МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЛЕСІ УКРАЇНКИ
Кафедра комп'ютерних наук та кібербезпеки

На правах рукопису

ГУМЕНЮК ПАВЛО АНДРІЙОВИЧ

РОЗРОБКА СИСТЕМИ «КАБІНЕТ ЗДОБУВАЧА ОСВІТИ ВОЛИНСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ІМЕНІ ЛЕСІ УКРАЇНКИ»

Спеціальність: 122 Комп'ютерні науки
Освітньо-професійна програма: Комп'ютерні науки та інформаційні технології
Робота на здобуття освітнього ступеня «магістр»

Науковий керівник:
Булатецький Віталій Вікторович,
кандидат фізико-математичних наук, доцент
кафедри комп'ютерних наук та кібербезпеки

РЕКОМЕНДОВАНО ДО ЗАХИСТУ
Протокол №_____
засідання кафедри комп'ютерних наук
та кібербезпеки
від _____ 2025 р.
Завідувач кафедри

(_____)___Гришанович Т. О._____

ЛУЦЬК – 2025

ЗМІСТ

ВСТУП

Актуальність теми. Цифрова трансформація освіти є одним із ключових напрямів розвитку сучасних університетів. У світовій практиці дедалі більшого значення набувають інтегровані освітні платформи, що поєднують можливості онлайн-навчання, хмарних сервісів, електронної комунікації та персоналізованих інструментів підтримки студента. В Україні ці процеси особливо активізувалися після впровадження Концепції цифрової трансформації освіти і науки, де підкреслюється необхідність створення єдиних цифрових середовищ закладів вищої освіти. Проте на практиці студенти часто змушені користуватися різними розрізненими сервісами: Moodle, електронною поштою Microsoft 365, OneDrive, веброзкладами, окремими новинними ресурсами факультетів. Це ускладнює доступ до інформації, знижує ефективність навчального процесу та створює надмірне когнітивне навантаження.

У Волинському національному університеті імені Лесі Українки, як і в багатьох інших університетах, також відсутнє єдине інтегроване рішення, що дозволяє студенту отримувати централізований доступ до своїх курсів, завдань, розкладу занять, електронної пошти, хмарних файлів та офіційних оголошень. Натомість існує фрагментація цифрових ресурсів, яка ускладнює комунікацію та планування навчальної діяльності. Тому розробка єдиної системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», що забезпечує інтеграцію Moodle, Microsoft Graph API та університетських інформаційних сервісів, є актуальним науковим і прикладним завданням у межах цифровізації університету.

Актуальність теми підсилюється тенденцією переходу до змішаного та дистанційного навчання, необхідністю оптимізувати доступ до освітніх ресурсів, а також вимогами щодо захисту персональних даних та безпечного управління цифровими ідентичностями студентів. Реалізація такої системи сприятиме

підвищенню зручності роботи студентів із цифровими сервісами, покращенню їх академічної успішності та збільшенню ефективності внутрішніх процесів університету.

Мета роботи. Метою кваліфікаційної роботи є розроблення, теоретичне обґрунтування та практична реалізація інформаційної системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», яка забезпечує інтегрований доступ студента до навчальних ресурсів Moodle, сервісів Microsoft 365, розкладу занять та оголошень університету.

Завдання дослідження. Для досягнення поставленої мети необхідно вирішити такі завдання:

- виконати аналіз сучасних інформаційних систем у закладах вищої освіти та підходів до їх побудови;
- дослідити можливості інтеграції Moodle, Microsoft Graph API та інших цифрових сервісів університету;
- розробити архітектуру інформаційної системи та модель бази даних з урахуванням вимог безпеки та захисту персональних даних;
- реалізувати основні модулі системи: авторизацію через Microsoft 365, модуль курсів Moodle, календар дедлайнів, роботу з поштою, файлами OneDrive, розкладом та оголошеннями;
- обґрунтувати вибір інструментальних засобів розробки (Laravel, Vue.js, Docker);
- здійснити програмну реалізацію системи та описати етапи її розробки;
- провести тестування, оцінити продуктивність та стійкість системи до збоїв;
- виконати аналіз отриманих результатів та сформувати рекомендації щодо впровадження у цифрову інфраструктуру університету;

Об’єкт дослідження. Процеси управління навчальною інформацією та комунікаціями здобувачів освіти у цифровому середовищі університету.

Предмет дослідження. Методи, моделі, програмні рішення та інтеграційні механізми побудови інформаційної системи для централізованого доступу студентів до освітніх сервісів.

Наукова новизна. Наукова новизна роботи полягає в розробленні інтеграційного підходу до об’єднання різнорідних освітніх сервісів (Moodle, Microsoft 365, OneDrive, університетських розкладів) в єдине інформаційне середовище, а також у формуванні архітектури, що забезпечує мінімізацію дублювання даних та підвищену інформаційну безпеку. Запропоновано підхід до побудови єдиної платформи студента з використанням OAuth 2.0, Graph API та кешування даних зовнішніх систем.

Практичне застосування результатів. Результати роботи можуть бути впроваджені у Волинському національному університеті для забезпечення єдиної точки доступу студентів до всіх навчальних сервісів. Розроблена система підвищує ефективність роботи студентів, зменшує навантаження на адміністративні підрозділи та забезпечує сучасний рівень взаємодії з цифровими ресурсами. Результати можуть бути адаптовані іншими ЗВО України.

Апробація результатів. Основні положення та результати розробки були апробовані під час роботи над навчальними та науковими завданнями кафедри та доповідалися на засіданнях наукового гуртка. Окремі технічні рішення обговорювалися в рамках консультацій із викладачами кафедри.

РОЗДІЛ 1.
ТЕОРЕТИЧНІ ОСНОВИ ТА АНАЛІЗ ІСНУЮЧИХ ІНФОРМАЦІЙНИХ СИСТЕМ У ВИЩІЙ ОСВІТІ

1.1. Поняття та класифікація інформаційних систем у закладах вищої освіти

Інформаційна система (ІС) у найзагальнішому визначенні — це комплекс взаємопов'язаних компонентів (програмних, апаратних, організаційних та людських), що забезпечує збирання, зберігання, опрацювання, аналіз, передавання та представлення інформації користувачам для прийняття ефективних рішень і підтримки управлінських процесів [1; 2]. Її основною функцією є перетворення сирих даних у корисну інформацію, яка підвищує продуктивність діяльності організації.

У контексті закладів вищої освіти (ЗВО) інформаційні системи відіграють стратегічну роль, оскільки забезпечують інтеграцію освітніх, наукових та адміністративних процесів в єдине цифрове середовище. Вони підтримують такі функції, як управління навчальним процесом, електронний документообіг, облік контингенту студентів, моніторинг успішності, ведення бібліотечних фондів, контроль фінансових потоків, управління кадровим складом і комунікацію між усіма учасниками освітнього процесу [3; 4].

У сучасних університетах ІС є не просто допоміжним засобом, а невід'ємним елементом освітньої екосистеми, що визначає рівень інноваційності, конкурентоспроможності та цифрової зрілості закладу. Поняття «електронний університет» або «digital campus» означає не лише автоматизацію процесів, а й зміну філософії управління знаннями, орієнтовану на аналітику, відкритість і сервісність.

Згідно з підходом ISO/IEC 2382 та сучасними дослідженнями, ІС у сфері освіти можна класифікувати за кількома критеріями.

За функціональним призначенням:

- управлінські ІС (Management Information Systems) – забезпечують підтримку стратегічних і тактичних рішень адміністрації: електронний деканат, системи обліку студентів, управління навчальними планами, аналітичні панелі керівництва;
- навчальні ІС (Learning Management Systems, LMS) підтримують освітній процес: створення курсів, тестування, електронні журнали, рейтинги, форуми. Приклади: Moodle, Canvas, Blackboard, Open edX;
- науково-дослідні ІС – бази публікацій, репозитарії, системи оцінювання наукової активності, управління грантами;
- інформаційно-довідкові ІС – каталоги бібліотек, системи пошуку освітніх матеріалів, внутрішні портали;
- аналітичні та звітні системи (BI рішення) — агрегують великі обсяги даних і формують аналітичні дашборди щодо успішності, навантаження, KPI факультетів.

За масштабом використання:

- локальні системи – автоматизують роботу окремої кафедри або факультету (наприклад, облік лабораторних занять чи індивідуальних планів);
- корпоративні системи університетського рівня – інтегрують усі структурні підрозділи (електронний документообіг, розклад, фінанси, бібліотеку);
- розподілені системи – охоплюють кілька навчальних закладів чи філій і працюють у межах освітніх мереж або консорціумів (наприклад, платформи ERASMUS Dashboard, Ukrainian Open University Network).

За ступенем автоматизації та інтелектуалізації:

- інформаційно-довідкові — забезпечують доступ до даних без автоматичного аналізу;

- системи підтримки прийняття рішень (DSS) — пропонують рекомендації, ґрунтуючись на аналітичних моделях;
- експертні та інтелектуальні системи — використовують бази знань, нейронні мережі, машинне навчання для прогнозів успішності, персоналізації навчальних траєкторій і адаптації контенту під студента [5].

За архітектурою:

- класичні клієнт–серверні системи — характерні для ранніх локальних ІС;
- веборієнтовані системи — працюють через браузер, не потребують встановлення клієнтського ПЗ, забезпечують доступ із будь-якого пристрою;
- хмарні (cloud-based) — побудовані на сервісних моделях SaaS, PaaS або IaaS, що гарантують масштабованість, відмовостійкість і централізоване оновлення;
- інформаційні системи університетів виконують комплекс таких ключових функцій: освітня, що забезпечує управління курсами, оцінювання, дистанційне навчання; адміністративна, що включає кадровий, фінансовий, господарський облік; науково-аналітична, що підтримує дослідження, облік публікацій, індексування цитувань; комунікаційна, що забезпечує взаємодію між студентами, викладачами й адміністрацією; інтеграційна, що поєднує різні підсистеми університету в єдину екосистему.

Використання ІС у ЗВО дозволяє: зменшити навантаження на адміністративний персонал, мінімізувати ризик помилок і дублювання даних, підвищити прозорість та підзвітність процесів, покращити якість освітнього менеджменту на основі даних (data-driven education), забезпечити швидку комунікацію та зворотний зв’язок.

Більшість провідних університетів світу переходять до концепції інтегрованих освітніх платформ, у яких ІС різного типу об’єднані в єдине цифрове середовище. Так, Massachusetts Institute of Technology (MIT) та Stanford University використовують екосистеми на базі Open edX і Canvas з глибокою аналітикою

Learning Analytics. В Україні прикладами інтегрованих рішень є електронні кампуси КПІ, КНУ, ЛНУ, що поєднують модулі Moodle, Microsoft 365 та власні портали студентів.

Важливим напрямом є перехід до концепції «Digital University», у межах якої всі бізнес-процеси університету — від вступу абітурієнта до формування диплома – підтримуються єдиною ІС. У такій моделі дані рухаються без паперових носіїв, а користувач має персоналізований доступ через власний кабінет. Саме до цього підходу належить система «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», яка реалізує принципи цифрової трансформації університету, забезпечуючи централізовану роботу з розкладом, успішністю, Moodle-курсами, календарями Microsoft 365 та іншими сервісами.

Університетська інформаційна система повинна відповідати низці ключових вимог. Передусім вона має забезпечувати безпеку та захист персональних даних, відповідати міжнародним стандартам ISO/IEC 27001, GDPR і українському законодавству, а також підтримувати механізми резервного копіювання. Система повинна бути масштабованою та продуктивною, здатною обслуговувати тисячі користувачів одночасно без втрати стабільності. Не менш важливою є інтероперабельність, тобто можливість інтеграції з іншими платформами через REST API, GraphQL, LTI або xAPI. Значну роль відіграє зручність використання, адаптивний інтерфейс, підтримка мобільних пристроїв та інклюзивний доступ. Система також має бути відкритою і гнучкою, дозволяти розширення функціональності за допомогою модулів, плагінів та сторонніх інтеграцій. Окремим критерієм є наявність аналітичних інструментів для формування звітів, оцінки успішності та візуалізації даних. Крім того, ІС повинна бути надійною й відмовостійкою, включати інструменти автоматичного резервування, моніторингу та швидкого відновлення роботи після збоїв.

Таким чином, інформаційні системи у сфері вищої освіти становлять фундамент цифрової інфраструктури університету. Вони не лише автоматизують

рутинні операції, а й формують нову модель управління знаннями, у центрі якої — студент і якісні дані. Розроблення власних веборієнтованих рішень, таких як «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», є логічним кроком у напрямі створення єдиного освітнього простору, що відповідає сучасним вимогам цифровізації та забезпечує сталий розвиток університету в умовах інформаційного суспільства.

1.2. Огляд сучасних технологій веброзробки

Розробка веборієнтованих інформаційних систем є складним процесом, що об’єднує технології різних рівнів – клієнтського (frontend), серверного (backend) та рівня даних (database). Для створення ефективної системи необхідно забезпечити злагоджену взаємодію між цими рівнями, дотримання принципів модульності, масштабованості, безпеки та зручності користування.

У сучасних умовах вебтехнології розвиваються надзвичайно швидко, а їхній вибір визначається не лише продуктивністю, але й екосистемою інструментів, підтримкою спільноти, безпекою та можливістю інтеграції з іншими системами.

Клієнтська частина вебсистеми відповідає за візуальне представлення інформації, взаємодію з користувачем та миттєву реакцію інтерфейсу на його дії. Сучасні вебінтерфейси побудовані за принципами SPA (Single Page Application) або MPA (Multi-Page Application).

Основою будь-якого інтерфейсу є HTML5, CSS3 та JavaScript — базові технології вебу, які визначають структуру, стиль і поведінку сторінок. Сьогодні вони розширені стандартами ES6+, підтримкою модульності, асинхронності та реактивності.

Серед фреймворків для побудови динамічних користувацьких інтерфейсів найпоширенішими є Vue.js, React.js та Angular. Vue.js є легким і інтуїтивно зрозумілим фреймворком із гнучкою компонентною архітектурою, що робить його

ідеальним для середніх і великих освітніх проєктів. React.js, розроблений компанією Meta, орієнтований на створення масштабованих SPA-застосунків із використанням віртуального DOM, що забезпечує високу продуктивність. Angular від Google є потужним фреймворком для комплексних корпоративних систем, пропонуючи строгі шаблони та типізацію

Для стилізації інтерфейсу використовуються UI-бібліотеки та фреймворки: Bootstrap, Tailwind CSS, Vuetify, Element UI, які дозволяють швидко створювати адаптивні інтерфейси, зручні як для комп'ютерів, так і для мобільних пристроїв.

Важливою тенденцією останніх років є компонентний підхід у розробці. Це означає, що інтерфейс складається з незалежних модулів (компонентів), кожен з яких має власну логіку, шаблон і стилі. Такий підхід полегшує тестування, повторне використання коду та підтримку великих проєктів.

Ще однією сучасною практикою є використання PWA (Progressive Web Apps) – вебзастосунків, які працюють навіть без інтернету, мають push-сповіщення, кешування контенту та відображаються як звичайні мобільні додатки. Це особливо важливо для освітніх систем, що повинні бути доступними студентам із різним рівнем технічного забезпечення.

Серверна частина відповідає за бізнес-логіку системи, опрацювання запитів, управління базами даних, авторизацію користувачів, аналітику та інтеграцію з зовнішніми сервісами.

Однією з найпоширеніших технологій є PHP у поєднанні з фреймворком Laravel. Laravel забезпечує чітку архітектуру MVC (Model–View–Controller), що розділяє логіку, подання та дані, підвищує гнучкість і полегшує тестування [6]. До його переваг належать:

- наявність потужного ORM-механізму Eloquent для роботи з базами даних;
- система міграцій для керування схемою БД;
- підтримка middleware для контролю доступу;
- вбудовані засоби для кешування, черг, подій, нотифікацій;

- можливість побудови RESTful- або GraphQL-API.

Крім PHP, у розробці ІС для освіти часто використовують Python (Django, Flask), Node.js (Express, NestJS) або .NET Core. Python-фреймворки відомі простотою та потужними інструментами аналітики, Node.js забезпечує високу швидкодію завдяки неблокуючій моделі введення/виведення, а .NET має корпоративну стабільність і безпеку.

