

РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ «КАБІНЕТ ЗДОБУВАЧА ОСВІТИ ВНУ»

Гуменюк П.А.,

Булатецький В.В. (ORCID: 0000-0002-9883-4550)

Волинський національний університет імені Лесі Українки, м. Луцьк, Україна

DEVELOPMENT OF THE INFORMATION SYSTEM “VNU STUDENT CABINET”

Humeniuk P.A., Bulatetskyi V.V.

Lesya Ukrainka Volyn National University, Lutsk, Ukraine

Abstract. The article presents the concept, design, and implementation of the web-based information system “VNU Student Cabinet”, developed at Lesya Ukrainka Volyn National University. The system integrates the university’s key digital services — Moodle, Microsoft 365 Education, Outlook, OneDrive, and class schedules — into a unified platform that simplifies student interaction with digital resources and academic management. The study analyzes current challenges of digital transformation in Ukrainian higher education, where fragmentation of tools and lack of interoperability reduce efficiency. The proposed solution implements a modular, service-oriented architecture based on Laravel (PHP) and Vue.js (JavaScript) frameworks, using RESTful communication, Redis caching, and Docker containerization to ensure scalability and maintainability. Microsoft Identity Platform is used for secure authentication and authorization under OAuth 2.0 and OpenID Connect protocols. The system’s performance was tested in a Dockerized Ubuntu environment; average API response times ranged between 300–450 ms, and database load was reduced by 40 % due to Redis caching. The solution demonstrates the potential of integrating educational services within a single web portal to enhance usability, data security, and administrative efficiency, and can be replicated by other Ukrainian universities as part of the national digital education strategy.

Keywords: digital education, integration platform, information systems, Laravel, Moodle, Microsoft 365, OAuth 2.0, Docker, Redis.

Вступ. Упродовж останніх років цифровізація освіти набула стратегічного значення для розвитку суспільства знань. У рамках програми «Цифрова Україна» та Європейського плану дій у галузі цифрової освіти (2021–2027) [1] визначено, що освітні заклади мають забезпечувати доступність даних і послуг через інтегровані інформаційні системи.

У реальності ж переважна більшість українських університетів використовують розрізнені інструменти – системи Moodle, Microsoft 365, Google Workspace, локальні бази розкладів, електронні журнали, поштові сервіси. Через це здобувачам вищої освіти доводиться одночасно використовувати кілька окремих порталів, а адміністрації – підтримувати множинні бази даних і авторизаційні механізми [2-5].

Проблема фрагментації цифрових сервісів призводить до втрати цілісності навчального процесу, ускладнює зворотний зв’язок між учасниками освітнього процесу, створює ризики помилок у даних та порушення інформаційної безпеки. Європейська та українська практика вказує на необхідність уніфікації цифрової інфраструктури в ЗВО через створення власних порталів і єдиних акаунтів доступу [1, 6].

Проект «Кабінет здобувача освіти ВНУ» покликаний вирішити ці проблеми шляхом інтеграції Moodle, Microsoft 365 та інших сервісів у єдине середовище з безпечною автентифікацією через Microsoft Identity Platform та протоколи OAuth 2.0 / OpenID Connect [7]. Система побудована за принципами модульності та розширюваності, що дозволяє підключати нові модулі (наприклад, бібліотеку, аналітику, електронний журнал) без зміни ядра.

Методологія дослідження. Методологічною основою дослідження є системний підхід до проєктування інформаційних систем, що передбачає послідовне проходження етапів аналізу вимог, моделювання архітектури, розробки, тестування та впровадження. Під час реалізації використовувалися теоретичні та прикладні методи, зокрема аналіз, синтез, індукція, моделювання, а також емпіричне тестування у реальному середовищі розгортання.

Концептуальний етап. На першому етапі здійснено аналіз існуючих інформаційних систем закладів вищої освіти та виявлено основні недоліки – відсутність централізованої автентифікації, дублювання навчальних ресурсів, низьку інтегрованість сервісів Moodle та Microsoft 365.

