НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БІОРЕСУРСІВ І
ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
Факультет інформаційних технологій

ПОГОДЖЕНО **Декан факультету**	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ** **Завідувач кафедри інформаційних систем і технологій**
_____**Ігор Болбот** (підпис)	_____**Михайло Швиденко** (підпис)
«___» _____2026 р.	«___» _____2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Система корпоративного навчання на основі веб-технологій»

Спеціальність «126 Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційні системи та технології»

Гарант освітньої програми
к.е.н., доцент _____Максим Мокрієв
(підпис)

Керівник бакалаврської кваліфікаційної роботи _____ **Сергій Саяпін**
(підпис)
к.е.н, доцент

Виконав _____ **Володимир Бортник**
(підпис)

Київ-2026

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри інформаційних систем та технологій
к.е.н., професор _____Михайло Швиденко
(підпис)

«___» _____2026 р.

ЗАВДАННЯ ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Бортнику Володимиру Володимировичу
Спеціальність «126 Інформаційні системи та технології»
Освітньо-професійна програма «Інформаційні системи та технології»
Тема бакалаврської кваліфікаційної роботи
«Система корпоративного навчання на основі веб-технологій»
затверджена наказом від «19 грудня» 2025 р. № 3205«С»

Термін подання завершеної роботи на кафедру «30» травня 2026 р.

Вихідні дані до бакалаврської кваліфікаційної роботи:

- Технічне завдання на розробку веб-орієнтованої системи корпоративного навчання;
- Вимоги до організації процесів L&D (Learning and Development) та рольової моделі доступу (4 типи акторів);
- Стек технологій розробки: мова програмування Python (серверна частина), мова JavaScript, HTML5/CSS3 (клієнтська частина).

Перелік питань, що підлягають дослідженню:

- Аналіз предметної області та системне проєктування
- Моделювання інформаційних потоків
- Об'єктно-орієнтований аналіз та архітектурне проєктування

Дата видачі завдання «25» грудня 2025 р.

Керівник бакалаврської кваліфікаційної роботи _____ **Сергій Саяпін**
(підпис)

Завдання взяв до виконання _____ **Володимир Бортник**
(підпис)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

1. API: програмний інтерфейс прикладної системи.
2. BPMN: нотація моделювання бізнес-процесів.
3. CSS: каскадні таблиці стилів.
4. CSV: формат табличних даних.
5. DB: база даних.
6. DFD: діаграма потоків даних.
7. ERP: система планування ресурсів підприємства.
8. HTML: мова гіпертекстової розмітки.
9. HTTP: протокол передавання гіпертексту.
10. HTTPS: захищений протокол передавання гіпертексту.
11. HR: управління персоналом.
12. HRIS: інформаційна система управління персоналом.
13. ID: унікальний ідентифікатор запису або об’єкта.
14. KPI: ключовий показник ефективності.
15. L&D: навчання та розвиток персоналу.
16. LMS: система управління навчанням.
17. OAuth2: протокол авторизації користувачів.
18. REST: архітектурний підхід до побудови веб-сервісів.
19. SAML: стандарт обміну даними автентифікації та авторизації.
20. SCORM: стандарт організації електронного навчального контенту.
21. SMS: сервіс коротких текстових повідомлень.
22. SQL: мова структурованих запитів до бази даних.
23. SQLite: вбудована реляційна система керування базами даних.
24. SSO: технологія єдиного входу користувача.
25. UI: користувацький інтерфейс.
26. UML: уніфікована мова моделювання.
27. URL: уніфікований покажчик ресурсу.

28. Web UI: веб-інтерфейс користувача.
29. БД: база даних.
30. ІС: інформаційна система.
31. ПЗ: програмне забезпечення.
32. СКН: система корпоративного навчання.
33. СУБД: система управління базами даних.

ВСТУП

В умовах цифрової трансформації бізнес-середовища та переходу організацій до економіки знань особливої актуальності набуває розвиток систем корпоративного навчання, орієнтованих на безперервне підвищення професійних компетентностей персоналу. Динамічні зміни ринку праці, поява нових технологій і зростання вимог до кваліфікації працівників зумовлюють необхідність упровадження ефективних інструментів управління навчальними процесами в межах підприємств і організацій. За цих умов традиційні форми навчання не завжди забезпечують достатню гнучкість, масштабованість та оперативність оновлення навчального контенту, що знижує їх ефективність у корпоративному середовищі.

Використання сучасних веб-технологій створює передумови для побудови інформаційних систем корпоративного навчання нового покоління, які забезпечують централізоване управління навчальними матеріалами, автоматизацію процесів призначення та проходження курсів, контроль навчальних результатів і формування аналітичної звітності. Веб-орієнтований підхід дозволяє реалізувати доступ до навчальних ресурсів незалежно від місця та часу, підтримувати різні ролі користувачів (співробітників, викладачів, менеджерів, адміністраторів), а також інтегрувати систему з іншими корпоративними інформаційними сервісами.

Зростання обсягів навчального контенту, різноманітність форматів навчання та необхідність персоналізації освітніх траєкторій працівників зумовлюють потребу в автоматизованих засобах моніторингу навчальної активності та результатів корпоративного навчання. Сучасні системи корпоративного навчання мають не лише фіксувати факт проходження курсів, а й забезпечувати аналіз прогресу користувачів, формування рекомендацій щодо подальшого навчання та підтримку управлінських рішень у сфері розвитку персоналу.

Метою кваліфікаційної роботи є проєктування та розробка системи корпоративного навчання на основі веб-технологій, яка забезпечує автоматизоване управління навчальними процесами, моніторинг результатів навчання та підтримку прийняття рішень щодо розвитку професійних компетентностей персоналу.

Для досягнення поставленої мети у роботі необхідно розв’язати такі **завдання:**

1. провести аналіз предметної області корпоративного навчання та існуючих програмних рішень;
2. сформувати функціональні, нефункціональні та безпекові вимоги до системи корпоративного навчання;
3. розробити архітектуру програмного забезпечення з урахуванням принципів модульності, масштабованості та веб-орієнтованості;
4. побудувати UML-моделі основних бізнес-процесів і компонентів системи;
5. реалізувати програмний прототип веб-системи корпоративного навчання;
6. здійснити тестування та оцінку ефективності функціонування розробленої системи.

Об’єктом дослідження є процес організації та управління корпоративним навчанням персоналу в організаціях.

Предметом дослідження є методи, моделі та програмні засоби побудови веб-орієнтованих інформаційних систем корпоративного навчання.

Наукова новизна роботи полягає у розробленні підходу до побудови системи корпоративного навчання, що поєднує веб-орієнтовану архітектуру, рольову модель доступу та механізми моніторингу навчальної активності в єдиному програмному рішенні. Запропоновано інтеграцію основних підсистем управління курсами, навчальним процесом і аналітикою в рамках єдиної

інформаційної платформи, що підвищує ефективність управління навчальними процесами в корпоративному середовищі.

Методи дослідження ґрунтуються на застосуванні системного аналізу предметної області, UML-моделювання для опису процесів і структури системи, а також методів об’єктно-орієнтованого проєктування при розробленні програмного забезпечення. Для реалізації системи використано клієнт–серверну архітектуру, веб-технології та REST-підхід до організації взаємодії між компонентами.

Практична значущість одержаних результатів полягає у можливості використання розробленої системи корпоративного навчання в організаціях різного типу для автоматизації процесів навчання персоналу, контролю результатів і формування аналітичної звітності. Система може бути інтегрована з існуючими корпоративними інформаційними системами та використана для підвищення ефективності управління розвитком персоналу.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

У межах предметної області система корпоративного навчання розглядається як веб-орієнтований програмно-інформаційний комплекс, призначений для автоматизованої підтримки процесів навчання персоналу, управління навчальним контентом, моніторингу результатів проходження курсів та формування аналітичної звітності. Система забезпечує централізований збір, зберігання й оброблення даних щодо навчальної активності співробітників, що створює передумови для підвищення ефективності управління розвитком професійних компетентностей у межах організації.

Функціональне призначення системи полягає в автоматизації ключових бізнес-процесів корпоративного навчання, зокрема призначення та проходження курсів, контролю виконання навчальних планів, оцінювання результатів навчання і формування сертифікатів. Система підтримує рольову модель доступу, що охоплює співробітників, викладачів (авторів курсів), менеджерів з навчання та адміністраторів, а також забезпечує інтеграцію з зовнішніми корпоративними інформаційними системами, такими як HRIS, системи єдиного входу (SSO) та сервіси сповіщень. Це дозволяє підтримувати актуальність даних і узгодженість навчальних процесів із кадровими та управлінськими процесами організації.

На рис. 1.1 наведено DFD-діаграму 0-го рівня системи корпоративного навчання, яка відображає взаємодію зовнішніх сутностей (Співробітник, Викладач/Автор, Менеджер/HR, Адміністратор, зовнішні корпоративні системи) з центральним процесом «0 Система корпоративного навчання». Основними потоками даних є навчальний контент, запити на призначення курсів, результати проходження навчання, сертифікати, аналітичні звіти та дані синхронізації з зовнішніми сервісами.

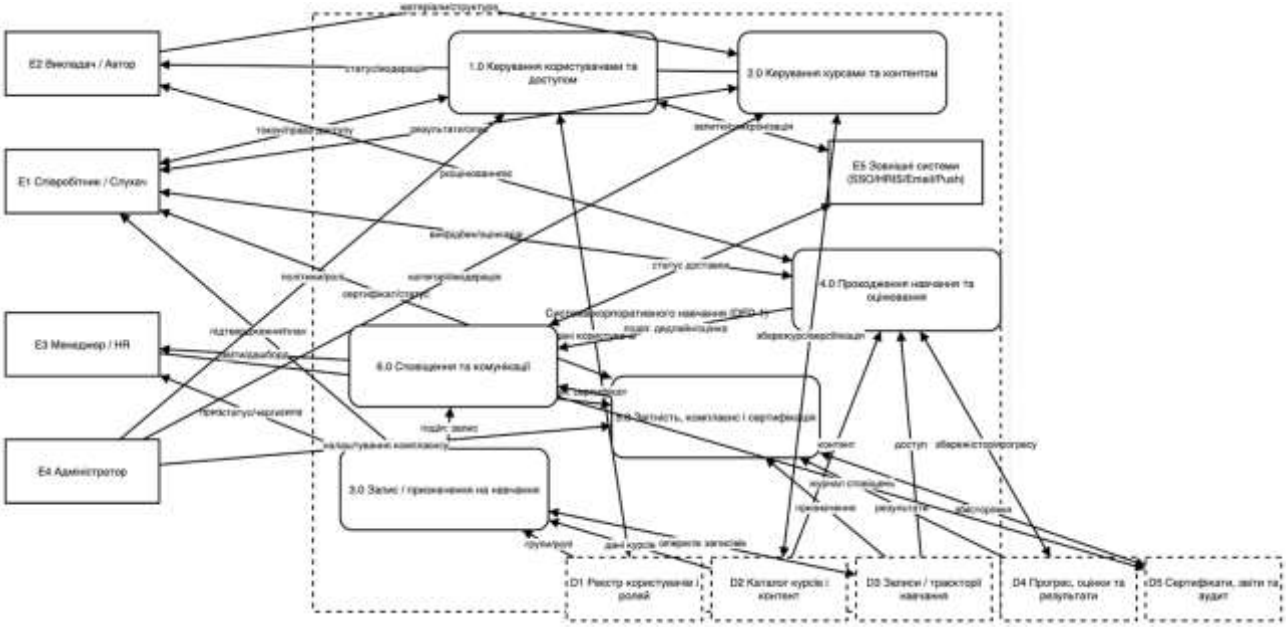

Рисунок 1.1 – DFD 0-рівня системи корпоративного навчання

Для деталізації функціональних процесів на рис. 1.2 подано BPMN-діаграму, що формалізує бізнес-процеси взаємодії користувачів із системою корпоративного навчання. Менеджер з навчання ініціює процес призначення курсу або навчального плану, після чого система перевіряє доступність курсу та права користувача. У разі успішної перевірки співробітнику надсилається сповіщення з доступом до навчального контенту. Співробітник проходить модулі курсу та складає підсумковий тест, результати якого фіксуються системою. За умови успішного завершення навчання система формує сертифікат і надає аналітичну інформацію менеджеру щодо результатів проходження курсу.

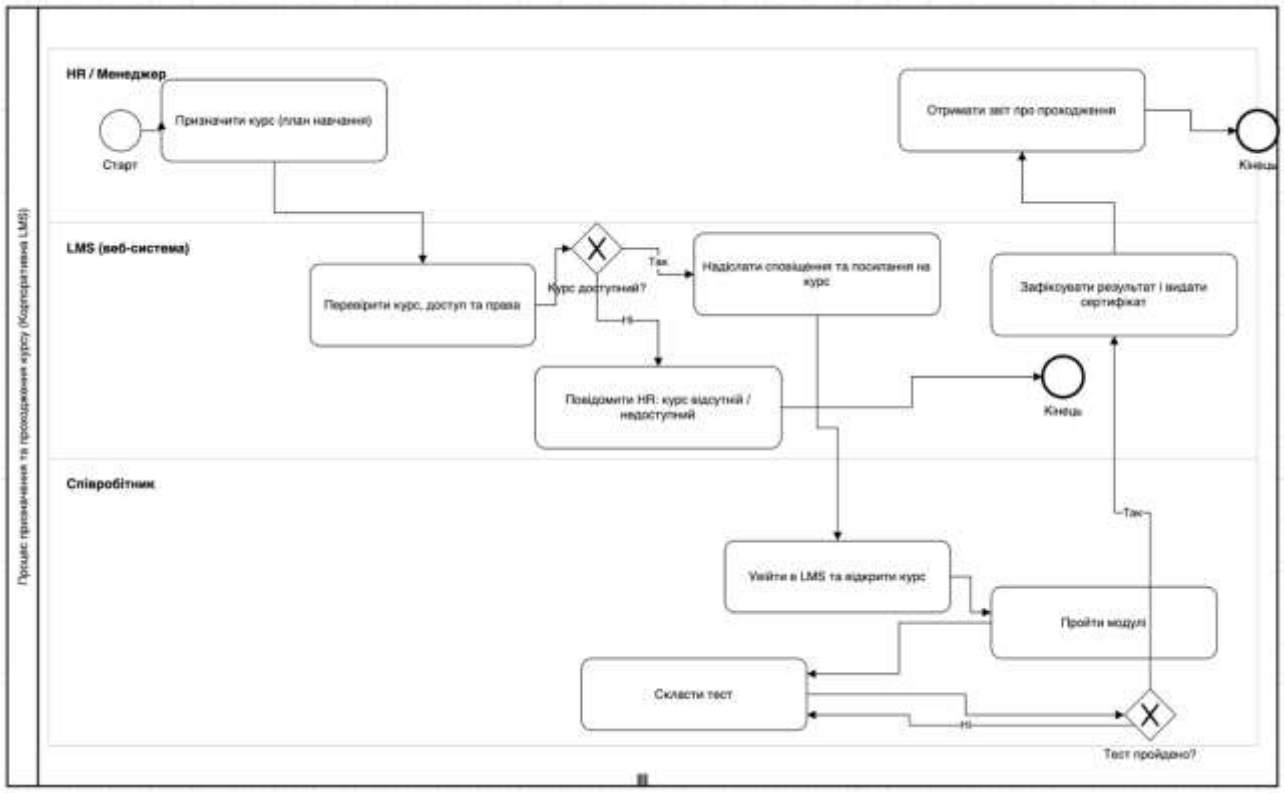

Рисунок 1.2 – BPMN-діаграма процесів системи корпоративного навчання

Для формалізації предметної області у табл. 1.1 наведено основні сутності системи корпоративного навчання, їх функціональне призначення та типи оброблюваних даних.

Таблиця 1.1

Основні сутності предметної області системи корпоративного навчання

Сутність	Опис	Функціональне призначення	Типи даних
Співробітник	Кінцевий користувач системи, який проходить корпоративне навчання	Проходження курсів, виконання тестів, отримання сертифікатів	Облікові дані, результати навчання, статус проходження
Викладач / Автор	Користувач, відповідальний за створення навчального контенту	Розробка та оновлення курсів, модулів і тестів	Навчальні матеріали, метадані курсів

Продовження таблиці 1.1

Менеджер / HR	Користувач, що керує процесами навчання персоналу	Призначення курсів, контроль виконання, аналіз звітів	Дані про навчальні плани, аналітичні показники
Адміністратор	Користувач із повними правами доступу	Керування користувачами, ролями, інтеграціями та безпекою	Системні журнали, конфігураційні дані
Система корпоративного навчання	Центральна програмна підсистема	Управління навчальними процесами, зберігання та оброблення даних	Навчальні дані, результати, сертифікати
Зовнішні системи (SSO, HRIS)	Корпоративні інформаційні сервіси	Автентифікація, синхронізація кадрових даних	Дані користувачів, службова інформація
Бази даних D1–D5	Сховища інформації системи	D1 – користувачі і ролі; D2 – курси і контент; D3 – записи і траєкторії; D4 – результати; D5 – сертифікати і звіти	Реляційні та документні структури

Предметна область системи корпоративного навчання характеризується централізованим управлінням навчальними ресурсами, автоматизованим контролем проходження курсів і можливістю формування аналітичної інформації для підтримки управлінських рішень. Застосування веб-орієнтованого підходу та формалізованих моделей бізнес-процесів дозволяє підвищити прозорість навчальної діяльності, скоротити витрати на організацію навчання та забезпечити безперервний розвиток професійних компетентностей персоналу.

1.2 Аналіз існуючих рішень

Розвиток сучасних систем корпоративного навчання супроводжується активним формуванням інтегрованих веб-орієнтованих середовищ, що

поєднують управління навчальним контентом, користувачами, процесами навчання та аналітикою результатів. Такі системи орієнтовані на підтримку безперервного професійного розвитку персоналу, масштабованість для великих організацій і інтеграцію з корпоративними інформаційними сервісами. Серед найбільш поширених рішень у сфері електронного та корпоративного навчання слід виділити Moodle, Google Classroom, Canvas LMS, Blackboard Learn та Blended Platform, кожне з яких реалізує власний архітектурний підхід до автоматизації навчального циклу.