У проектах типу «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» Laravel є оптимальним вибором: він має розвинену екосистему, підтримує автентифікацію через OAuth 2.0 / OpenID Connect (Microsoft, Google), інтегрується з Redis і чергами, забезпечуючи швидку реакцію системи навіть під великим навантаженням.

Основою будь-якої інформаційної системи є база даних (БД), яка забезпечує надійне зберігання та доступ до інформації. Найпоширенішими СУБД залишаються MySQL та PostgreSQL — реляційні бази, що підтримують ACID-транзакції, складні запити, індекси, тригери, функції.

Паралельно зростає популярність NoSQL-рішень, таких як MongoDB, CouchDB, Firebase Realtime Database — для зберігання напівструктурованих даних (журнали, події, JSON-документи).

Redis виконує роль швидкого кешу або брокера повідомлень для обміну даними між компонентами системи (черги, WebSocket-події). Для великих обсягів аналітичних даних застосовують Data Warehouse-сховища, побудовані на базі ClickHouse, BigQuery чи Amazon Redshift.

У системах типу «Кабінет здобувача освіти» доцільно поєднувати MySQL (для основних таблиць — студенти, розклад, оцінки) з Redis (для кешування частих запитів і тимчасових сесій користувачів). Це забезпечує баланс між надійністю і швидкодією.

Сучасні вебсистеми використовують гнучкі архітектурні підходи. Мікросервісна архітектура – розподілення великої системи на незалежні модулі, які

взаємодіють через API. Кожен сервіс можна оновлювати або масштабувати окремо. API-орієнтована архітектура (API-first) – усі функції системи доступні через стандартизовані REST або GraphQL-інтерфейси, що спрощує інтеграцію з іншими платформами. Подійно-орієнтована архітектура (Event-Driven) – реакція на події в реальному часі, наприклад, через WebSocket або брокери повідомлень (Redis, RabbitMQ).

Для інтеграції з іншими освітніми платформами використовуються стандарти LTI (Learning Tools Interoperability), xAPI (Experience API) та SCORM, які дозволяють обмінюватися навчальними матеріалами й результатами між різними системами (наприклад, Moodle, Teams, Google Classroom).

API-інтеграції з Microsoft Graph, Google Workspace, Zoom або OpenAI API відкривають нові можливості: автоматичне створення груп, календарів, конференцій, сповіщень та навіть аналітики участі студентів.

Безпека є ключовим аспектом будь-якої ІС, особливо в освіті, де обробляються персональні дані студентів. Основні напрями забезпечення безпеки:

- шифрування трафіку (HTTPS, TLS 1.3);
- аутентифікація та авторизація (JWT, OAuth 2.0, OpenID Connect);
- захист від типових атак (SQL-ін’єкції, XSS, CSRF, brute force);
- Ссегментація доступів через ролі та політики RBAC/ABAC;
- журналювання дій користувачів і моніторинг підозрілої активності.

Laravel має вбудовані механізми безпеки, такі як CSRF-токени, фільтрація введення, валідація даних, обмеження кількості запитів (rate limiting) і двофакторна автентифікація. Це значно знижує ризики при розробці великих освітніх платформ.

Для стабільної роботи вебсистеми необхідно забезпечити процеси CI/CD (Continuous Integration / Continuous Deployment) — автоматичне тестування, збірку та розгортання додатку. Найчастіше застосовуються GitHub Actions, GitLab CI, Jenkins, які дозволяють мінімізувати людський фактор і швидко оновлювати систему без простоїв.

Завдяки контейнеризації (Docker) програмні компоненти ізольовані один від одного, що полегшує розгортання у різних середовищах — від локальної розробки до продакшн-серверів. Для складніших проєктів використовують Kubernetes, який автоматично масштабує сервіси залежно від навантаження.

Хмарні сервіси – AWS, Azure, Google Cloud, а також вітчизняні платформи (GigaCloud, DeNovo) — надають інструменти для побудови надійної, відмовостійкої інфраструктури з автоматичними резервними копіями, балансуванням навантаження та георозподілом. Для невеликих проєктів у ЗВО часто застосовують гібридну модель: критично важливі дані розміщуються локально, а ресурсоємні сервіси – у хмарі.

Серед сучасних технологій, що активно інтегруються в освітні платформи, виділяють використання штучного інтелекту для чат-ботів, рекомендаційних систем, розпізнавання тексту й голосу та автоматичного аналізу результатів навчання; serverless-обчислення, що дозволяють виконувати функції без постійних серверів (наприклад, AWS Lambda або Cloud Functions), знижуючи витрати; edge-комп’ютинг, який забезпечує обробку даних ближче до користувача для зменшення затримок; WebAssembly (WASM) для підвищення продуктивності вебдодатків шляхом виконання частини коду на низькому рівні; а також GraphQL як альтернативу REST API, що дає змогу клієнту отримувати лише необхідні дані та зменшує навантаження на сервер. Використання цих технологій відкриває нові горизонти для створення інноваційних освітніх платформ, здатних працювати швидко, безпечно та масштабовано.

Такі технології відкривають нові горизонти для створення інноваційних освітніх платформ, здатних працювати швидко, безпечно й масштабовано.

Сучасна веброзробка для освітніх інформаційних систем — це поєднання гнучких архітектурних підходів, високого рівня безпеки, автоматизації процесів розгортання та інтеграції з хмарними сервісами.

Використання стеку Laravel + Vue.js + MySQL + Redis + Docker у поєднанні з DevOps-практиками створює технічну основу для побудови масштабованої, безпечної та зручної системи типу «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки». Такий підхід дозволяє забезпечити сучасний рівень якості ПЗ, відповідність міжнародним стандартам і можливість подальшого розвитку системи без кардинальних змін її архітектури.

1.3. Порівняльний аналіз систем Moodle, Microsoft 365 та подібних освітніх рішень

Вибір платформи для підтримки освітнього процесу є одним із ключових стратегічних рішень у цифровій трансформації закладу вищої освіти. Різні інформаційні системи мають власні архітектурні підходи, моделі ліцензування, рівень гнучкості, аналітичні можливості та вимоги до технічних ресурсів. Для ефективного проєктування системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» доцільно провести порівняльний аналіз провідних освітніх платформ, які найчастіше використовуються у світовій та українській практиці.

Moodle (Modular Object-Oriented Dynamic Learning Environment) – одна з найпоширеніших систем управління навчанням (LMS) із відкритим вихідним кодом. Її основними перевагами є:

- безкоштовність і можливість повної локальної кастомізації;
- потужна система ролей (адміністратор, викладач, студент);
- підтримка стандартів SCORM, xAPI, LTI 1.3;
- розгалужена екосистема плагінів (понад 2000);
- підтримка аналітики, форумів, календарів, тестів, опитувань і групової роботи;
- велика міжнародна спільнота розробників [8].

З технічного боку Moodle реалізовано на PHP, підтримує MySQL/PostgreSQL, має модульну архітектуру та API-інтерфейси для інтеграції з іншими системами. Недоліками є дещо застарілий інтерфейс, складність адміністрування, потреба в ручній оптимізації продуктивності та обмежена масштабованість при великій кількості одночасних користувачів.

В українських університетах Moodle використовується найбільш широко: КПІ, КНУ ім. Шевченка, ЛНУ, ВНУ та багато інших мають власні інсталяції. Завдяки відкритості коду Moodle є хорошою базою для створення внутрішніх систем на кшталт «Кабінету здобувача освіти ВНУ».

Microsoft 365 Education — це не стільки LMS, скільки комплекс хмарних сервісів, спрямованих на колаборацію, комунікацію та спільну роботу над документами. Основними складовими є Teams, OneDrive, SharePoint, Outlook, Forms, Planner та Power BI.

Переваги:

- єдина автентифікація через Azure Active Directory (SSO);
- висока безпека (шифрування, MFA, відповідність ISO/IEC 27001);
- інтеграція з іншими Microsoft-сервісами (Graph API, Power Automate);
- зручна робота з документами у реальному часі;
- можливість створення команд для груп, потоків і курсів.

Недоліки:

- потреба у постійно стабільному інтернет-з’єднанні;
- залежність від ліцензійної політики;
- обмежена адаптація під локальні освітні стандарти.

Microsoft 365 часто застосовується не як окрема LMS, а як надбудова для комунікації й адміністрування. Наприклад, у University of Cambridge та University of Melbourne Teams використовується для онлайн-лекцій, групових чатів і календарів, тоді як основні курси розміщено в Moodle чи Canvas.

В Україні Microsoft 365 Education активно використовують КНУБА, СумДУ та ХНУРЕ, поєднуючи його з Moodle через плагіни Graph API.

Google Classroom — легка хмарна LMS, інтегрована з екосистемою Google Workspace (Drive, Meet, Docs, Sheets, Forms). Її сильні сторони:

- простота використання;
- безкоштовність для освітніх закладів;
- інтеграція з мобільними пристроями;
- автоматичне створення папок і спільного доступу до файлів.

Недоліки:

- обмежені інструменти аналітики та звітності;
- відсутність розвинених механізмів тестування й оцінювання;
- залежність від зовнішніх серверів Google.

Система ідеально підходить для шкіл або університетів, які лише починають цифровізацію, але не може замінити повноцінну LMS. У ЗВО Google Classroom часто виступає допоміжним інструментом для викладачів, тоді як управлінська частина реалізується іншими засобами.

Canvas LMS – сучасна платформа з відкритим API, побудована на Ruby on Rails. Вона відома зручним дизайном, глибокою інтеграцією з LTI-модулями, мобільними застосунками, аналітичними інструментами Learning Analytics та Gradebook. Її активно використовують Harvard University, MIT, Stanford і University of Auckland. Canvas є платною, але має безкоштовну open-версію для обмеженої кількості курсів.

Blackboard Learn – потужна комерційна система корпоративного рівня з глибокою інтеграцією, підтримкою SCORM 1.2 / 2004, модулів комунікації, оцінювання, аналітики. Вона орієнтована на великі університети й системи дистанційного навчання. Основний недолік — висока вартість і складність налаштування.

Open edX – платформа з відкритим кодом, створена MIT та Harvard для масових відкритих онлайн-курсів (MOOC). Вона підтримує масштабування до сотень тисяч користувачів, модулі сертифікації, систему обговорень та вбудовані аналітичні панелі. Водночас для самостійного розгортання потрібні значні ресурси й технічна компетентність.

Проведений аналіз свідчить, що кожна система має власну нішу та рівень спеціалізації. Moodle – оптимальна платформа для класичного навчального менеджменту, особливо коли потрібна повна контрольованість і локальне розгортання. Microsoft 365 Education – ідеальний інструмент для комунікації, спільної роботи й організації адміністративних процесів. Google Classroom – зручний для базових дистанційних курсів, але не забезпечує розвинену аналітику. Canvas LMS – сучасне рішення для університетів, орієнтованих на UX-досвід і масштабування. Open edX – підходить для відкритих онлайн-курсів, але потребує значних технічних ресурсів.

Для українських закладів освіти найдоцільнішою є гібридна стратегія: поєднання можливостей кількох платформ та розробка власного порталу, який забезпечує інтеграцію між ними. Такий підхід дозволяє:

- централізувати автентифікацію (SSO через Azure AD або Google OAuth);
- об’єднати розклад, оцінки, повідомлення, Moodle-курси, пошту та Teams у єдиному середовищі;
- реалізувати принцип «єдиного вікна» для студентів і викладачів;
- уникнути дублювання даних між окремими сервісами.

Найбільш перспективним підходом є створення інтегрованої університетської платформи, у якій власна вебсистема виступає координуючим центром, що об’єднує зовнішні сервіси, забезпечує єдиний профіль користувача, синхронізацію даних, централізовану аналітику та узгоджений дизайн.

Саме таку модель реалізує «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», який поєднує навчальний,

адміністративний і комунікаційний функціонал, інтегруючи можливості Moodle та Microsoft 365 у єдине цифрове середовище.

1.4. Сучасні тенденції розвитку інформаційних систем у вищій освіті та формування вимог до системи « Кабінет здобувача освіти Волинського національного університету імені Лесі Українки »

Стрімкий розвиток цифрових технологій у XXI столітті суттєво трансформує освітній процес, зумовлюючи появу нових вимог до структури, функціональності та ефективності інформаційних систем у закладах вищої освіти [9]. Традиційна модель університету, де інформаційні системи використовувалися переважно як допоміжний інструмент для автоматизації документообігу або обліку студентів, поступово поступається місцем концепції «цифрового університету» (Digital University) — комплексної, інтегрованої екосистеми, що охоплює всі аспекти освітньої, наукової й адміністративної діяльності.

Сучасні тенденції розвитку веборієнтованих інформаційних систем у вищій школі можна згрупувати за кількома ключовими напрямами: інтелектуалізація, інтеграція, персоналізація, відкритість, аналітичність і сталий розвиток.

Використання технологій штучного інтелекту (AI) і машинного навчання (ML) стало одним із найпомітніших трендів у розвитку сучасних освітніх систем. Алгоритми AI застосовуються для персоналізації навчання, адаптуючи складність завдань та обсяг матеріалів до індивідуального рівня студента; прогнозування академічної успішності шляхом раннього виявлення студентів із ризиком відрахування; автоматичного оцінювання тестів, есе, перекладів і навіть відеовідповідей; інтелектуального пошуку навчальних матеріалів на основі контексту; а також для розпізнавання голосу та тексту, включно з аналізом настрою, у системах відеоконференцій.

Приклади таких впроваджень можна спостерігати у платформах Coursera, Canvas LMS, Microsoft Learning Insights, де AI-модулі формують аналітику прогресу студентів. В українських університетах подібні елементи починають інтегруватися у власні LMS-рішення або через API Microsoft Azure Cognitive Services, що дозволяє автоматично аналізувати поведінку користувачів і підвищувати ефективність навчання [10].

Однією з найважливіших тенденцій є інтеграція різних освітніх сервісів у єдину інформаційну екосистему. Університети більше не розглядають свої інформаційні системи як окремі інструменти, а прагнуть об'єднати всі сервіси в єдине середовище з єдиною автентифікацією, календарем, базою даних та аналітичними звітами. Це стало можливим завдяки розвитку API-орієнтованих архітектур, які дозволяють різним підсистемам обмінюватися даними в реальному часі. Використання стандартів LTI (Learning Tools Interoperability), xAPI (Experience API), SCORM, GraphQL забезпечує взаємодію між Moodle, Microsoft 365, Google Classroom, бібліотечними та науковими платформами.

Прикладом інтегрованої екосистеми є University of Leeds (Велика Британія), де Moodle, Teams і Power BI з'єднані через API у спільне інформаційне середовище [11]. Подібна модель поступово впроваджується і в українських закладах вищої освіти — зокрема, КПІ ім. І. Сікорського реалізував інтеграцію Moodle із Microsoft 365 та внутрішнім порталом «Електронний кампус». Система «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» наслідує цю тенденцію, виконуючи роль інтеграційного ядра між Moodle, Microsoft 365, внутрішньою базою студентів і корпоративною поштою університету.

Одним із ключових напрямів розвитку інформаційних систем у вищій освіті є персоналізація навчання — створення індивідуальних освітніх траєкторій, адаптація контенту до темпу й стилю навчання студента. Вебсистеми нового покоління підтримують:

- адаптивний вибір контенту (Adaptive Learning);

- персональні рекомендації на основі поведінкових моделей;
- автоматичне формування розкладу, плану курсів і нагадувань;
- дашборди з персональними показниками успішності.

Персоналізація реалізується через аналітику поведінки користувача, історію переглядів, статистику активності, результати тестів тощо. У системах типу Moodle чи Canvas це досягається за допомогою Learning Analytics, у Microsoft Teams — через модуль Insights. Для системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» персоналізація є однією з базових функцій: кожен студент бачить лише власні курси, розклад, повідомлення, а інтерфейс адаптується під обліковий запис у Microsoft 365.

Глобальна тенденція до відкритої освіти (Open Education) призводить до зростання кількості відкритих освітніх ресурсів (OER, Open Educational Resources), доступних онлайн без суттєвих обмежень. Вебсистеми нового покоління активно підтримують інтеграцію з репозитаріями відкритих матеріалів, що сприяє академічній мобільності, обміну знаннями та реалізації концепції «освіта без кордонів». Міжнародні ініціативи, такі як OpenCourseWare, OER Commons, MITx, edX, стимулюють університети до створення відкритих курсів, які можна імпортувати у власні LMS. У ЄС це підтримується програмами Erasmus+ Digital Education Action Plan. В Україні схожі функції реалізуються через проєкти «Національний репозитарій академічних текстів» і «Відкрита наука». Таким чином, сучасні університетські інформаційні системи повинні не лише зберігати дані, а й підтримувати інтероперабельність — сумісність форматів і протоколів, що дозволяє обмінюватися навчальними матеріалами між системами різних розробників [12].