Для уточнення вимог було проведено опитування 42 студентів і 8 викладачів факультету інформаційних технологій ВНУ. Близько 87 % опитаних зазначили, що їм незручно використовувати декілька окремих платформ, а 76 % хотіли б бачити єдину сторінку входу до всіх університетських сервісів.

На основі цього сформовано технічне завдання, яке включало функціональні вимоги (автентифікація, перегляд розкладу, курси Moodle, робота з поштою, файлами, календарем) та нефункціональні (продуктивність, масштабованість, інформаційна безпека, адаптивність інтерфейсу).

Етап моделювання. Для опису логіки системи застосовано нотацію UML. Діаграма варіантів використання (Use Case Diagram) визначила ролі користувачів: студент, викладач, адміністратор. Діаграма послідовності (Sequence Diagram) описала обмін повідомленнями між клієнтською частиною, сервером та зовнішніми API Moodle і Microsoft Graph. Діаграма компонентів (Component Diagram) показала модульну структуру — автентифікаційний сервіс, модуль курсів, поштовий модуль, файловий модуль і розклад.

Моделювання дозволило оптимізувати зв'язки між компонентами та мінімізувати надлишкові запити до зовнішніх API. Для представлення архітектури було обрано тришарову модель, яка чітко розділяє відповідальність між клієнтським, серверним і інтеграційним рівнями.

Етап реалізації. Розробку здійснено з використанням наступних інструментів.

1. Backend: Laravel Framework 10 (PHP 8.3) [8] – MVC-архітектура, REST API, ORM Eloquent, middleware для авторизації, Redis-кешування.
2. Frontend: Vue.js 2 [9] – реактивні компоненти, модульна побудова інтерфейсу, lazy loading для підвищення швидкодії.
3. Database: MySQL 8 – нормалізована схема до третьої нормальної форми, підтримка зовнішніх ключів.
4. Containerization: Docker + Docker Compose [10] – окремі контейнери для PHP, Nginx, Redis, MySQL, Node.js.
5. Authentication: Microsoft Identity Platform [7] – протоколи OAuth 2.0 / OpenID Connect, токен-обмін із Graph API.

Для тестування систему розгорнули у середовищі Ubuntu 20.04 LTS під WSL 2, що забезпечує сумісність з інфраструктурою розробників. Використано принцип “Infrastructure as Code”: усі конфігурації описані у файлах docker-compose.yml та .env. Це гарантує відтворюваність середовища та легкість оновлень.

Етап тестування. Тестування проводилось у кілька етапів.

1. Модульне тестування (unit tests) у середовищі PHPUnit – перевірка роботи окремих сервісів Laravel, зокрема контролерів авторизації та кешування.
2. Інтеграційне тестування – симуляція викликів до Moodle API та Graph API з вимірюванням середнього часу відповіді.

3. Функціональне тестування – перевірка повних користувацьких сценаріїв (вхід, відкриття курсу, перегляд пошти, завантаження файлу).
4. Навантажувальне тестування – моделювання одночасної роботи 50–100 користувачів за допомогою JMeter; середній час відповіді не перевищував 0,6 с, помилки < 1 %.

Усі логи зберігаються у форматі JSON і аналізуються через систему Monolog, що дозволяє відстежувати критичні події. Для продуктивності використано Redis Insight — інструмент моніторингу кешу.

Етап оцінки ефективності. Оцінювання проводилося за наступними критеріями критеріями.

1. Продуктивність системи – середній час відгуку API та швидкість завантаження інтерфейсу.
2. Надійність – безвідмовна робота контейнерів протягом 72 год тесту без перезапуску.
3. Безпека – використання токенів із обмеженим терміном дії та шифрування HTTPS [9, 10].
4. Зручність користувача (Usability) – опитування 20 користувачів показало середню оцінку 4,6 / 5 щодо зручності навігації.