На рис. 1.3 подано інтерфейс системи Moodle, яка є однією з найпоширеніших платформ з відкритим програмним кодом для організації дистанційного та корпоративного навчання. Moodle побудована за клієнт–серверною веб-архітектурою та підтримує розширення функціональності за рахунок модульної системи плагінів. Платформа забезпечує створення курсів, тестування, форуми, базову аналітику успішності та інтеграцію зі сторонніми сервісами. Основними перевагами Moodle є масштабованість і широка екосистема, проте інтерфейс системи часто потребує додаткової адаптації для використання в корпоративному середовищі, а розширена аналітика вимагає встановлення додаткових модулів.

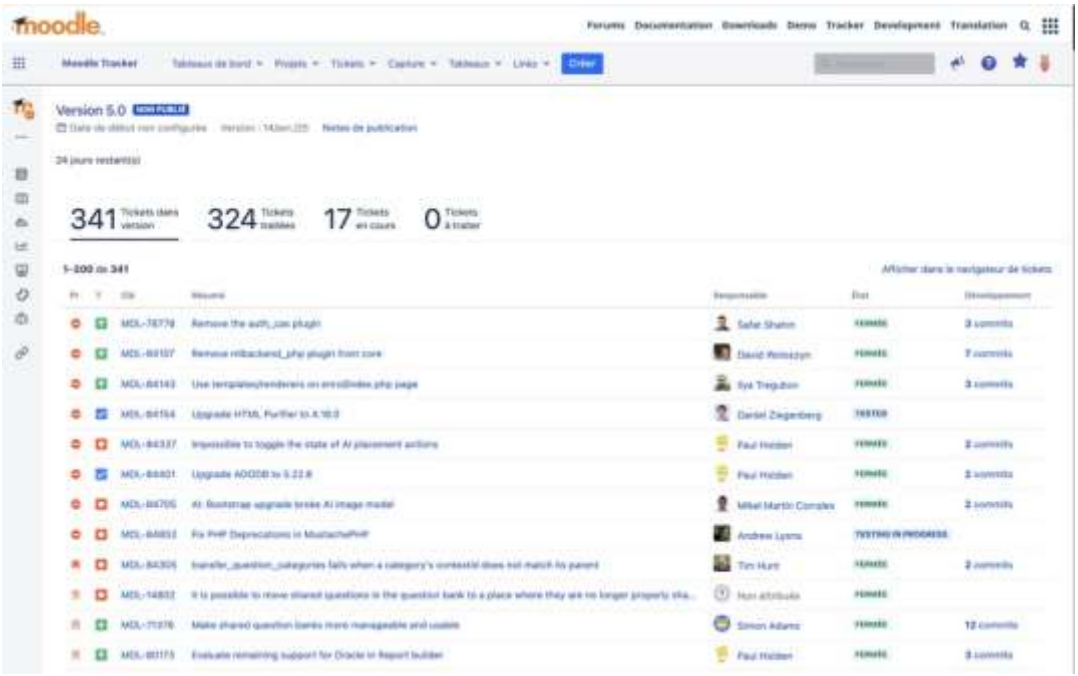

Рисунок 1.3 – Інтерфейс системи Moodle в середовищі корпоративного навчання

Наступним поширеним рішенням є Google Classroom, що реалізує хмарну модель взаємодії між користувачами та навчальним контентом із використанням сервісів Google Workspace. На рис. 1.4 зображено типову панель курсів Google Classroom, орієнтовану на простоту використання та швидке розгортання навчальних матеріалів. Система підтримує призначення завдань, комунікацію та базовий контроль виконання, однак має обмежені можливості аналітики й адаптації навчальних траєкторій, що знижує її ефективність у корпоративному навчанні з підвищеними вимогами до звітності та контролю компетентностей.

Рисунок 1.4 – Інтерфейс курсів Google Classroom

На рис. 1.5 представлено систему Canvas LMS, розроблену компанією Instructure. Canvas LMS характеризується сучасною веб-архітектурою з підтримкою REST API, стандартів SCORM та LTI, а також розвиненими інструментами створення навчальних модулів і тестів. Платформа має потужні засоби аналітики та візуалізації прогресу користувачів, що робить її придатною для корпоративного навчання. Водночас її впровадження потребує стабільної серверної інфраструктури та значних ресурсів, що може ускладнювати використання в невеликих організаціях.

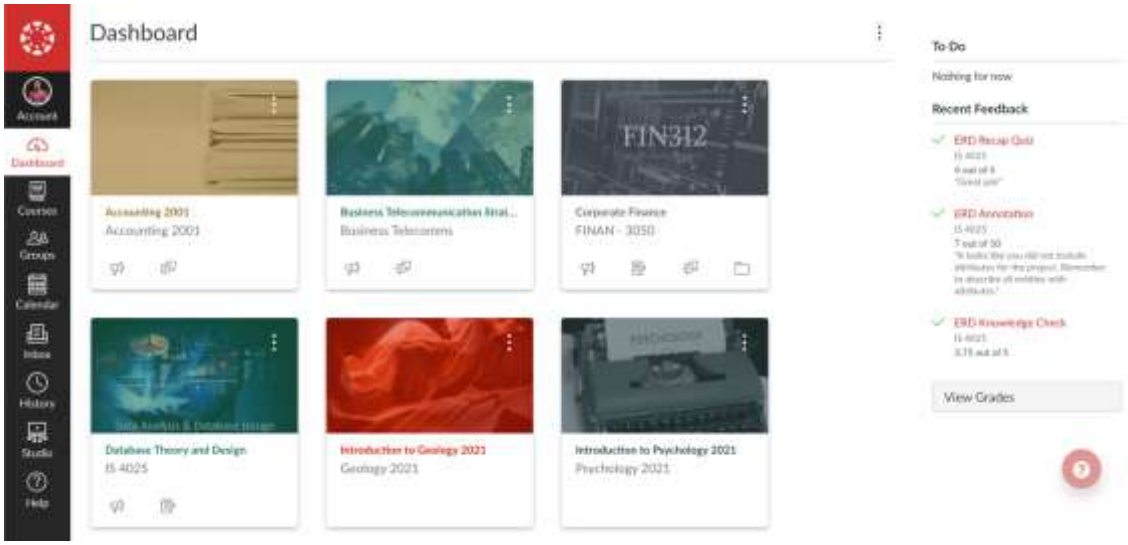

Рисунок 1.5 – Інтерфейс модулів і завдань у Canvas LMS

Система Blackboard Learn, наведена на рис. 1.6, орієнтована на централізоване управління навчальними процесами з акцентом на адміністрування, аналітику та контроль. Blackboard широко застосовується як в академічному, так і в корпоративному середовищі завдяки розвиненим засобам звітності, управління курсами та ролями користувачів. Основними недоліками платформи є складність користувацького інтерфейсу та висока вартість ліцензій, що обмежує її доступність для середніх і малих компаній.

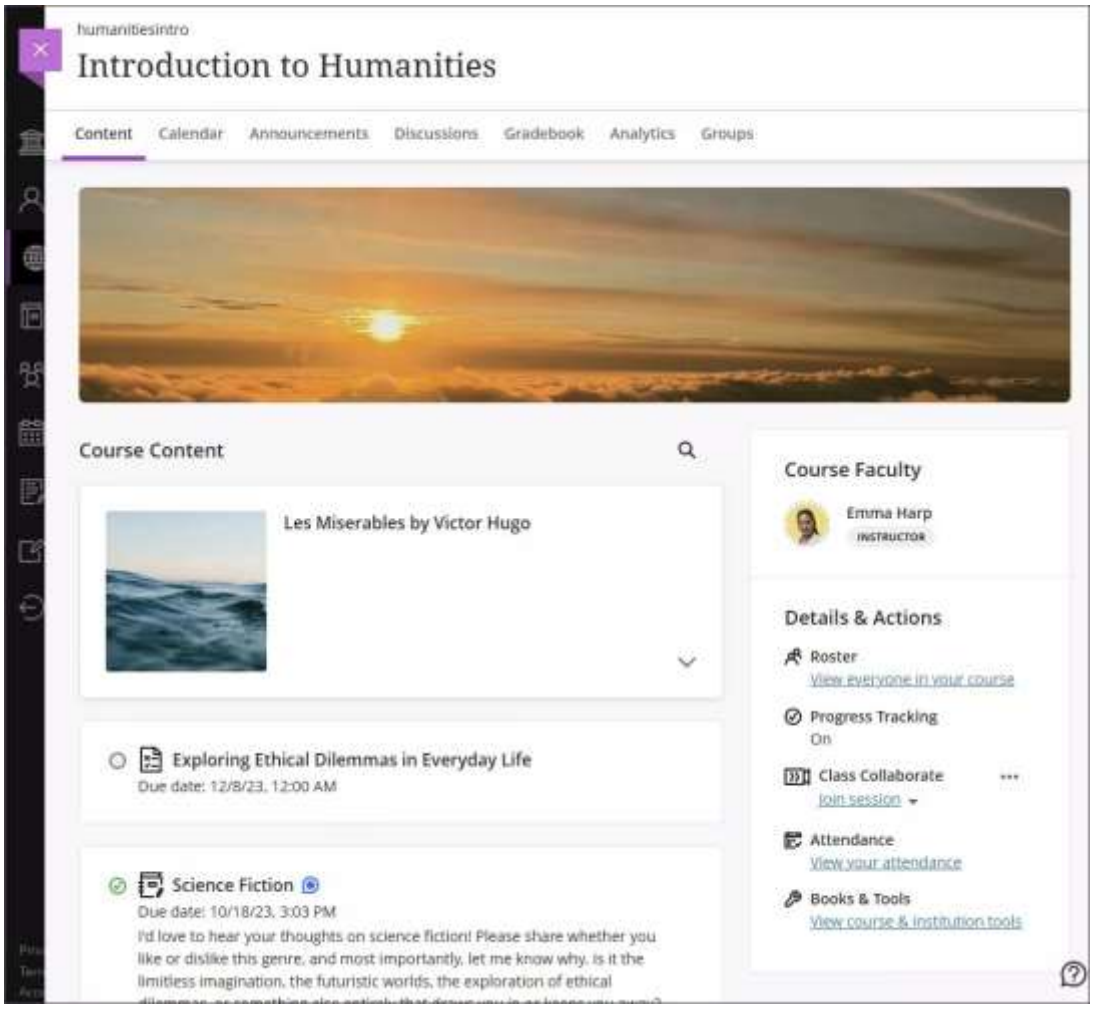

Рисунок 1.6 – Навчальний курс і аналітичні інструменти в Blackboard Learn

Платформа Blend-ed Platform, зображена на рис. 1.7, є сучасним SaaS-рішенням, орієнтованим переважно на корпоративних клієнтів. Архітектура системи базується на мікросервісному підході з підтримкою REST API, що забезпечує гнучке масштабування та інтеграцію з HR-системами. Платформа реалізує елементи гейміфікації, персоналізовані рекомендації та розширену аналітику навчальної активності. Водночас залежність від хмарної інфраструктури та обмежені можливості локального налаштування можуть бути критичними для організацій із підвищеними вимогами до контролю даних.

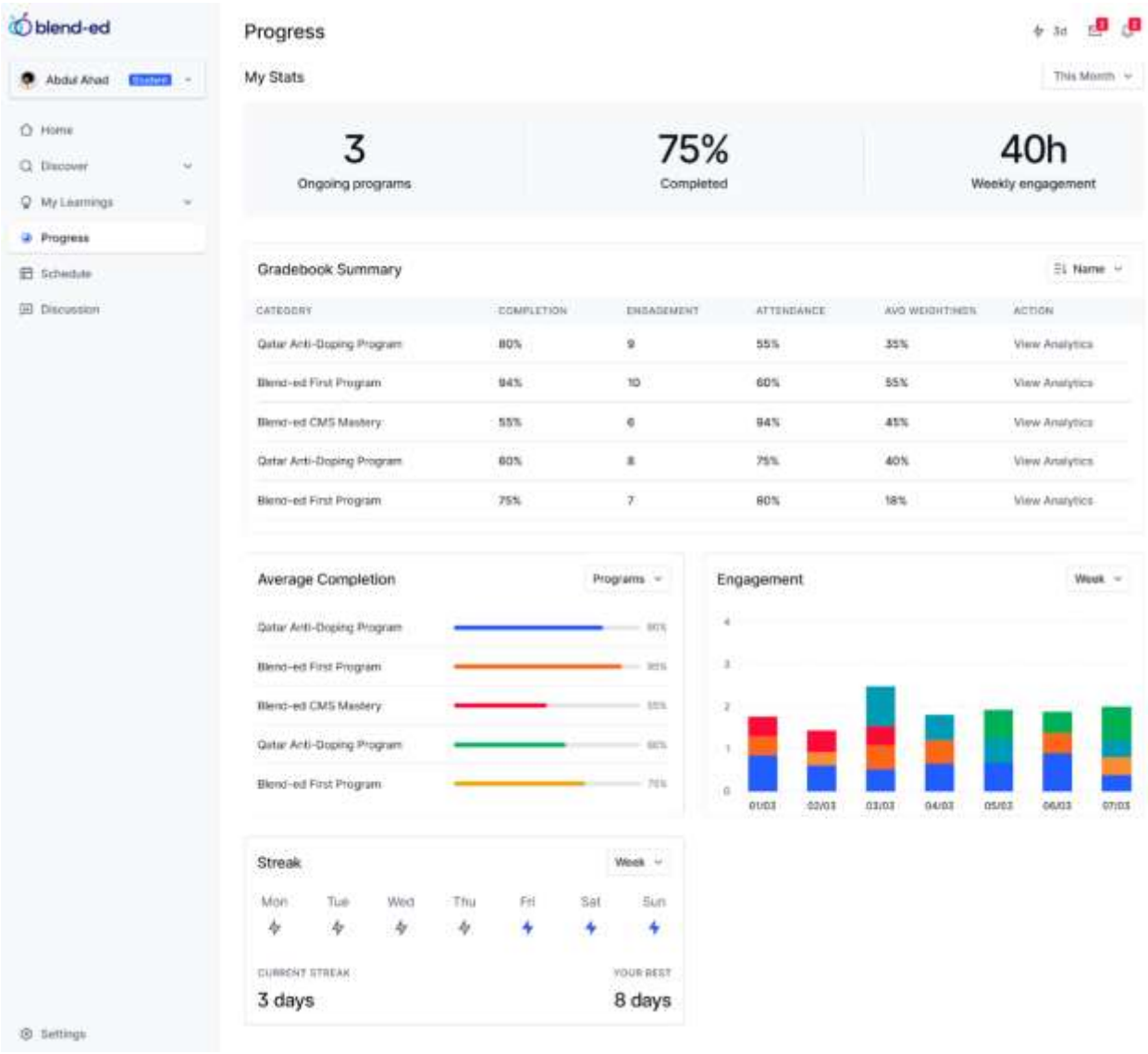

Рисунок 1.7 – Персоналізована аналітична панель користувача в Blend-ed Platform

Порівняльний аналіз існуючих систем корпоративного навчання наведено в табл. 1.2. Аналіз показує, що більшість рішень орієнтована на веб- або хмарну архітектуру з фокусом на браузерну взаємодію та інтеграцію з корпоративними сервісами.

Таблиця 1.2

Порівняння існуючих систем корпоративного навчання та розроблюваної веб-системи

Система	Тип архітектури	Адаптивність навчання	Аналітика	Інтеграція з HR/SSO	Призначення
Moodle	Веб / клієнт–сервер	Обмежена (плагіни)	Базова / розширювана	Часткова	Освітні та корпоративні середовища
Google Classroom	Хмарна	Обмежена	Обмежена	Через Google Workspace	Навчальні курси
Canvas LMS	Веб / REST API	Розширена	Потужна	Так	Університети, корпоративне навчання
Blackboard Learn	Централізована	Розширена	Потужна	Так	Великі організації
Blend-ed Platform	SaaS / мікросервісна	Розширена	Потужна	Так	Корпоративне навчання
Веб-система (розробка)	Веб / клієнт–сервер	Розширена	Вбудована	Так	Корпоративне навчання

Результати аналізу свідчать, що існуючі системи корпоративного навчання здебільшого орієнтовані на універсальні сценарії використання та не завжди забезпечують гнучке налаштування навчальних процесів відповідно до специфіки організації. У межах даної роботи запропоновано підхід до розробки веб-орієнтованої системи корпоративного навчання, яка поєднує керування курсами, навчальними траєкторіями та аналітичними модулями в єдиній архітектурі.

1.3 Моделювання програмної системи

Моделювання предметної області системи корпоративного навчання спрямоване на формалізацію основних процесів, ролей користувачів та взаємодії між компонентами системи з метою подальшого проєктування її архітектури та програмної реалізації. Для цього застосовано UML-моделювання, яке дозволяє наочно відобразити функціональні можливості системи, сценарії взаємодії користувачів і логіку виконання бізнес-процесів корпоративного навчання.

На рис. 1.8 наведено UML-діаграму прецедентів системи корпоративного навчання, яка відображає основних акторів і їх взаємодію з функціональними можливостями системи. До акторів належать співробітник (слухач), викладач (автор навчального контенту), менеджер з навчання (L&D/HR), адміністратор, а також зовнішні інформаційні системи, зокрема сервіси автентифікації (SSO), HRIS/ERP та сервіси сповіщень. Діаграма прецедентів демонструє, що система забезпечує автентифікацію користувачів, перегляд каталогу курсів, запис і призначення навчання, проходження навчальних матеріалів, тестування, отримання сертифікатів, а також керування користувачами, ролями та навчальним контентом. Окремо виділено сценарії з обмеженим доступом до курсів, для яких передбачено механізм погодження з боку менеджера з навчання.