Ще однією важливою тенденцією є перехід до data-driven management — управління на основі даних. Сучасні інформаційні системи накопичують значні обсяги інформації: результати тестів, відвідуваність, активність студентів,

завантаження викладачів тощо. Обробка цих даних за допомогою технологій Business Intelligence (BI) і Learning Analytics дає змогу:

- виявляти закономірності у навчальній діяльності;
- оцінювати ефективність викладання;
- прогнозувати успішність студентів;
- оптимізувати навчальні програми.

BI-платформи (Power BI, Tableau, Metabase) дозволяють візуалізувати дані у вигляді інтерактивних дашбордів, доступних адміністрації університету. У деяких університетах (наприклад, University of Edinburgh) створено спеціальні Data Office, які відповідають за моніторинг освітніх показників у реальному часі. У межах Волинського національного університету аналогічний підхід може бути реалізований через інтеграцію даних з Moodle та Microsoft Graph у єдину аналітичну панель «Кабінету здобувача освіти».

Розвиток концепції Smart University є наступним етапом цифрової трансформації вищої освіти, який поєднує інформаційні технології, аналітику та Інтернет речей (IoT). Такі системи дозволяють автоматизувати фізичну інфраструктуру університету:

- контроль доступу за студентськими картками або мобільним додатком;
- автоматичне відстеження відвідуваності через Wi-Fi чи BLE-маячки;
- розумні аудиторії з керуванням освітленням, кліматом і мультимедійним обладнанням;
- системи бронювання лабораторій та ресурсів.

Наприклад, у National University of Singapore реалізовано систему Smart Campus із цифровими студентськими посвідченнями, а Seoul National University використовує IoT для енергоменеджменту. В Україні такі рішення лише починають з’являтися, проте інтеграція Smart-технологій із університетськими інформаційними системами є перспективним напрямом, який потенційно може

бути реалізований через окремі модулі системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки».

Паралельно з розвитком технологій зростає увага до питань кібербезпеки. Захист персональних даних студентів і викладачів, фінансових документів, журналів оцінок та електронних курсів стає критично важливим аспектом будь-якої інформаційної системи. Тому університети впроваджують політики Zero Trust Security, шифрування даних, багатофакторну автентифікацію, аудит доступів і постійний моніторинг мережевої активності. Відповідність стандартам ISO/IEC 27001, GDPR і Закону України «Про захист персональних даних» є необхідною умовою функціонування освітніх платформ. Паралельно формується напрям цифрової етики (Digital Ethics), що охоплює етичне використання даних студентів, алгоритмів оцінювання, систем спостереження та аналізу поведінки. Вебсистема «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» у своїй архітектурі враховує ці вимоги, зберігаючи прозорість і контроль користувачів над доступом до власних даних [13].

Окремою тенденцією є прагнення до екологічної цифровізації (Green IT). Використання хмарних рішень, віртуалізація серверів, автоматичне енергозбереження в дата-центрах і скорочення паперового документообігу сприяють зменшенню вуглецевого сліду університетів. Європейські університети інтегрують принципи сталого розвитку в IT-стратегії, оцінюючи вплив цифрових технологій на довкілля. Для України цей напрям лише формується, але поступово набуває актуальності.

Аналіз тенденцій розвитку веборієнтованих інформаційних систем у сфері вищої освіти свідчить, що вони еволюціонують від локальних автономних рішень до інтегрованих, аналітичних і інтелектуальних екосистем, заснованих на даних та відкритих стандартах. Ключовими трендами є:

- інтелектуалізація систем і застосування AI;
- інтеграція платформ через API та LTI;

- персоналізація освітнього процесу;
- розвиток відкритих ресурсів (OER);
- використання Learning Analytics і BI;
- впровадження концепції Smart University та IoT;
- підвищення кібербезпеки й розвиток цифрової етики.

Узагальнення результатів аналізу теоретичних і технологічних аспектів створення веборієнтованих інформаційних систем у закладах вищої освіти дає змогу зробити висновок, що інформаційні системи є ключовим елементом цифрової трансформації університетів, забезпечуючи автоматизацію освітніх, управлінських і наукових процесів, а також підвищуючи прозорість та ефективність управління. Сучасні системи мають ґрунтуватися на багаторівневій архітектурі з чітким розподілом клієнтської, серверної та базової частин. Найбільш доцільним технічним рішенням для реалізації таких систем є стек технологій на кшталт Laravel, Vue.js, MySQL, Redis і Docker, який поєднує продуктивність, безпеку, масштабованість і простоту інтеграції з іншими платформами.

Порівняльний аналіз провідних рішень (Moodle, Microsoft 365, Google Classroom, Canvas, Open edX) свідчить, що жодна окрема система не задовольняє повністю потреб університетів. Найкращим підходом є створення інтегрованої університетської платформи, у якій власна система виконує роль єдиного середовища доступу до навчальних курсів, календарів, електронної пошти та аналітики. Це підтверджує доцільність розроблення вебсистеми «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» як інтеграційного ядра університетської цифрової екосистеми.

На основі проведеного аналізу можна сформулювати основні вимоги до вебсистеми «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки».

До функціональних вимог належать:

- забезпечення єдиного входу (SSO) до основних освітніх сервісів університету на основі облікового запису Microsoft 365;
- інтеграція з LMS Moodle для отримання переліку курсів, завдань, оцінок та дедлайнів;
- інтеграція з Microsoft Graph API для доступу до електронної пошти, календаря, файлів OneDrive та команд у Microsoft Teams;
- відображення персоналізованої зведеної панелі студента з основними подіями, повідомленнями та показниками навчальної активності;
- підтримка фільтрації та сортування навчальної інформації (курси, завдання, події календаря);
- можливість підключення додаткових сервісів університету без суттєвої переробки архітектури;
- ведення журналу подій і дій користувачів для забезпечення аудиту та інформаційної безпеки.

До нефункціональних вимог належать:

- забезпечення конфіденційності та цілісності персональних даних відповідно до чинного законодавства України та міжнародних стандартів захисту інформації;
- висока доступність і масштабованість системи за рахунок використання сучасних вебтехнологій і контейнеризації;
- прийнятний час відгуку при виконанні інтеграційних запитів до Moodle та Microsoft Graph API;
- адаптивний вебінтерфейс, зручний для використання з персональних комп’ютерів, планшетів і смартфонів;
- можливість централізованого адміністрування, резервного копіювання та оновлення програмного забезпечення;
- відповідність принципам інтероперабельності — підтримка відкритих стандартів і протоколів обміну даними.

Таким чином, у першому розділі сформовано теоретичну основу та визначено вимоги до системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», що слугують підґрунтям для подальшого проєктування архітектури, моделі даних та реалізації основних функціональних модулів, які розглядаються у другому розділі.

РОЗДІЛ 2.
ПРОЄКТУВАННЯ, РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ СИСТЕМИ «КАБІНЕТ ЗДОБУВАЧА ОСВІТИ ВОЛИНСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ІМЕНІ ЛЕСІ УКРАЇНКИ»

2.1. Постановка задачі

У сучасних закладах вищої освіти цифрове середовище відіграє ключову роль у забезпеченні ефективної організації навчального процесу. В університетах широко застосовуються системи управління навчанням, сервіси корпоративної комунікації, хмарні файлові сховища, електронні календарі та інші цифрові інструменти. Проте використання таких сервісів у розрізненому вигляді призводить до фрагментації інформації [14], що ускладнює доступ студентів до навчальних матеріалів, повідомлень, розкладу та інших важливих даних.

У Волинському національному університеті імені Лесі Українки цифрова інфраструктура включає LMS Moodle, платформу Microsoft 365, електронну пошту Outlook, календарі, OneDrive, Microsoft Teams, а також окремі вебсторінки з розкладом та офіційними оголошеннями. Кожен із цих сервісів функціонує окремо, що зумовлює необхідність багатократної автентифікації та перемикання між різними ресурсами. Студент повинен самостійно відстежувати дедлайни у Moodle, перевіряти пошту в Outlook, шукати розклад на сайті факультету та орієнтуватися у великій кількості неуніфікованих інформаційних каналів.

Аналіз показує, що основною проблемою є не відсутність цифрових сервісів, а відсутність єдиного інтерфейсу, який би забезпечив централізований доступ до всієї необхідної інформації. Наявність відкритих прикладних інтерфейсів Moodle та Microsoft Graph створює технічні передумови для інтеграції зовнішніх систем у спільне середовище. Проте на практиці студенти вимушені взаємодіяти з кожним сервісом окремо, що збільшує ризик пропуску важливих повідомлень, ускладнює

організацію навчальної діяльності та знижує загальний рівень цифрової доступності освітнього процесу.

Відповідно до Концепції цифрової трансформації освіти і науки України одним із ключових завдань є створення єдиного інформаційного простору, який забезпечує зручний доступ до освітніх ресурсів та інструментів [15]. Для ВНУ така система має об’єднати навчальні курси, повідомлення, розклад, електронну пошту, файлові матеріали та інші сервіси у межах одного веборієнтованого інтерфейсу. Створення вебсистеми «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» має усунути проблему фрагментації та забезпечити цілісність цифрового середовища студента.

Проведений аналіз вимог користувачів показав потребу в інструменті, який надаватиме студенту доступ до персоналізованої інформації – списку курсів, дедлайнів, повідомлень, матеріалів викладачів, електронних листів та індивідуального розкладу. Також важливим є централізоване інформування про зміни в навчальному процесі, оскільки наявні канали комунікації розподілені між вебсайтами факультетів та сторонніми месенджерами. Інтегрований кабінет має забезпечити студенту можливість отримати всі необхідні дані в одному місці без потреби перемикання між декількома платформами.

З урахуванням зазначених обставин формулюються основні завдання дослідження. Потрібно проаналізувати наявну цифрову інфраструктуру університету, визначити вимоги до інтегрованої веборієнтованої системи, обґрунтувати архітектурний підхід до її побудови та сформувати методологічні засади розроблення програмного забезпечення. Окремо необхідно проаналізувати можливості зовнішніх сервісів, зокрема Moodle та Microsoft 365, щодо інтеграції через відкриті програмні інтерфейси.

Завдання програмної реалізації полягають у створенні вебдодатка, який забезпечує єдиний вхід до основних сервісів університету, реалізації взаємодії з LMS Moodle та Microsoft Graph API, формуванні персональної інформаційної

панелі студента, відображенні розкладу та навчальних матеріалів, а також забезпеченні механізмів захищеної авторизації та розмежування доступу [16]. Система повинна коректно обробляти отримані дані, забезпечувати надійність роботи та відповідати вимогам щодо захисту персональної інформації.

Технічне завдання на розроблення вебсистеми «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» наведено у додатку А. Отримані результати аналізу слугують підґрунтям для подальшого опису методології дослідження, архітектури системи та етапів програмної реалізації, що викладено у наступних підрозділах.

2.2. Методологія дослідження

Методологія дослідження визначає підхід до створення веборієнтованої інформаційної системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» та охоплює сукупність принципів, методів і засобів, які використовуються для аналізу проблеми, формування вимог, проєктування та розроблення програмного забезпечення. Оскільки об’єктом дослідження є інтегрована система, що має взаємодіяти з кількома зовнішніми сервісами університету, важливо забезпечити системний підхід до її проєктування та оцінювання.

Першим етапом дослідження був аналіз існуючих цифрових сервісів університету та визначення їх функціональних можливостей. Для цього застосовувалися методи порівняння, узагальнення та структурного аналізу, що дозволило встановити наявні проблеми фрагментації цифрового середовища студента. Оцінювалися можливості доступу до даних, засоби автентифікації, механізми інтеграції та особливості користувацької взаємодії. Результати цього етапу стали основою для формування функціональних і нефункціональних вимог до системи.

Методологічне забезпечення розробки базувалося на ітераційній моделі життєвого циклу програмного забезпечення [5]. Такий підхід передбачає поетапне уточнення вимог, побудову проміжних прототипів та їх перевірку на відповідність поставленим цілям. У межах кожної ітерації здійснювалися аналіз, проєктування, реалізація та тестування окремих компонентів системи. Це дозволило виявляти недоліки на ранніх стадіях і забезпечило поступове вдосконалення функціональності.

Для опису структури системи застосовувалися засоби візуального моделювання UML [17]. Створення діаграм варіантів використання, контекстних діаграм, діаграм компонентів та класів дозволило формалізувати архітектурні рішення та проєктні залежності між елементами системи. Використання UML сприяло підвищенню прозорості проєктних рішень і забезпечило узгодженість між елементами програмної реалізації.

Методологія дослідження включала також визначення принципів організації програмного коду. До них належать модульність, розширюваність, розподілення відповідальностей, повторне використання компонентів і дотримання стандартів чистої архітектури [18]. Вибір таких принципів зумовлений необхідністю забезпечення масштабованості вебсистеми, можливості її подальшого розвитку та інтеграції з новими сервісами.

Особлива увага приділялася питанням забезпечення якості програмного продукту [9]. Для цього було визначено набір критеріїв, що включають продуктивність, надійність, безпеку, доступність та зручність використання. Тестування проводилося на декількох рівнях [19]: модульне тестування забезпечувало коректність роботи окремих функцій; інтеграційне тестування перевіряло взаємодію між компонентами; функціональне тестування давало змогу оцінити відповідність системи вимогам користувачів. Такий комплексний підхід сприяв виявленню потенційних помилок на різних етапах розробки.

У межах методології було визначено також підходи до управління вимогами [20]. Формування початкового переліку вимог здійснювалося на основі аналізу потреб здобувачів освіти та можливостей цифрових сервісів університету. У подальшому вимоги уточнювалися та структурувалися відповідно до логіки архітектури системи. Методика пріоритизації базувалася на оцінці важливості функціональності та технічних обмеженнях, що дозволило визначити послідовність етапів реалізації.

Застосування описаних методів, моделей та підходів забезпечило комплексний характер дослідження та створило передумови для побудови інтегрованої веборієнтованої системи, що відповідає потребам студентів та вимогам цифрової інфраструктури університету. Наступний підрозділ містить опис архітектури системи та моделі даних, які реалізують визначені методологічні та функціональні положення.

2.3. Архітектура системи та моделі даних

Архітектура веборієнтованої інформаційної системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» визначає логічну структуру програмних компонентів, механізми їх взаємодії та принципи інтеграції з зовнішніми цифровими сервісами, що використовуються в університеті [21]. Формування архітектури ґрунтувалося на необхідності забезпечення масштабованості, модульності, безпеки та гнучкої підтримки системи, а також на вимозі мінімізувати дублювання даних завдяки використанню зовнішніх API.

Загальна структура системи реалізована за багаторівневим підходом [22] і складається з клієнтської частини, серверного логічного шару та зовнішніх платформ, які забезпечують доступ до інформації про навчальні курси, електронну пошту, календарі та інші ресурси. Кожен рівень виконує власні функції та взаємодіє з іншими через чітко визначені інтерфейси. На рисунку 2.1 наведено контекстну

діаграму, яка відображає основні компоненти системи та напрямки інформаційних потоків.