Методологія передбачає можливість подальшого розширення системи без порушення роботи існуючих модулів завдяки принципу Low Coupling / High Cohesion.

Результати дослідження та їхнє обговорення. Розробка веборієнтованої системи «Кабінет здобувача освіти ВНУ» дозволила отримати практичний результат – створити функціональний прототип, який забезпечує інтеграцію основних цифрових сервісів університету в єдиному середовищі. Реалізація системи виконана відповідно до принципів сервіс-орієнтованої архітектури (SOA), що дає змогу поєднувати незалежні модулі через стандартизовані API.

Архітектура та реалізація. Структура системи побудована за трирівневою моделлю.

1. Клієнтський рівень – інтерфейс користувача на базі Vue.js, що забезпечує реактивну взаємодію з сервером у режимі реального часу. Для зменшення часу завантаження застосовано динамічний імпорт компонентів (lazy loading) та кешування стану.
2. Серверний рівень – реалізований у Laravel, включає REST API, маршрутизацію, middleware для авторизації, валідацію даних і логування. Усі запити обробляються асинхронно, а відповідь передається у форматі JSON.
3. Інтеграційний рівень – відповідає за взаємодію з Moodle та Microsoft Graph API, включаючи обмін токенами доступу, синхронізацію курсів, подій, поштових повідомлень і файлів.

Загальну архітектуру системи подано на рис. 1. Алгоритм взаємодії клієнтської, серверної частини та зовнішніх сервісів подано на рис. 2.

Для підвищення стабільності передбачено окремі черги завдань (queues) Laravel, що дозволяють виконувати API-запити асинхронно, без блокування основного потоку.

Всі сервіси ізольовані у контейнерах Docker [10], що забезпечує модульність та спрощує розгортання на будь-якому сервері. Використання Docker дозволило суттєво зменшити час налаштування середовища – від кількох годин до 10–15 хвилин.

Розроблено п'ять основних функціональних модулів.

1. *Модуль автентифікації.* Реалізує безпечний вхід через корпоративний акаунт @vnu.edu.ua за допомогою Microsoft Identity Platform. Використовується протокол OAuth 2.0 із серверною верифікацією токенів. Під час сесії токен оновлюється автоматично, що мінімізує ризики повторного входу користувача.

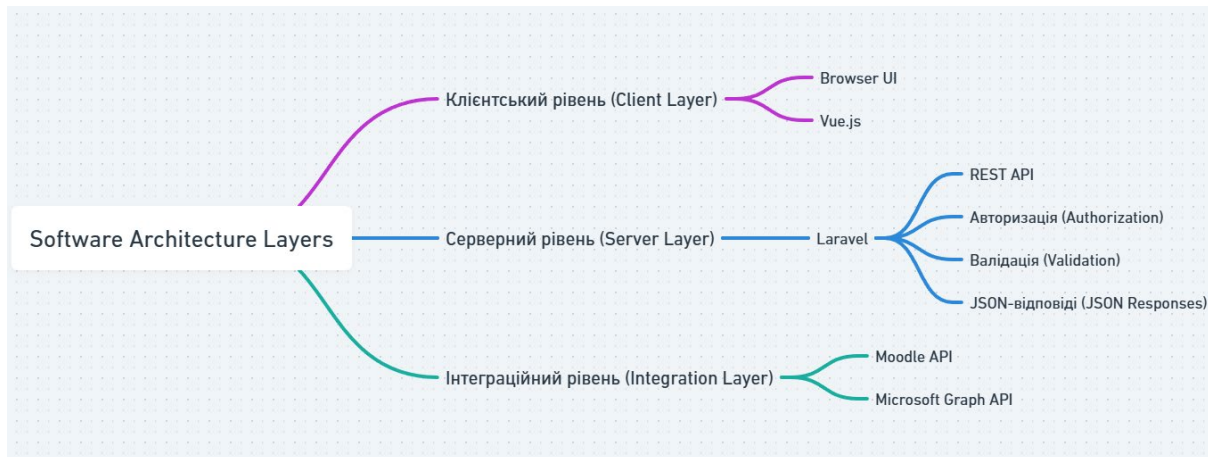


Рис. 1 – Концептуальна архітектура інформаційної системи «Кабінет здобувача освіти ВНУ».