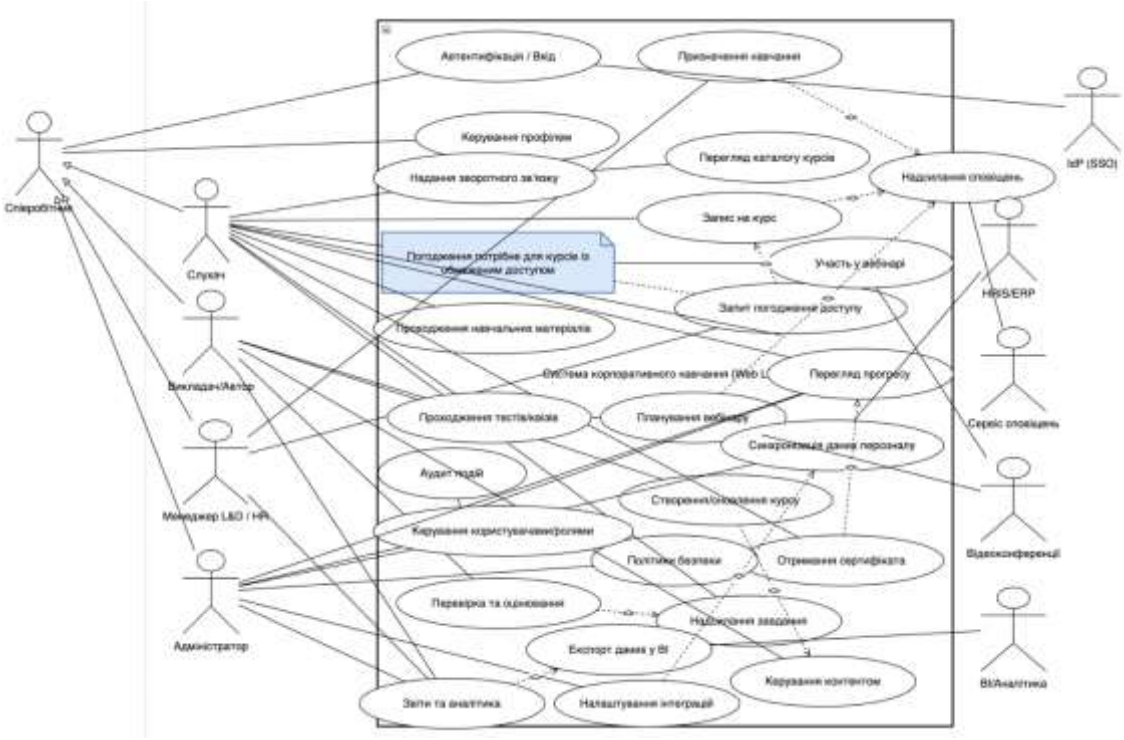

Рисунок 1.8 – UML-діаграма прецедентів системи корпоративного навчання

Для деталізації динамічної взаємодії між компонентами системи на рис. 1.9 подано UML-діаграму послідовності, що описує типовий сценарій запису співробітника на курс, його проходження та отримання сертифіката. Діаграма відображає обмін повідомленнями між користувачем, веб-клієнтом системи, сервісом автентифікації, сервісами курсів, запису, навчання і тестування, сертифікації, сховищем даних та сервісом сповіщень. Послідовність повідомлень ілюструє логіку автентифікації через SSO, отримання даних курсу, створення запису на навчання, фіксацію результатів тестування та генерацію сертифіката з подальшим інформуванням користувача.

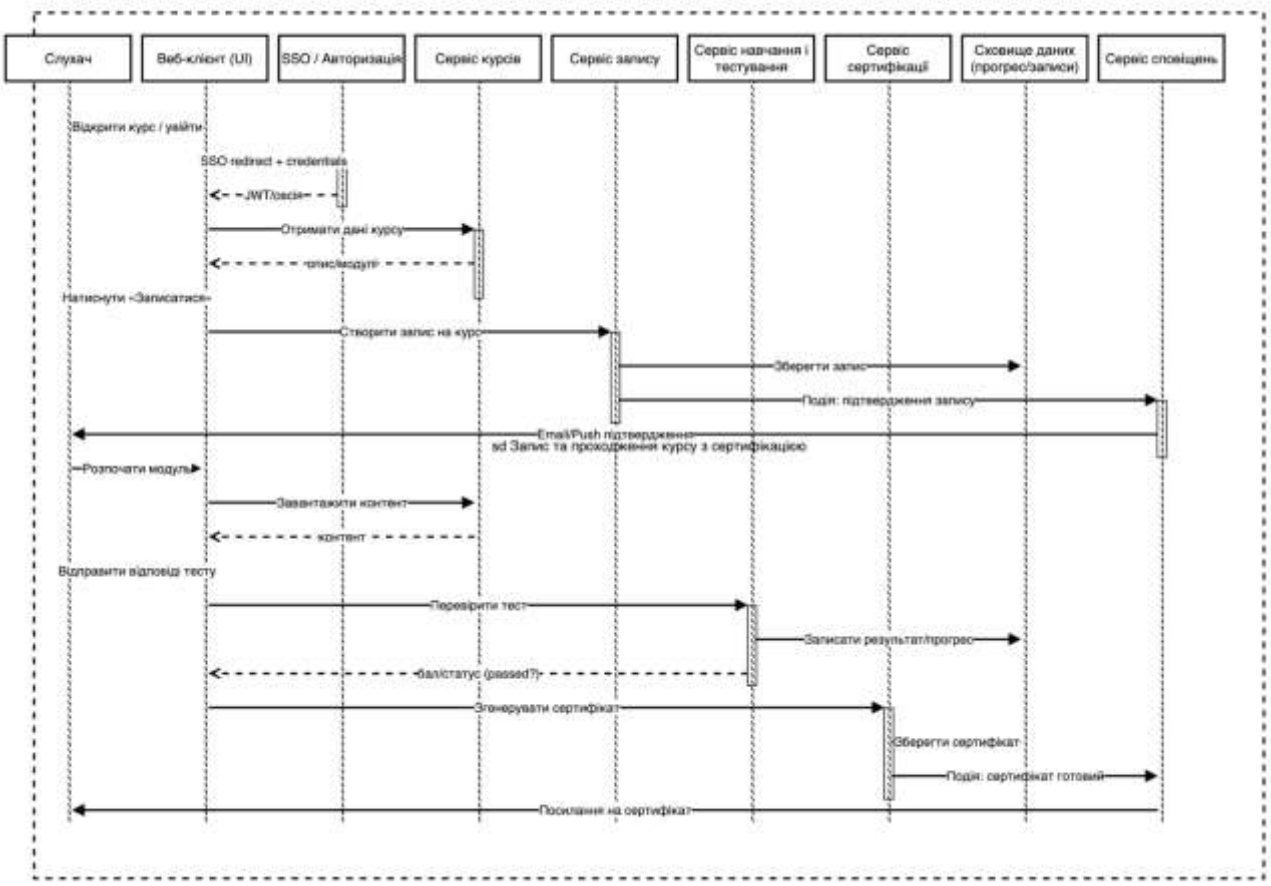

Рисунок 1.9 – UML-діаграма послідовності процесу запису та проходження курсу

Логіку виконання бізнес-процесу корпоративного навчання відображено на рис. 1.10 у вигляді UML-діаграми активності з використанням партицій. Діаграма структурована за ролями «Слухач», «Система» та «Менеджер/HR», що дозволяє чітко розмежувати відповідальність учасників процесу. Процес починається з автентифікації користувача та вибору курсу, після чого система перевіряє правила доступу та вимоги курсу. У разі необхідності погодження система надсилає відповідний запит менеджеру. Після підтвердження створюється запис на курс, забезпечується проходження навчальних матеріалів і складання тесту. За результатами оцінювання передбачено можливість перездачі або генерації сертифіката про успішне завершення навчання.

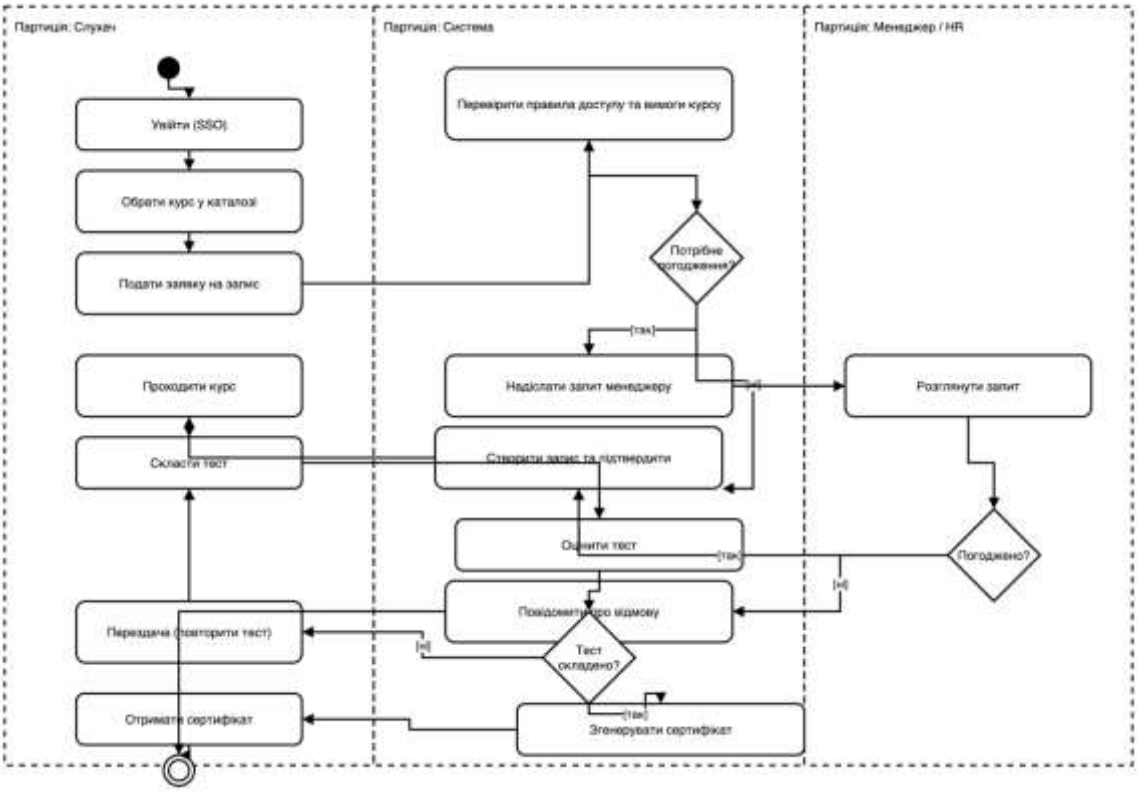

Рисунок 1.10 – UML-діаграма активності процесу корпоративного навчання

Застосування UML-діаграм прецедентів, послідовності та активності дозволяє комплексно описати предметну область системи корпоративного навчання, відобразити її функціональні можливості, сценарії взаємодії користувачів і внутрішню логіку виконання основних бізнес-процесів. Отримані моделі створюють формальну основу для подальшого проєктування архітектури програмного забезпечення та реалізації окремих функціональних модулів системи.

1.4 Аналіз вимог інформаційної системи

Проєктована система розглядається як веб-орієнтований програмний продукт, що функціонує в клієнт–серверному середовищі та забезпечує доступ користувачів через браузер або корпоративний веб-клієнт. Такий підхід зумовлює вимоги до стабільної роботи серверної частини, швидкодії інтерфейсу, масштабованості та безпечної обробки навчальних і персональних даних

співробітників. Система орієнтована на інтеграцію з корпоративними сервісами автентифікації та кадровими інформаційними системами, що забезпечує узгодженість навчальних процесів із загальною IT-інфраструктурою організації.

Функціональні вимоги визначають перелік дій і операцій, які система корпоративного навчання повинна реалізовувати для підтримки користувачів різних ролей. Основними користувачами системи є співробітник (слухач), викладач або автор навчального контенту, менеджер з навчання (L&D/HR) та адміністратор. Система повинна забезпечувати автентифікацію користувачів, керування навчальними курсами, запис і призначення навчання, контроль проходження курсів, оцінювання результатів та формування аналітичних звітів. Узагальнений перелік функціональних вимог, сформований на основі UML-моделей предметної області, наведено в таблиці 1.3.

Таблиця 1.3

Функціональні вимоги системи корпоративного навчання

№	Вимога	Опис	Роль користувача
1	Автентифікація користувачів	Вхід до системи з використанням корпоративного облікового запису або SSO	Усі користувачі
2	Перегляд каталогу курсів	Отримання інформації про доступні навчальні програми та модулі	Співробітник
3	Запис і призначення навчання	Створення запису на курс або призначення навчального плану	Система, менеджер
4	Проходження навчання	Доступ до навчальних матеріалів і модулів курсу	Співробітник
5	Тестування та оцінювання	Перевірка знань і фіксація результатів	Система
6	Формування сертифікатів	Генерація сертифікатів про завершення навчання	Система
7	Аналітика та звітність	Формування звітів про результати навчання персоналу	Менеджер, адміністратор
8	Керування користувачами	Створення, редагування та керування ролями	Адміністратор
9	Сповіщення користувачів	Надсилання повідомлень про події навчального процесу	Система

Нефункціональні вимоги визначають якісні характеристики системи, що впливають на стабільність її роботи, зручність використання та можливість масштабування. З огляду на веб-орієнтований характер програмного забезпечення основний акцент зроблено на продуктивності серверної частини, надійності зберігання даних, сумісності з різними платформами та можливості інтеграції з корпоративними інформаційними системами. Узагальнення нефункціональних вимог наведено в таблиці 1.4.

Таблиця 1.4

Нефункціональні вимоги системи корпоративного навчання

№	Категорія	Вимога	Показник
1	Продуктивність	Час відгуку веб-інтерфейсу	≤ 1 с
2	Надійність	Доступність системи	≥ 99,5 %
3	Масштабованість	Підтримка зростання кількості користувачів	Передбачено
4	Сумісність	Робота в сучасних браузерах	Повна
5	Інтерфейс	Зручність та адаптивність UI	Висока
6	Інтеграція	Підключення HRIS, SSO, сервісів сповіщень	Передбачено

Окрему групу становлять вимоги до безпеки, оскільки система корпоративного навчання оперує персональними даними співробітників і результатами їх навчальної діяльності. Система повинна забезпечувати захист інформації від несанкціонованого доступу, контроль прав користувачів і збереження цілісності навчальних даних. Основні вимоги до безпеки подано в таблиці 1.5.

Таблиця 1.5

Вимоги до безпеки системи корпоративного навчання

№	Вимога	Опис
1	Контроль доступу	Автентифікація та авторизація користувачів
2	Рольова модель	Розмежування прав доступу за ролями
3	Захист даних	Використання захищених каналів передачі
4	Журнали подій	Реєстрація дій користувачів і системних подій
5	Цілісність даних	Запобігання несанкціонованим змінам
6	Захист звітів	Обмеження доступу до аналітичної інформації

Проведений аналіз вимог дозволяє сформувати узгоджену специфікацію системи корпоративного навчання, орієнтованої на автоматизацію навчальних процесів і підтримку управління розвитком персоналу. Запропонована система зосереджена на ефективному управлінні навчальними курсами, контролі результатів та формуванні аналітичної інформації для менеджерів. Використання веб-технологій забезпечує доступність системи, масштабованість і можливість інтеграції з корпоративною IT-інфраструктурою, що створює передумови для її практичного впровадження в умовах реального корпоративного середовища.

1.5 Постановка завдання

На основі проведеного аналізу предметної області, результатів моделювання процесів корпоративного навчання та сформованих вимог визначається завдання розроблення системи корпоративного навчання на основі веб-технологій. Розроблювана система повинна забезпечувати автоматизацію основних процесів корпоративного навчання, зокрема управління навчальними курсами, контролю проходження навчання персоналом, оцінювання результатів і формування аналітичної інформації для підтримки управлінських рішень у межах організації.

У межах кваліфікаційної роботи необхідно виконати формалізацію бізнес-процесів корпоративного навчання з використанням UML-моделей, спроєктувати структуру веб-орієнтованої інформаційної системи з урахуванням клієнт–серверної архітектури та реалізувати механізми автентифікації й авторизації користувачів на основі рольової моделі доступу. Система повинна підтримувати керування навчальними курсами, модулями та навчальними матеріалами, забезпечувати процеси запису й призначення навчання, проходження курсів, тестування та формування сертифікатів про завершення навчання. Окрему увагу необхідно приділити реалізації аналітичної підсистеми, що забезпечує моніторинг результатів навчання персоналу та формування

зведених звітів, а також інтеграції із зовнішніми корпоративними сервісами, зокрема системами єдиного входу, кадровими інформаційними системами та сервісами сповіщень. Завершальним етапом є тестування функціонування системи та перевірка її відповідності сформованим вимогам.

У процесі функціонування система корпоративного навчання оперує визначеним набором вхідних і вихідних даних. До вхідних даних належать облікові дані користувачів, інформація про співробітників, їх ролі та належність до організаційних підрозділів, навчальні курси, модулі та матеріали, параметри навчальних програм і правил доступу, результати тестування та дії користувачів у процесі проходження навчання, а також службові дані, що надходять із зовнішніх корпоративних інформаційних систем. Вихідними даними системи є інформація про записи співробітників на курси та статус їх проходження, результати оцінювання знань, сформовані сертифікати про успішне завершення навчання, аналітичні звіти та узагальнені показники навчальної активності персоналу, а також повідомлення користувачам про події навчального процесу.

Сформульована постановка завдання визначає функціональні межі та логіку роботи системи корпоративного навчання, склад оброблюваних даних і напрям подальшого проєктування архітектури програмного забезпечення, що створює основу для реалізації програмного прототипу та його практичного застосування в корпоративному середовищі.