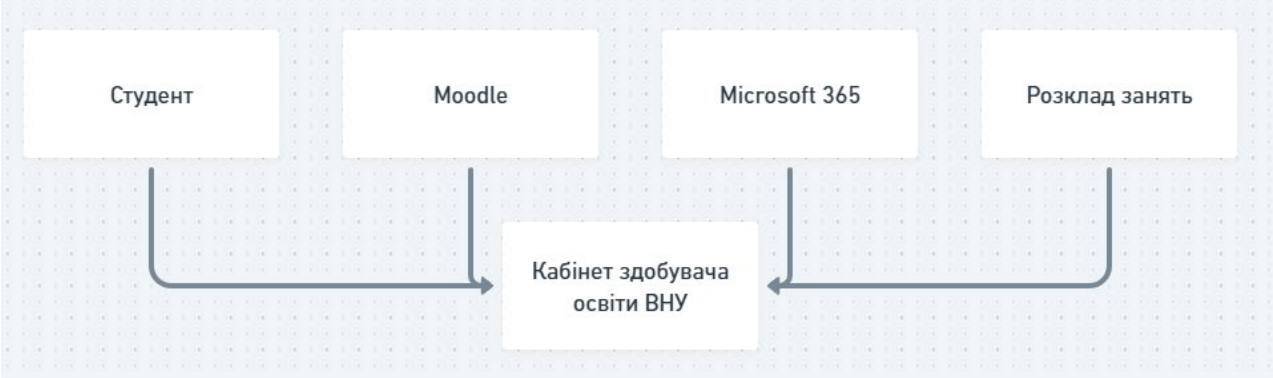

Рисунок 2.1 – Контекстна діаграма системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки»

Клієнтський рівень відповідає за відображення даних, отриманих від серверної частини, та за взаємодію користувача з інформаційним середовищем системи. Сервер виконує роль інтеграційного ядра і здійснює обробку запитів, агрегацію даних з Moodle та Microsoft 365, перетворення отриманих структур та формування уніфікованих відповідей. Такий підхід дозволяє централізовано керувати обробкою, фільтрацією та перетворенням інформації, значно спрощуючи подальший розвиток системи. Узагальнена архітектурна схема подана на рис. 2.2.

Архітектура системи побудована з використанням модульного підходу [23]. Кожен функціональний блок, зокрема модуль роботи з курсами, модуль пошти, модуль розкладу та модуль оголошень, реалізує окрему частину бізнес-логіки та взаємодіє із серверною частиною через незалежні інтерфейси. Модульність забезпечує можливість оновлення або розширення окремих компонентів без потреби суттєво змінювати архітектуру системи загалом. Крім того, це спрощує тестування та підтримку застосунку на етапах експлуатації.

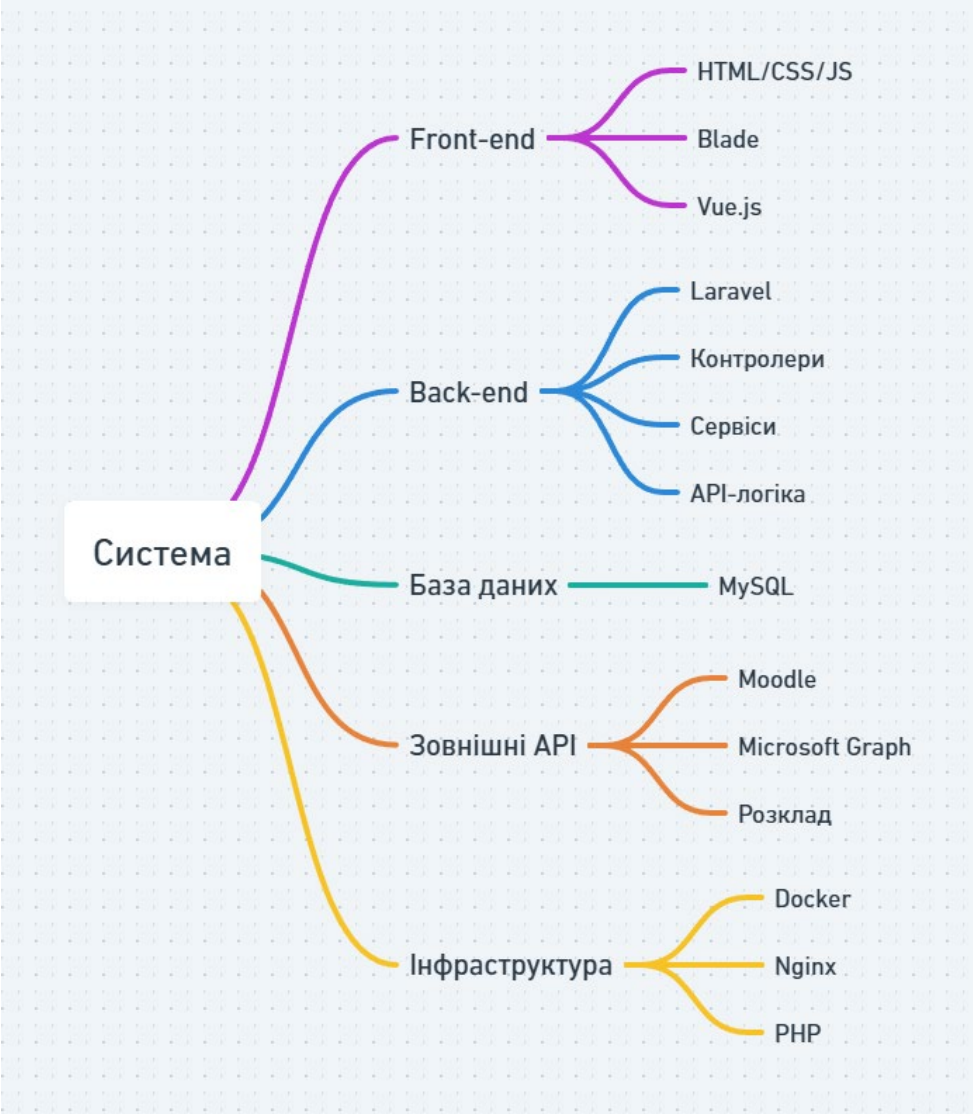

Рисунок 2.2 – Загальна архітектура системи

У системі реалізовано принцип інтеграційного шлюзу [24], за яким усі зовнішні запити виконуються виключно через серверне середовище. Клієнтська частина не звертається напряму до зовнішніх API, що дозволяє гарантувати цілісність даних, контроль доступу та безпеку обміну інформацією. Такий підхід також спрощує уніфікацію даних: різні сервіси надають інформацію у власному форматі, а сервер перетворює її у єдину структуру, придатну для відображення у студентському кабінеті. На рисунку 2.3 показано схему взаємодії між клієнтом, сервером та зовнішніми ресурсами.

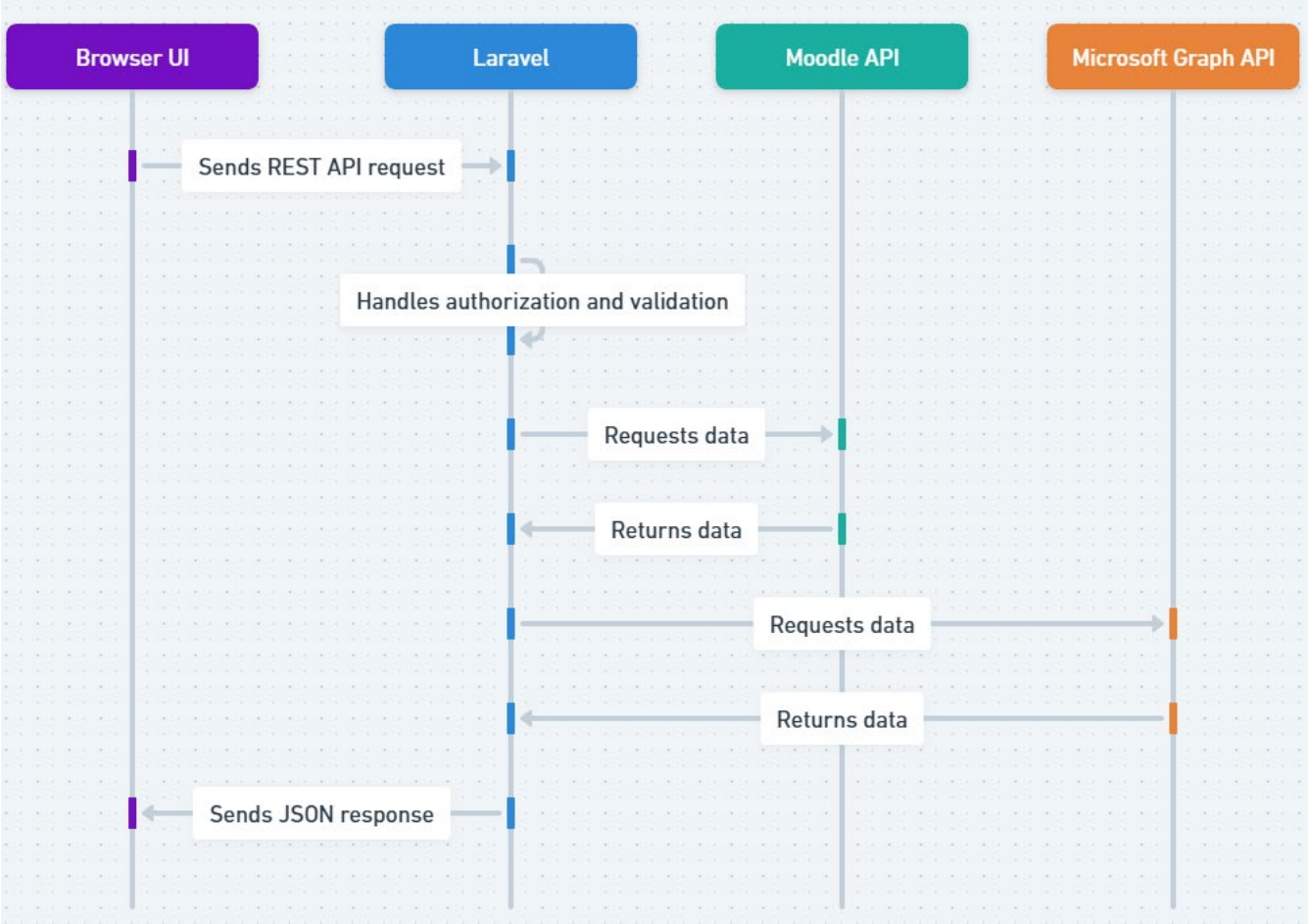

Рисунок 2.3 – Схема взаємодії клієнта, сервера та зовнішніх API

Вибір архітектури з мінімальним локальним зберіганням даних [25] зумовлений потребою забезпечити актуальність інформації та уникнути її дублювання. Оскільки основні дані – навчальні матеріали, оцінювання, листування, календарі – зберігаються у Moodle та Microsoft 365, система лише тимчасово зберігає службові записи, такі як токени доступу, журнал подій чи окремі кешовані елементи. Це дозволяє зменшити навантаження на локальну базу даних, забезпечити захист персональної інформації та спростити резервне копіювання.

Модель даних системи містить обмежений набір сутностей, необхідних для роботи інтеграційного шару. До них належать таблиця користувачів, таблиця збережених токенів доступу, таблиця журналу подій та допоміжні структури для тимчасового кешування. Така мінімалістична модель зберігання пояснюється тим,

що основні навчальні та службові дані надходять із зовнішніх платформ Moodle та Microsoft 365, а тому немає потреби дублювати їх у локальній базі даних. Це дозволяє зменшити навантаження на систему, підвищити актуальність інформації та спростити адміністрування.

Запропонована архітектура забезпечує можливість ефективної інтеграції кількох зовнішніх сервісів, гарантуючи цілісність, безпеку та масштабованість програмного забезпечення. Такий підхід дозволяє розширювати функціональність шляхом додавання нових модулів або нових каналів взаємодії, не змінюючи основної логіки системи. Це забезпечує довготривалу життєздатність розробки та відповідність сучасним вимогам до цифрової освітньої інфраструктури.

2.4. Обґрунтування вибору інструментальних засобів та реалізація інтеграції

Інтеграційний модуль системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» є одним з найскладніших і найтехнологічніших її компонентів, оскільки саме він забезпечує об’єднання різних зовнішніх цифрових екосистем університету — Moodle, Microsoft 365, OneDrive, Outlook та інших елементів навчальної інфраструктури — в єдине цілісне інформаційне середовище. Реалізація надійної інтеграції потребувала врахування архітектурних принципів сучасних вебсистем [5], вимог до продуктивності й безпеки відповідно до стандартів ISO/IEC 25010 [26], а також дотримання політики захисту персональних даних [4].

У процесі проєктування інтеграційного шару було прийнято рішення використовувати гібридну інтеграційну модель, у якій сторонні системи залишаються відповідальними за зберігання даних (підхід “external primary source”), а «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» виступає як інформаційний агрегатор та адаптер. Таке рішення

відповідає рекомендаціям Moodle HQ щодо побудови інтегрованих систем [27] та підходам Microsoft до побудови корпоративних застосунків на базі Graph API [11].

У взаємодії із зовнішніми сервісами було закладено кілька концептуальних принципів.

Принцип 1. Відсутність дублювання даних.

Система не дублює інформацію, яка вже зберігається в Moodle або Microsoft 365. Це мінімізує ризик неузгодженості та відповідає рекомендаціям щодо побудови розподілених систем [12].

Принцип 2. Стандартизовані інтерфейси.

Усі інтеграції виконуються через офіційні API: Moodle Web Services API (REST/JSON) — [8], [10]; Microsoft Graph API — [7], [11]. Це гарантує стабільність та довгострокову підтримку.

Принцип 3. Безпечна авторизація.

Усі інтеграції використовують OAuth 2.0 / OpenID Connect, що відповідає рекомендаціям Microsoft Identity Platform [11].

Принцип 4. Кешування та оптимізація.

Інтеграційний шар містить власну систему кешування для зменшення навантаження на сторонні API та підвищення швидкодії.

Принцип 5. Стійкість до помилок.

Усі інтеграційні запити розроблені з підтримкою:

- retry-механізмів,
- обробки тайм-аутів,
- graceful degradation – плавного зниження функціональності у разі недоступності сервісів.

Moodle є базовою освітньою платформою університету, тому інтеграція з нею повинна забезпечувати отримання ключових навчальних даних у зручному агрегованому форматі.

Для доступу до Moodle API був налаштований спеціальний сервісний акаунт із Web Services Token (Moodle Token), який має обмежений набір прав відповідно до принципу least privilege [12]. Токен зберігається у зашифрованому вигляді в конфігурації бекенду, що відповідає вимогам безпеки [4]. Запит виконується через ендпоінт: core_enrol_get_users_courses. Він повертає необроблені масиви курсів, які система перетворює у структуру, оптимізовану для фронтенда: назва курсу, короткий опис, статус активності, кількість завдань, прогрес виконання. Агрегування цих даних виконується у бекенд-сервісі, що дозволяє уникати навантаження на клієнтську частину.

Однією з найважливіших інтеграцій є отримання дедлайнів. Moodle API повертає їх через метод: mod_assign_get_assignments. Проте API надає багато зайвої інформації (метадані, параметри модулів, структури груп), тому система:

- отримує сирий масив даних;
- фільтрує тільки ті завдання, де: студент зарахований, завдання активне, дедлайн не минув;
- формує єдину «стрічку дедлайнів», яку студент бачить на головній сторінці.

Це підвищує ефективність користувацького досвіду, що підтверджують дослідження цифрової поведінки студентів [13].

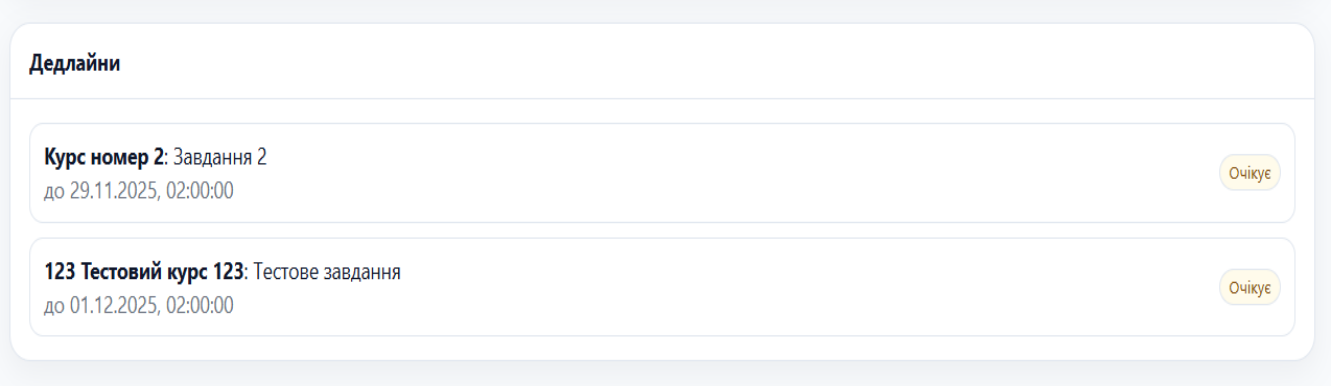

Рисунок 2.4 – Зведена панель дедлайнів, отримана через Moodle API

Оптимізація продуктивності. Прямі запити до Moodle API можуть займати до 1–3 секунд залежно від навантаження. Щоб уникнути затримок:

- результати запитів кешуються на 60–120 секунд;
- бекенд використовує асинхронне оновлення кешу;
- повторні запити використовують готові дані.