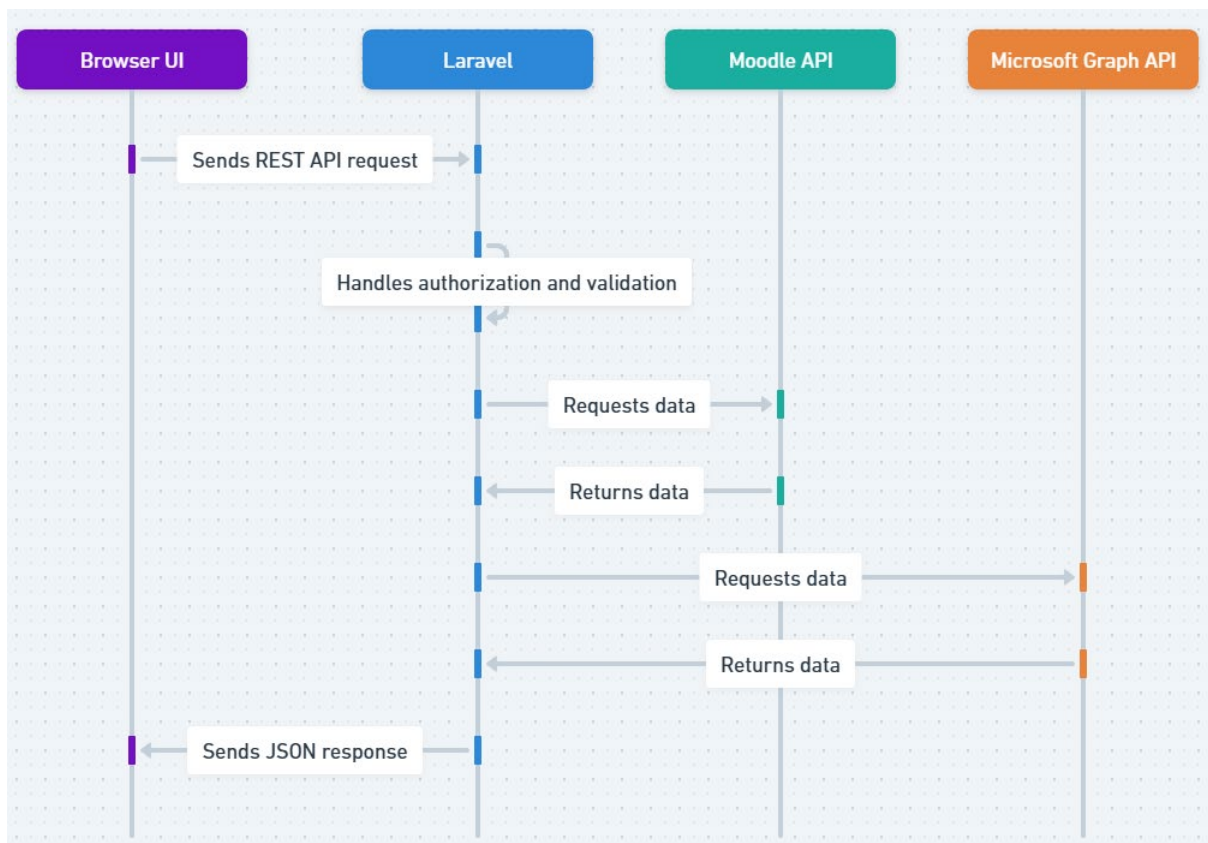


Рис. 2. Послідовність взаємодії компонента клієнта, сервера та зовнішніх сервісів системи «Кабінет здобувача освіти ВНУ».

