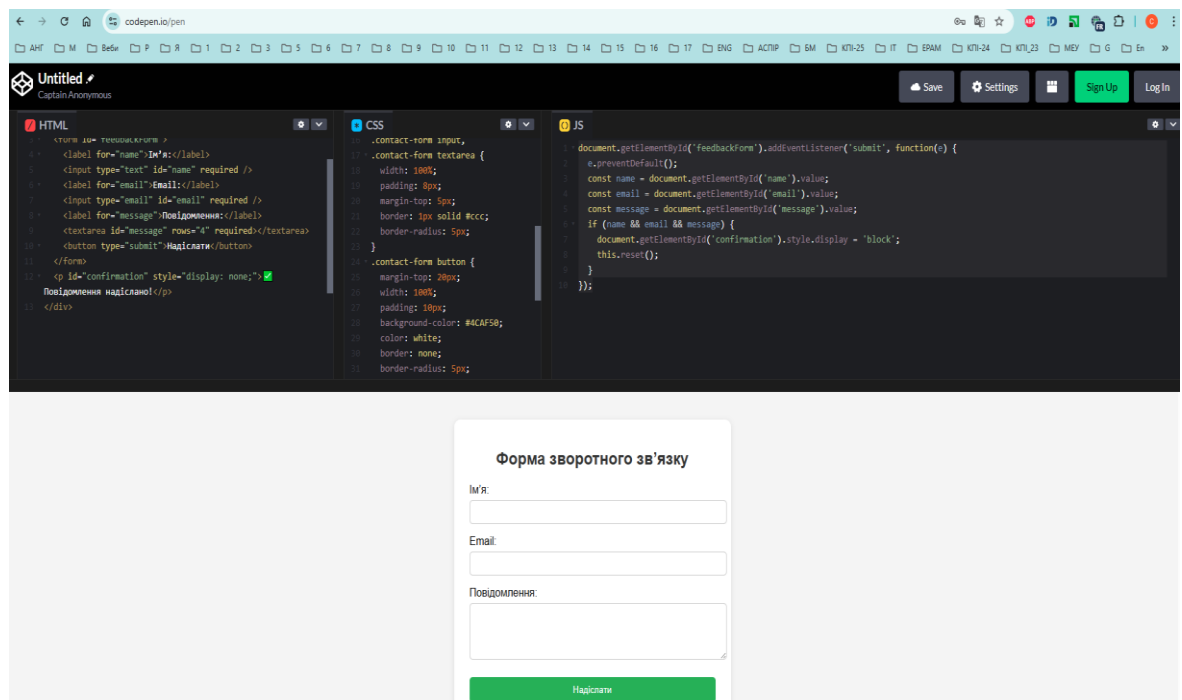


Рис.Е.2. Головне вікно CodePen

ДОДАТОК К

Таблиця К.1

Вивчення мов програмування у вітчизняних ЗВО

Назви закладів вищої освіти України, факультетів та кафедр	МОВИ ПРОГРАМУВАННЯ, ЩО ВИВЧАЛИ У ВЗО									
	БД	C++	Е	Java	C#	Java Script	PHP	QA	Python	Objective-C
РЕСПОНДЕНТИ	81%	74%	60%	44%	37%	30%	26%	15%	9%	4%
Національний університет «Києво-Могилянська академія» (НаУКМА)	97%	95%	83%	90%	32%	41%	41%	25%	8%	10%
Київський національний університет ім.Шевченка (КНУ м.Шевченка)	70%	75%	57%	33%	40%	23%	8%	9%	13%	3%
Харківський національний університет радіоелектроніки (ХНУРЕ)	87%	77%	53%	58%	48%	44%	36%	28%	11%	3%
Чорноморський національний університет ім.Петра Могили (ЧГУ ім.Петра Могили)	95%	91%	56%	67%	40%	81%	67%	49%	40%	16%
Сумський державний університет (СумДУ)	82%	77%	68%	42%	36%	36%	62%	25%	3%	3%
НТУУ "Київський політехнічний інститут ім.І.Сікорського" (НТУУ	80%	77%	67%	49%	44%	27%	21%	15%	15%	4%
Львівський національний університет ім.Франка (ЛНУ ім.Франка)	75%	71%	22%	40%	58%	25%	12%	5%	3%	0%
Дніпроперовський національний університет ім. Гончара (ДНУ	72%	75%	52%	35%	32%	35%	22%	5%	4%	2%
Львівська Політехніка	85%	77%	82%	30%	35%	27%	19%	13%	6%	6%
Одеський національний політехнічний університет	75%	78%	68%	54%	10%	21%	21%	16%	2%	3%

ДОДАТОК Л

Таблиця Л.1

Континентальна модель: дисципліни бакалаврської програми

https://www.ifi.lmu.de/sommersemester-2025/fruehere_semester/wintersemester-2022/index.html

	Зимовий семестр 2023/24	Wintersemester 2023/24
	Дисципліни бакалаврської програми	Lehrveranstaltungen im Bachelorstudium
1	Вступ до програмування , Стрікрот	Einführung in die Programmierung, Strickroth
2	Цифрові медіа , Шмідт	Digitale Medien, Schmidt
3	Вступ до біоінформатики I , Список	Einführung in die Bioinformatik I, List
4	Операційні системи , Linnhoff-Popien	Betriebssysteme, Linnhoff-Popien
5	Програмні технології , Байєр, Вінтер	Softwaretechnik, Beyer, Winter
6	Системи баз даних , Зайдль	Datenbanksysteme, Seidl
7	Формальна специфікація та верифікація , Ернст	Formale Spezifikation und Verifikation, Ernst
8	Мультимедіа в Інтернеті , Піркер	Multimedia im Netz, Pirker
9	Алгоритмічна біоінформатика II , Фрідель	Algorithmische Bioinformatik II, Friedel
10	Досвід користувача 1 , Вітофф	User Experience 1, Wiethoff
11	Досвід користувача 2 , Ульріх	User Experience 2, Ullrich
12	Основи машинного навчання , Хюллермаєр	Grundlagen des Maschinellen Lernens, Hüllermeier
13	Стажування з розробки програмного забезпечення , Шуберт	Softwareentwicklungspraktikum, Schubert
14	Стажування з розробки програмного забезпечення Computer Vision & Deep Learning , Ommer	Softwareentwicklungspraktikum Computer Vision & Deep Learning, Ommer
15	Стажування систем , Kranzlmüller	Systempraktikum, Kranzlmüller
16	Стажування з програмування біоінформатики , Ціммер, Гойн, Фрідель	Programmierpraktikum Bioinformatik, Zimmer, Heun, Friedel
17	Управління юридичними IT-проєктами , Сарре	Juristisches IT-Projektmanagement, Sarre
18	Компактний семінар: Процесно-орієнтоване управління IT-послугами , Кранцльмюллер, Бреннер, Куліг	Kompaktseminar: Prozessorientiertes IT-Service-Management, Kranzlmüller, Brenner, Kuhlig
19	Мультимедійний проєкт з компетенції: Майя , Бутц	Projektkompetenz Multimedia: Maya, Butz
20	Семінар «Новітні теми машинного навчання та штучного інтелекту» , Kranzlmüller	Seminar "Emerging Topics in Machine Learning and AI", Kranzlmüller