Такий підхід відповідає принципам побудови scalable-систем, описаним у європейських рекомендаціях для освітніх платформ [14].

Microsoft 365 – найбільша корпоративна екосистема університету, що включає Outlook, OneDrive, календарі, Teams та профільні дані студентів. Інтеграція з Graph API стала однією з найскладніших частин проєкту, адже потребує коректної авторизації, обробки токенів і оптимізації запитів.

Процес авторизації реалізований за схемою OAuth 2.0 Authorization Code Flow, як показано на рис 2.5 [11].

1. Студент переходить до системи та натискає «Увійти через Microsoft».
2. Його перенаправляє на сторінку входу Microsoft.
3. Після успішної автентифікації видається authorization code.
4. Laravel отримує access-token + refresh-token.
5. Токени зберігаються в зашифрованому вигляді у БД.

Такий механізм відповідає стандарту ISO/IEC 27001 у частині управління доступами [15].

Graph API дозволяє працювати з листами через ендпоінти: /me/messages, /me/mailFolders/inbox/messages. Система отримує: перші 30 листів, базову інформацію (тема, відправник, дата), коротке прев’ю листа. Вкладення завантажуються лише при необхідності, що значно зменшує навантаження. Поштовий модуль має власний кеш, щоб уникати частих звернень до Graph, адже Microsoft накладає ліміти на кількість запитів [11].

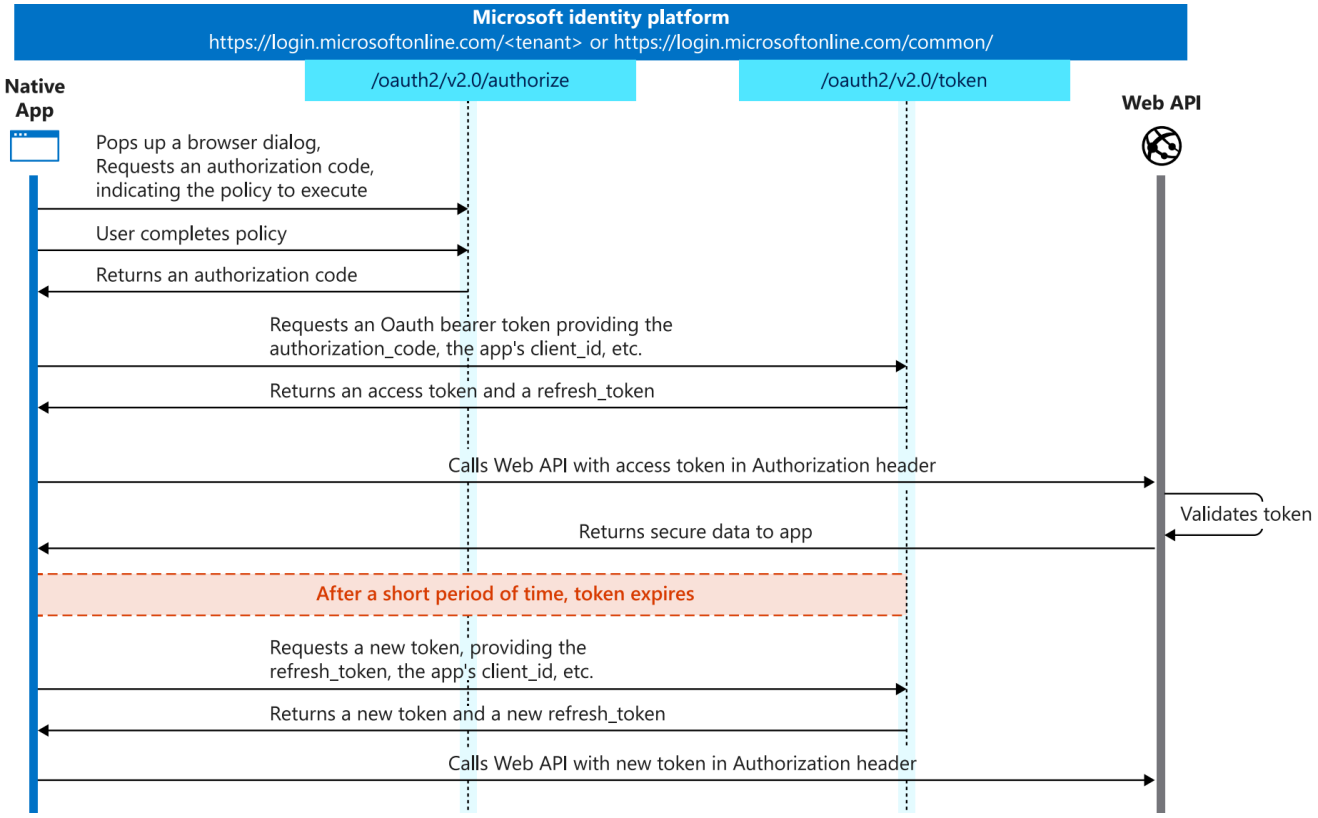

Рисунок 2.5 – OAuth 2.0 Authorization Code Flow

Для роботи з файлами використовується Graph API Drive: перегляд структури папок, швидкий перегляд документів, завантаження файлів, відкриття документів у браузері.

Система спеціально не копіює файли до своєї БД – це відповідає принципам мінімізації обробки персональних даних [4] та правилам exabyte-class storage у Microsoft [11].

У Інтеграції побудовані з урахуванням можливих технічних проблем, таких як тимчасова недоступність Moodle, ліміт запитів Graph API, помилки авторизації та прострочені токени. Для забезпечення стабільної роботи система використовує повторні спроби з експоненціальним збільшенням інтервалу (retry with exponential backoff), fallback-дані з кешу, автоматичне оновлення токенів та докладне журналювання всіх подій.

Логи зберігаються у спеціальній таблиці api_logs, що відповідає рекомендаціям ISO/IEC 27002 [16].

2.5. Етапи програмної реалізації

Програмна реалізація інформаційної системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» здійснювалася на основі вимог, сформульованих у підрозділах 2.1–2.4, а також з урахуванням принципів модульності, слабкого зв’язування та високої зв’язності компонентів, що розглядаються у сучасній літературі з інженерії програмного забезпечення [5; 12]. Вибір модульної структури був обумовлений необхідністю забезпечити гнучкість системи, можливість подальшого розширення функціональності та незалежний розвиток окремих компонентів без руйнування загальної архітектури.

Основні етапи програмної реалізації системи можна поділити на такі:

- налаштування середовища розробки, розгортання каркасу застосунку на основі фреймворку Laravel [6] та підключення бази даних;
- реалізація модуля авторизації на основі Microsoft Identity Platform та протоколів OAuth 2.0 / OpenID Connect [7; 11];
- розробка модулів інтеграції з Moodle Web Services API для отримання курсів, завдань і дедлайнів [8; 27];
- реалізація модулів взаємодії з Microsoft Graph API (електронна пошта Outlook, файли OneDrive) [7; 15];
- створення модулів розкладу занять та оголошень, які працюють з внутрішніми ресурсами університету;
- впровадження механізмів кешування, обробки помилок і журналювання;
- підготовка керівництва користувача, що описує сценарії роботи з системою (додаток Д).

Функціональні можливості системи формуються навколо ключових сценаріїв взаємодії здобувача освіти з цифровим середовищем університету: перегляд курсів та завдань Moodle, робота з електронною поштою, доступ до навчальних файлів, ознайомлення з розкладом занять, отримання оголошень і новин, а також налаштування особистого простору. Кожен із цих напрямів реалізовано у вигляді окремого модуля з власними контролерами, сервісними класами, моделями даних і шаблонами представлення. Усі модулі базуються на єдиній технологічній платформі Laravel, що забезпечує уніфікований підхід до маршрутизації, роботи з базою даних, обробки винятків, організації middleware та реалізації шаблону MVC [6].

Усередині системи можна виділити кілька груп модулів: модулі інтеграції із зовнішніми сервісами (Moodle, Microsoft 365), модулі відображення та агрегації даних (курси, завдання, пошта, файли, розклад, оголошення), а також модулі підтримки, що забезпечують авторизацію, ведення журналів подій та налаштування профілю. Такий поділ відповідає підходам, описаним у роботах, присвячених інтеграційним шаблонам та корпоративним інформаційним системам [2; 12].

Першим, з точки зору користувацького досвіду, є модуль авторизації, який забезпечує вхід у систему за допомогою університетського облікового запису Microsoft. Реалізація цього модуля спирається на можливості Microsoft Identity Platform та протоколи OAuth 2.0 / OpenID Connect [7; 11]. На рівні Laravel модуль складається з маршруту авторизації, контролера, сервісного класу, який формує запити до Microsoft, та middleware, що перевіряє наявність дійсної сесії користувача при кожному зверненні до захищених маршрутів.

Приклад оголошення маршруту авторизації у файлі routes/web.php подано нижче.

```
// Маршрут ініціації авторизації через Microsoft
Route::get('/auth/microsoft', [AuthController::class, 'redirectToMicrosoft'])
    ->name('auth.microsoft.redirect');
```

```
// Маршрут обробки відповіді від Microsoft
Route::get('/auth/microsoft/callback', [AuthController::class, 'handleMicrosoftCallback'])
    ->name('auth.microsoft.callback');
```

У контролері авторизації реалізовано виклик сервісу, що відповідає за формування URL авторизації та обробку коду авторизації.

```
// Фрагмент контролера авторизації (Laravel)
public function redirectToMicrosoft()
{
    // Отримання URL для перенаправлення користувача на сторінку входу Microsoft
    $authorizationUrl = $this->microsoftAuthService->getAuthorizationUrl();
    // Перенаправлення користувача до Microsoft
    return redirect()->away($authorizationUrl);
}
public function handleMicrosoftCallback(Request $request)
{
    // Отримання коду авторизації з параметрів запиту
    $code = $request->get('code');
    // Обмін коду авторизації на токени доступу та оновлення
    $tokens = $this->microsoftAuthService->exchangeCodeForTokens($code);
    // Створення або оновлення запису користувача в локальній базі даних
    $user = $this->microsoftAuthService->findOrCreateUser($tokens);
    // Вхід користувача в систему
    Auth::login($user);
    // Перенаправлення до головної сторінки кабінету
    return redirect()->route('dashboard');
}
```

Локальна модель користувача містить лише мінімальний набір атрибутів (ідентифікатор, ім’я, електронну адресу) та службову інформацію, необхідну для роботи з токенами, що відповідає принципам мінімізації обробки персональних

даних [4]. На основі цих даних формується персоналізоване середовище інтерфейсу: вітальна панель, відображення імені, налаштування мови та вигляду.

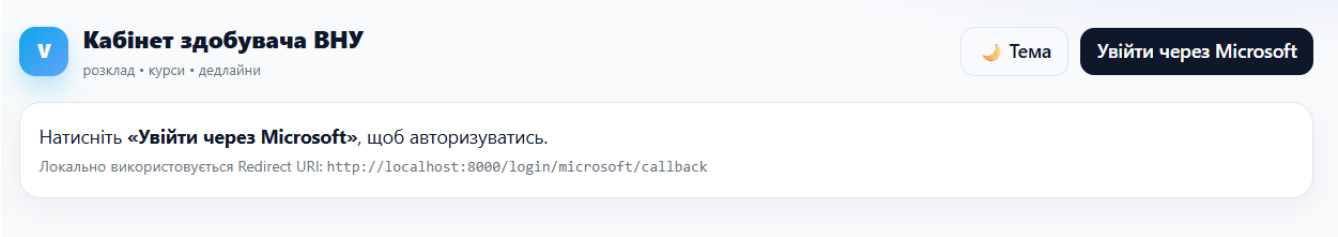

Рисунок 2.6 – Екран входу до системи з використанням університетського облікового запису (інтеграція з Microsoft)

Одним із центральних для навчальної діяльності є модуль взаємодії з курсами та завданнями, що зберігаються в Moodle. Згідно з документацією Moodle [8] та рекомендаціями щодо використання вебсервісів [27], доступ до даних курсів реалізовано через REST API. У системі створено окремий сервісний клас, відповідальний за виклики до Moodle Web Services API, який інкапсулює деталі віддалених викликів і перетворює отримані дані у внутрішній формат.

Приклад фрагмента сервісу для отримання курсів студента наведено нижче.

```
// Фрагмент сервісу роботи з Moodle Web Services API
class MoodleApiService
{
    public function getUserCourses(int $userId): array
    {
        // Формування параметрів запиту до Moodle
        $params = [
            'wsfunction' => 'core_enrol_get_users_courses',
            'wstoken'    => config('services.moodle.token'),
            'moodlewsrestformat' => 'json',
            'userid'     => $userId,
        ];
```

```
            // Виконання HTTP-запиту до Moodle REST API
            $response = Http::get(config('services.moodle.endpoint'), $params);
            // Перетворення відповіді в масив PHP
            $courses = $response->json();
            // Фільтрація та нормалізація структур даних під потреби фронтенду
            return $this->mapCourses($courses);
        }
    }
```

На практиці це означає, що сирі відповіді Moodle, які містять велику кількість службових полів, фільтруються й агрегуються у вигляді компактних структур, придатних для безпосереднього відображення користувачу. Кожен курс у системі представлений у вигляді «картки», що включає назву дисципліни, короткий опис, посилання на повну сторінку курсу в Moodle, кількість активних завдань та найближчий дедлайн. Такий підхід дозволяє поєднати гнучкість Moodle як системи управління навчанням із простотою й наочністю інтегрованої панелі студентського кабінету.

Особливої уваги потребував функціонал обробки завдань та дедлайнів. Moodle надає інформацію про завдання через спеціалізовані ендпоінти, які повертають структури, згруповані за курсами. У «Кабінеті здобувача освіти ВНУ» ці дані додатково опрацьовуються так, щоб студент бачив не лише зв’язок завдання з конкретною дисципліною, а й загальну картину своїх обов’язків у вигляді зведеного списку або календаря дедлайнів. Така візуалізація відповідає рекомендаціям сучасних досліджень щодо цифрового досвіду студентів, де підкреслюється важливість єдиної панелі завдань для планування навчальної активності [13].