2 ПРОЄКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ КОРПОРАТИВНОГО НАВЧАННЯ

2.1 Логічна модель даних системи корпоративного навчання

Логічна модель даних системи корпоративного навчання розроблена з метою формалізації структури інформації, яка використовується під час реєстрації користувачів, керування навчальними курсами, призначення навчання, проходження модулів, перевірки знань і формування сертифікатів. Побудована модель відображає основні інформаційні об’єкти системи, їх атрибути та зв’язки між ними. Вона є основою для подальшого переходу до фізичної моделі бази даних і реалізації таблиць у вибраній системі керування базами даних.

Логічна модель даних наведена на рисунку 2.1. У моделі виділено сім основних сутностей: Role, User, Course, CourseModule, Enrollment, TestResult та Certificate. Такий склад сутностей є достатнім для реалізації базового функціоналу веб-системи корпоративного навчання та не перевантажує модель другорядними об’єктами. Основний акцент зроблено на збереженні даних про користувачів, ролі, навчальні курси, модулі курсів, факти запису на навчання, результати тестування та сертифікати.

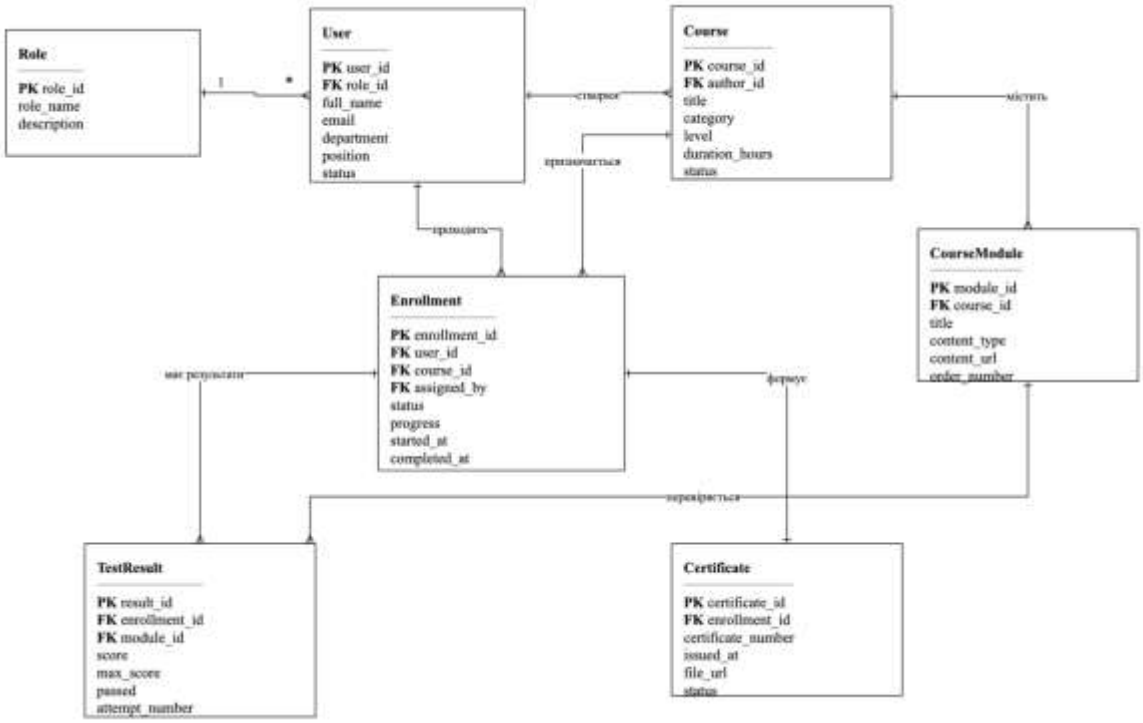

Рисунок 2.1 – Логічна модель даних системи корпоративного навчання

Центральною сутністю моделі є User, яка використовується для зберігання інформації про всіх користувачів системи. До таких користувачів належать співробітники, автори навчального контенту, менеджери з навчання та адміністратори. Сутність User містить первинний ключ user_id, зовнішній ключ role_id, а також атрибути full_name, email, department, position і status. Атрибут role_id пов’язує користувача з відповідною роллю в системі, що дає змогу реалізувати розмежування прав доступу.

Сутність Role призначена для опису ролей користувачів. Вона містить первинний ключ role_id, назву ролі role_name та опис description. Через зв’язок Role - User одна роль може бути призначена багатьом користувачам, тоді як кожен користувач належить до однієї основної ролі. Такий підхід забезпечує просту й зрозумілу рольову модель доступу, яка відповідає вимогам системи корпоративного навчання.

Навчальний контент у моделі представлено сутностями Course та CourseModule. Сутність Course описує навчальний курс і містить первинний ключ course_id, зовнішній ключ author_id, назву курсу title, категорію category,

рівень складності level, тривалість duration_hours і статус status. Зовнішній ключ author_id вказує на користувача, який створює або супроводжує курс. Це дозволяє пов’язати навчальний матеріал із конкретним автором або викладачем.

Сутність CourseModule деталізує структуру курсу. Один курс може містити декілька навчальних модулів, що відображено зв’язком Course - CourseModule. Сутність CourseModule має первинний ключ module_id, зовнішній ключ course_id, назву модуля title, тип навчального контенту content_type, посилання або шлях до матеріалу content_url та порядковий номер order_number. Атрибут order_number використовується для визначення послідовності проходження навчальних матеріалів у межах курсу.

Процес призначення або самостійного запису користувача на курс реалізується через сутність Enrollment. Вона є проміжною сутністю між User і Course та усуває зв’язок «багато до багатьох», оскільки один користувач може проходити багато курсів, а один курс може бути призначений багатьом користувачам. Сутність Enrollment містить первинний ключ enrollment_id, зовнішні ключі user_id, course_id і assigned_by, а також атрибути status, progress, started_at і completed_at. Атрибут assigned_by використовується для фіксації користувача, який призначив навчання, наприклад менеджера або адміністратора. Атрибут progress дозволяє відстежувати відсоток проходження курсу.

Результати перевірки знань зберігаються в сутності TestResult. Вона пов’язана із сутностями Enrollment та CourseModule, оскільки результат тестування залежить як від конкретного запису користувача на курс, так і від навчального модуля, за яким виконувалась перевірка. Сутність TestResult містить первинний ключ result_id, зовнішні ключі enrollment_id і module_id, оцінку score, максимальну кількість балів max_score, ознаку успішного проходження passed та номер спроби attempt_number. Така структура дає змогу зберігати історію спроб користувача та аналізувати якість засвоєння навчального матеріалу.

Після успішного завершення курсу система формує сертифікат, дані про який зберігаються в сутності Certificate. Вона містить первинний ключ certificate_id, зовнішній ключ enrollment_id, номер сертифіката certificate_number, дату видачі issued_at, посилання на файл file_url і статус status. Зв’язок між Enrollment і Certificate показує, що за одним записом на курс може бути сформований один сертифікат після виконання всіх умов проходження навчання.

Основні сутності логічної моделі даних наведено в таблиці 2.1.

Таблиця 2.1

Основні сутності логічної моделі даних

Сутність	Призначення	Основні атрибути
Role	Зберігання ролей користувачів системи	role_id, role_name, description
User	Облік користувачів системи корпоративного навчання	user_id, role_id, full_name, email, department, position, status
Course	Зберігання інформації про навчальні курси	course_id, author_id, title, category, level, duration_hours, status
CourseModule	Опис навчальних модулів у межах курсу	module_id, course_id, title, content_type, content_url, order_number
Enrollment	Фіксація запису користувача на курс і стану проходження	enrollment_id, user_id, course_id, assigned_by, status, progress, started_at, completed_at
TestResult	Зберігання результатів тестування за модулями	result_id, enrollment_id, module_id, score, max_score, passed, attempt_number
Certificate	Зберігання даних про сформовані сертифікати	certificate_id, enrollment_id, certificate_number, issued_at, file_url, status

Зв’язки між сутностями визначають логіку роботи системи. Сутність Role має зв’язок один до багатьох із User, оскільки одна роль може бути призначена багатьом користувачам. Сутність User пов’язана з Course через поле author_id, що відображає створення курсів авторами або викладачами. Сутність Course має зв’язок один до багатьох із CourseModule, тому що кожен курс складається з декількох навчальних модулів.

Зв’язок між User і Course реалізовано через сутність Enrollment. Це дозволяє зберігати не лише факт належності користувача до курсу, а й додаткові дані про стан проходження, прогрес, дату початку та дату завершення навчання. Сутність Enrollment також пов’язана із TestResult, оскільки кожен запис на курс може мати декілька результатів тестування. Крім того, Enrollment пов’язана із Certificate, що забезпечує формування сертифіката після успішного завершення курсу.

Розроблена логічна модель відповідає принципам нормалізації даних. Дані про ролі, користувачів, курси, модулі, результати тестування та сертифікати розділено на окремі сутності, що зменшує дублювання інформації та спрощує її супровід. Первинні ключі забезпечують однозначну ідентифікацію записів, а зовнішні ключі підтримують цілісність зв’язків між таблицями. Така структура дає змогу реалізувати основні функції системи корпоративного навчання, зокрема автентифікацію користувачів, керування курсами, призначення навчання, контроль проходження, оцінювання знань, формування сертифікатів і підготовку аналітичної інформації.

2.2 Діаграма класів і кооперації системи корпоративного навчання

Діаграма класів є одним з основних елементів об’єктно-орієнтованого проєктування програмної системи, оскільки вона відображає статичну структуру майбутнього програмного забезпечення, склад основних класів, їх атрибути, методи та зв’язки між об’єктами. Для системи корпоративного навчання діаграма класів побудована на основі логічної моделі даних, функціональних вимог і сценаріїв взаємодії користувачів із системою. Основну увагу приділено класам, які забезпечують керування користувачами, ролями, навчальними курсами, модулями, записами на навчання, результатами тестування та сертифікатами.

На рисунку 2.2 наведено UML-діаграму класів системи корпоративного навчання.

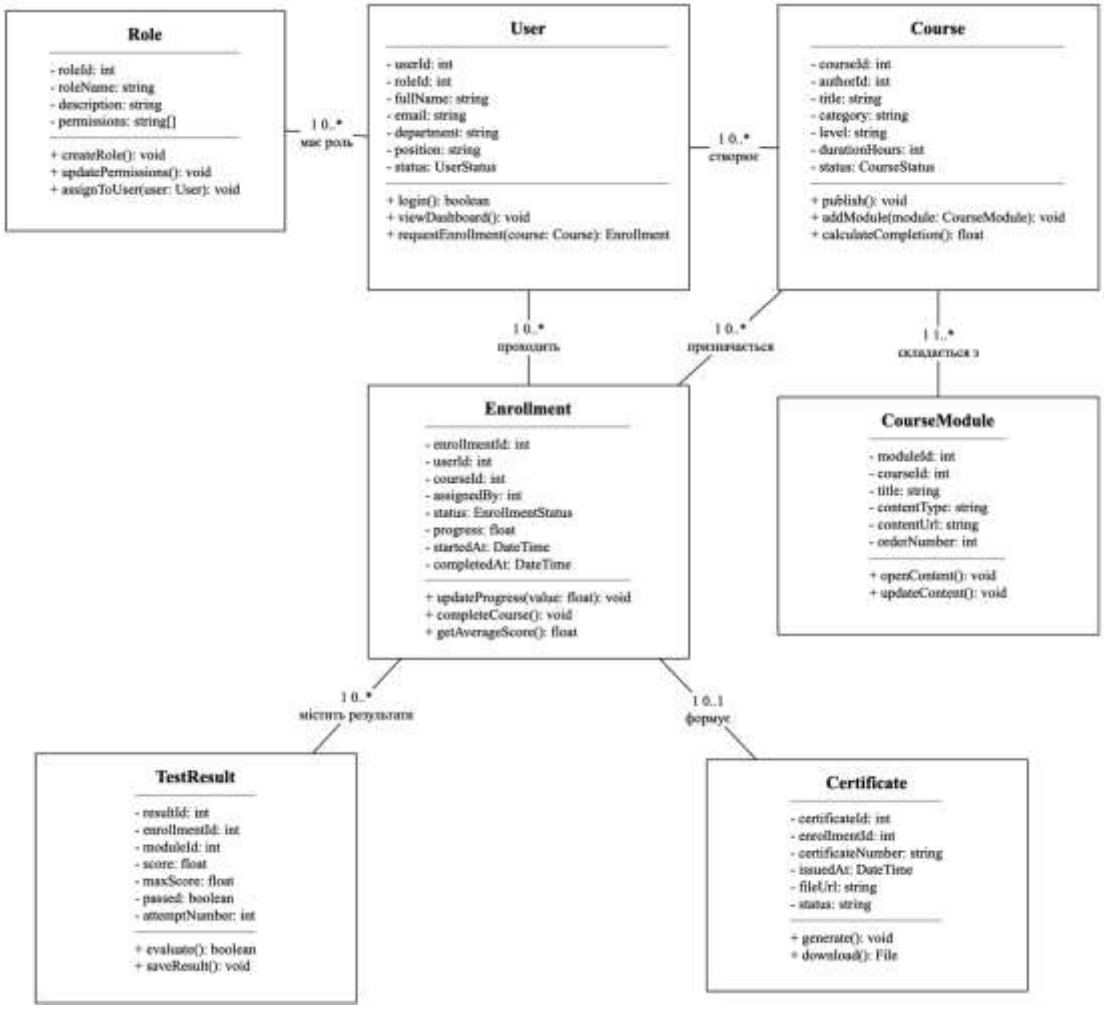

Рисунок 2.2 – UML-діаграма класів системи корпоративного навчання

У структурі діаграми виділено сім основних класів: Role, User, Course, CourseModule, Enrollment, TestResult та Certificate. Такий склад класів є достатнім для реалізації основних функціональних можливостей системи без надмірного ускладнення архітектури. Кожен клас відповідає окремій сутності предметної області та має власну зону відповідальності.

Клас Role призначений для опису ролей користувачів у системі. Він містить атрибути role_id, role_name та description. Через цей клас реалізується розмежування прав доступу між співробітником, автором курсу, менеджером з навчання та адміністратором. Наявність окремого класу Role дозволяє централізовано керувати правами користувачів і не дублювати інформацію про доступ у класі User.

Клас User є одним із центральних класів системи. Він описує користувача веб-платформи корпоративного навчання та містить атрибути user_id, role_id, full_name, email, department, position і status. Об’єкти цього класу можуть представляти різні категорії користувачів: співробітників, викладачів, менеджерів або адміністраторів. Клас User пов’язаний із класом Role, оскільки кожен користувач має визначену роль у системі. Крім того, User може бути автором навчального курсу або слухачем, який проходить призначене навчання.

Клас Course описує навчальний курс. Його основними атрибутами є course_id, author_id, title, category, level, duration_hours і status. Поле author_id вказує на користувача, який створив або супроводжує курс. Клас Course пов’язаний із класом CourseModule, оскільки кожен курс складається з одного або кількох навчальних модулів. Також Course пов’язаний із Enrollment, оскільки користувачі можуть записуватися на курс або отримувати призначення на його проходження.

Клас CourseModule деталізує структуру навчального курсу. Він містить атрибути module_id, course_id, title, content_type, content_url та order_number. Цей клас використовується для зберігання навчальних матеріалів, відео, текстових блоків, тестових завдань або інших елементів курсу. Атрибут order_number визначає порядок проходження модулів, що важливо для коректної побудови навчальної траєкторії.

Клас Enrollment реалізує зв’язок між користувачем і навчальним курсом. Він містить атрибути enrollment_id, user_id, course_id, assigned_by, status, progress, started_at і completed_at. Наявність цього класу дозволяє відстежувати, який користувач проходить конкретний курс, хто призначив навчання, у якому стані перебуває проходження та який відсоток матеріалу вже опрацьовано. Саме через Enrollment реалізується зв’язок «багато до багатьох» між User і Course, оскільки один користувач може проходити декілька курсів, а один курс може бути призначений багатьом користувачам.

Клас TestResult призначений для збереження результатів тестування. Він містить атрибути result_id, enrollment_id, module_id, score, max_score, passed і attempt_number. Цей клас пов’язаний із Enrollment та CourseModule, оскільки результат тестування належить конкретному користувачу в межах певного запису на курс і стосується певного навчального модуля. Наявність attempt_number дозволяє зберігати інформацію про кількість спроб складання тесту.

Клас Certificate відповідає за збереження інформації про сертифікат, сформований після успішного завершення курсу. Його атрибутами є certificate_id, enrollment_id, certificate_number, issued_at, file_url і status. Клас Certificate пов’язаний із Enrollment, оскільки сертифікат формується лише після завершення конкретного запису користувача на курс.

Основні класи системи та їх призначення наведено в таблиці 2.2.

Таблиця 2.2 – Основні класи системи корпоративного навчання

Клас	Призначення	Основні атрибути
Role	Опис ролей користувачів і розмежування доступу	role_id, role_name, description
User	Зберігання даних про користувачів системи	user_id, role_id, full_name, email, department, position, status
Course	Опис навчального курсу	course_id, author_id, title, category, level, duration_hours, status
CourseModule	Опис навчального модуля в межах курсу	module_id, course_id, title, content_type, content_url, order_number
Enrollment	Фіксація запису користувача на курс і стану проходження	enrollment_id, user_id, course_id, assigned_by, status, progress, started_at, completed_at
TestResult	Збереження результатів тестування	result_id, enrollment_id, module_id, score, max_score, passed, attempt_number
Certificate	Збереження даних про сформований сертифікат	certificate_id, enrollment_id, certificate_number, issued_at, file_url, status

Зв’язки між класами відображають логіку функціонування системи. Клас Role має зв’язок один до багатьох із класом User, оскільки одна роль може бути призначена багатьом користувачам. Клас User пов’язаний із Course через атрибут

author_id, що відображає створення курсів автором або викладачем. Клас Course має композиційний зв’язок із CourseModule, оскільки модулі є складовими частинами курсу й не мають самостійного змісту без нього.