	Зимовий семестр 2023/24	Wintersemester 2023/24
21	Семінар «Тенденції розвитку мобільних та розподілених систем» , Ліннхофф-Попієн	Seminar "Trends in Mobilen und Verteilten Systemen" , Linnhoff-Popien
22	Семінар «Геоінформаційні системи в контексті зміни клімату» , Шуберт	Seminar "Geoinformationssysteme im Kontext des Klimawandels" , Schubert
23	Семінар "Інтелектуальний аналіз даних" , NN	Seminar "Data Mining" , NN
24	Семінар "Верифікація програм з циклами та масивами" , Ернст	Seminar "Verifikation von Programmen mit Schleifen und Arrays" , Ernst
25	Семінар «Перевірка результатів верифікації» , Якобс	Seminar "Validation of Verification Results" , Jakobs
26	Семінар «Алгоритми? Алгоритми!» , Майстер Кедр	Seminar "Algorithmen? Algorithmen!" , Majster-Cederbaum
27	Семінар «Моделювання динамічних та адаптивних систем» , Вірзінг, Гессен	Seminar "Modellierung dynamischer und adaptiver Systeme" , Wirsing, Hesse
28	Семінар "Аналіз шкідливого програмного забезпечення" , Діти	Seminar "Malware-Analyse" , Kinder
29	Семінар "Основи доведення теорем за допомогою Coq" , Бланшетт, Лімперг	Seminar "Basics of Theorem Proving using Coq" , Blanchette, Limperg
30	Семінар «Функціональні перли» , Бланшетт, Майо	Seminar "Functional Pearls" , Blanchette, Maio
31	Семінар «Вибрані теми штучного інтелекту» , Хюллермаєр	Seminar "Ausgewählte Themen der Künstliche Intelligenz" , Hüllermeier
32	Проблемне навчання , Хойн	Problembasiertes Lernen , Heun
33	Проблемне навчання , Фрідель	Problembasiertes Lernen , Friedel
34	Проблемне навчання , кімната	Problembasiertes Lernen , Zimmer
35	Проблемне навчання , Фрішманн	Problembasiertes Lernen , Frischmann
36	Просемінар з медіаінформатики , Бутц, Піркер, Майєр, Шмідт	Proseminar Medieninformatik , Butz, Pirker, Mayer, Schmidt
	Імпорт з інших відділів для основного предмета	Import aus anderen Fachbereichen für das Hauptfach
37	Аналіз для комп'ютерних науковців , Філіп	Analysis für Informatiker , Philip
38	Лінійна алгебра для комп'ютерних науковців , Райхерт	Lineare Algebra für Informatiker , Reichert
39	Статистика I для медіаінформатики , Хенш	Statistik I für Medieninformatiker , Haensch
40	Семінар з бізнес-планування (соціальна та особистісна компетентність) , Центр підприємництва LMU	Seminar Geschäftsplanung (Soziale und Persönliche Kompetenz) , LMU Entrepreneurship Center
	Додаткові пропозиції	Zusatzangebote
41	Python для початківців , Jakobs, Gaida	Python für Anfänger , Jakobs, Gaida
42	Пропедевтична біоінформатика , Берхтольд	Propädeutikum Bioinformatik , Berchtold

ДОДАТОК М

Опис веборієнтованого застосунку «Інформаційна система «E-student»»

Перед початком роботи студент повинен пройти процедуру реєстрації в системі. Після введення необхідних даних на електронну пошту користувача надсилається лист для підтвердження реєстрації. Лише після підтвердження акаунту студент отримує доступ до навчальних матеріалів, тестів і власних результатів

Викладач у системі виконує розширену роль адміністратора-викладача. Він має можливість керувати користувачами, при необхідності додавати нових студентів, редагувати їх дані або видаляти акаунти. Викладач бачить акаунти своїх студентів, їхні результати навчання, оцінки та статистику успішності, крім того передбачено можливість за потреби працювати в ролі студента для перевірки коректності тестів або навчальних матеріалів

Після запуску з'являється головне вікно, яке представлено на рис. Н.1, у якому надається можливість вибору потрібного університету.

Після вибору навчального закладу у вікні викладач вибирає курс, наприклад третій, на якому буде створена нова дисципліна, наприклад «Веборієнтовані технології. Frontend розробка».

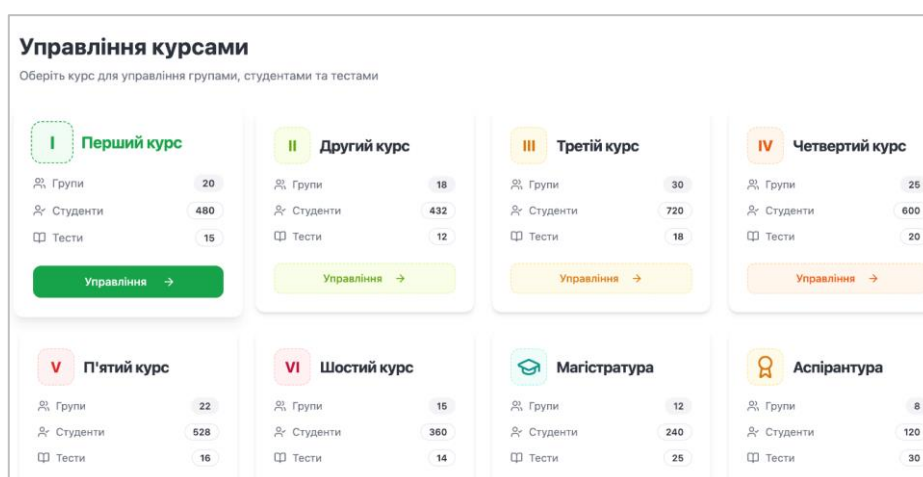


Рис.Н.1. Вікно вибору дисципліни в ІС «E-student»

Для створення нової дисципліни заповнюємо поля у вікні, яке представлено на рис. Н.2.

×

Створити нову дисципліну

Назва дисципліни *

WEB-орієнтовані технології. Frontend розробка

Опис дисципліни *

Запрошуємо на дисципліну Frontend! Навчившись створювати сучасні вебсторінки та інтерфейси: HTML, CSS, JavaScript, адаптивна верстка та базова практика на реальних прикладах. Підійде для старту в IT.

URL зображення

Введіть URL зображення (необов'язково)

Півріччя

Перше півріччя

Навчальний рік

2025-2026

Тип контролю


Залік

Скасувати

Створити дисципліну

Рис. Н 2. Вікно створення нової дисципліни у програмі «Е-student»

Результат створення дисципліни продемонстровано на рис. Н.3-Н.4.


E-STUDENT

Курси


Користувачі

Про викладача

Svitlana Proskura

Адміністратор

Назад до дисциплін



TS

jQuery

Завіт

Активна

WEB-орієнтовані технології. Frontend розробка

Запрошуємо на дисципліну Frontend! Навчисись створювати сучасні вебсторінки та інтерфейси: HTML, CSS, JavaScript, адаптивна верстка та базова практика на реальних прикладах. Підійде для старту в IT.

0

Студентів

0

Груп

0

Тестів

Розділи

Групи

Журнал

Тести

Рис. Н.3. Вікно створення нової дисципліни у програмі «Е-student»

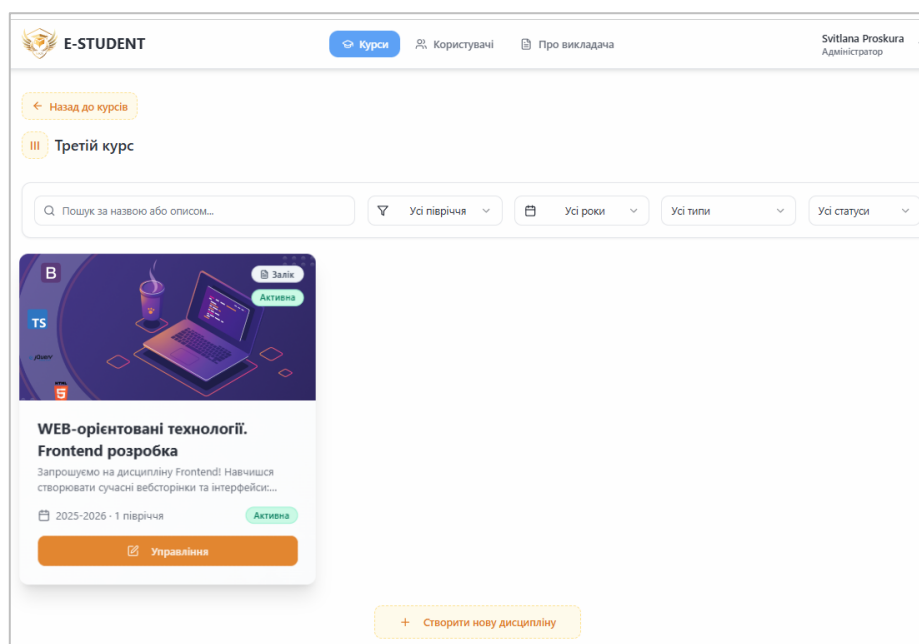


Рис.Н.4. Вікно переліку дисциплін 3 курсу у програмі «E-student»

Перелік всіх дисциплін вибраного курсу показано на рис.4.6. До кожної нової дисципліни викладач має можливість додати назви розділів (рис. 4.7). і відповідно у кожному розділі додати лекції (рис. 4.8), оголошення, протоколи виконання комп'ютерних практикумів(практичних), додаткові матеріали для ознайомлення, тести (рис. Н.5).