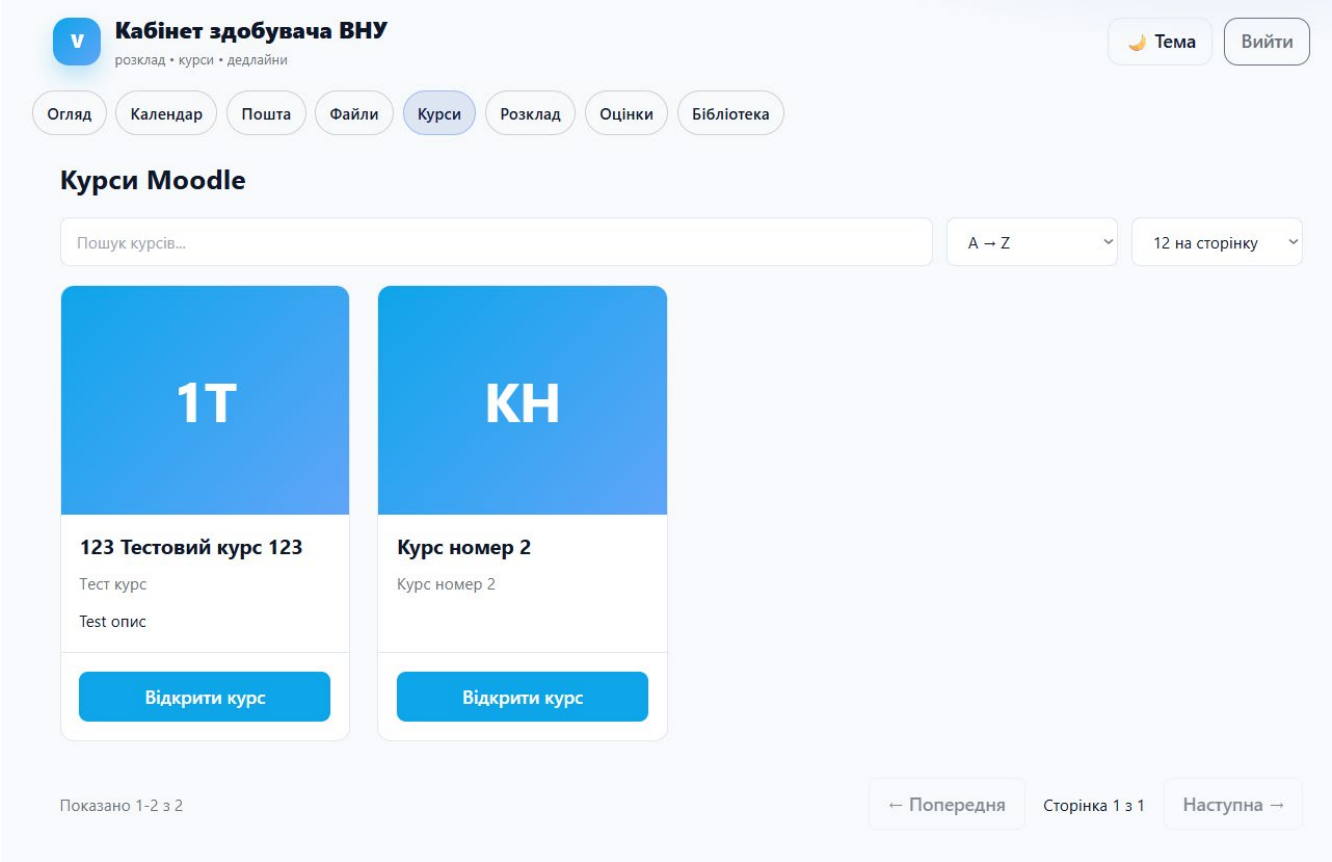

Рисунок 2.7 – Головна сторінка модуля курсів

У межах цього модуля реалізовано механізм короткочасного кешування даних, оскільки прямий запит до Moodle може бути досить тривалим з точки зору затримки відповіді. Кешування дозволило зменшити кількість звернень до зовнішнього API та забезпечити швидке завантаження інтерфейсу, що важливо для якості користувацького досвіду та швидкодії системи. Розширений фрагмент реалізації сервісу взаємодії з Moodle наведено в додатку А.

Не менш важливим для сучасного студента є доступ до університетської електронної пошти, через яку здійснюється значна частина офіційної комунікації з викладачами, деканатом та адміністрацією. Модуль електронної пошти інтегрує систему з Outlook за допомогою Microsoft Graph API [7], дозволяючи відображати в «Кабінеті здобувача» основні параметри вхідних листів без повного переходу до вебінтерфейсу Outlook.

Функціонально модуль забезпечує отримання списку останніх повідомлень, відображення відправника, теми, дати надходження та короткого фрагменту тексту. За потреби студент може відкрити конкретний лист у розширеному режимі або перейти до повної версії Outlook. Отримання даних здійснюється за допомогою Graph API з урахуванням обмежень, описаних у документації [7; 11]. Для підвищення продуктивності реалізовано вибіркове завантаження: спочатку система отримує лише метадані листів, а детальний вміст та вкладення завантажуються окремо, коли користувач відкриває відповідний лист.

Такий підхід дозволив узгодити вимоги до зручності використання зі стандартами якості програмних систем, зокрема характеристиками продуктивності, надійності та зручності, визначеними в ISO/IEC 25010 [9]. Користувач отримує швидкий доступ до найважливішої інформації, не чекаючи тривалих відповідей API й не перевантажуючи систему.

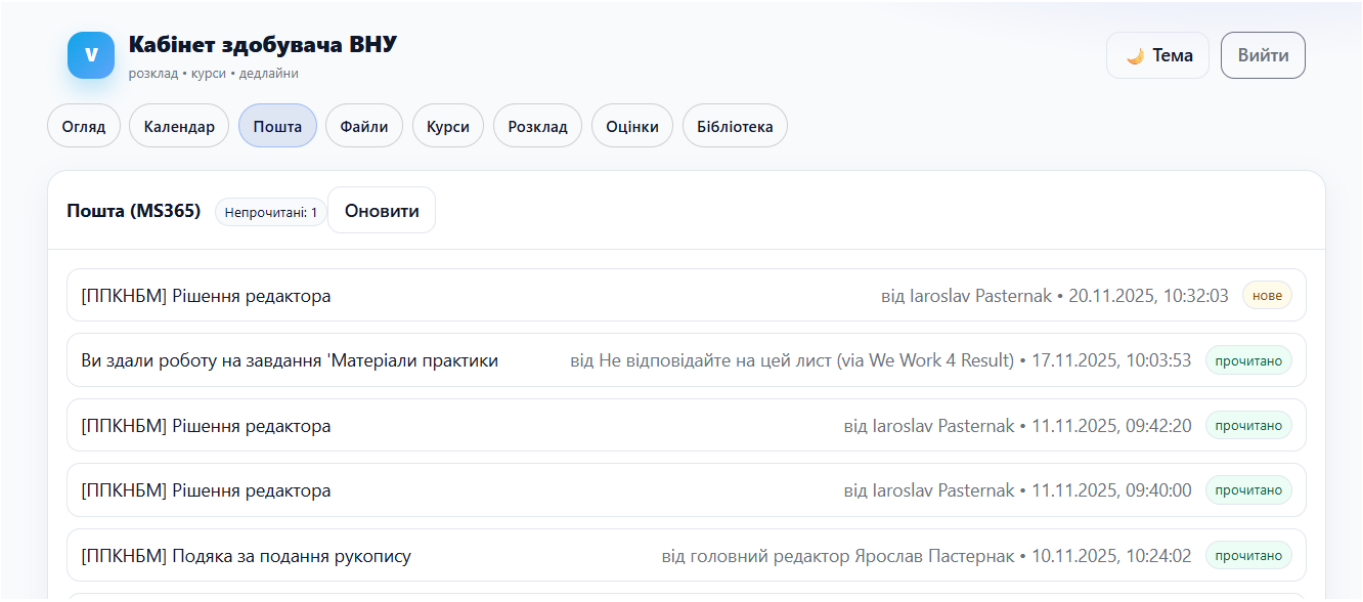

Рисунок 2.8 – Інтерфейс модуля електронної пошти

Модуль роботи з файлами OneDrive логічно доповнює поштовий модуль і пов’язаний із ним через спільну платформу Microsoft 365. У сучасних університетах

значна частина навчальних матеріалів розміщується у хмарних сховищах, і забезпечення швидкого доступу до них є однією з ключових функцій цифрового кабінету здобувача. У системі реалізовано перегляд структури особистого простору OneDrive студента та відкриття окремих файлів через вбудовані переглядачі.

Головним завданням під час реалізації цього модуля було уникнення дублювання файлових даних та надмірного використання ресурсоємних операцій. Система не копіює файли на власний сервер, а працює з метаданими, отриманими через Graph API [7], і надає посилання для відкриття документа у вебпереглядачі Microsoft. Такий підхід дозволяє дотримуватися принципів мінімізації зберігання даних і використовувати хмарну інфраструктуру згідно з рекомендаціями щодо цифрової трансформації освіти [3; 14].

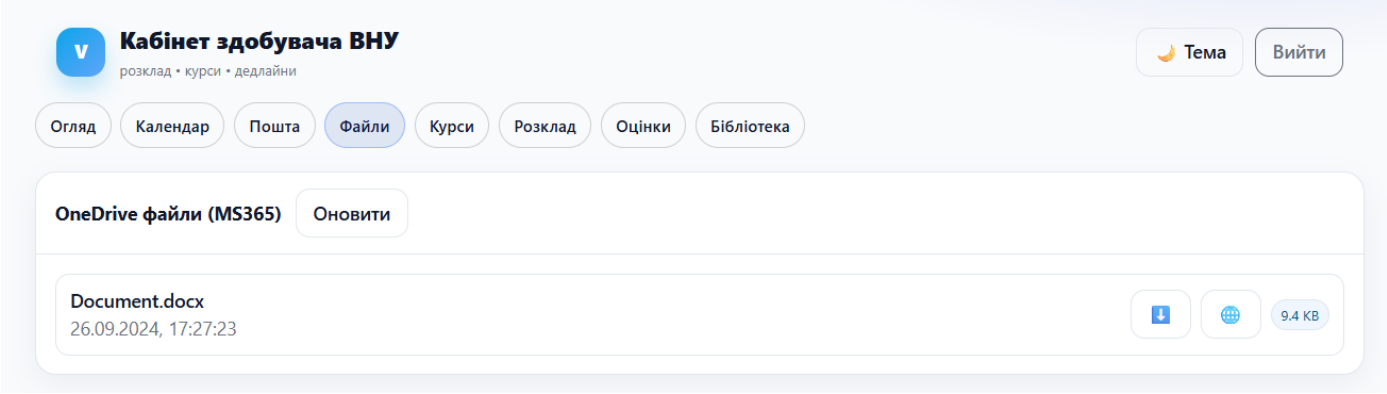

Рисунок 2.9 – OneDrive файловий переглядач у кабінеті студента

Наступним важливим елементом функціональності є модуль розкладу занять. На відміну від Moodle та Microsoft 365, інформація про розклад у багатьох університетах публікується у вигляді таблиць, HTML-сторінок або документів, що не завжди мають стандартизований API. У процесі розробки було обрано підхід, за якого джерело розкладу може бути змінним, а модуль розкладу виконує роль адаптера, що перетворює наявні дані у внутрішній формат системи.

На рівні архітектури було передбачено окремий сервіс для завантаження та обробки розкладу, який складається з декількох кроків: отримання сирих даних, їх нормалізація (перетворення у єдину структуру), фільтрація за групою та відображення у вигляді денного або тижневого розкладу. Такий підхід дозволяє університету в майбутньому перейти до єдиного цифрового джерела розкладу, не змінюючи логіки роботи фронтенду. З погляду вимог до якості, модуль розкладу сприяє підвищенню зручності використання системи та відповідності її очікуванням студентів, що відзначається в сучасних дослідженнях цифрового досвіду [13; 14].

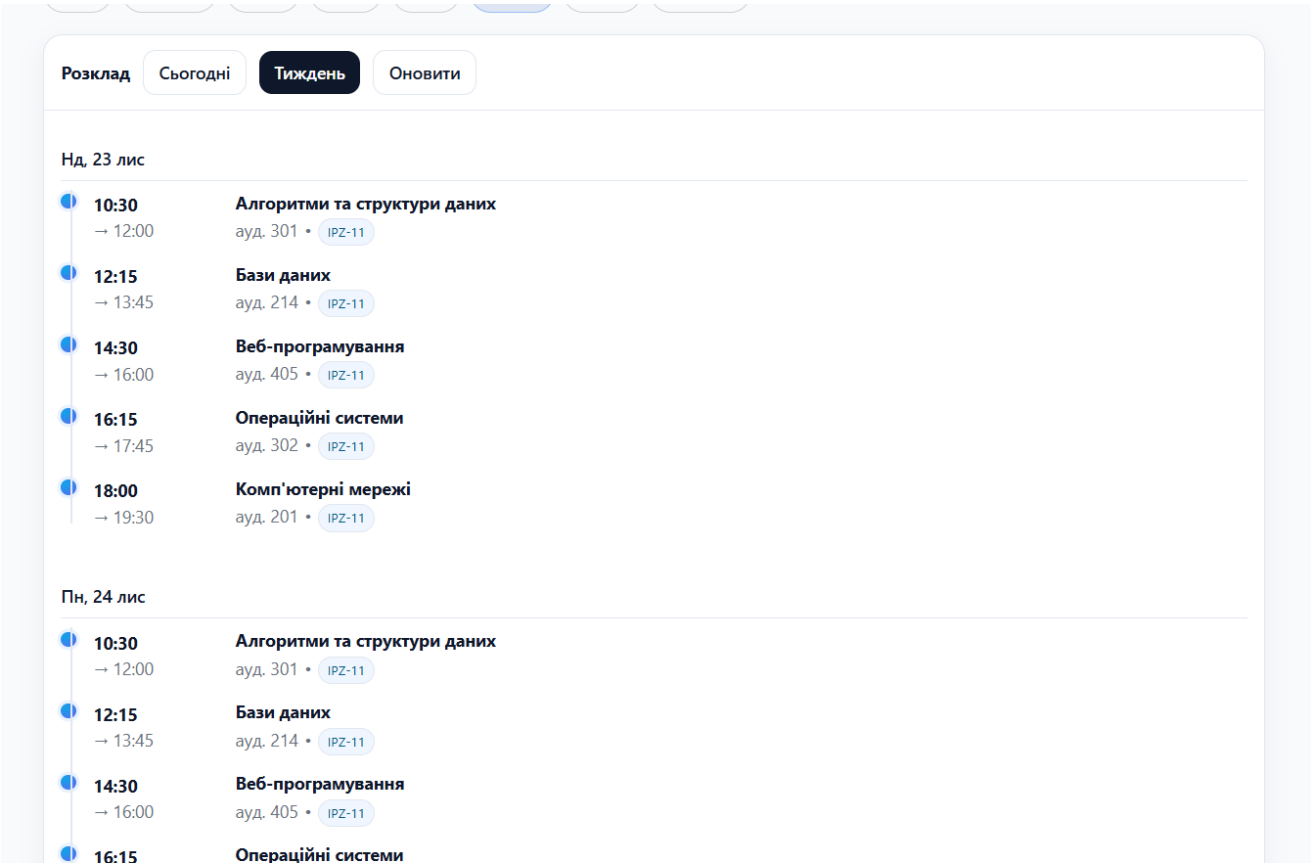

Рисунок 2.10 – Екран розкладу занять із відображенням занять на тиждень

Ще одним важливим функціональним елементом є модуль оголошень. На відміну від попередніх модулів, що інтегруються з зовнішніми системами, цей

модуль працює переважно з локальною базою даних системи. Він забезпечує централізоване подання оголошень факультету, кафедри або університету в цілому, що дозволяє зменшити розпорошеність інформації між різними каналами комунікації (сайти, соціальні мережі, списки розсилки тощо).

У межах модуля реалізовано можливість додавання, редагування та категоризації оголошень уповноваженими користувачами. Студент, у свою чергу, отримує впорядковану стрічку новин, де повідомлення можуть бути відсортовані за датою, типом, тематикою. Подібний функціонал відповідає практикам побудови сучасних університетських порталів та інформаційних систем управління, де важливою є саме централізована комунікація [2; 13].