Зв’язок між User і Course реалізовано через клас Enrollment. Це дозволяє зберігати не лише сам факт проходження курсу, а й додаткові параметри: статус, прогрес, дату початку, дату завершення та особу, яка призначила навчання. Клас Enrollment пов’язаний із TestResult, оскільки за одним записом користувача на курс може бути отримано декілька результатів тестування. Також Enrollment пов’язаний із Certificate, оскільки сертифікат формується після успішного завершення курсу.

Для деталізації поведінки класів у межах основних сценаріїв роботи системи побудовано три кооперації. Кооперації відображають взаємодію об’єктів під час виконання конкретних процесів і доповнюють діаграму класів динамічним описом роботи системи.

Перша кооперація описує процес запису співробітника на навчальний курс. У ній беруть участь об’єкти employee:User, course:Course, enrollment:Enrollment та manager:User. Користувач переглядає каталог курсів, обирає потрібний курс, після чого система перевіряє його доступність. Якщо курс активний і доступний для проходження, створюється об’єкт enrollment:Enrollment, у якому фіксуються user_id, course_id, статус призначення та початковий прогрес. Після цього користувач отримує доступ до навчальних матеріалів.

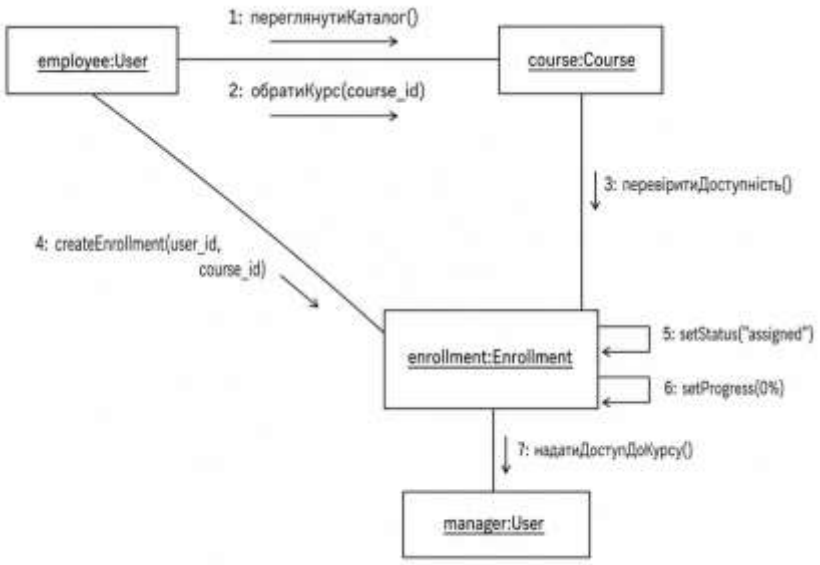

Рисунок 2.3 – Кооперація запису співробітника на навчальний курс

Друга кооперація відображає створення та наповнення навчального курсу. У ній задіяні об’єкти author:User, role:Role, course:Course, module1:CourseModule та module2:CourseModule. Спочатку перевіряється роль користувача, оскільки створювати навчальний контент може лише автор, менеджер або адміністратор. Після перевірки прав доступу створюється об’єкт course:Course, якому надається початковий статус draft. Далі до курсу додаються навчальні модулі, визначається їх порядок, оновлюється навчальний контент і після завершення підготовки курс переводиться у статус active.

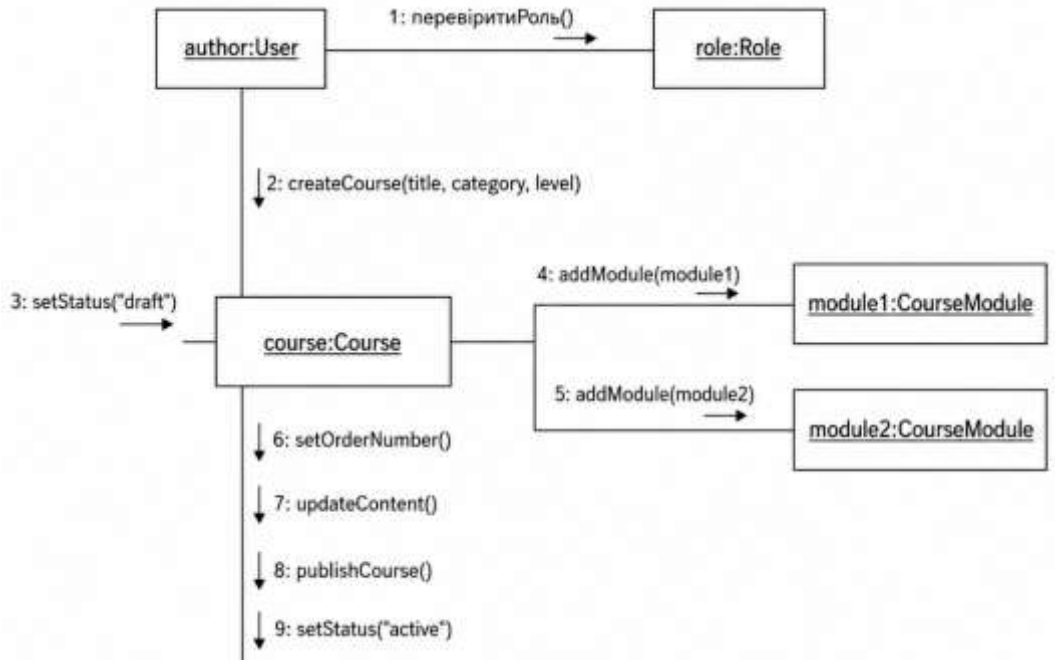

Рисунок 2.4 – Кооперація створення та наповнення навчального курсу

Третя кооперація описує процес тестування та формування сертифіката. У ній беруть участь об’єкти employee:User, enrollment:Enrollment, module:CourseModule, result:TestResult та certificate:Certificate. Користувач проходить навчальний модуль і надсилає тест на перевірку. Після цього створюється об’єкт result:TestResult, у якому зберігаються отримані бали, максимальна кількість балів і результат проходження. Якщо тест складено успішно, об’єкт enrollment:Enrollment оновлює прогрес користувача. Після завершення всіх необхідних модулів створюється об’єкт certificate:Certificate, який зберігає номер сертифіката, дату видачі та посилання на файл сертифіката.

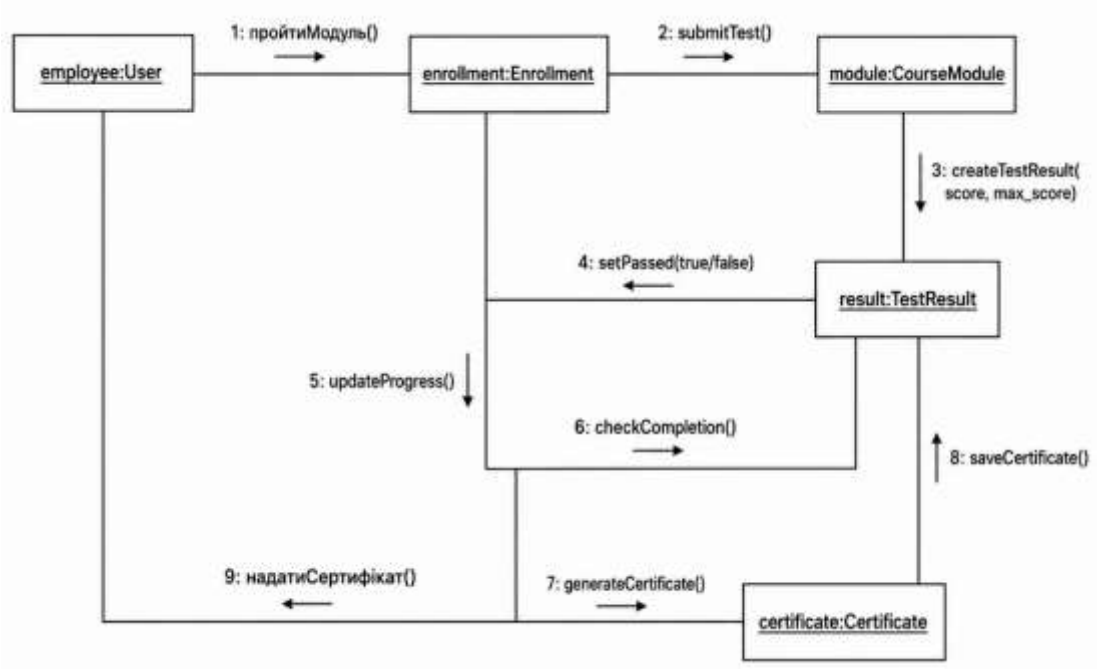

Рисунок 2.5 – Кооперація тестування та формування сертифіката

Результати побудови кооперацій наведено в таблиці 2.3.

Таблиця 2.3

Характеристика кооперацій системи корпоративного навчання

Кооперація	Задіяні класи	Результат виконання
Запис співробітника на курс	User, Course, Enrollment	Створюється запис про проходження курсу, користувач отримує доступ до навчання

Створення та наповнення курсу	Role, User, Course, CourseModule	Створюється курс, додаються модулі, курс переводиться у стан активного
Тестування та формування сертифіката	Enrollment, CourseModule, TestResult, Certificate	Зберігається результат тестування, оновлюється прогрес і формується сертифікат

Розроблена діаграма класів і побудовані кооперації підтверджують узгодженість об’єктної моделі з функціональними вимогами системи корпоративного навчання. Класи охоплюють основні інформаційні сутності, необхідні для реалізації веб-застосунку, а кооперації деталізують взаємодію об’єктів у межах ключових бізнес-процесів. Запропонована модель забезпечує основу для подальшої реалізації програмних модулів керування користувачами, курсами, навчальними модулями, записами на навчання, тестуванням, сертифікацією та аналітикою результатів корпоративного навчання.

2.3 Діаграма компонентів

Діаграма компонентів використовується для подання програмної системи корпоративного навчання на рівні основних функціональних модулів та зв’язків між ними. Вона показує, які частини входять до складу веб-застосунку, як вони взаємодіють між собою та які зовнішні сервіси використовуються під час роботи системи.

На рисунку 2.6 наведено діаграму компонентів системи корпоративного навчання.

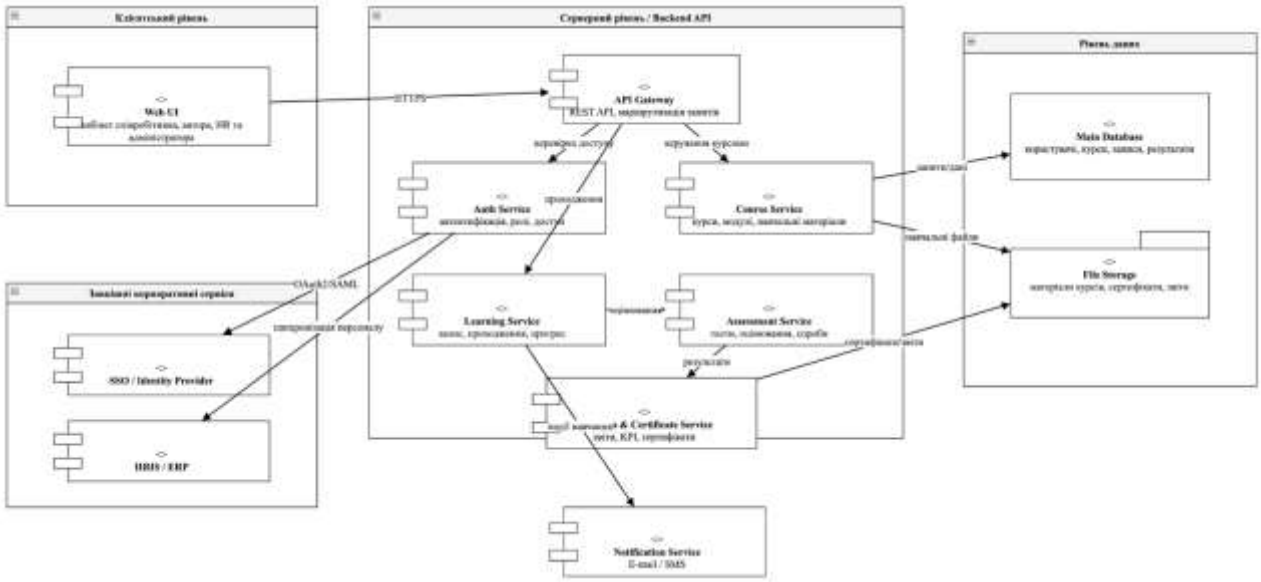

Рисунок 2.6 – Діаграма компонентів системи корпоративного навчання

У діаграмі виділено чотири основні рівні: клієнтський рівень, серверний рівень, рівень даних та зовнішні корпоративні сервіси. Клієнтський рівень представлений компонентом Web UI, через який працюють співробітник, автор курсу, HR-менеджер та адміністратор. Взаємодія клієнта із сервером здійснюється через захищений HTTPS-запит до API Gateway.

Компонент API Gateway є єдиною точкою входу до серверної частини. Він приймає запити від веб-інтерфейсу, виконує маршрутизацію до відповідних сервісів та забезпечує первинну перевірку доступу. Такий підхід дозволяє не звертатися напряму до окремих сервісів і спрощує контроль безпеки.

Серверний рівень складається з окремих функціональних сервісів. Auth Service відповідає за автентифікацію користувачів, перевірку ролей та прав доступу. Він взаємодіє із зовнішнім SSO / Identity Provider через OAuth2 або SAML, а також отримує службову інформацію про персонал із HRIS / ERP. Course Service забезпечує керування навчальними курсами, модулями та навчальними матеріалами. Learning Service відповідає за запис користувачів на курси, проходження навчання та фіксацію прогресу.

Assessment Service реалізує функції тестування, оцінювання та обліку спроб. Після проходження тестів результати передаються до Analytics & Certificate Service, який формує звіти, KPI-показники та сертифікати. Notification

Service використовується для надсилання повідомлень користувачам через e-mail або SMS, наприклад про призначення курсу, завершення навчання або видачу сертифіката.

Основні компоненти системи наведено в таблиці 2.4.

Таблиця 2.4

Компоненти системи корпоративного навчання

Компонент	Призначення
Web UI	Веб-інтерфейс для співробітника, автора курсу, HR та адміністратора
API Gateway	Приймання REST-запитів, маршрутизація та контроль доступу
Auth Service	Автентифікація, ролі користувачів, перевірка прав доступу
Course Service	Керування курсами, модулями та навчальними матеріалами
Learning Service	Запис на курси, проходження навчання, облік прогресу
Assessment Service	Тести, оцінювання, збереження результатів і спроб
Analytics & Certificate Service	Формування звітів, KPI та сертифікатів
Notification Service	Надсилання повідомлень користувачам
Main Database	Зберігання користувачів, курсів, записів і результатів
File Storage	Зберігання матеріалів курсів, сертифікатів і звітів
SSO / Identity Provider	Корпоративна автентифікація користувачів
HRIS / ERP	Синхронізація даних про персонал

Рівень даних складається з Main Database та File Storage. Main Database зберігає структуровані дані системи: користувачів, ролі, курси, записи на навчання, результати тестування та службові журнали. File Storage використовується для зберігання файлів навчальних матеріалів, сформованих сертифікатів і звітів. Розділення бази даних і файлового сховища є доцільним, оскільки текстові й табличні дані обробляються окремо від великих навчальних файлів.

Зовнішні корпоративні сервіси представлені компонентами SSO / Identity Provider та HRIS / ERP. SSO забезпечує єдиний вхід користувачів до системи, а HRIS / ERP використовується для синхронізації даних про співробітників, їхні посади, підрозділи та організаційну належність. Це дає змогу підтримувати актуальність облікових даних без ручного дублювання інформації.

Запропонована компонентна структура забезпечує модульність системи. Кожен сервіс виконує окрему функцію і може розроблятися, тестуватися та оновлюватися незалежно. Такий підхід спрощує супровід веб-застосунку, підвищує масштабованість і дозволяє в майбутньому розширювати систему новими модулями, наприклад адаптивним навчанням, рекомендаціями курсів або розширеною аналітикою результатів навчання.

2.4 Діаграма пакетів системи

Діаграма пакетів використовується для відображення логічної структуризації програмної системи корпоративного навчання. Вона показує, на які основні пакети поділяється програмне забезпечення, які функції виконує кожен пакет та як між ними організовані залежності. Такий підхід дозволяє подати архітектуру системи не на рівні окремих класів, а на рівні більших програмних модулів.

На рисунку 2.7 наведено діаграму пакетів системи корпоративного навчання.