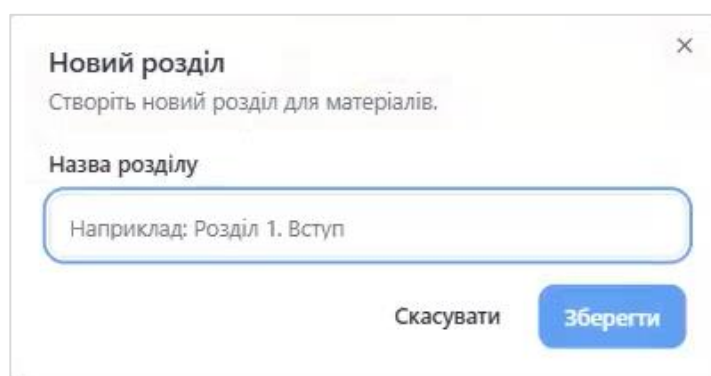


Рис. Н.5 Вікно створення розділу у програмі «E-student»

ДОДАТИ МАТЕРІАЛ

РОЗДІЛ 1 ОСНОВИ FRONTEND РОЗРОБКИ. ПОЗИЦІОНУВАННЯ ЕЛЕМЕНТІВ. АДАПТИВНІСТЬ WEB-ОРІЄНТОВАНОГО ЗАСТОСУНКУ. СИСТЕМА КОНТРОЛЯ ВЕРСІЙ

Лекція
Матеріали лекції + (опційно) відео до заняття.

Оголошення
Опис, важлива інформація та файли оголошення.

Практична
Завдання, інструкції або підбірка вправ.

Матеріал для ознайомлення
Додаткові статті, файли та джерела.

Тест
Тип тесту, максимальні бали та дедлайн.

Назва

Наприклад: Лекція №1 — HTML/CSS/JS

Короткий опис

Що буде на занятті або в матеріалі.

Файли лекції / презентація

Перетягніть файли або натисніть
PDF, DOCX або зображення (JPG, PNG, WEBP), сумарно до 20 МБ
Сумарний розмір: 0 Б / 20.0 МБ

Посилання на відео (опційно)

[https://youtube.com/...](https://youtube.com/)

Скасувати Зберегти

Рис. Н.6. Вікно створення лекції у програмі «Е-student»

ДОДАТИ МАТЕРІАЛ

РОЗДІЛ 1 ОСНОВИ FRONTEND РОЗРОБКИ. ПОЗИЦІОНУВАННЯ ЕЛЕМЕНТІВ. АДАПТИВНІСТЬ WEB-ОРІЄНТОВАНОГО ЗАСТОСУНКУ. СИСТЕМА КОНТРОЛЯ ВЕРСІЙ

Лекція
Матеріали лекції + (опційно) відео до заняття.

Оголошення
Опис, важлива інформація та файли оголошення.

Практична
Завдання, інструкції або підбірка вправ.

Матеріал для ознайомлення
Додаткові статті, файли та джерела.

Тест
Тип тесту, максимальні бали та дедлайн.

Назва

Асинхронність. Проміси

Короткий опис

Цей тест визначає рівень компетентності студентів за темою "Асинхронності. Проміси" з використанням таксономії Блума

Створити новий тест
Заповніть параметри тесту. Максимальна кількість балів завжди 100.

Дедлайн

17 лют. 2026 р., 01:09

Скасувати Зберегти

Рис. Н.7. Вікно створення теста у програмі «Е-student»

Створити тест Тест 100 балів

Назва * Курс

Дисципліна * Кінцевий термін *

Обмеження часу (хв) *

Статус

Опис *

Рис. Н.8. Вікно встановлення кінцевого терміну виконання тесту

На рис.Н.9. представлено вікно, у якому викладач має можливість ввести питання, вибрати тип питання, ввести максимальний бал, вибрати рівень за таксономією Блума, написати варіанти відповідей, обов'язково вказавши правильну відповідь. Кожен рівень оцінюється визначеною кількістю балів, яку задає сам викладач.

Питання Додати питання

ПИТАННЯ 1 Перетягнути Видалити

Як створити новий Promise?

Тип * Макс. бали * Рівень за таксономією Блума

Текст питання *

Варіанти відповідей * Додати варіант

let p = Promise();	Правильний	Видалити
let p let p = new Promise();= new Promise();	Правильний	Видалити
let p = new Promise((resolve, reject) => { /* ... */ });	Правильний	Видалити
let p = createPromise(resolve, reject);	Правильний	Видалити

Рис.Н.9. Вікно формування питання тесту

Студент, під час проходження тесту, має можливість обрати правильну відповідь, подивитись до якого рівня за таксономією Блума відноситься поточне питання і якою максимальною кількістю балів буде оцінено це питання (рис. Н.10).

Питання 6: Яка різниця між цими функціями? `function fetchData() { return fetch("url").then(res => res.json()); } async function fetchDataAsync() { const res = await fetch("url"); return res.json(); }` 6 балів

Застосування

- ☒ Поведінка однакова
- ☐ fetchDataAsync() повертає json, а fetchData()
- ☐ fetchData() швидше
- ☐ fetchDataAsync() не обробляє помилки

Питання 7: Яка помилка в наступному коді? `async function f() { await Promise.reject("Помилка"); console.log("Ніколи не буде виведено"); } f();` 8 балів

Аналіз

- ☐ Нічого не станеться
- ☐ "Помилка" буде викинута але не перехоплена
- ☐ Буде виведено "Ніколи не буде виведено"
- ☐ Код не працює, бо await не можна використовувати з reject

Рис.Н.10. Вікно проходження тесту

Результати проходження тесту студент у відсотках може побачити у вікні, які представлені на рис. Н.11; у вигляді кругової діаграми на рис. Н.12 і у вигляді лінійної діаграми на рис. Н.13. Також ці результати може побачити викладач у режимі адміністратора.

E-STUDENT ▶ Відеоуроки 📖 Тести 📄 Про викладача Svitlana Proskura
Адміністратор

Робота студента Назад

Інформація про спробу

Студент	Тест	Статус	Результат
Тарас Шевченко taras@edu.kpi.ua Група: IM-33	Асинхронність. Проміси 12 питань	Оцінено	96/100 96%