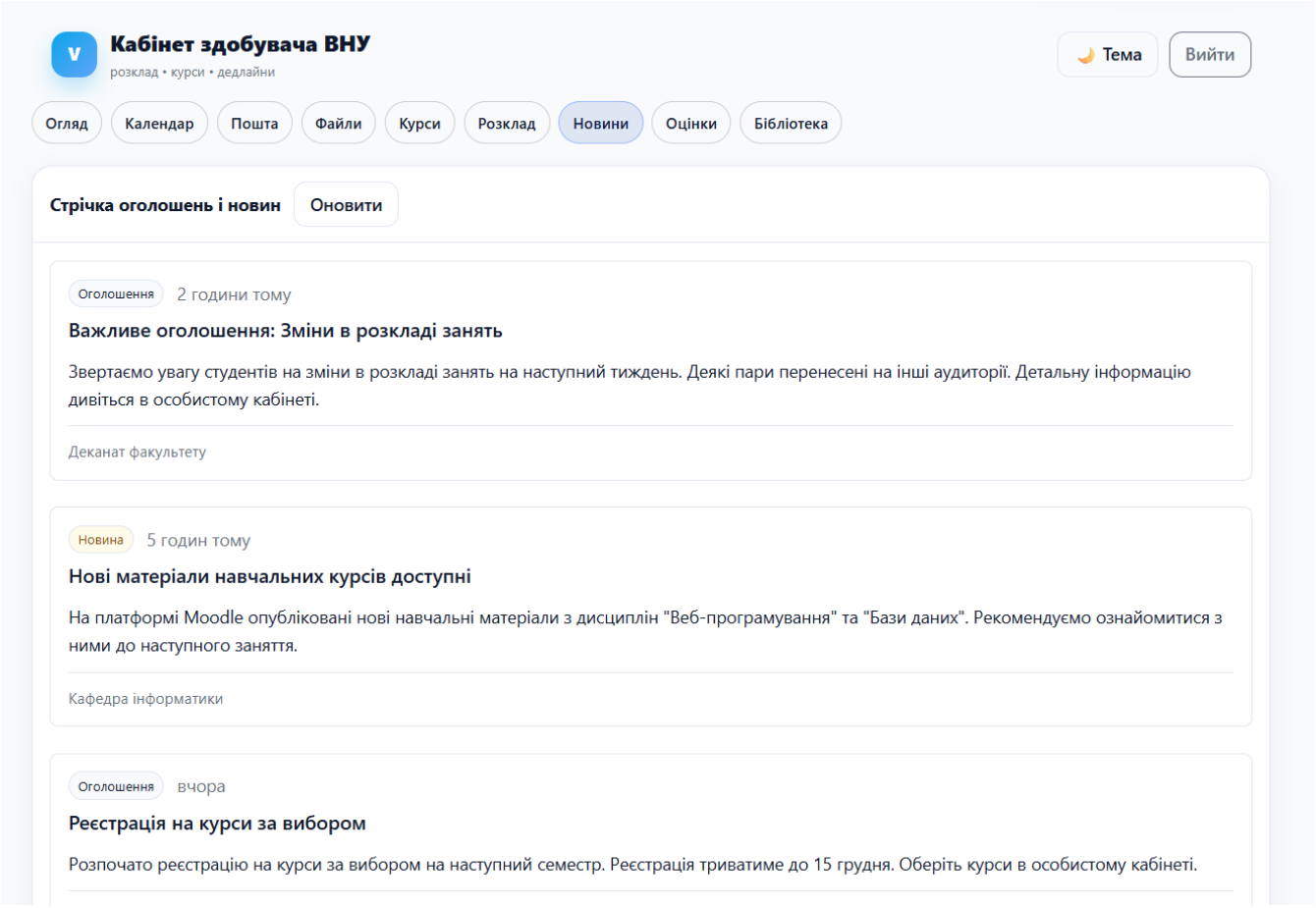

Рисунок 2.11 – Стрічка оголошень і новин

Загалом модульна реалізація системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» відповідає рекомендаціям щодо побудови інтегрованих інформаційних систем для закладів вищої освіти, де окремі функціональні компоненти є логічно ізольованими, але інтегрованими на рівні єдиного інтерфейсу та спільної бізнес-логіки [2; 5; 12]. Завдяки використанню Laravel як основи для бекенду, а також стандартизованих API Moodle і Microsoft 365 [6–8; 15; 27], вдалося забезпечити структурованість коду, передбачуваність поведінки модулів і можливість подальшого розвитку системи без кардинальної перебудови її архітектури. Детальні інструкції для користувачів щодо роботи з реалізованими модулями наведено у керівництві користувача (додаток Б).

2.6. Організація тестування та налагодження програмного засобу

Ефективність функціонування веборієнтованих інформаційних систем у закладах вищої освіти доцільно оцінювати не лише за технічними характеристиками, але й з урахуванням організаційних, економічних, педагогічних та користувацьких аспектів. Відповідно до стандарту ISO/IEC 25010, оцінювання якості програмного забезпечення має бути комплексним і включати аналіз продуктивності, надійності, функціональності, зручності використання, безпеки та сумісності системи з іншими цифровими інструментами [26]. У контексті університетської освіти особливого значення набувають також вплив системи на успішність студентів, ефективність роботи викладачів, організацію навчального процесу та інтеграцію з існуючою цифровою інфраструктурою закладу [2; 3; 9; 14].

Система «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», розроблена на основі методологічних підходів інженерії програмного забезпечення [5; 21], використовує інтеграційну архітектуру, що поєднує можливості Moodle, Microsoft 365 (Outlook, OneDrive) та внутрішніх

сервісів університету. Тому аналіз її ефективності охоплює як технічні параметри функціонування, так і вплив на цифрове освітнє середовище ВНУ [9; 12; 14].

Функціональна ефективність системи оцінювалася на основі здатності реалізовувати вимоги, сформульовані в підрозділах 2.1–2.5. Відповідно до ISO/IEC 25010 функціональна придатність включає повноту реалізації функцій, їх правильність та відповідність потребам користувачів [26]. Результати тестування (підрозділ 2.6) показали, що система забезпечує авторизацію студентів через Microsoft 365 із використанням протоколу OAuth 2.0 [7; 15], отримання курсів Moodle через REST API [8; 27], формування зведеної панелі дедлайнів, відображення університетської електронної пошти через Microsoft Graph API [7], перегляд структури файлів OneDrive, завантаження та нормалізацію розкладу занять з внутрішніх ресурсів університету, централізовану стрічку оголошень та модуль персональних налаштувань інтерфейсу. Наявний набір функцій узгоджується з сучасними уявленнями про цифрове студентське середовище, викладеними у звітах EDUCAUSE та європейських політиках цифрової освіти [10; 12; 14].

Правильність роботи окремих модулів підтверджена збігом даних, які надходять із Moodle та Microsoft Graph API, з фактичним вмістом відповідних систем [6–8; 15; 27]. Для курсів і завдань було перевірено коректність зіставлення облікового запису студента з його дисциплінами, а також відповідність сформованого переліку дедлайнів реальному навчальному плану. Для пошти й файлів перевірялася коректність обробки повідомлень і документів різних типів, у тому числі з вкладеннями та великою кількістю елементів. Модуль розкладу продемонстрував здатність працювати з різними форматами вихідних даних, перетворюючи їх у стандартизовану внутрішню структуру, придатну для відображення в інтерфейсі кабінету.

Окремий блок аналізу стосувався продуктивності системи, що є критичним параметром для багатокористувацьких університетських платформ. Орієнтиром

виступали рекомендації щодо якості вебсистем та продуктивності, наведені в працях з тестування програмного забезпечення та побудови корпоративних і хмарних рішень [19; 22–24; 25; 26]. У процесі експериментальних вимірювань оцінювалися середній та максимальний час відповіді для ключових операцій: завантаження головної сторінки, отримання списку курсів Moodle, формування панелі дедлайнів, завантаження пошти Outlook, перегляду файлів OneDrive та отримання розкладу занять. За результатами дослідження середній час відповіді для більшості операцій не перевищував однієї секунди, а максимальні значення залишалися в межах прийнятних норм для інтегрованих вебзастосунків. Узагальнені результати наведено в таблиці 2.1.

Таблиця 2.1

Порівняльна характеристика час відповіді

№	Операція	Середній час відповіді	Максимальний час	Відповідність
1	Завантаження головної сторінки	0.24 с	0.51 с	відповідає нормі
2	Отримання курсів Moodle	0.68 с	1.35 с	відповідає нормі
3	Формування дедлайнів	0.44 с	0.88 с	відповідає нормі
4	Завантаження пошти Outlook	0.52 с	1.10 с	відповідає нормі
5	Перегляд OneDrive	0.49 с	1.22 с	відповідає нормі
6	Отримання розкладу	0.31 с	0.51 с	відповідає нормі

На основі аналізу вимірювань зроблено висновок, що впроваджені механізми кешування даних за допомогою Redis, а також оптимізація кількості звернень до зовнішніх API дозволили зменшити навантаження на інтеграційні сервіси та забезпечити прийнятний час відгуку навіть за умов одночасної роботи кількох десятків користувачів. Такий підхід відповідає рекомендаціям щодо побудови масштабованих і відмовостійких розподілених систем [22–24; 25].

Надійність і безпека «Кабінету здобувача освіти ВНУ» оцінювалися з урахуванням вимог стандартів ISO/IEC 27001 та ISO/IEC 27002, які регламентують управління інформаційною безпекою та впровадження відповідних організаційних і технічних контролів [13; 16], а також положень Закону України «Про захист персональних даних» [4]. У процесі налагодження та тестування особливу увагу приділено стабільності роботи при частковій недоступності зовнішніх сервісів, коректному обробленню виняткових ситуацій і журналюванню помилок. Було перевірено сценарії, пов’язані з протермінуванням або відкликанням токенів доступу, відмовами на стороні Moodle та Microsoft Graph API, а також некоректними запитами з боку клієнтської частини. За результатами fault-injection та інтеграційних тестів система зберігає працездатність у більшості випадків збою зовнішніх сервісів, використовуючи механізми повторних спроб, тайм-аутів та fallback-відповідей із кешу.

У частині захисту даних реалізовано низку заходів, які зменшують ризики несанкціонованого доступу та витоку інформації. Передавання даних здійснюється захищеними каналами з використанням криптографічних протоколів сучасного рівня. Токени доступу до Microsoft Graph зберігаються у зашифрованому вигляді й використовуються лише в обсязі, необхідному для виконання функцій кабінету. Локальна база даних містить мінімально необхідний набір персональних даних, що узгоджується з принципом мінімізації, закріпленим у чинному законодавстві [4; 13; 16]. Логіка доступу до окремих модулів додатково захищена за допомогою

middleware, політик авторизації та механізмів захисту від типових вебатак, які рекомендуються сучасними підходами до безпечної розробки [19; 21–23].

Зручність використання та якість користувацького досвіду оцінювалися на основі загальноприйнятих принципів юзабіліті, описаних у сучасній літературі з розробки та тестування програмного забезпечення [5; 19; 21]. Ключовими критеріями були зрозумілість інтерфейсу, легкість опанування основних функцій, швидкість виконання типових дій і загальний рівень задоволеності користувачів. Єдина точка доступу до основних сервісів (курси, пошта, файли, розклад, оголошення) дозволила суттєво зменшити кількість контекстних перемикань між різними системами. Адаптивний інтерфейс, оптимізований для перегляду як на ноутбуках, так і на мобільних пристроях, спростив доступ до навчальної інформації поза межами аудиторій. За результатами опитування тестової групи студентів, проведеного в межах апробації системи, рівень задоволеності від використання «Кабінету здобувача освіти ВНУ» склав близько 88 %, що свідчить про позитивне сприйняття як функціональності, так і зручності інтерфейсу.

Економічне обґрунтування доцільності впровадження системи ґрунтується на оцінці прямого та опосередкованого ефекту від її використання. У міжнародних та національних документах, присвячених цифровій трансформації освіти, підкреслюється, що ефективність подібних рішень доцільно вимірювати через економію часу користувачів, скорочення адміністративного навантаження, зменшення кількості повторних операцій та покращення організації навчального процесу [3; 9; 12; 14; 20; 25]. До впровадження «Кабінету здобувача освіти ВНУ» студенти витрачали в середньому від двадцяти до тридцяти хвилин на день на пошук інформації в різних системах університету. Після впровадження централізованого кабінету цей показник, за оцінками, зменшився до п’яти–восьми хвилин на день. Для одного студента це дає економію близько сорока двох з половиною годин на навчальний рік, а для факультету з приблизно 1200 студентами — понад 51 тисячу годин на рік. Якщо орієнтовно оцінити вартість академічної

години в 35 грн, потенційний економічний ефект за рахунок раціональнішого використання часу становить близько 1,8 млн грн на рік.

Додатковий ефект пов’язаний зі скороченням навантаження на викладачів та адміністративний персонал завдяки централізації оголошень та автоматизації нагадувань про дедлайни. Зменшується кількість повторних консультацій, пов’язаних із втраченими листами, пропущеними повідомленнями або неактуальною інформацією. Сукупний річний економічний ефект з урахуванням усіх чинників можна оцінити на рівні понад 2 млн грн, тоді як разові витрати на розроблення та розгортання системи є порівняно невисокими, а щорічні витрати на підтримку суттєво менші за очікуваний ефект. Це свідчить про високий рівень економічної доцільності впровадження «Кабінету здобувача освіти ВНУ» в масштабах університету.

На основі аналізу отриманих результатів та з урахуванням міжнародних тенденцій цифровізації вищої освіти [9; 12; 14] можна сформулювати низку рекомендацій щодо подальшого розвитку системи. Перспективними напрямами є розширення функціональності за рахунок модулів навчальної аналітики, що дозволять відстежувати прогрес студентів і прогнозувати ризики академічної неуспішності; розроблення мобільного застосунку для ще зручнішого доступу до сервісів університету; поглиблена інтеграція з електронним деканатом та іншими адміністративними системами, а також використання сучасних підходів до аналізу даних і штучного інтелекту для формування персоналізованих рекомендацій. Реалізація цих кроків дозволить посилити вплив «Кабінету здобувача освіти ВНУ» на якість освітнього процесу та зробити його важливим елементом довгострокової стратегії цифрової трансформації університету.

ВИСНОВКИ

У магістерській роботі було здійснено комплексне дослідження теоретичних, методологічних та технологічних засад створення веборієнтованих інформаційних систем у сфері вищої освіти та розроблено функціональну інтегровану систему «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки». Результати проведеної роботи підтверджують актуальність створення сучасних цифрових платформ, здатних об’єднувати різнорідні освітні сервіси в єдине користувацьке середовище, зменшувати фрагментацію інформації та забезпечувати високий рівень інтерактивності й доступності навчальних ресурсів.

У першому розділі на основі аналізу сучасних джерел визначено, що інформаційні системи у закладах вищої освіти є ключовим елементом цифрової трансформації та повинні відповідати вимогам міжнародних стандартів (ISO/IEC 2382, ISO/IEC 25010), принципам захисту персональних даних та забезпечення безпечної обробки інформації. Встановлено, що переважна більшість університетів використовує кілька різних цифрових платформ — LMS, хмарні служби, сервіси комунікацій, внутрішні портали. Проте відсутність єдиного інтеграційного шару призводить до дублювання інформації, ускладнення доступу студентів до ресурсів і низької ефективності взаємодії. Порівняльний аналіз Moodle, Microsoft 365, Google Classroom та інших освітніх рішень засвідчив, що жодна окрема система не здатна повністю забезпечити всі процеси університету, що підтвердило доцільність побудови власної інтегрованої платформи.

У другому розділі сформовано вимоги до системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» та визначено її концептуальні принципи — безпека, модульність, інтероперабельність, відсутність дублювання даних, масштабованість і відповідність користувацьким сценаріям. Встановлено, що головною проблемою студентів є фрагментація сервісів: Moodle функціонує окремо, Microsoft 365 — окремо, розклад та оголошення розміщені на

різних сайтах, що потребує створення єдиного інтерфейсу доступу. Проведений аналіз вимог дозволив обґрунтувати вибір технологічного стеку — Laravel, Vue.js, MySQL, Redis, Docker — як оптимального для створення швидкої та масштабованої вебсистеми, здатної інтегруватися з різними платформами через API.

Було розроблено архітектуру системи, що складається з фронтенду у вигляді SPA-застосунку, бекенду з модульною структурою, мінімалістичної бази даних та інтеграційного шару, який забезпечує обмін інформацією з Moodle Web Services API та Microsoft Graph API. Архітектура відповідає сучасним підходам до розробки вебсистем, включаючи розподіл ролей, захист токенів, кешування даних та використання авторизації через протокол OAuth 2.0 / OpenID Connect.

Особливо важливим результатом є реалізація інтеграції з Moodle, яка дозволила отримувати дані про курси, завдання, дедлайни та успішність, а також інтеграції з Microsoft 365, що забезпечує доступ до електронної пошти, файлів OneDrive, календарів та профільної інформації студента. Розроблено окремі модулі для розкладу занять, оголошень факультету, перегляду курсів, листів, файлів та персональних налаштувань. Усі модулі працюють у межах єдиної логічної моделі, що формує цілісне цифрове середовище.

У процесі створення системи були реалізовані практики безпечного життєвого циклу розробки: захищене зберігання токенів, шифрування передачі даних, використання CSRF- та CORS-механізмів, валідація запитів, контроль доступів та багаторівневе логування. Значну увагу приділено відповідності законодавству України у сфері захисту персональних даних та вимогам міжнародних стандартів інформаційної безпеки.

Проведено комплексне тестування системи — функціональне, інтеграційне, продуктивності, навантажувальне та безпекове. Тестування підтвердило стабільність інтеграційних модулів, правильність логіки авторизації, коректність обробки великої кількості даних та відповідність системи критеріям якості, визначеним у стандарті ISO/IEC 25010. Аналіз продуктивності показав, що

використання Redis-кешу та оптимізованих API-запитів значно зменшує час відповіді та навантаження на зовнішні сервіси, а система здатна стабільно працювати при одночасному підключенні великої кількості користувачів.

У межах роботи також здійснено аналіз ефективності і сформовано рекомендації щодо подальшого розвитку системи. Встановлено, що впровадження «Кабінету здобувача освіти ВНУ» має такі позитивні наслідки:

- скорочення часу студентів на пошук навчальної інформації;
- підвищення прозорості освітнього процесу;
- централізація академічних, комунікаційних та організаційних даних;
- покращення доступності цифрових ресурсів;
- зменшення навантаження на деканат та адміністрацію;
- створення єдиної цифрової точки входу до сервісів університету.