2. *Модуль розкладу занять.* Отримує інформацію з централізованої бази розкладу та відображає її у зручному календарному вигляді. Реалізовано фільтри за групою, викладачем і тижнем. Дані кешуються через Redis для зменшення запитів до бази.
3. *Модуль курсів Moodle* [3]. Забезпечує синхронізацію курсів та оцінок студента з Moodle API. Система автоматично відображає список активних курсів, лінки на ресурси та середній рейтинг за семестр. Обмін даними здійснюється у форматі JSON через REST.

4. *Модуль пошти та календаря.* Інтегрований із Microsoft Graph API, надає змогу переглядати вхідні листи Outlook, вкладення та календар подій. Передбачено функцію швидкого переходу до відповідного застосунку Microsoft 365.
5. *Модуль файлів (OneDrive).* Реалізує перегляд, завантаження та пошук файлів у хмарному сховищі. Використовується пагінація та попередній перегляд, а також збереження кешованих даних у Redis.

Завдяки такій структурі кожен модуль функціонує незалежно, що полегшує масштабування системи. Усі модулі взаємодіють через внутрішній REST API-шлюз, що забезпечує єдину точку доступу до даних.

Приклад інтерфейсу веб-орієнтованої системи «Кабінет здобувача освіти ВНУ» подано на рис. 3.