Подано: 12.02.2026, 19:31:49
Оцінено: 12.02.2026, 19:31:49

Рис. Н.11. Вікно результату проходження тесту студента

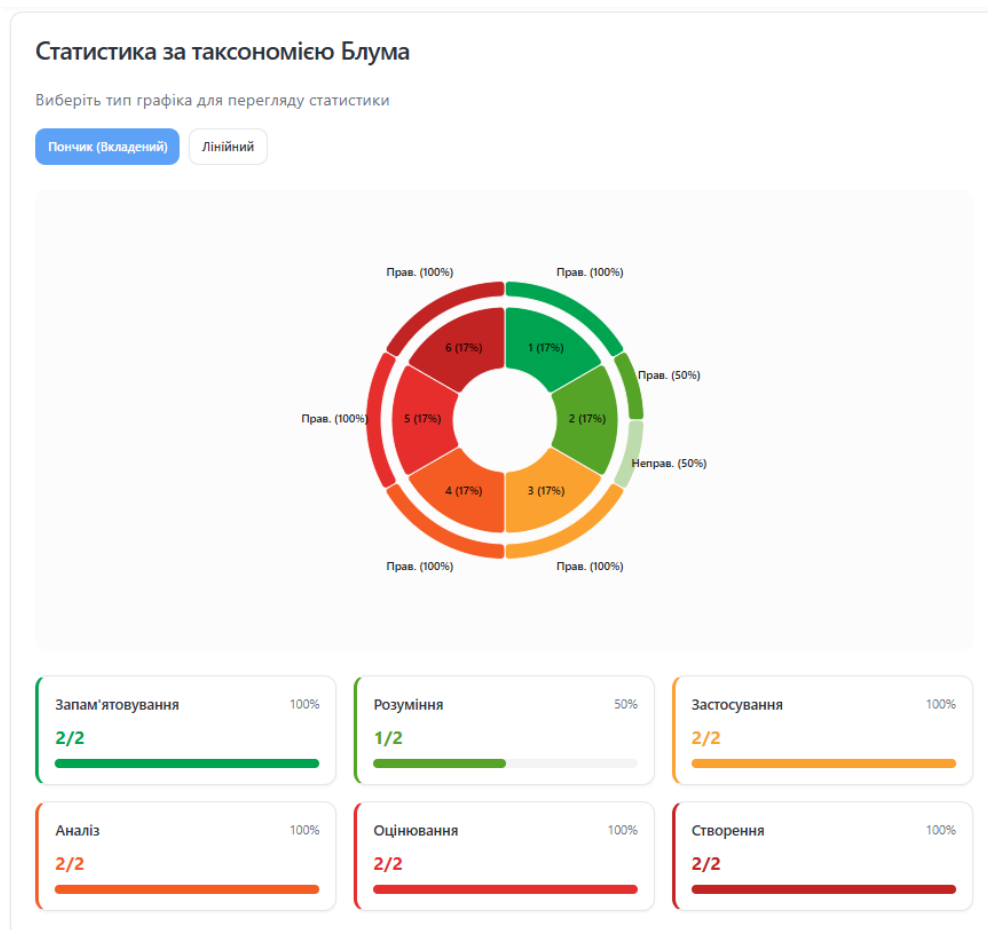


Рис. Н.12. Вікно результату проходження тесту у вигляді кругової діаграми

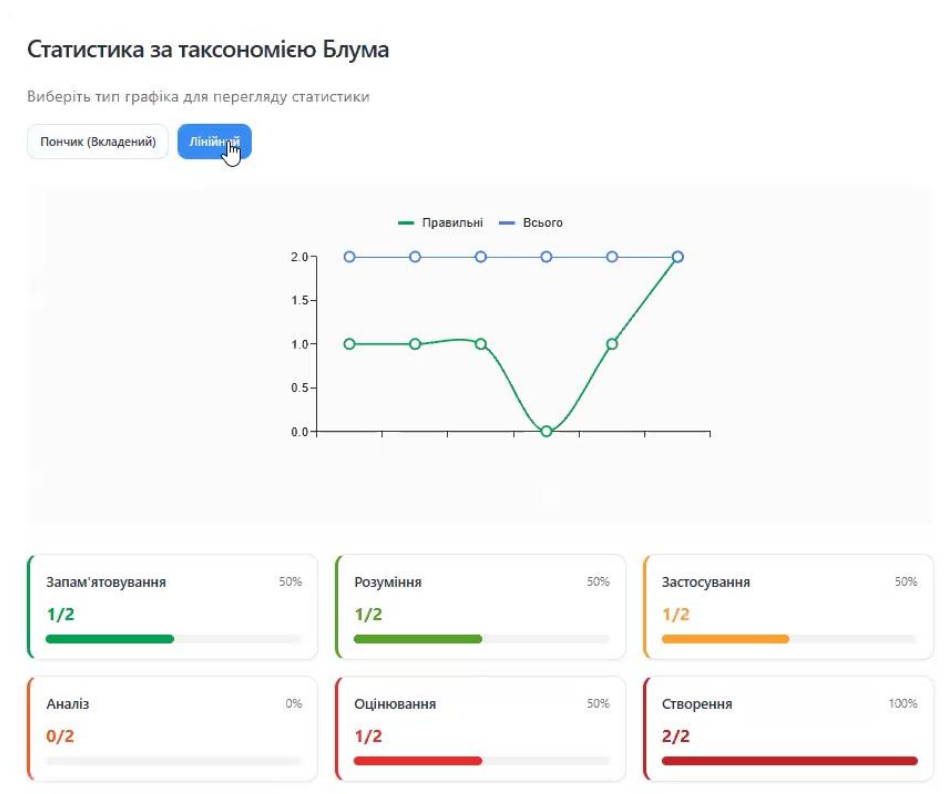


Рис. Н.13. Вікно результату проходження тесту у вигляді кругової діаграми

Також студент або викладач має можливість побачити на круговій діаграмі вікна «Статистика за таксономією Блума» більш детальну інформацію вибраного сектора рівня

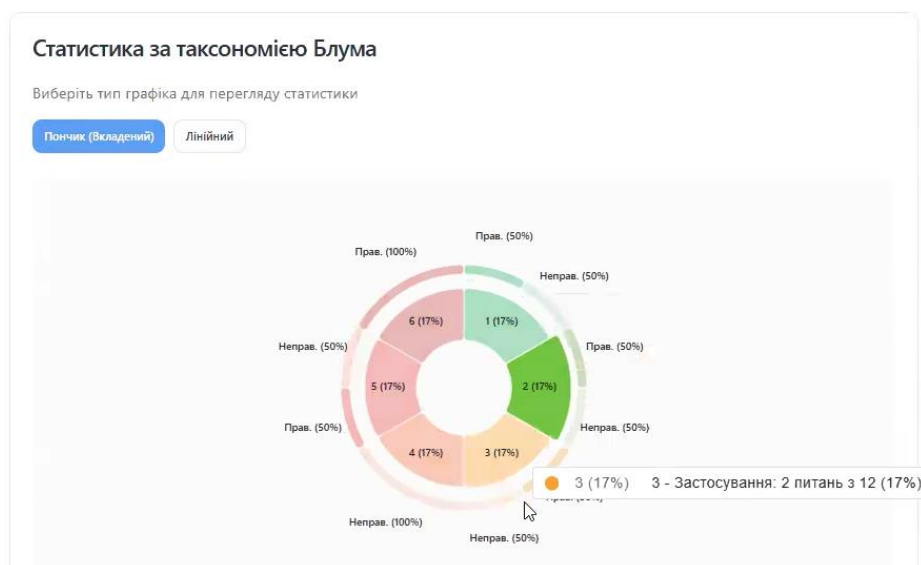


Рис.Н.14. Вікно результату проходження тесту у вигляді кругової діаграми

Після проходження тестування програма автоматично аналізує результати вона визначає рівень сформованості компетентностей студента за кожним рівнем таксономії Блума та обчислює підсумкову оцінку. Результати подаються як у розрізі когнітивних рівнів так і за Болонською системою. У разі незадовільного результату, система формує рекомендації щодо покращення знань, вказуючи які теми або рівні потребують додаткового опрацювання.

Програма передбачає швидкий пошук студента шляхом вибору перших літер прізвища та імені. На рис. Н.15 представлено вікно, яке демонструє цей вибір, наприклад вибір студента Тараса Шевченко, а також відразу виводиться повна інформація студента, а саме: назва тесту, кількість балів для кожного рівня, загальна кількість балів, назва літери за ECTS, рівень за таксономією Блума. Після проходження визначеної кількості тестів надається середнє значення вище перерахованих показників. У полі «Режим перегляду» вибрати рядок «за студентом»

E-STUDENT Курси Користувачі Про викладача Svitlana Proskura
Адміністратор

Фільтри журналу

Режим перегляду
За студентом

Фільтр за ім'ям
Всі А Б В Г Г' Д Е Є Ж З И І І' Й К Л М Н О П Р С **Т** У Ф Х Ц Ч
Ш Щ Ъ Ю Я

Фільтр за прізвищем
Всі А Б В Г Г' Д Е Є Ж З И І І' Й К Л М Н О П Р С Т У Ф Х Ц Ч
Ш Щ Ъ Ю Я

Тарас Шевченко
taras@educ.kpi.ua
Група: KH-11

Журнал студента

Email: taras@educ.kpi.ua Студент: Тарас Шевченко Група: KH-11

Тест	Рівні за таксономією Блума (бали)												Бали	ECTS	Рівень	Шкала 0-6						
	1		2		3		4		5		6					0	1	2	3	4	5	6
	бали	з	бали	з	бали	з	бали	з	бали	з	бали	з										
Тест №1: Асинхронність. Проміси	2	2	4	4	6	6	8	8	10	10	20	20	100	A	6							100
Всього													100	A	6							
Середній бал													100	A	6							

Результат: За семестр студент отримає 100 балів, що відповідає 6 рівню за таксономією Блума.