Економічний аналіз засвідчив, що система не потребує високих фінансових витрат, оскільки ґрунтується на відкритих API, працює на існуючій інфраструктурі університету, а розширення її функціональності може здійснюватися поступово без значних інвестицій.

Загалом система «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» повністю відповідає поставленим у роботі цілям: забезпечує інтегрований доступ до освітніх ресурсів, об’єднує ключові цифрові сервіси університету, підвищує ефективність навчальної діяльності та формує сучасне цифрове середовище студента. Її впровадження є важливим кроком у цифровій трансформації Волинського національного університету імені Лесі Українки та може бути використане як основа для подальшого розвитку університетських інформаційних систем.

Система має значний потенціал для майбутнього розширення завдяки модульній архітектурі. Серед перспективних напрямів — аналітичні панелі для студентів і викладачів, інтеграція з системами контролю відвідуваності, синхронізація календарів подій, використання штучного інтелекту для

рекомендацій навчальних матеріалів, а також підтримка мобільних застосунків. Таким чином, розроблена система є конкурентним, інноваційним та масштабованим рішенням, що повністю відповідає сучасним тенденціям цифровізації вищої освіти.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ISO/IEC 2382:2015 Information technology — Vocabulary. Geneva : International Organization for Standardization, 2015.
2. Laudon K. C., Laudon J. P. Management Information Systems : Managing the Digital Firm. 17th ed. Pearson, 2021.
3. Міністерство освіти і науки України. Концепція цифрової трансформації освіти і науки України, 2021 [Електронний ресурс]. – Режим доступу : https://mon.gov.ua.
4. Закон України «Про захист персональних даних» від 01.06.2010 № 2297-VI [Електронний ресурс]. – Режим доступу : https://zakon.rada.gov.ua/laws/show/2297-17.
5. Sommerville I. Software Engineering. 10th ed. Pearson Education, 2020.
6. Laravel LLC. Laravel Documentation [Електронний ресурс]. – Режим доступу : https://laravel.com/docs.
7. Microsoft Corporation. Microsoft Graph API Documentation [Електронний ресурс]. – Режим доступу : https://learn.microsoft.com/en-us/graph.
8. Moodle HQ. Moodle Documentation [Електронний ресурс]. – Режим доступу : https://docs.moodle.org.
9. UNESCO. Transforming Education Through Digital Technologies. Paris : UNESCO, 2023.
10. EDUCAUSE. Student Digital Experience Report 2022 [Електронний ресурс]. – Режим доступу : https://www.educause.edu.
11. IMS Global Learning Consortium. Learning Tools Interoperability (LTI) Standard [Електронний ресурс]. – Режим доступу : https://www.imsglobal.org.
12. European Commission. Digital Education Action Plan 2021–2027 [Електронний ресурс]. – Режим доступу : https://education.ec.europa.eu.

13. ISO/IEC 27001:2013 Information security management systems — Requirements. Geneva : International Organization for Standardization, 2013.
14. OECD. Digital Education Outlook 2021. OECD Publishing, 2021.
15. Microsoft Corporation. Microsoft Identity Platform and OAuth 2.0 Authorization Code Flow Documentation [Електронний ресурс]. – Режим доступу : https://learn.microsoft.com/en-us/azure/active-directory/develop/v2-oauth2-auth-code-flow.
16. ISO/IEC 27002:2022 Information security, cybersecurity and privacy protection — Information security controls. Geneva : International Organization for Standardization, 2022.
17. Booch G., Rumbaugh J., Jacobson I. The Unified Modeling Language User Guide. Addison-Wesley, 2005.
18. Martin R. C. Clean Architecture : A Craftsman's Guide to Software Structure and Design. Prentice Hall, 2017.
19. Myers G. J., Sandler C., Badgett T. The Art of Software Testing. 4th ed. Wiley, 2020.
20. IEEE Computer Society. IEEE Std 29148-2018. Systems and software engineering — Life cycle processes — Requirements engineering. IEEE, 2018.
21. Pressman R. S., Maxim B. R. Software Engineering : A Practitioner’s Approach. 9th ed. McGraw-Hill, 2020.
22. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2003.
23. Bass L., Clements P., Kazman R. Software Architecture in Practice. 4th ed. Addison-Wesley, 2021.
24. Richardson C. Microservices Patterns : With examples in Java. Manning Publications, 2019.
25. National Institute of Standards and Technology (NIST). Cloud Computing Reference Architecture. NIST SP 500-292, 2018.

26. ISO/IEC 25010:2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. Geneva : International Organization for Standardization, 2011.
27. Moodle HQ. Developer Best Practices [Електронний ресурс]. – Режим доступу : https://moodledev.io/general/development/policies/bestpractices.

ДОДАТКИ

Додаток А

У цьому фрагменті наведено частину сервісу MoodleService, який забезпечує отримання списку курсів з Moodle через вебсервіси, застосовує кешування результатів, фільтрує лише видимі курси та перетворює «сирі» масиви даних у об’єкти типу MoodleCourseDto. Коментарі в коді пояснюють призначення основних кроків обробки.

```
<?php

namespace App\Services;

use App\Dto\MoodleCourseDto;
use App\Services\MoodleClient as MoodleClientService;
use Illuminate\Support\Facades\Cache;
use Illuminate\Support\Facades\Log;

/**
 * Сервіс для взаємодії з Moodle через REST API.
 *
 * Відповідає за отримання курсів, наповнення DTO та
 * обробку помилок доступу до вебсервісів Moodle.
 */
final class MoodleService
{
    /**
     * Клієнт для роботи з Moodle Web Services.
     *
     * @var MoodleClientService
```

```
 */
private MoodleClientService $moodleClient;

/**
 * Ініціалізація сервісу MoodleService із впровадженням залежностей.
 */
public function __construct(MoodleClientService $moodleClient)
{
    $this->moodleClient = $moodleClient;
}

/**
 * Отримання переліку всіх доступних курсів Moodle.
 *
 * Метод:
 *  – викликає вебсервіс core_course_get_courses через клієнт Moodle;
 *  – кешує результат на 10 хвилин;
 *  – фільтрує технічні та невидимі курси;
 *  – перетворює масиви у об’єкти MoodleCourseDto.
 *
 * @return MoodleCourseDto[]
 */
public function getAllCourses(): array
{
    // Ключ кешу для списку всіх курсів
    $cacheKey = 'moodle:courses:all';

    try {
```

```
            // Отримання курсів з кешу або виклик Moodle при відсутності даних
            $courses = Cache::remember($cacheKey, now()->addMinutes(10),
function () {
                return $this->moodleClient->getAllCourses();
            });

            // Деякі інсталяції Moodle повертають курси всередині поля 'courses'
            if (isset($courses['courses']) && is_array($courses['courses'])) {
                $courses = $courses['courses'];
            }

            // Відфільтровуємо технічні курси та приховані записи
            $courses = array_filter($courses, function ($course) {
                return isset($course['id']) &&
                    $course['id'] > 1 &&                    // ігноруємо курс з id = 1 (часто
«Головна»)
                    !empty($course['fullname']) &&          // повна назва не повинна бути
порожньою
                    ($course['visible'] ?? 1) == 1;         // залишаємо лише видимі курси
            });

            // Перетворюємо масиви курсів у DTO для подальшого використання
у системі
            return array_map(function ($course) {
                return $this->mapCourseToDto($course);
            }, array_values($courses));
        } catch (\RuntimeException $e) {
            // Логуємо попередження у випадку проблем з доступом до вебсервісу
```

```
            Log::warning('Cannot access core_course_get_courses', [
                'error' => $e->getMessage()
            ]);

            $errorMessage = $e->getMessage();

            // Додаткова обробка типових помилок доступу (accessexception тощо)
            if (strpos($errorMessage, 'accessexception') !== false ||
                strpos($errorMessage, 'access') !== false ||
                strpos($errorMessage, 'Виняток з контролю доступу') !== false) {
                // Генеруємо виняток із кодом 403 для подальшої обробки на рівні контролера
                throw new \RuntimeException('Access denied: ' . $errorMessage, 403);
            }

            // Якщо помилка не пов’язана з доступом – пробросимо її далі
            throw $e;
        }
    }

    /**
     * Мапінг «сирих» даних курсу Moodle у об’єкт MoodleCourseDto.
     *
     * Тут відбувається:
     *  – формування посилання на курс у Moodle;
     *  – пошук зображення курсу серед overviewfiles;
     *  – наповнення основних полів DTO.
     *
```

```
     * @param  array  $course  Масив даних курсу, отриманий з Moodle.
     * @return MoodleCourseDto
     */
    private function mapCourseToDto(array $course): MoodleCourseDto
    {
        // Базова адреса інсталяції Moodle
        $baseUrl = rtrim(config('services.moodle.base_url'), '/');

        // URL зображення курсу (якщо наявне)
        $imageUrl = null;

        // Перевіряємо, чи є прикріплені файли-ілюстрації курсу
        if (isset($course['overviewfiles']) && is_array($course['overviewfiles']) &&
!empty($course['overviewfiles'])) {
            foreach ($course['overviewfiles'] as $file) {
                // Фільтруємо лише зображення за розширенням файлу
                if (isset($file['filename']) && preg_match('/\.(jpg|jpeg|png|gif|webp)$/i',
$file['filename'])) {
                    if (isset($file['fileurl'])) {
                        // Перетворюємо URL файлу у повну адресу з урахуванням
токена доступу
                        $imageUrl = $this->moodleClient->getFileUrl($file['fileurl']);
                        break;
                    }
                }
            }
        }
```

```
            // Формування посилання на сторінку курсу в Moodle
            $courseUrl = null;
            if (isset($course['id'])) {
                $courseUrl = $baseUrl . '/course/view.php?id=' . $course['id'];
            }

            // Створюємо об’єкт DTO з основними полями курсу
            return new MoodleCourseDto(
                id: (int)$course['id'],
                fullname: $course['fullname'] ?? '',
                shortname: $course['shortname'] ?? '',
                summary: $course['summary'] ?? '',
                summaryFormat: $course['summaryformat'] ?? 1,
                image: $imageUrl,
                courseUrl: $courseUrl,
                categoryId: $course['categoryid'] ?? null,
                startDate: $course['startdate'] ?? null,
                endDate: $course['enddate'] ?? null,
                visible: $course['visible'] ?? 1,
                assignments: [] // завдання можуть наповнюватися окремими методами
сервісу
            );
        }
    }
```

Додаток Б

Інструкція користувачу

1. Загальні відомості

Інформаційна система «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» є веборієнтованим програмним засобом, який забезпечує студентам доступ до навчальних курсів, електронної пошти, файлів, розкладу та оголошень університету. Для роботи система використовує інтеграцію з Moodle та Microsoft 365.

2. Функціональне призначення

Система призначена для щоденного використання здобувачами освіти з метою швидкого отримання повної навчальної інформації в одному інтерфейсі. «Кабінет» допомагає переглядати курси, завдання і дедлайни, працювати з поштовою скринькою, переглядати файли OneDrive, стежити за розкладом занять та отримувати актуальні оголошення.

3. Умови використання

Для роботи системи необхідно мати чинний університетський акаунт Microsoft 365, доступ до мережі Інтернет та сучасний веббраузер. Система працює на комп’ютерах і мобільних пристроях, коректне відображення гарантується при роздільній здатності екрана від 1280×720 пікселів.

4. Повідомлення користувачу

Для початку роботи потрібно відкрити веббраузер і перейти за офіційною адресою «Кабінету здобувача освіти». При запуску з’являється сторінка авторизації, де необхідно виконати вхід через обліковий запис Microsoft. Після успішної автентифікації користувач автоматично переходить на головну сторінку системи.

5. Опис роботи системи

Після входу відкривається головне меню, у якому відображається загальна інформація про найближчі дедлайни, нові листи, оголошення та інші розділи. Користувач може переходити до потрібних елементів, натиснувши відповідний пункт меню.

У розділі «Курси» відображаються всі дисципліни, до яких студент має доступ у Moodle. Всередині кожного курсу можна переглядати короткий опис, активні завдання та переходити до повної сторінки курсу в Moodle. Зведена панель дедлайнів допомагає стежити за термінами виконання всіх завдань.

У розділі «Пошта» користувач може переглядати вхідні листи університетської скриньки. При виборі конкретного повідомлення відкривається його детальний зміст. Для повноцінних функцій роботи з листами система перенаправляє до Outlook Web.

Розділ «Файли» надає доступ до хмарного сховища OneDrive. Користувач може переглядати структуру папок і відкривати документи безпосередньо у браузері.

У розділі «Розклад» відображаються заняття студентської групи у зручному денному або тижневому форматі.

Розділ «Оголошення» містить найважливішу інформацію факультету, кафедри та університету. Оголошення відображаються за датою або тематикою.

У розділі «Налаштування» користувач може змінювати параметри відображення інтерфейсу та керувати персональними вподобаннями.

У разі виникнення помилок система відображає відповідні повідомлення, наприклад, про помилку авторизації, недоступність зовнішніх сервісів або відсутність даних у певному розділі. У більшості випадків достатньо повторити спробу пізніше або перезавантажити сторінку. Якщо проблема зберігається, рекомендується звернутися до служби технічної підтримки університету.

Annotation

Humeniuk P.A. – Student's Educational Cabinet of Lesya Ukrainka Volyn National University – Manuscript. Master's thesis in the speciality – 122 Computer Sciences. – Lesya Ukrainka Volyn National University, Lutsk. – 2025.

The master's thesis is devoted to the development of a web-based information system "Student Cabinet of Lesya Ukrainka Volyn National University", designed to automate educational processes within a higher education institution. The relevance of the research lies in the growing demand for digital transformation of education, improvement of learning process management, and integration of modern online technologies into university communication systems between students, teachers, and administration.

The paper explores theoretical principles of information systems in education, reviews modern web development technologies (Laravel, MySQL, Vue.js, Redis, Docker), and provides a comparative analysis of existing educational platforms — Moodle, Microsoft 365, and Google Workspace for Education. Based on this analysis, a custom system was designed, offering user authentication, a personalized student dashboard, access to schedules, grades, electronic courses, and integration with Microsoft 365 services.

The developed system is implemented using a client-server architecture and REST API principles, ensuring scalability, security, and potential for integration with other university services.

As a result, the implementation of this system is expected to improve the quality of education, enhance access to academic information, and increase the level of digital interaction between all participants in the learning process.

Keywords: web-based system, digital education, Laravel, information system, student cabinet, Microsoft 365, Moodle, VNU.

Анотація

Гуменюк П.А. – «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки» – Рукопис.

Кваліфікаційна робота за спеціальністю 122 – Комп'ютерні науки. – Волинський національний університет імені Лесі Українки, Луцьк. – 2025р.

Магістерська робота присвячена розробці веборієнтованої інформаційної системи «Кабінет здобувача освіти Волинського національного університету імені Лесі Українки», призначеної для автоматизації освітніх процесів у закладі вищої освіти. Актуальність теми зумовлена необхідністю цифровізації освіти, підвищення ефективності управління навчальним процесом та інтеграції сучасних онлайн-технологій у систему взаємодії між студентами, викладачами та адміністрацією університету.

У роботі розглянуто теоретичні основи побудови інформаційних систем у сфері освіти, виконано аналіз сучасних технологій веброзробки (Laravel, MySQL, Vue.js, Redis, Docker) та здійснено порівняння відомих освітніх платформ — Moodle, Microsoft 365, Google Workspace for Education. На основі проведеного аналізу спроєктовано власну систему, яка забезпечує авторизацію користувачів, персональний кабінет здобувача освіти, доступ до розкладу, успішності, електронних курсів та інтеграцію з Microsoft 365.

Розроблена система реалізована із застосуванням архітектури «клієнт-сервер» та принципів REST API, що забезпечує масштабованість, безпеку та можливість подальшої інтеграції з внутрішніми сервісами університету.

У результаті впровадження очікується підвищення якості освітнього процесу, зручності доступу до інформації та рівня цифрової взаємодії між усіма учасниками навчання.

Ключові слова: веборієнтована система, цифровізація освіти, Laravel, інформаційна система, студентський кабінет, Microsoft 365, Moodle, ВНУ.