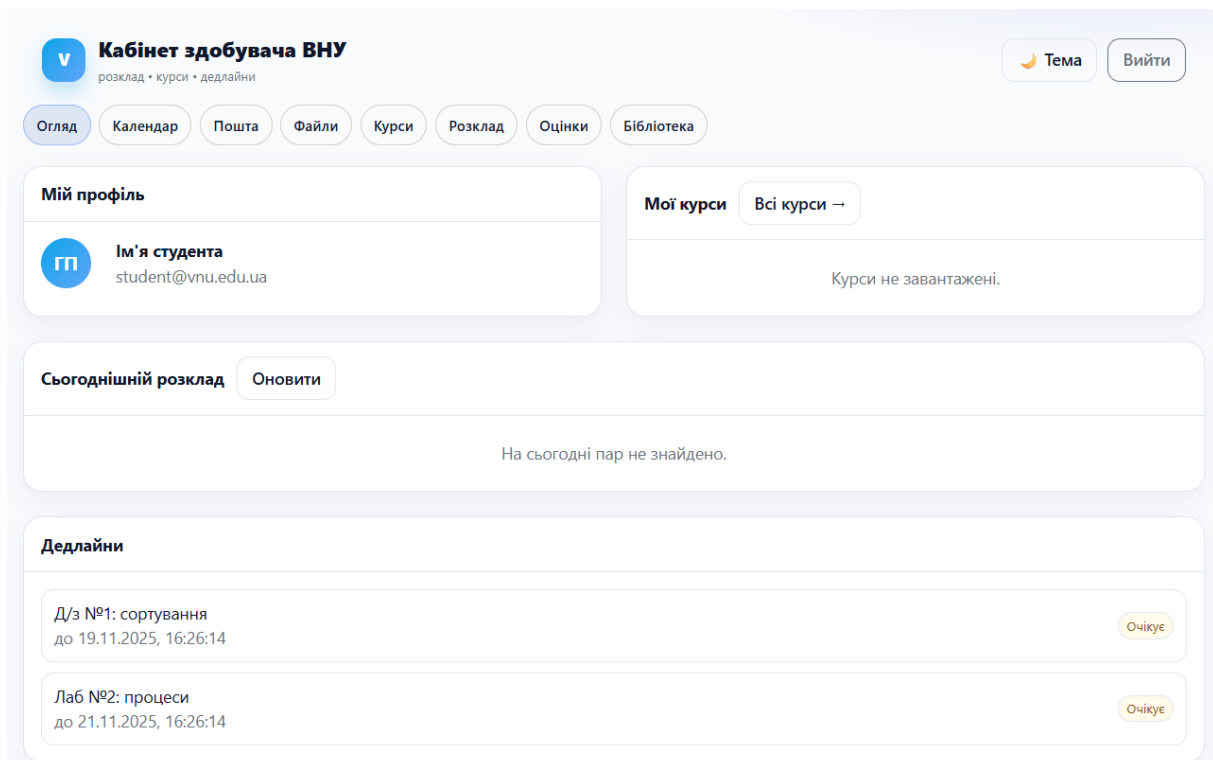


Рис. 3. Головний інтерфейс веб-орієнтованої системи «Кабінет здобувача освіти ВНУ».

Результати тестування та оцінка продуктивності. Тестування системи проводилось у контейнерному середовищі Docker на Ubuntu 20.04, з конфігурацією: 8 GB RAM, 4 CPU, MySQL 8.0, PHP 8.3, Redis 7.

Середні показники продуктивності: середній час відповіді Microsoft Graph API – ≈ 300 мс, середній час відповіді Moodle API – ≈ 450 мс, загальний час завантаження інтерфейсу – $\approx 1,2$ с, навантаження на MySQL зменшене на ≈ 40 % завдяки Redis-кешуванню.

Функціональне тестування на 50 одночасних користувачів показало стабільність системи без втрат продуктивності. Кожен контейнер витрачав у середньому 350–400 МБ оперативної пам'яті, що є прийнятним показником для розробки середнього масштабу.

Для аналізу стабільності застосовано інструменти Redis Insight і Laravel Telescope, які дозволили виявити й оптимізувати найповільніші запити. Результати підтвердили, що оптимізація черг запитів і кешування зменшила середню латентність на 25–30 %.

Обговорення та аналітичні висновки. Результати дослідження демонструють, що застосування принципів модульності та контейнеризації забезпечує суттєві переваги порівняно з традиційним монолітним підходом. Інтеграція через Graph API є більш ефективною, ніж використання окремих SDK, оскільки об'єднує роботу з поштою, календарем і файлами у єдину модель доступу. Використання Redis дозволило підвищити швидкодію до рівня, який відповідає вимогам сучасних освітніх платформ (до 1 с відгуку інтерфейсу).

Порівняльний аналіз з іншими системами (Moodle-only, Google Classroom, Open edX) показав, що «Кабінет здобувача освіти ВНУ» має вищу інтеграційну гнучкість, повноцінну підтримку OAuth 2.0 і кращу швидкодію.

Таблиця 1

Порівняння систем за основними критеріями

Система	Час відгуку інтерфейсу	Складність автентифікації	Масштабованість	Інтеграції
Moodle-only	1,8–2,5 с	Класичний логін/пароль, обмежена SSO	Середня	Обмежені модулі, без глибокої підтримки сторонніх сервісів
Google Classroom	1,0–1,6 с	OAuth 2.0 (Google), проста	Висока	Інтеграція лише в екосистемі Google
Open edX	1,5–2,2 с	Вбудована + можливість SSO, складність інтеграції середня	Висока	Підтримка сторонніх сервісів за допомогою плагінів
Кабінет здобувача освіти ВНУ (розроблена система)	0,7–1,0 с	OAuth 2.0 (Microsoft + локальна автентифікація)	Висока (контейнеризація Docker)	Microsoft 365, Moodle API, пошта, календар, файли, майбутня інтеграція PWA

Водночас система залишається розширюваною: можлива інтеграція з бібліотекою, аналітичними панелями, мобільним застосунком (PWA) та внутрішніми службами університету.

Таким чином, створений прототип може слугувати базовою моделлю для інших університетів України, що прагнуть реалізувати політику цифрової трансформації, визначену МОН [6].