Рис. Н.15. Вікно журналу студента

Також у цьому вікні є можливість переглянути показники всієї групи одночасно, вказавши у полі «Режим перегляду» рядок «за текстом»

Фільтри журналу

Режим перегляду
За тестом

Тест
Асинхронність. Проміси

Фільтр за ім'ям
Всі А Б В Г Г' Д Е Є Ж З И І І' Й К Л М Н О П Р С Т У Ф Х Ц Ч
Ш Щ Ъ Ю Я

Фільтр за прізвищем
Всі А Б В Г Г' Д Е Є Ж З И І І' Й К Л М Н О П Р С Т У Ф Х Ц Ч
Ш Щ Ъ Ю Я

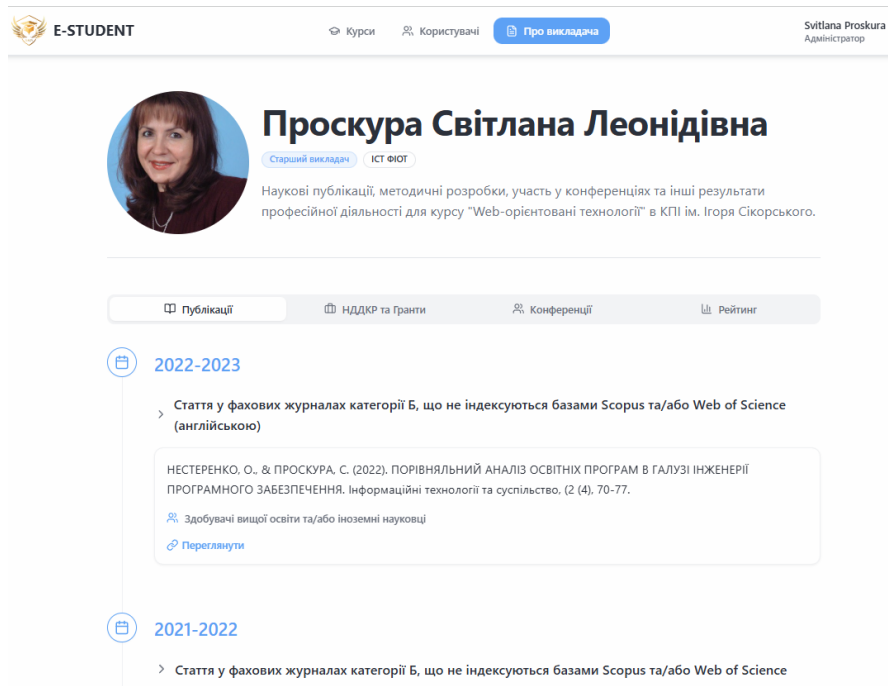
Журнал тесту

Тест: Асинхронність. Проміси Оцінюємо: 2 студентів

Студент	Рівні за таксономією Блума (бали)												Бали	ECTS	Рівень	Шкала 0-6						
	1		2		3		4		5		6					0	1	2	3	4	5	6
	бали	з	бали	з	бали	з	бали	з	бали	з	бали	з										
Петро Цибуля petro@educ.kpi.ua Група: KH-11	0	2	0	4	0	6	0	0	0	10	20	20	62	E	2			62				
Тарас Шевченко taras@educ.kpi.ua Група: KH-11	2	2	4	4	6	6	8	8	10	10	20	20	100	A	6							100

Рис. Н.16. Вікно журналу групи студентів

Програма також наводить дані про викладача (рис. Н.17)



E-STUDENT Курси Користувачі Про викладача Svitlana Proskura Адміністратор

Проскура Світлана Леонідівна
Старший викладач ІСТ ФІОТ

Наукові публікації, методичні розробки, участь у конференціях та інші результати професійної діяльності для курсу "Web-орієнтовані технології" в КПІ ім. Ігоря Сікорського.

Публікації НДДКР та гранти Конференції Рейтинг

2022-2023

> Стаття у фахових журналах категорії Б, що не індексуються базами Scopus та/або Web of Science (англійською)

НЕСТЕРЕНКО, О., & ПРОСКУРА, С. (2022). ПОРІВНЯЛЬНИЙ АНАЛІЗ ОСВІТНІХ ПРОГРАМ В ГАЛУЗІ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ. Інформаційні технології та суспільство, (2 (4), 70-77.

Здобувачі вищої освіти та/або іноземні науковці

Переглянути

2021-2022

> Стаття у фахових журналах категорії Б, що не індексуються базами Scopus та/або Web of Science

Рис. Н.17. Вікно даних про викладача

Особливістю програми є визначення рівня компетентності студента на основі результатів тестувань за таксономії Блума що забезпечує більш глибокий і об'єктивний аналіз навчальних досягнень порівняно з традиційними підходами до оцінювання.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Розділи монографії

1. Oleksandr Burov, Evgeniy Lavrov, Svitlana Lytvynova, Olha Pinchuk, **Svitlana Proskura**, Oleksii Tkachenko, Natalia Kovalenko, Yana Chybiriak & Yana Dolgikh. Cognitive and Perceptual Reliable Performance: Comparison of Psychophysiological Limitations. *HCI International 2025 Posters. Communications in Computer and Information Science. Cham: Springer, 2025. Vol. 2523. P. 3-13.* DOI: https://doi.org/10.1007/978-3-031-94153-5_1 (Scopus).

2. **Проскура С. Л.**, Литвинова С. Г., Кронда О. П. Засоби організації дистанційного навчання в період карантину 2020 року в закладах вищої освіти України. *Екстрене дистанційне навчання в Україні* : колективна монографія / за ред. В. М. Кухаренка, В. В. Бондаренка. Харків : КП "Міська друкарня", 2020. С. 299–313.

URL: https://duan.edu.ua/images/News/UA/Departments/Management/2020/monograph_ekstr_dyst_navch.pdf

Статті у наукометричних базах Scopus

1. Lytvynova S. H., Rashevskaya N. V., **Proskura S. L.** The use of artificial intelligence in teaching students programming languages *Proceedings of the IX International Workshop on Professional Retraining and Life-Long Learning using ICT: Person-oriented Approach (3L-Person 2024), co-located with ICTERI 2024. CEUR Workshop Proceedings, 2024. Vol. 3781. P. 10–29.* URL: <https://ceur-ws.org/Vol-3781/paper01.pdf>

2. **Proskura S. L.**, Lytvynova S. H., Kronta O. P. Students' academic achievement assessment in higher education institutions. *ICTERI 2020: Proceedings of the 16th International Conference on ICT in Education, Research and Industrial Applications. CEUR Workshop Proceedings, 2020. Vol. 2732. P. 734–746* DOI: URL: <http://ceur-ws.org/Vol-2732/20200734.pdf>

3. **Proskura S. L.**, Lytvynova S. H., Kronda O. P. Demeshkant N. Mobile learning approach as a supplementary approach in the organization of the studying process in educational institutions. *ICTERI 2020: Proceedings of the 16th International Conference on ICT in Education, Research and Industrial Applications. CEUR Workshop Proceedings*, 2020. P. 650–664. URL: <http://ceur-ws.org/Vol-2732/20200650.pdf>
4. **Proskura S. L.**, Lytvynova S. H. The approaches to Web-based education of computer science bachelors in higher education institutions. *Cloud technologies in education. CTE-2019. CEUR Workshop Proceedings*, 2020. 2643. P.609-625. .URL: <https://ceur-ws.org/Vol-2643/paper36.pdf>
5. **Proskura S. L.**, Lytvynova S. H. Organization of independent studying of future bachelors in computer science within higher education institutions of Ukraine. *ICTERI 2018 (3L-Person 2018). CEUR Workshop Proceedings*, 2018. P. 348–358. URL: http://ceur-ws.org/Vol-2104/paper_160.pdf