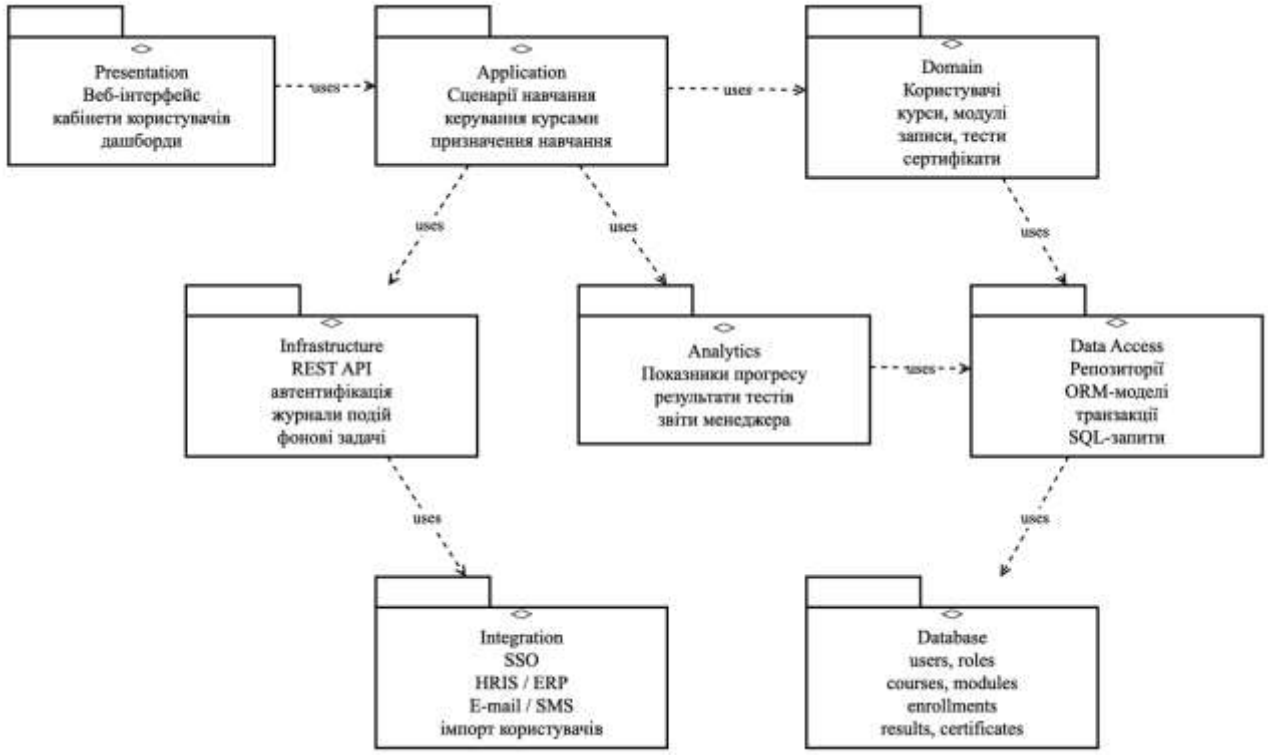

Рисунок 2.7 – Діаграма пакетів системи корпоративного навчання

У діаграмі виділено сім основних пакетів: Presentation, Application, Domain, Infrastructure, Analytics, Data Access, Database та Integration. Кожен пакет відповідає за окрему частину функціональності системи та має чітко визначену роль у загальній архітектурі веб-застосунку.

Пакет Presentation відповідає за клієнтський рівень системи. До нього належать веб-інтерфейс, особисті кабінети користувачів і інформаційні панелі. Через цей пакет із системою взаємодіють співробітники, автори курсів, HR-менеджери та адміністратори. Presentation не містить бізнес-логіки, а лише передає дії користувача до прикладного рівня та відображає отримані результати.

Пакет Application є центральним прикладним рівнем системи. Він містить сценарії навчання, керування курсами та призначення навчання користувачам. Саме цей пакет координує роботу інших частин системи: приймає запити з веб-інтерфейсу, звертається до доменної моделі, використовує інфраструктурні сервіси та передає дані до аналітичного модуля.

Пакет Domain містить основні об’єкти предметної області: користувачів, курси, модулі, записи на навчання, тести та сертифікати. Він відображає базову логіку корпоративного навчання та пов’язаний із класами, які були визначені в діаграмі класів. До цього пакета належать ключові правила роботи системи: хто може створювати курс, як користувач записується на навчання, як фіксується прогрес і за яких умов формується сертифікат.

Пакет Infrastructure забезпечує службові механізми роботи системи. До нього входять REST API, автентифікація, журнали подій та фонові задачі. Цей пакет використовується прикладним рівнем для обробки запитів, перевірки доступу, реєстрації дій користувачів і виконання службових операцій.

Пакет Analytics відповідає за обробку навчальних результатів. Він формує показники прогресу, аналізує результати тестів і готує звіти для менеджера. Цей пакет використовує дані з основних доменних об’єктів і звертається до Data Access для отримання інформації з бази даних.

Пакет Data Access реалізує доступ до даних. До нього належать репозиторії, ORM-моделі, транзакції та SQL-запити. Його призначення полягає у відокремленні бізнес-логіки від прямої роботи з базою даних. Завдяки цьому прикладний і доменний рівні не залежать від конкретної реалізації збереження даних.

Пакет Database відповідає за фізичне збереження даних системи. У ньому розміщуються таблиці users, roles, courses, modules, enrollments, results та certificates. Цей пакет забезпечує збереження інформації про користувачів, ролі, курси, навчальні модулі, записи на курси, результати тестування та сформовані сертифікати.

Пакет Integration призначений для взаємодії із зовнішніми корпоративними сервісами. Він охоплює інтеграцію з SSO, HRIS / ERP, сервісами E-mail / SMS та механізмами імпорту користувачів. Через цей пакет система може отримувати актуальні дані про персонал, виконувати єдиний вхід користувачів і надсилати повідомлення про події навчального процесу.

Основні пакети системи наведено в таблиці 2.5.

Таблиця 2.5

Пакети програмної архітектури системи корпоративного навчання

Пакет	Призначення
Presentation	Веб-інтерфейс, кабінети користувачів, дашборди
Application	Сценарії навчання, керування курсами, призначення навчання
Domain	Користувачі, курси, модулі, записи, тести, сертифікати
Infrastructure	REST API, автентифікація, журнали подій, фонові задачі
Analytics	Показники прогресу, результати тестів, звіти менеджера
Data Access	Репозиторії, ORM-моделі, транзакції, SQL-запити
Database	Таблиці users, roles, courses, modules, enrollments, results, certificates
Integration	SSO, HRIS / ERP, E-mail / SMS, імпорт користувачів

Залежності між пакетами побудовано за принципом багаторівневої архітектури. Пакет Presentation використовує Application, оскільки всі дії користувача передаються до прикладного рівня. Application використовує Domain для роботи з основними сутностями системи, а також Infrastructure та Analytics для службової обробки запитів і формування показників навчання. Domain використовує Data Access для збереження та отримання інформації. Data Access звертається до Database, де розміщені основні таблиці системи. Infrastructure взаємодіє з Integration, оскільки автентифікація, синхронізація персоналу та надсилання повідомлень виконуються через зовнішні корпоративні сервіси.

Запропонована пакетна структура є компактною та придатною для реалізації веб-застосунку корпоративного навчання. Вона забезпечує розділення відповідальності між модулями, спрощує супровід програмного коду та дозволяє в подальшому розширювати систему без суттєвої зміни її базової архітектури. Наприклад, до пакета Analytics можна додати розширені звіти, до Integration - нові корпоративні сервіси, а до Presentation - окремі інтерфейси для мобільних пристроїв.

2.5 Вибір стеку технологій

Для реалізації системи корпоративного навчання обрано стек технологій, який забезпечує просте розгортання, роботу через браузер, підтримку бази даних і можливість подальшого розширення функціоналу. Оскільки система орієнтована на використання у корпоративному середовищі, основними критеріями вибору були стабільність, доступність інструментів, невисокі вимоги до серверного обладнання, простота супроводу та можливість швидкого запуску програмного прототипу.

У межах розроблення системи використано веб-орієнтований підхід. Користувачі працюють із системою через браузер, а серверна частина відповідає за обробку запитів, авторизацію, роботу з курсами, записами на навчання, тестуванням, сертифікатами та аналітичними даними. Для збереження інформації використовується локальна реляційна база даних.

Основні технології реалізації системи наведено в таблиці 2.6.

Таблиця 2.6

Обраний стек технологій для реалізації системи

Компонент системи	Обрана технологія	Призначення
Серверна частина	Python 3.9+	Обробка запитів, бізнес-логіка, маршрутизація сторінок
База даних	SQLite	Зберігання користувачів, ролей, курсів, записів, результатів і сертифікатів
Клієнтська частина	HTML5, CSS3, JavaScript	Побудова веб-інтерфейсу та інтерактивних елементів
Обмін даними	HTTP / REST-підхід	Передавання запитів між клієнтською та серверною частинами
Авторизація	Сесії користувачів, хешування паролів	Захист доступу до системи та розмежування ролей
Звіти	CSV-файли	Експорт результатів навчання та аналітичних даних
Середовище запуску	Локальний веб-сервер	Запуск системи на робочому комп’ютері або сервері організації

Для серверної частини обрано мову Python, оскільки вона має простий синтаксис, достатню кількість стандартних бібліотек і добре підходить для створення навчальних та прикладних веб-прототипів. Python використовується для маршрутизації запитів, перевірки авторизації, обробки форм, взаємодії з базою даних і формування сторінок системи. Додатковою перевагою є можливість запуску застосунку без складного налаштування серверного середовища.

Як систему керування базою даних використано SQLite. Такий вибір є доцільним для програмного прототипу, оскільки SQLite не потребує окремого серверу бази даних і зберігає всю інформацію в одному файлі. У базі даних зберігаються облікові записи користувачів, ролі, навчальні курси, модулі, записи на проходження навчання, результати тестування, сертифікати, сповіщення та журнали подій. За потреби в подальшому SQLite може бути замінено на PostgreSQL або MySQL без зміни загальної логіки системи.

Клієнтська частина реалізується засобами HTML5, CSS3 та JavaScript. HTML5 використовується для структури сторінок, CSS3 - для оформлення інтерфейсу, а JavaScript - для базової інтерактивності. Інтерфейс системи передбачає сторінку входу, реєстрацію, особистий кабінет, каталог курсів, сторінку проходження навчання, блок тестування, сертифікати, панель аналітики та адміністративні сторінки. Дизайн виконано у стриманому корпоративному стилі без надмірних декоративних елементів.

Для організації взаємодії між клієнтською та серверною частинами використовується HTTP із REST-підходом. Запити користувача надходять із браузера на сервер, після чого сервер перевіряє права доступу, виконує необхідну операцію та повертає оновлену сторінку або дані. Такий підхід відповідає компонентній структурі системи, у якій API Gateway, сервіси автентифікації, курсів, навчання, тестування та аналітики можуть бути виділені в окремі модулі.

Для захисту доступу до системи передбачено авторизацію користувачів за електронною поштою та паролем. Паролі зберігаються не у відкритому вигляді, а у вигляді хешованих значень. Після входу користувача система створює сесію, яка використовується для визначення активного користувача та його ролі. Рольова модель дозволяє розмежувати доступ до функцій співробітника, автора курсу, HR-менеджера та адміністратора.

Обраний стек технологій є достатнім для реалізації основних функцій системи корпоративного навчання: реєстрації й авторизації користувачів, керування курсами, створення навчальних модулів, запису на курси, проходження навчання, тестування, формування сертифікатів, створення звітів і перегляду аналітики. Водночас архітектура не є надмірно складною, що спрощує запуск, тестування та подальше доопрацювання системи.

2.6 Алгоритмізація програмних модулів системи

Алгоритмізація програмних модулів системи корпоративного навчання виконана для формального опису основних процесів роботи веб-застосунку. На основі функціональних вимог, діаграми класів, компонентної та пакетної структури було виділено три ключові алгоритми: авторизація користувача, запис і проходження навчального курсу, тестування та формування сертифіката.

Перший алгоритм описує роботу модуля авторизації та визначення ролі користувача. Його блок-схему наведено на рисунку 2.8.

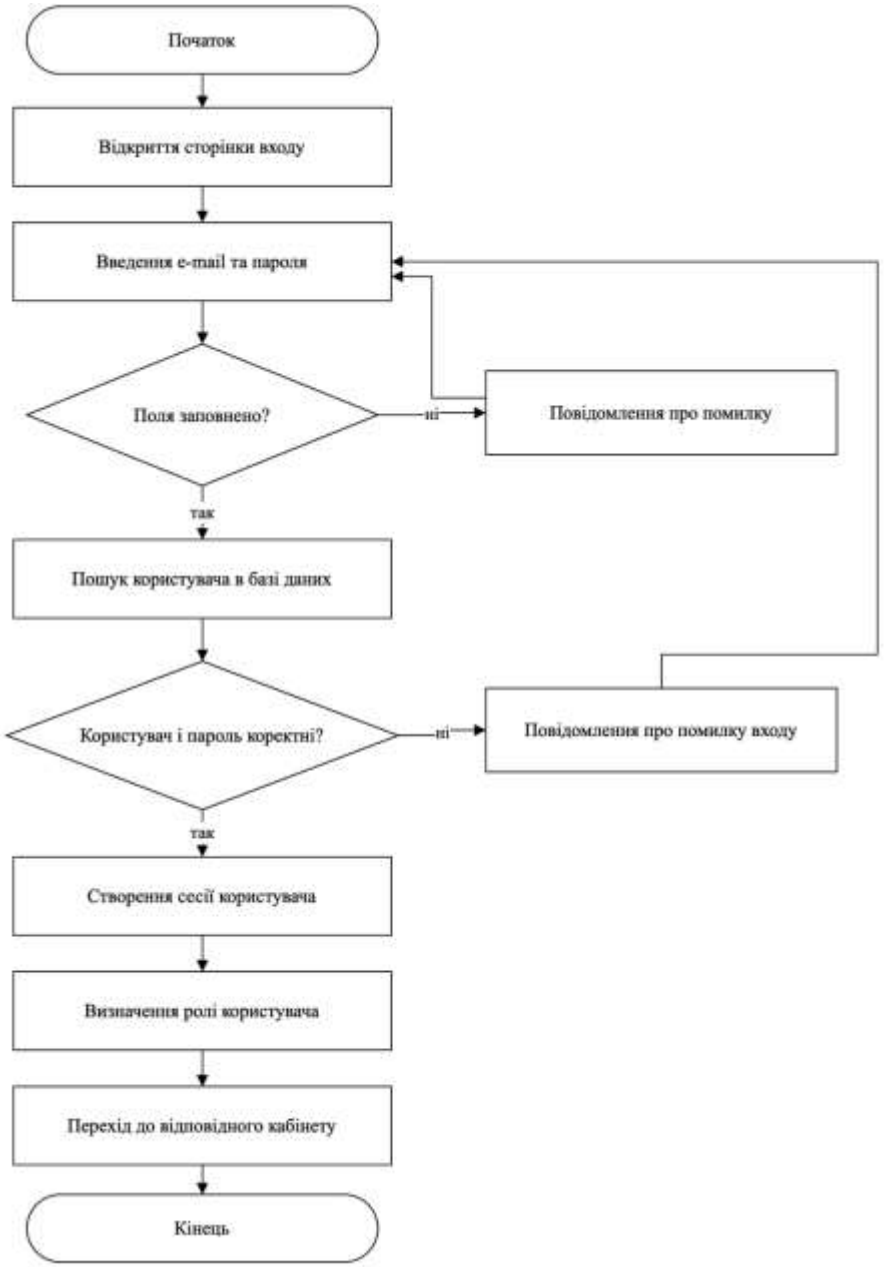

Рисунок 2.8 – Блок-схема алгоритму авторизації та визначення ролі користувача

Алгоритм починається з відкриття сторінки входу. Користувач вводить e-mail і пароль, після чого система перевіряє, чи всі поля заповнені. Якщо обов’язкові дані відсутні, формується повідомлення про помилку, і користувач повертається до форми введення. Якщо поля заповнені, система виконує пошук користувача в базі даних і перевіряє коректність облікових даних.

У разі помилкового e-mail або пароля система виводить повідомлення про помилку входу. Якщо дані правильні, створюється сесія користувача,

визначається його роль і виконується перехід до відповідного кабінету. Для співробітника відкривається кабінет проходження навчання, для автора - сторінки керування курсами, для HR-менеджера - панель призначення навчання й аналітики, для адміністратора - розділи керування користувачами та системними даними.

Другий алгоритм описує запис користувача на курс і проходження навчальних модулів. Блок-схему цього процесу наведено на рисунку 2.9.

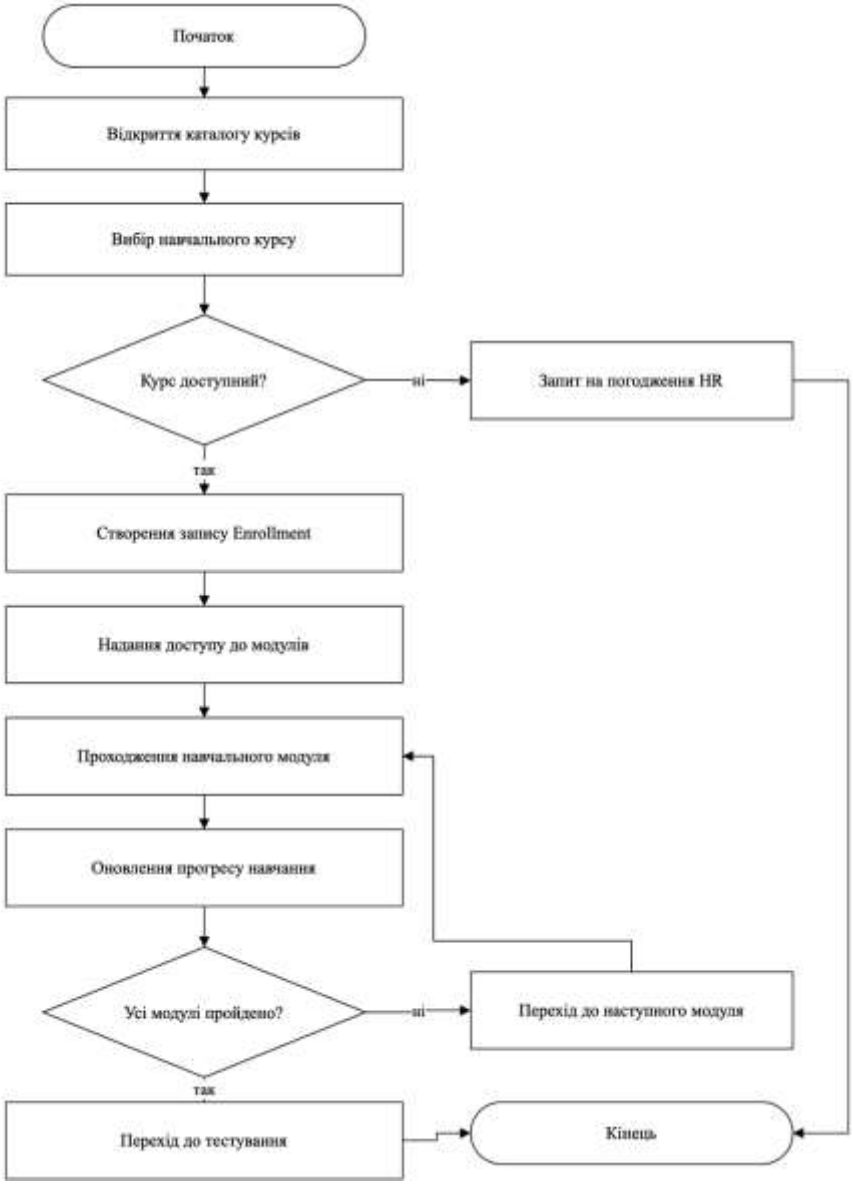

Рисунок 2.9 – Блок-схема алгоритму запису та проходження навчального курсу

Після відкриття каталогу курсів користувач обирає потрібний навчальний курс. Система перевіряє його доступність. Якщо курс недоступний або потребує погодження, формується запит до HR-менеджера. Якщо курс доступний, створюється запис Enrollment, який пов’язує користувача з обраним курсом. У цьому записі фіксується статус проходження, дата початку навчання та початковий прогрес.