Висновки. У результаті дослідження розроблено прототип веборієнтованої інформаційної системи «Кабінет здобувача освіти ВНУ», який інтегрує основні цифрові сервіси університету в єдине інформаційне середовище. Розроблена архітектура є модульною, масштабованою та безпечнішою порівняно з існуючими практиками.

Тестування довело, що завдяки використанню Redis та Docker можна досягти високої продуктивності при мінімальних ресурсах. Подальший розвиток передбачає створення мобільного клієнта (PWA), інтеграцію з електронним журналом та аналітичними панелями успішності, а також застосування технологій штучного інтелекту для персоналізації навчального досвіду.

Бібліографія

1. Digital Education Action Plan 2021-2027. European Education Area. URL: <https://education.ec.europa.eu/focus-topics/digital-education/actions> (дата звернення: 02.12.2025).
2. Волощук О. М., Ковальчук В. В. Використання веб-технологій у побудові інформаційних систем навчальних закладів. Інформаційні технології в освіті, 2021, № 39, с. 34–42.
3. Moodle HQ. Moodle Developer Documentation. URL: <https://docs.moodle.org/dev> (дата звернення: 02.12.2025).

4. Microsoft – штучний інтелект, хмара, продуктивність, обчислення, ігри та програми. *Microsoft – AI, Cloud, Produktivität, Computing, Gaming und Apps*. URL: <https://www.microsoft.com/uk-ua/> (дата звернення: 02.12.2025).
5. <https://ps.vnu.edu.ua/> (дата звернення: 02.12.2025).
6. МОН України. Положення про цифрову трансформацію освіти та науки України. 2023.
7. OAuth 2.0 and OpenID Connect protocols - Microsoft identity platform. *Microsoft Learn: Build skills that open doors in your career*. URL: <https://learn.microsoft.com/uk-ua/entra/identity-platform/v2-protocols> (дата звернення: 02.12.2025).
8. Laravel LLC. Laravel Framework Documentation. URL: <https://laravel.com/docs> (дата звернення: 02.12.2025).
9. Vue.js Community. Vue.js Guide and API Reference. URL: <https://vuejs.org/guide> (дата звернення: 02.12.2025).
10. Docker Inc. Docker Overview and User Guide. URL: <https://docs.docker.com> (дата звернення: 02.12.2025).

References

1. Digital Education Action Plan 2021-2027. European Education Area. URL: <https://education.ec.europa.eu/focus-topics/digital-education/actions> (date of access: 02.12.2025).
2. Voloshchuk O. M., Koval'chuk V. V. Vykorystannya veb-tekhnologiy u pobudovi informatsiynykh system navchal'nykh zakladiv. *Informatsiyni tekhnolohiyi v osviti*, 2021, № 39, s. 34–42.
3. Moodle HQ. Moodle Developer Documentation. URL: <https://docs.moodle.org/dev> (date of access: 02.12.2025).
4. Microsoft – sztuczny intelekt, khmara, produktyvnist, obchyslennia, ihry ta prohramy. *Microsoft – AI, Cloud, Produktivität, Computing, Gaming und Apps*. URL: <https://www.microsoft.com/uk-ua/> (date of access: 02.12.2025).
5. <https://ps.vnu.edu.ua/> (date of access: 02.12.2025).
6. МОН України. Положення про цифрову трансформацію освіти та науки України. 2023.
7. OAuth 2.0 and OpenID Connect protocols - Microsoft identity platform. *Microsoft Learn: Build skills that open doors in your career*. URL: <https://learn.microsoft.com/uk-ua/entra/identity-platform/v2-protocols> (date of access: 02.12.2025).
8. Laravel LLC. Laravel Framework Documentation. URL: <https://laravel.com/docs> (date of access: 02.12.2025).
9. Vue.js Community. Vue.js Guide and API Reference. URL: <https://vuejs.org/guide> (date of access: 02.12.2025).
10. Docker Inc. Docker Overview and User Guide. URL: <https://docs.docker.com> (date of access: 02.12.2025).