Статті у фахових наукових виданнях України

1. **Proskura S. L.**, Lytvynova S. H., Kronda O. P. The use of WEB-oriented technologies in the process of WEB-programming teaching for technical universities students. *Educational Dimension*, 2021. №5. URL: <https://lib.iitta.gov.ua/id/eprint/728607/1/Proskura-Lytvynova-Kronda-2021.pdf> DOI: <https://doi.org/10.31812/educdim.4724>
2. **Проскура С. Л.**, Литвинова С. Г. Формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*. 2019. № 2 (20). DOI: <https://doi.org/10.31110/2413-1571-2019-020-2-022> URL: https://fmo-journal.fizmatsspu.sumy.ua/journals/2019-v2-20/2019_2-20-Proskura_Lytvynova_FMO.pdf
3. **Проскура С. Л.** Модель формування професійної компетентності майбутніх бакалаврів комп'ютерних наук. *Фізико-математична освіта*, 2019. № 3 (21). DOI: <https://doi.org/10.31110/2413-1571-2019-021-3-016>

URL: https://fmo-journal.fizmatsspu.sumy.ua/journals/2019-v3-21/2019_3-21-Proskura_FMO.pdf

4. **Проскура С. Л.,** Литвинова С. Г. Підготовка фахівців з інформаційних технологій у закладах вищої освіти: стан, проблеми і перспективи. *Інформаційні технології в освіті*, 2018. № 2(35). Р. 72–88. DOI: <https://doi.org/10.14308/ite000668> (дата звернення: 27.03.2026).

5. **Проскура С. Л.** Застосування інтелект-карт для підвищення якості та ефективності навчання студентів курсу програмування вищих навчальних закладів. *Актуальні питання природничо-математичної освіти*, 2017. С. 220–228. URL: https://fizmat.sspu.edu.ua/images/NAUKA/APPMO/Arhiv/APPMO_7-8_2016_1c655.pdf

Матеріали апробаційного характеру

1. **Проскура С. Л.** Педагогічні умови використання WEB-орієнтованих технологій у підготовці бакалаврів з інформаційних технологій та систем. *Цифрова трансформація науково-освітніх середовищ в умовах воєнного* : збірник матеріалів. Звітна наукова конференція Інституту цифровізації освіти НАПН України, 23 лютого 2024 р., м. Київ / упоряд.: О. П. Пінчук, Н. В. Яськова. Київ : ІЦО НАПН України, 2024. С. 130–132. URL: https://lib.iitta.gov.ua/id/eprint/740554/1/Збірник_тез_звітної_2024_v2.pdf

2. **Проскура С. Л.,** Литвинова С. Г. Моніторинг використання WEB-орієнтованих технологій бакалаврами комп'ютерних наук в процесі вивчення програмування. *Інформаційно-цифровий освітній простір України: трансформаційні процеси і перспективи розвитку* : матеріали методологічного семінару НАПН України, 4 квітня 2019 р.. Київ : НАПН України, 2019. С. 211–219. URL: <https://lib.iitta.gov.ua/id/eprint/717123/>

3. **Проскура С. Л.,** Таксономія Блума в оцінюванні результатів освітньої діяльності студентів. Звітна наукова конференція Інституту інформаційних технологій і засобів навчання НАПН України. 2020. С.83-89.

URL: <https://lib.iitta.gov.ua/id/eprint/720537/1/Збірник%20тез%20звітної%20конференції%20ІТЗН%20НАПН%20України%202020.pdf>

4. Kronda O. P., **Proskura S. L.** The level of digital technologies use in higher education institutions in the conditions of distance and blended learning. *Наукова молодь-2020* : матеріали VIII Всеукр. наук.-практ. конф., м.Київ, 2020. С. 129-131

С.URL: https://lib.iitta.gov.ua/id/eprint/722327/1/ЗБІРНИК%20НАУКОВА%20МОЛОДЬ%202020_1.pdf

5. **Проскура С. Л.** Використання Web-орієнтованих технологій в закладах вищої освіти. *Наукова молодь. 2019*: матеріали VII Всеукраїнської науково-практичної конференції. 2019. С.39-42

URL: https://lib.iitta.gov.ua/id/eprint/718530/2/Збірник%20Наукова%20молодь%202019.pdf?utm_source

6. **Проскура С. Л.,** Литвинова С. Г., Кронда О. П. Оцінювання рівня знань бакалаврів комп'ютерних наук у закладах вищої освіти. *Сучасні тенденції та фактори розвитку педагогічних та психологічних наук в Україні та країнах ЄС* : матеріали міжнародної наук.-практ. конф., Люблін. Польща, 2020. С.250-254.

URL: <http://www.baltijapublishing.lv/omp/index.php/bp/catalog/download/68/1534/3505-1>

7. **Проскура С. Л.,** Особливості організації змішаного навчання майбутніх бакалаврів комп'ютерних наук у закладах вищої освіти. *Звітна наукова конференція Інституту інформаційних технологій і засобів навчання НАПН України.* 2019. С 133 – 137.

URL: <https://lib.iitta.gov.ua/id/eprint/715956/1/Збірник%20тез%20звітної%202019.pdf>

8. **Проскура С. Л.,** Литвинова С. Г. Особливості підготовки майбутніх бакалаврів комп'ютерних наук в університетах США. *Наукова молодь-2018* : матеріали VI Всеукраїнської наук.-практ. конф. молодих учених. Інститут

інформаційних технологій і засобів навчання, м. Київ, Україна, 16 листоп. 2018.
С. 21-24 URL: https://lib.iitta.gov.ua/id/eprint/717127/?utm_source

9. **Проскура С. Л.,** Литвинова С. Г. Огляд компетентностей майбутніх бакалаврів комп'ютерних наук. *Звітна наукова конференція Інституту інформаційних технологій і засобів навчання НАПН України: збірник матеріалів наукової конференції.* 2018. С.32-34.
URL: <http://lib.iitta.gov.ua/711730/1/Збірник%20тез%20звітна%202018-output.pdf>

10. **Проскура С. Л.** Розвиток радіантного мислення студентів вищих навчальних закладів. *Концептуальні шляхи розвитку науки* : матеріали міжнародної науково-методичної конференції, 2017. С.65-66.
URL: https://lib.iitta.gov.ua/id/eprint/712031/1/proskura_rad_mysl.pdf?utm_source

11. **Проскура С. Л.** Сучасні підходи до викладання мов програмування в закладах вищої освіти. *Наукова молодь-2017* : матеріали V Всеукраїнської науково-практичної конференції. Київ, 2017. С. 313–315. URL: http://lib.iitta.gov.ua/709994/1/Збірник конф_Наукова молодь 2017.pdf

12. **Проскура С. Л.** Інтелект-карти як засіб організації самостійної роботи студентів-програмістів. *Розвиток інтелектуальних умінь і творчих здібностей учнів та студентів у процесі навчання дисциплін природничо-математичного циклу «ІТМ» плюс – 2017»* : матеріали II Міжнародної дистанційної науково-методичної конференції, 2017. м.Суми: у 2 ч. Ч.2 /упорядн. Чашечникова О. С. С. 38-39.
URL: https://lib.iitta.gov.ua/id/eprint/712029/1/Проскура_ІнтелектКартиОрганізаціяРоботиСтуд.pdf

Відомості про апробацію результатів дослідження.