Після створення запису користувачу надається доступ до навчальних модулів. Система послідовно відкриває модулі курсу, фіксує факт їх проходження та оновлює прогрес навчання. Після кожного модуля виконується перевірка, чи всі модулі курсу пройдено. Якщо залишилися непройдені модулі, користувач переходить до наступного навчального блоку. Якщо всі модулі завершено, система переводить користувача до етапу тестування.

Третій алгоритм описує перевірку знань і формування сертифіката. Його блок-схему наведено на рисунку 2.10.

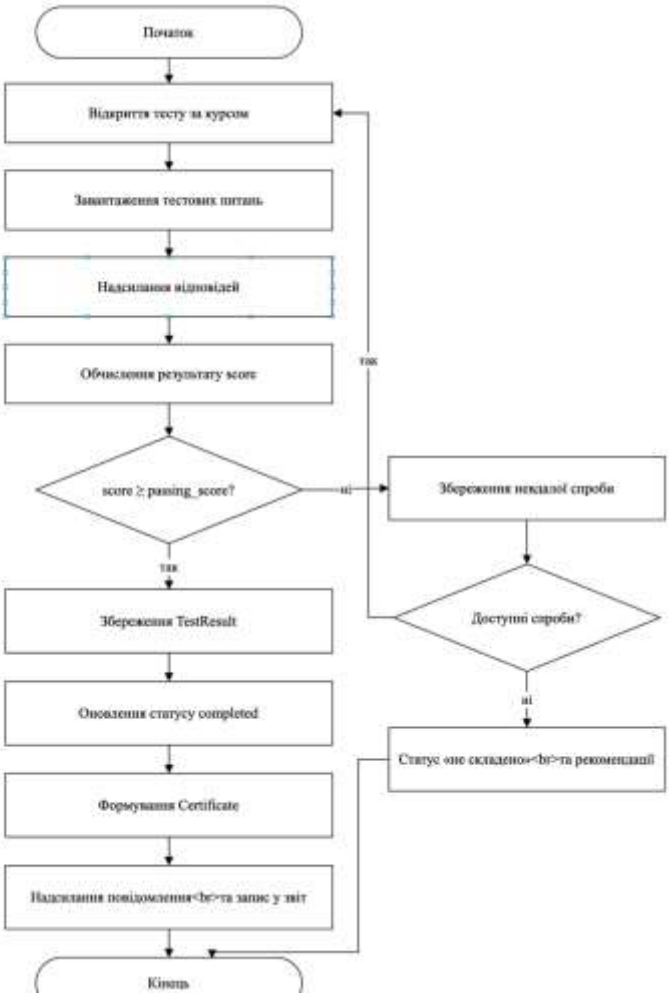

Рисунок 2.10 – Блок-схема алгоритму тестування та формування сертифіката

Алгоритм починається з відкриття тесту за курсом. Система завантажує тестові питання, після чого користувач надсилає відповіді. Далі виконується автоматичне обчислення результату score і порівняння його з мінімальним прохідним значенням passing_score. Якщо результат є достатнім, система зберігає запис TestResult, оновлює статус проходження курсу на completed, формує Certificate, надсилає користувачу повідомлення та додає інформацію до звітності.

Якщо результат нижчий за мінімальний прохідний бал, система зберігає невдалу спробу та перевіряє, чи залишилися доступні повторні спроби. Якщо спроби доступні, користувач повертається до повторного проходження тесту. Якщо ліміт спроб вичерпано, системі призначається статус «не складено», а

користувач отримує повідомлення з рекомендаціями щодо повторного опрацювання навчального матеріалу.

Основні алгоритми програмних модулів системи наведено в таблиці 2.7.

Таблиця 2.7

Алгоритми програмних модулів системи корпоративного навчання

Алгоритм	Задіяний модуль	Вхідні дані	Результат виконання
Авторизація та визначення ролі	Auth Service	E-mail, пароль користувача	Створення сесії та перехід до відповідного кабінету
Запис і проходження курсу	Learning Service	Дані користувача, обраний курс	Створення Enrollment, відкриття модулів, оновлення прогресу
Тестування та сертифікація	Assessment Service, Certificate Service	Відповіді користувача, прохідний бал	Збереження TestResult, формування Certificate або фіксація невдалої спроби

Запропоновані алгоритми охоплюють основний цикл роботи системи корпоративного навчання: вхід користувача, отримання доступу до навчальних матеріалів, проходження курсу, перевірку знань і формування підтвердження результату. Така алгоритмічна структура є достатньою для реалізації програмного прототипу та відповідає розробленим UML-діаграмам, логічній моделі даних і компонентній архітектурі системи.

2.7 Висновки до другого розділу

У другому розділі виконано проєктування інформаційного та програмного забезпечення системи корпоративного навчання на основі веб-технологій. На цьому етапі було сформовано основні моделі, які визначають структуру даних, логіку взаємодії програмних об’єктів, компонентну організацію системи, пакетну архітектуру та алгоритми роботи ключових модулів.

Розроблено логічну модель даних системи, у якій визначено основні сутності предметної області: Role, User, Course, CourseModule, Enrollment, TestResult та Certificate. Запропонована модель відображає зв’язки між користувачами, ролями, навчальними курсами, модулями, записами на навчання,

результатами тестування та сертифікатами. Така структура забезпечує цілісність даних, зменшує дублювання інформації та створює основу для подальшої реалізації бази даних.

На основі логічної моделі побудовано UML-діаграму класів, яка деталізує об’єктну структуру програмної системи. У межах цього підрозділу також розроблено три кооперації: запис співробітника на курс, створення та наповнення навчального курсу, тестування і формування сертифіката. Побудовані кооперації показують взаємодію основних класів під час виконання ключових бізнес-процесів і підтверджують узгодженість діаграми класів із функціональними вимогами системи.

Для визначення програмної архітектури побудовано діаграму компонентів. У ній систему подано як сукупність взаємопов’язаних модулів: Web UI, API Gateway, Auth Service, Course Service, Learning Service, Assessment Service, Analytics & Certificate Service, Notification Service, Main Database, File Storage та зовнішніх корпоративних сервісів. Така структура дозволяє розмежувати відповідальність між компонентами, спростити супровід системи та забезпечити можливість її подальшого розширення.

Пакетна структура системи подана через пакети Presentation, Application, Domain, Infrastructure, Analytics, Data Access, Database та Integration. Вона відображає логічний поділ програмного забезпечення на рівні інтерфейсу, прикладної логіки, предметної області, доступу до даних, інтеграцій та аналітики. Такий підхід відповідає принципам багаторівневої архітектури та дозволяє уникнути змішування інтерфейсної, бізнесової та інфраструктурної логіки.

У розділі також обґрунтовано вибір стеку технологій для реалізації системи. Для серверної частини обрано Python 3.9+, для збереження даних - SQLite, для клієнтської частини - HTML5, CSS3 і JavaScript. Обраний стек є достатнім для реалізації веб-застосунку корпоративного навчання, не потребує складного розгортання та забезпечує підтримку основних функцій: авторизації,

керування курсами, проходження навчання, тестування, сертифікації та формування звітів.

Окремо виконано алгоритмізацію основних програмних модулів. Побудовано блок-схеми алгоритмів авторизації та визначення ролі користувача, запису і проходження навчального курсу, тестування та формування сертифіката. Ці алгоритми описують основний цикл роботи системи від входу користувача до завершення навчання та отримання результату.

Отже, у другому розділі сформовано повну проєктну основу для реалізації системи корпоративного навчання. Розроблені моделі даних, UML-діаграми, компонентна й пакетна структура, вибраний стек технологій та алгоритми програмних модулів забезпечують перехід від аналізу предметної області до практичної реалізації веб-застосунку. Отримані результати є базою для подальшого розроблення програмного забезпечення, створення бази даних, реалізації інтерфейсу користувача та тестування системи.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 План тестування програмних модулів та методика оцінювання результатів

Тестування системи корпоративного навчання виконується з метою перевірки коректності роботи основних програмних модулів, відповідності функціональним вимогам і стабільності взаємодії між клієнтською частиною, серверною логікою та базою даних. Основна увага приділяється тим модулям, які забезпечують авторизацію користувачів, керування навчальними курсами, запис на навчання, проходження модулів, тестування, формування сертифікатів та перегляд аналітичної інформації.

Тестування проводиться для веб-застосунку, який працює через браузер і використовує базу даних для збереження користувачів, ролей, курсів, записів на навчання, результатів тестування та сертифікатів. Перевірка виконується за основними сценаріями використання системи: вхід користувача, відкриття особистого кабінету, перегляд каталогу курсів, запис на курс, проходження навчальних модулів, складання тесту, отримання сертифіката та перегляд результатів у панелі адміністратора або HR-менеджера.

План тестування програмних модулів наведено в таблиці 3.1.

Таблиця 3.1 – План тестування програмних модулів системи корпоративного навчання

№	Модуль системи	Що перевіряється	Очікуваний результат
1	Модуль авторизації	Вхід користувача за e-mail і паролем, перевірка порожніх полів, неправильних даних і ролей	Користувач входить до системи лише з коректними даними, після входу відкривається відповідний кабінет
2	Модуль ролей і доступу	Обмеження доступу для співробітника, автора, HR-менеджера та адміністратора	Кожна роль бачить лише дозволені функції та сторінки

3	Модуль курсів	Створення, редагування, перегляд і публікація навчальних курсів	Курс коректно зберігається в базі даних і відображається в каталозі
4	Модуль навчальних матеріалів	Додавання модулів до курсу, відображення структури курсу та порядку проходження	Модулі відображаються у правильній послідовності та доступні користувачу

Продовження таблиці 3.1

5	Модуль запису на курс	Створення запису Enrollment, перевірка статусу курсу та доступу користувача	Після запису користувач отримує доступ до матеріалів курсу
6	Модуль проходження навчання	Перехід між модулями, фіксація проходження, оновлення прогресу	Прогрес навчання змінюється відповідно до кількості пройдених модулів
7	Модуль тестування	Завантаження тесту, надсилання відповідей, обчислення результату	Система правильно визначає кількість балів і статус проходження тесту
8	Модуль сертифікації	Формування сертифіката після успішного завершення курсу	Сертифікат створюється лише після виконання умов проходження
9	Модуль аналітики	Відображення кількості курсів, користувачів, результатів і сертифікатів	Дані в аналітичній панелі відповідають записам у базі даних
10	Модуль сповіщень і журналу подій	Фіксація основних дій користувача та системних повідомлень	Події зберігаються та доступні для перегляду адміністратором

Методика тестування передбачає поетапну перевірку кожного модуля окремо, а потім перевірку їх спільної роботи в межах повного сценарію навчання. Спочатку виконується функціональне тестування, під час якого перевіряється правильність виконання окремих дій користувача. Наприклад, для модуля авторизації перевіряється вхід із правильними та неправильними даними, а для модуля курсів - створення нового курсу, його збереження та відображення в каталозі.

Після цього проводиться інтеграційне тестування. Його мета полягає в перевірці взаємодії між модулями. Наприклад, після створення курсу автором система повинна коректно показати його співробітнику в каталозі, дозволити запис на курс, створити запис Enrollment, оновити прогрес проходження та

передати користувача до тестування. Така перевірка дозволяє переконатися, що окремі частини системи працюють узгоджено.

Окремо перевіряється робота з базою даних. Для цього контролюється правильність створення, оновлення та зчитування записів у таблицях користувачів, ролей, курсів, модулів, записів на навчання, результатів тестування та сертифікатів. Після виконання кожної важливої операції перевіряється, чи зберігаються дані без дублювання та чи не порушуються зв’язки між таблицями.

Методика оцінювання результатів тестування ґрунтується на порівнянні фактичного результату з очікуваним. Тест вважається успішним, якщо система виконує дію без помилок, дані коректно зберігаються в базі, інтерфейс відображає правильну інформацію, а користувач отримує доступ лише до тих функцій, які відповідають його ролі. Якщо результат відрізняється від очікуваного, фіксується помилка, визначається модуль, у якому вона виникла, після чого виконується повторна перевірка після виправлення.

Для оцінювання якості роботи системи використовуються такі критерії: коректність виконання функцій, стабільність роботи веб-інтерфейсу, правильність збереження даних, відповідність ролей правам доступу, швидкість відкриття основних сторінок і відсутність критичних помилок під час проходження повного сценарію навчання. Для програмного прототипу достатнім вважається результат, за якого основні сценарії виконуються без збоїв, а користувач може пройти шлях від авторизації до отримання сертифіката.

Запропонований план тестування охоплює основні програмні модулі системи корпоративного навчання та дозволяє перевірити її працездатність на рівні окремих функцій і повного бізнес-процесу. Це створює основу для подальшого тестування інтерфейсу, аналізу результатів перевірки та оцінювання ефективності розробленого веб-застосунку.

3.2 Тестування програмної системи

Тестування програмної системи корпоративного навчання виконувалося після реалізації основних модулів веб-застосунку. Перевірка проводилася вручну через браузер за основними ролями користувачів: співробітник, автор курсів, HR-менеджер та адміністратор. Метою тестування було підтвердити, що система коректно виконує авторизацію, відкриває відповідні кабінети, зберігає дані в базі SQLite, відображає курси, дозволяє запис на навчання, підтримує керування курсами й тестами та формує аналітичні показники.

Першим етапом перевірено сторінку входу до системи, наведену на рис. 3.1. На сторінці реалізовано введення e-mail і пароля, а також швидкий вибір демонстраційної ролі користувача: співробітник, автор, HR або адміністратор. Після введення коректних облікових даних система створює сесію користувача та відкриває відповідний кабінет. У разі некоректних або порожніх даних користувач не повинен отримати доступ до внутрішніх сторінок системи.

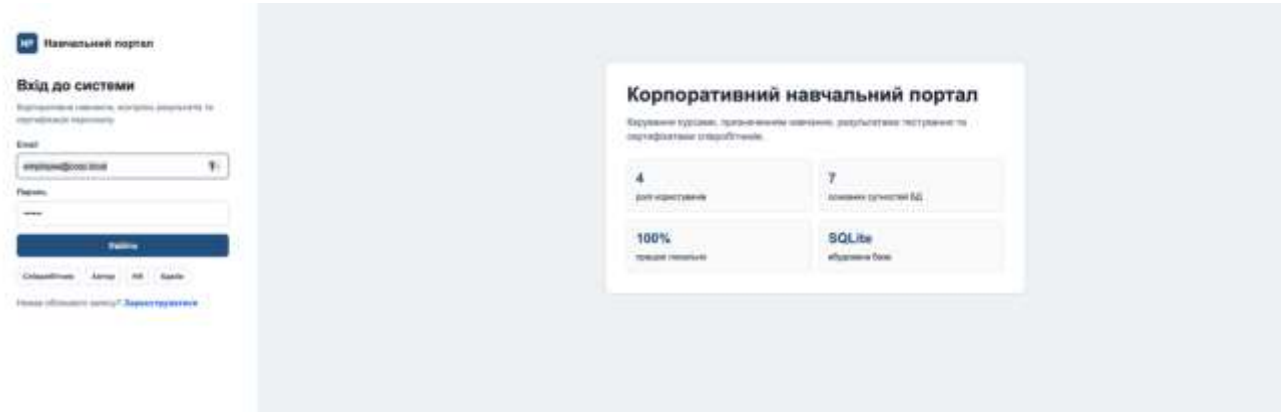

Рисунок 3.1 – Сторінка входу до системи корпоративного навчання

Після авторизації співробітника перевірено роботу головної панелі керування, показаної на рис. 3.2. На сторінці відображаються основні показники: кількість доступних курсів, активних навчань, завершених курсів і отриманих сертифікатів. Також реалізовано блок поточного прогресу, швидкі дії та рекомендовані курси. Перевірка підтвердила, що кнопки навігації відкривають відповідні розділи, а інформація на панелі відповідає даним, збереженим у базі.