Основні теоретичні й практичні результати, а також концептуальні ідеї та узагальнені висновки дисертаційного дослідження було оприлюднено у формі доповідей та тез на: міжнародних конференціях: 14-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Київ, 2018 р.); міжнародній науково-практичній конференції «Змішане навчання — інновація XXI сторіччя» (Харків, 2018 р.); 19-й міжнародній конференції «Workshop on Cloud Technologies in Education», CTE (Кривий Ріг, 2019 р.); 16-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Київ, 2020 р.); міжнародній науково-практичній конференції «Сучасні тенденції та фактори розвитку педагогічних та психологічних наук в Україні та країнах ЄС», (Люблін, Польща, 2020 р.); 17-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Херсон, 2021 р.); 19-й міжнародній конференції «ICT in Education, Research, and Industrial Applicat», ICTERI (Львів, 2024 р.); 14-й міжнародній науково-практичній конференції з інформаційних систем та технологій «Infocom Advanced Solutions 2026» (Київ, 2026 р.); всеукраїнських конференціях: Всеукраїнській науково-практичній конференції молодих вчених «Наукова молодь» (Київ, 2017–2020 рр.); IV Всеукраїнській науково-практичній конференції молодих вчених та студентів «Інформаційні системи та технології управління (Київ, 2020 р.); XV всеукраїнській науково-практичній конференції з міжнародною участю «Неперервна освіта: актуальні дискусії» (Ужгород, 2021 р.); звітній науковій конференції Інституту цифровізації освіти НАПН України, (Київ, 2018–2023 рр.); семінарах: міжнародному семінарі «Ключові питання шкільної освіти ЄС», у межах Модуля Жана Моне «Україна – ЄС: крос-культурні порівняння в освітніх дослідженнях» (Київ, 2018 р.); всеукраїнському методологічному семінарі «Інформаційно-комунікаційні технології в освіті та наукових дослідженнях» (Київ, 2018 р.); всеукраїнському науково-методологічному семінарі «Система

навчання і освіти в комп'ютерно орієнтованому середовищі» (Київ, 2018 -2026 рр.); всеукраїнському науково-практичному семінарі «Цифрова компетентність сучасного вчителя нової української школи» (Київ, 2018 р.); міжнародному семінарі «Вступ до обчислювального мислення», за участю професора Університету Дукесне Джозефа Куша (США, 2019 р.); міжнародному семінарі «Інтеграція штучного інтелекту в освіту – виклики та можливості» від провідних британських професорів (Київ, 2024 р.); у роботі оргкомітету по підготовці та проведенню XI Всеукраїнської олімпіади з інформатики та комп'ютерної техніки серед студентів закладів вищої освіти I-II рівнів акредитації (Ужгород, 2018 р.); під час професійного стажування в компанії EPAM «IT Ukraine Association Teacher's Internship 2024 held by EPAM» (Київ, 2024 р.).



УКРАЇНА

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
(КП ім. Ігоря Сікорського)

Факультет інформатики та обчислювальної техніки

03056, м. Київ, пр-т Берестейський, 37; тел. (044) 204 80 02 тел./факс (044) 204 94 55
<https://www.mmi.kpi.ua> e-mail: mmi@kpi.ua ЄДРПОУ 02070921

ДОВІДКА

**про впровадження результатів дисертаційного дослідження
Проскури Світлани Леонідівни
за темою «Використання веборієнтованих технологій у навчанні
програмування майбутніх бакалаврів комп'ютерних наук»**

Результати дисертаційного дослідження Проскури С.Л. впроваджено на кафедрі інформаційних систем та технологій факультету інформатики та обчислювальної техніки НТУУ «Київський політехнічний інститут ім. Ігоря Сікорського».

Аспіранткою виконано дисертаційне дослідження, присвячене використанню веборієнтованих технологій у навчанні програмування бакалаврів галузі знань F «Інформаційні технології».

У межах дослідження розроблено та впроваджено методику застосування веборієнтованих технологій у навчанні програмування бакалаврів, яка реалізовувалася під час вивчення фахових дисциплін з програмування. Методика передбачає комплексне використання сучасних веборієнтованих засобів навчання, інтерактивних онлайн-платформ, систем управління навчанням, вебсервісів для організації практичної діяльності студентів та автоматизованого оцінювання результатів навчання.

Результати дисертаційного дослідження використанні в освітньому процесі під час підготовки бакалаврів, зокрема при проєктуванні та реалізації навчальних курсів з програмування, організації практичної та самостійної роботи студентів, а також здійснення поточного та підсумкового контролю навчальних досягнень.

Результати упровадження підтверджують статистично значущу позитивну динаміку рівнів сформованості компетентності з програмування у студентів експериментальних груп порівняно з контрольними, зростання якості навчальних досягнень та доцільність використання веборієнтованих

технологій як ефективного засобу підвищення результативності фахової підготовки майбутніх бакалаврів галузі знань F «Інформаційні технології».

Отримані результати можуть бути рекомендовані для використання в освітньому процесі закладів вищої освіти, що здійснюють підготовку фахівців у галузі інформаційних технологій.

Результати апробації дисертаційного дослідження Проскури С.Л. були обговорені на засіданні кафедри інформаційних систем та технологій факультету інформатики та обчислювальної техніки (протокол № 14 від « 18» березня 2026 року).

Декан факультету
інформатики та обчислювальної техніки
КПІ ім. Ігоря Сікорського
доктор технічних наук, професор



Ярослав КОРНАГА

Завідувач кафедри
інформаційних систем та технологій
КПІ ім. Ігоря Сікорського
доктор технічних наук, професор

Олександр РОЛІК

**МІЖНАРОДНИЙ ЄВРОПЕЙСЬКИЙ
УНІВЕРСИТЕТ**

проспект Академіка Глушкова, буд.42, м. Київ, Україна – 03187
+ 38 (044) 338 00 42; +38 (044) 338 00 43
rector@ieu.edu.ua; www.ieu.edu.ua


**INTERNATIONAL EUROPEAN
UNIVERSITY**

42 Academician Glushkov Avenue, Kyiv, Ukraine – 03187
+ 38 (044) 338 00 42; +38 (044) 338 00 43
rector@ieu.edu.ua; www.ieu.edu.ua

~ 129-03/26
big 20.03.26

ДОВІДКА
про впровадження результатів дисертаційного дослідження
Проскури Світлани Леонідівни
на тему «Використання веборієнтованих технологій у навчанні
програмування майбутніх бакалаврів комп'ютерних наук»

Актуальність дослідження.

Актуальність дисертаційного дослідження зумовлена сучасними тенденціями цифровізації вищої освіти, стрімким розвитком веборієнтованих технологій та зростанням вимог до рівня професійної підготовки майбутніх бакалаврів комп'ютерних наук. Умови дистанційного та змішаного навчання, необхідність забезпечення безперервності освітнього процесу й орієнтація на компетентнісний підхід актуалізують потребу в удосконаленні методик навчання програмування з використанням сучасних веборієнтованих засобів. У цьому контексті впровадження результатів дисертаційного дослідження Проскури С.Л. відповідає актуальним потребам закладів вищої освіти та спрямоване на підвищення якості підготовки фахівців у галузі інформаційних технологій.

Упродовж 2025–2026 навчального року на кафедрі інформаційних технологій Європейської школи бізнесу Міжнародного європейського університету Проскурою Світланою Леонідівною було проведено дисертаційне дослідження, присвячене використанню веборієнтованих технологій у навчанні програмування майбутніх бакалаврів галузі знань F «Інформаційні технології».

У межах дослідження було розроблено та впроваджено **методику застосування веборієнтованих технологій у навчанні програмування**, яка реалізовувалася під час вивчення фахових дисциплін з програмування для здобувачів першого (бакалаврського) рівня вищої освіти. Методика передбачала інтеграцію веборієнтованих освітніх ресурсів, сервісів для спільної роботи, онлайн-платформ для виконання та перевірки практичних завдань, а також засобів формувального й підсумкового оцінювання навчальних досягнень студентів.

У межах дослідження розроблено та апробовано **методику використання веборієнтованих технологій у навчанні програмування**, яка базується на поєднанні традиційних і цифрових засобів навчання, інтерактивної взаємодії викладача і студентів, а також застосуванні сучасних онлайн-платформ для організації навчальної діяльності, виконання практичних завдань і здійснення контролю результатів навчання.

Наукова новизна результатів дослідження.

Наукова новизна дисертаційного дослідження полягає в тому, що: вперше розроблено та впроваджено **модель веборієнтованого навчання програмування бакалаврів**, яка поєднує навчальний контент, інтерактивну взаємодію та систему оцінювання в єдиному веборієнтованому освітньому середовищі; удосконалено **модель формування компетентності з програмування майбутніх бакалаврів комп'ютерних наук** шляхом системного використання веборієнтованих технологій; створено авторську **систему оцінювання навчальних досягнень бакалаврів з програмування**, відповідно до рівнів таксономії Блума, яка орієнтована на поетапне формування знань, умінь і навичок; розроблено та апробовано

У результаті впровадження встановлено, що запропонована методика забезпечує підвищення рівня навчальної мотивації здобувачів освіти, активізацію їхньої пізнавальної діяльності, розвиток алгоритмічного та критичного мислення, а також сприяє формуванню професійних компетентностей з програмування.

Використання запропонованих Проскурою С.Л. методичних і програмних розробок сприяло підвищенню якості навчання програмування, зростанню показників успішності студентів, удосконаленню організації освітнього процесу та розширенню можливостей застосування веборієнтованих технологій у підготовці майбутніх бакалаврів комп'ютерних наук.

Використання запропонованих Проскурою С. Л. методики, моделей і програмного засобу в освітньому процесі сприяло підвищенню ефективності навчання програмування, зростанню рівня навчальної мотивації студентів, активізації їхньої пізнавальної діяльності, формуванню стійких практичних навичок розв'язування програмних задач, розвитку алгоритмічного мислення, а також підвищенню рівня самостійності та відповідальності студентів за результати власного навчання.

Результати апробації дисертаційного дослідження Проскури Світлани Леонідівни були обговорені на засіданні кафедри на кафедрі інформаційних технологій Європейської школи бізнесу Міжнародного європейського університету (протокол № 8 від «27» _лютого_ 2026 року).

Завідувач кафедри
інформаційних технологій
Міжнародного європейського університету
доктор технічних наук, професор



Олександр НЕСТЕРЕНКО



ДОВІДКА

**про впровадження результатів дисертаційного дослідження
Проскури Світлани Леонідівни за темою «Використання веборієнтованих
технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук»**

Комісія у складі:

Голова комісії: декан математичного факультету ЗНУ, доктор технічних наук, професор Гоменюк С.І.

Члени комісії: завідувач кафедри програмної інженерії, кандидат фізико-математичних наук, доцент Лісняк А.О.; професор кафедри програмної інженерії, кандидат фізико-математичних наук, доцент Кудін О.В.

засвідчує, що результати дисертаційного дослідження Проскури Світлани Леонідівни було впроваджено в освітній процес Запорізького національного університету на математичному факультеті, кафедрі комп'ютерних наук у межах підготовки здобувачів вищої освіти першого (бакалаврського) рівня спеціальності 122 «Комп'ютерні науки».

У процесі професійної підготовки майбутніх бакалаврів комп'ютерних наук було використано методику застосування веборієнтованих технологій у навчанні програмуванню, спрямовану на підвищення ефективності засвоєння навчального матеріалу, активізацію пізнавальної діяльності студентів та формування стійких професійних умінь і навичок у галузі програмування.

Також було впроваджено модель веборієнтованого навчання бакалаврів комп'ютерних наук, яка дозволила забезпечити інтеграцію традиційних і цифрових форм організації освітнього процесу, використання сучасних вебінструментів, навчальних платформ та онлайн-сервісів для підтримки навчальної та самостійної роботи студентів.

Одночасно застосовувалася модель формування компетентності з програмування майбутніх бакалаврів комп'ютерних наук у процесі використання веборієнтованих технологій, спрямована на поетапний розвиток професійних компетентностей, формування алгоритмічного та критичного мислення, здатності до розв'язання практичних і професійно орієнтованих завдань з програмування.

Зазначена модель містить систему оцінювання навчальних досягнень майбутніх бакалаврів комп'ютерних наук з програмування, у межах якої, за допомогою авторського програмного засобу «E-Student», здійснювалося оцінювання виконання комп'ютерних практикумів за таксономією Блума.

Використання запропонованих Проскурою С.Л. методики, моделей і системи оцінювання сприяло підвищенню об'єктивності та прозорості контролю результатів навчання, зростанню навчальної мотивації студентів, розвитку їх самостійності, рефлексивних умінь, критичного й алгоритмічного мислення, а також покращенню якості виконання практичних завдань і засвоєння програмного матеріалу.

Узагальнення результатів упровадження продемонструвало педагогічну виваженість і високу ефективність запропонованого науково-методичного інструментарію в практиці закладів вищої освіти. Це переконливо засвідчує своєчасність, актуальність і вагомому практичну значущість дисертаційної роботи Проскури Світлани Леонідівни.

Результати апробації дисертаційного дослідження Проскури С.Л. були обговорені на засіданні кафедри програмної інженерії математичного факультету Запорізького національного університету (протокол № 9 від «10» березня 2026 року).

Голова комісії:

Сергій ГОМЕНЮК

Члени комісії:

Андрій ЛІСНЯК

Олексій КУДІН

ДОВІДКА
про впровадження результатів дисертаційного дослідження
ПРОСКУРИ Світлани Леонідівни
за темою «Використання веборієнтованих технологій у навчанні
програмування майбутніх бакалаврів комп'ютерних наук» на здобуття
наукового ступеня доктора філософії за спеціальністю 011 Освітні,
педагогічні науки, освітньо-науковою програмою: Інформаційно-
комунікаційні технології в освіті

Упродовж 2024–2025 навчального року на кафедрі програмних засобів факультету комп'ютерних наук і технологій Національного університету «Запорізька політехніка» здійснювалося впровадження результатів дисертаційного дослідження ПРОСКУРИ Світлани Леонідівни в освітній процес.

Впровадження результатів наукового дослідження відбувалось в межах дисциплін професійної підготовки майбутніх бакалаврів комп'ютерних наук галузі знань 12 «Інформаційні технології», спеціальності 122 «Комп'ютерні науки». Акцент був зроблений на дисциплінах, пов'язаних з програмуванням, зокрема вебпрограмуванням.

У процесі дослідно-експериментальної роботи були використані: **методика використання веборієнтованих технологій у навчанні програмування майбутніх бакалаврів комп'ютерних наук; модель формування компетентності з програмування бакалаврів комп'ютерних наук в умовах використання веборієнтованих технологій**, які забезпечили системну інтеграцію змісту навчання, форм, методів і засобів освітньої діяльності. Реалізація зазначеної моделі була спрямована на поетапне формування як загальних професійних компетентностей здобувачів вищої освіти, так і спеціальної компетентності з програмування, з урахуванням рівня підготовки студентів, їхніх індивідуальних освітніх потреб та особливостей навчальної діяльності в цифровому освітньому середовищі.

У межах реалізації зазначеної моделі та відповідної методики було здійснено **апробацію авторської системи оцінювання навчальних досягнень студентів з програмування, розробленої на основі таксономії Блума**, яка передбачає диференційоване оцінювання рівнів сформованості знань, умінь і навичок здобувачів освіти (знання, розуміння, застосування, аналіз, оцінювання, створення) із застосуванням веборієнтованих технологій. Такий підхід забезпечив узгодженість цілей навчання, змісту освітніх завдань і критеріїв оцінювання результатів навчальної діяльності студентів.

Запропоновані ПРОСКУРОЮ С.Л. методика, модель та система оцінювання сприяли не лише підвищенню ефективності навчання програмування бакалаврів з використанням веборієнтованих технологій, прозорості контролю результатів навчання, а й створенню умов для **індивідуалізації освітнього процесу, формуванню здатності студентів до самооцінювання та рефлексії власних навчальних досягнень**. Їх використання забезпечило зростання навчальної мотивації, активізацію пізнавальної діяльності, розвиток критичного й алгоритмічного мислення, а також підвищення рівня самостійності студентів у розв'язанні практичних і

професійно орієнтованих завдань з програмування, що позитивно позначилося на якості засвоєння програмного матеріалу.

Аналіз результатів апробації підтвердив **педагогічну доцільність та ефективність впровадження розроблених науково-методичних засобів в освітній процес закладів вищої освіти**, а також засвідчив їх відповідність сучасним вимогам підготовки фахівців у галузі інформаційних технологій, актуальність і практичну значущість дисертаційного дослідження ПРОСКУРИ Світлани Леонідівни.

Результати апробації моделі формування компетентності з програмування, методики використання веборієнтованих технологій у навчанні програмуванні майбутніх бакалаврів комп'ютерних наук авторської системи оцінювання студентів за таксономією Блума були обговорені на засіданні кафедри програмних засобів факультету комп'ютерних наук і технологій Національного університету «Запорізька політехніка» (протокол № 6 від «11» лютого 2026 року).

Завідувач кафедри програмних засобів
Національного університету
«Запорізька політехніка»,
доктор технічних наук, професор

 Сергій СУББОТІН

Підпис 
Засідуючий
Меч. 