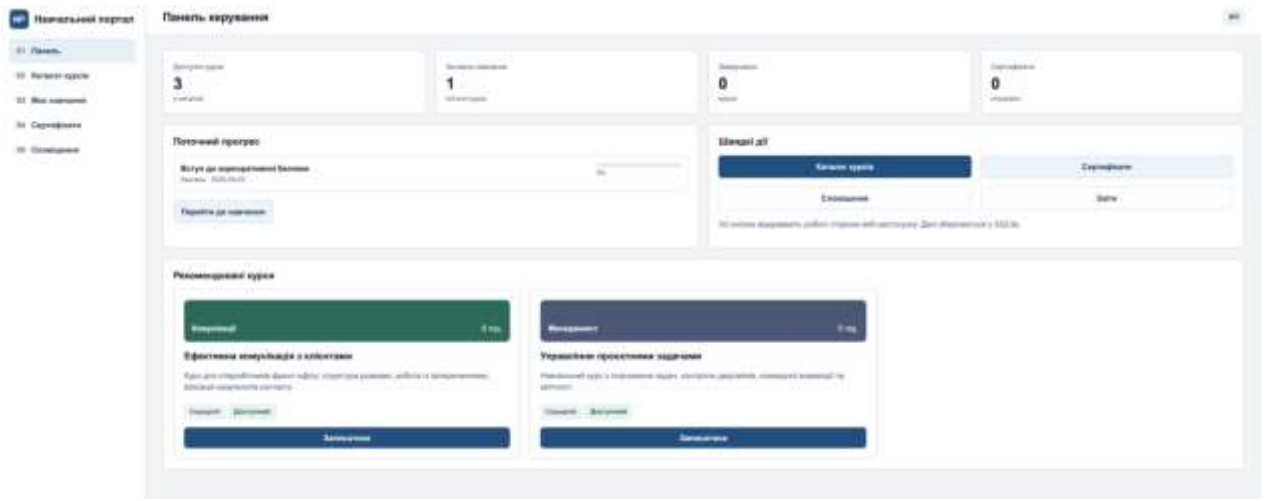

Рисунок 3.2 – Панель керування співробітника

На наступному етапі протестовано каталог курсів, наведений на рис. 3.3. У каталозі відображаються доступні навчальні курси з назвою, категорією, рівнем, тривалістю, описом і кнопкою запису або відкриття курсу. Додатково перевірено поле пошуку та фільтр за категорією. Під час тестування встановлено, що курси коректно завантажуються з бази даних, а після запису користувача змінюється стан курсу та з’являється можливість перейти до навчання.

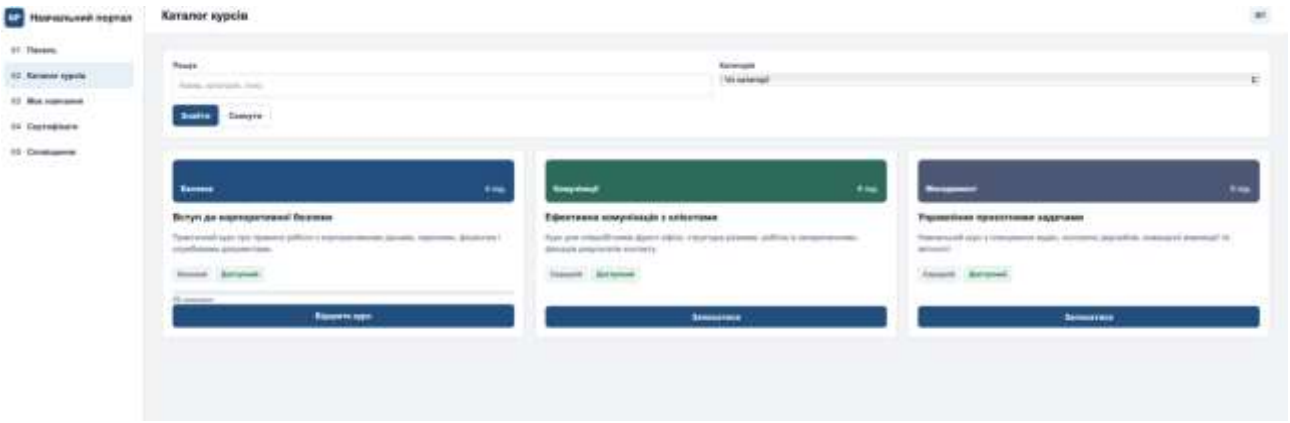

Рисунок 3.3 – Каталог навчальних курсів

Окремо перевірено сторінку «Моє навчання», зображену на рис. 3.4. На ній відображаються курси, на які користувач уже записаний. Для кожного курсу показано статус, рівень, прогрес проходження та кнопку продовження навчання. Тестування підтвердило, що створений запис Enrollment коректно пов’язує користувача з курсом, а прогрес навчання відображається у відповідному блоці.

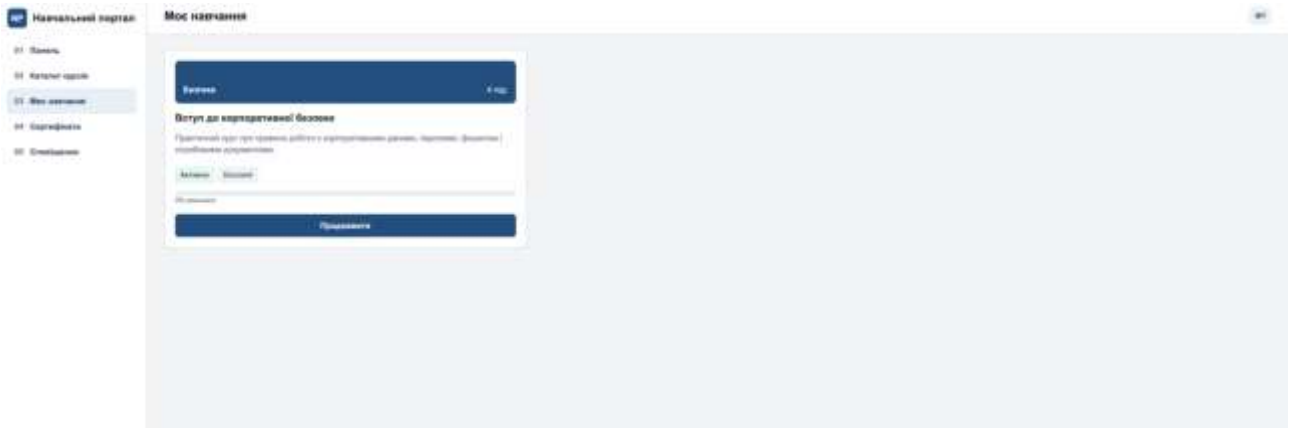

Рисунок 3.4 – Сторінка проходження навчання користувача

Для ролі автора курсу перевірено сторінку керування курсами, наведену на рис. 3.5. У цьому розділі реалізовано створення нового курсу, заповнення назви, опису, категорії, рівня, тривалості та статусу. Також відображається список уже створених курсів із кнопками перегляду, редагування та видалення. Під час тестування перевірено додавання нового курсу, відображення його в таблиці та збереження даних у SQLite.

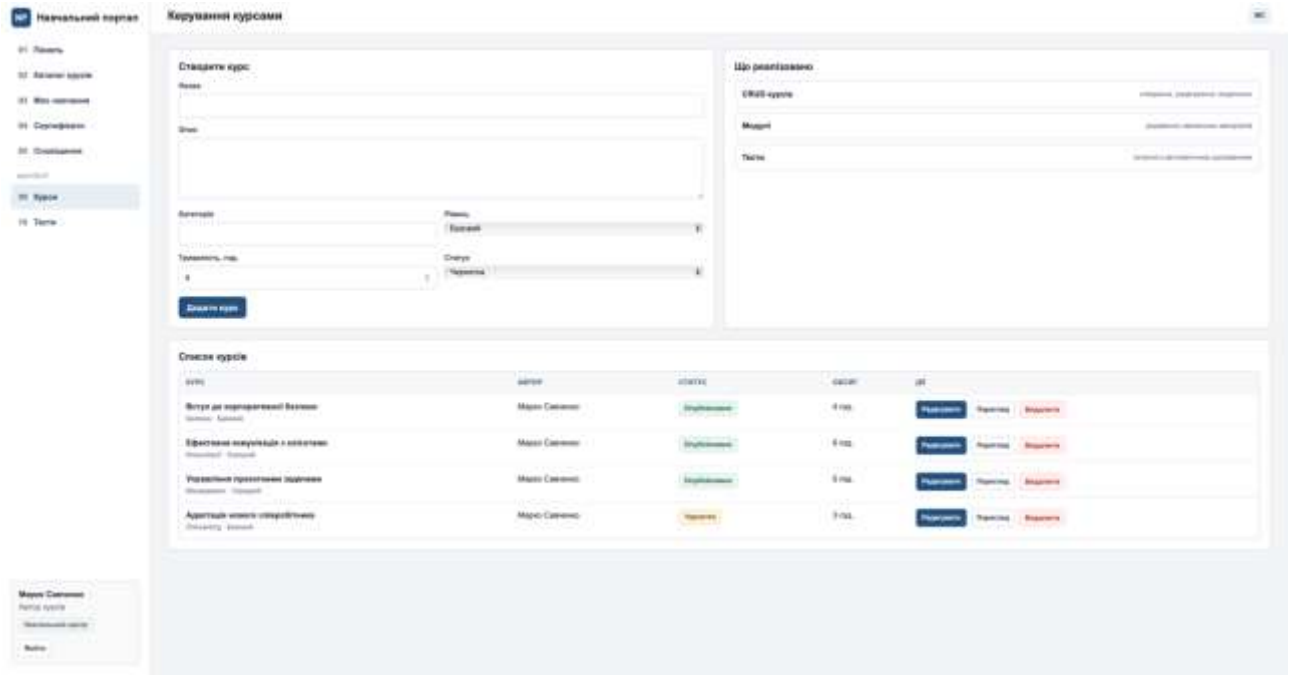

Рисунок 3.5 – Сторінка керування навчальними курсами

Також протестовано модуль керування тестами, показаний на рис. 3.6. Автор курсу може вибрати курс, додати тестове питання, вказати варіанти відповідей і правильну відповідь. На цій сторінці також подано правила

оцінювання: кожне питання має 1 бал, а тест вважається складеним, якщо користувач набрав не менше 60 % від максимальної кількості балів. Перевірка підтвердила, що форма приймає дані, а структура тестових питань відповідає логіці подальшого автоматичного оцінювання.

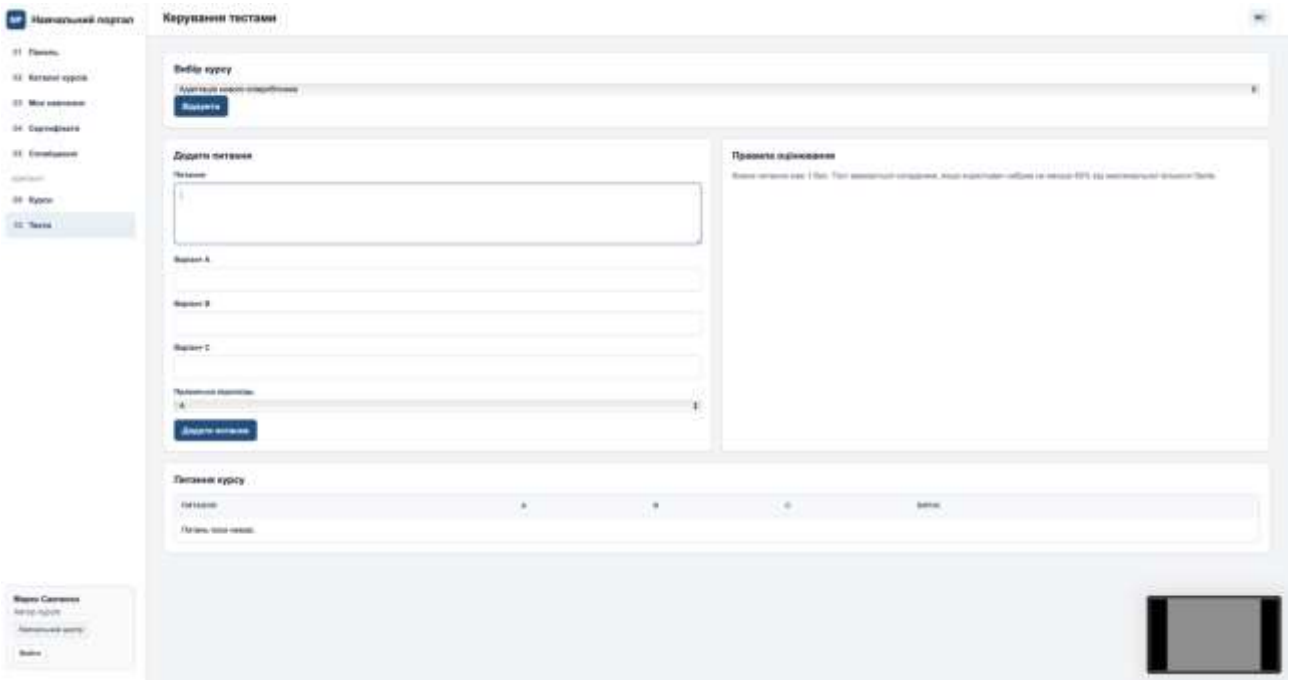

Рисунок 3.6 – Сторінка керування тестовими питаннями

Для HR-менеджера протестовано аналітичну панель, наведену на рис. 3.7. На сторінці відображаються загальні показники системи: кількість користувачів, курсів, записів на навчання та сертифікатів. Також показано статистику записів на курси, дані за підрозділами та ефективність курсів. Під час тестування перевірено, що аналітичні дані формуються на основі записів у базі даних, а таблиці й графічні блоки оновлюються відповідно до стану системи.

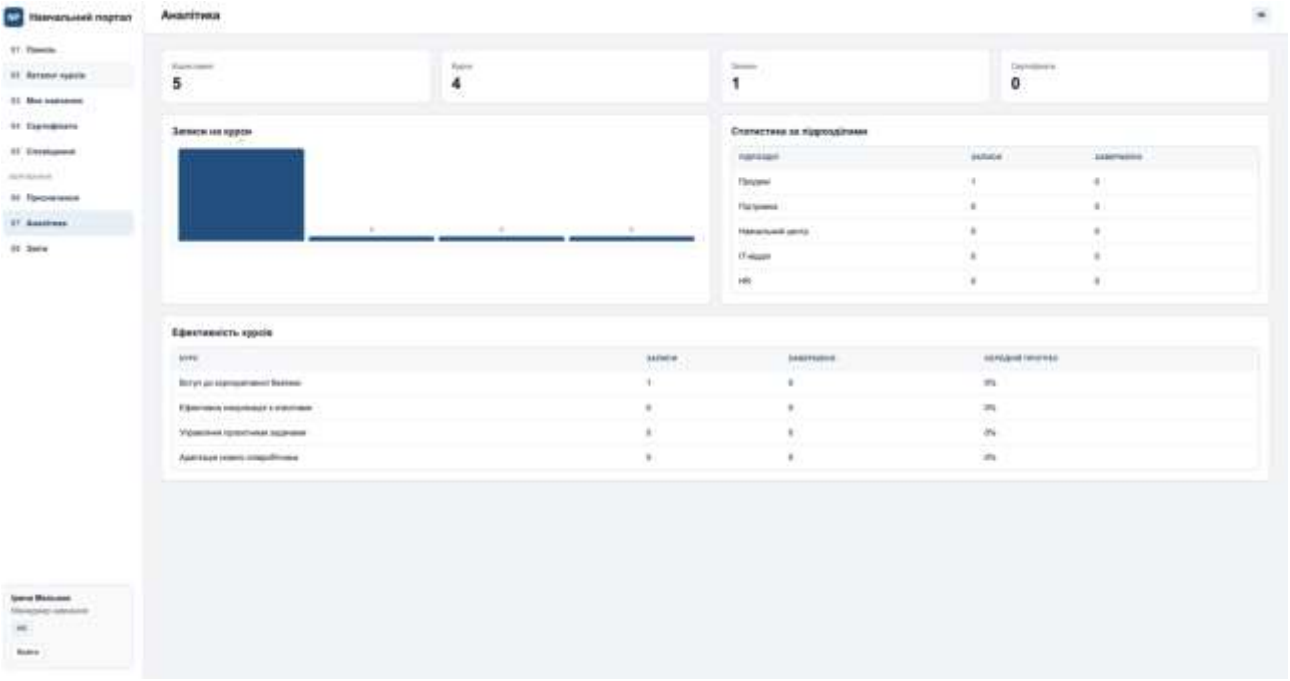

Рисунок 3.7 – Аналітична панель HR-менеджера

Результати тестування основних функцій системи наведено в таблиці 3.2.

Таблиця 3.2 – Результати тестування програмної системи

№	Сценарій тестування	Очікуваний результат	Фактичний результат	Статус
1	Вхід користувача за e-mail і паролем	Система відкриває кабінет відповідної ролі	Кабінет користувача відкрито	Успішно
2	Перехід між пунктами бокового меню	Відкриваються потрібні сторінки системи	Навігація працює коректно	Успішно
3	Перегляд каталогу курсів	Відображаються доступні курси з бази даних	Курси завантажені та показані картками	Успішно
4	Запис співробітника на курс	Створюється запис Enrollment	Курс з’являється у розділі «Моє навчання»	Успішно
5	Створення курсу автором	Новий курс зберігається та додається до списку	Курс відображається в таблиці курсів	Успішно
6	Додавання тестового питання	Питання зберігається для вибраного курсу	Форма тесту працює коректно	Успішно
7	Перегляд аналітики HR-менеджером	Показники формуються за даними системи	Дані відображаються в таблицях і графіку	Успішно

Проведене тестування показало, що основні модулі веб-застосунку працюють узгоджено. Авторизація забезпечує доступ до потрібного кабінету, каталог курсів коректно відображає навчальні матеріали, модуль запису створює зв’язок між користувачем і курсом, а сторінки автора та HR-менеджера підтримують керування курсами, тестами й аналітичними даними.

За результатами перевірки критичних помилок, які перешкоджають роботі системи, не виявлено. Програмна система виконує базові функції корпоративного навчання та може бути використана як робочий прототип для демонстрації процесів авторизації, керування курсами, проходження навчання, тестування й аналізу результатів.

3.3 Розгортання системи, склад інсталяційного пакету

Розгортання системи корпоративного навчання виконується з метою підготовки веб-застосунку до практичного використання в локальному або корпоративному середовищі. Розроблена система має клієнт-серверну структуру: користувач працює через браузер, серверна частина обробляє запити, виконує авторизацію, керує навчальними курсами, тестами, записами на навчання та сертифікатами, а база даних SQLite забезпечує збереження інформації.

На рисунку 3.8 наведено діаграму розгортання системи корпоративного навчання.

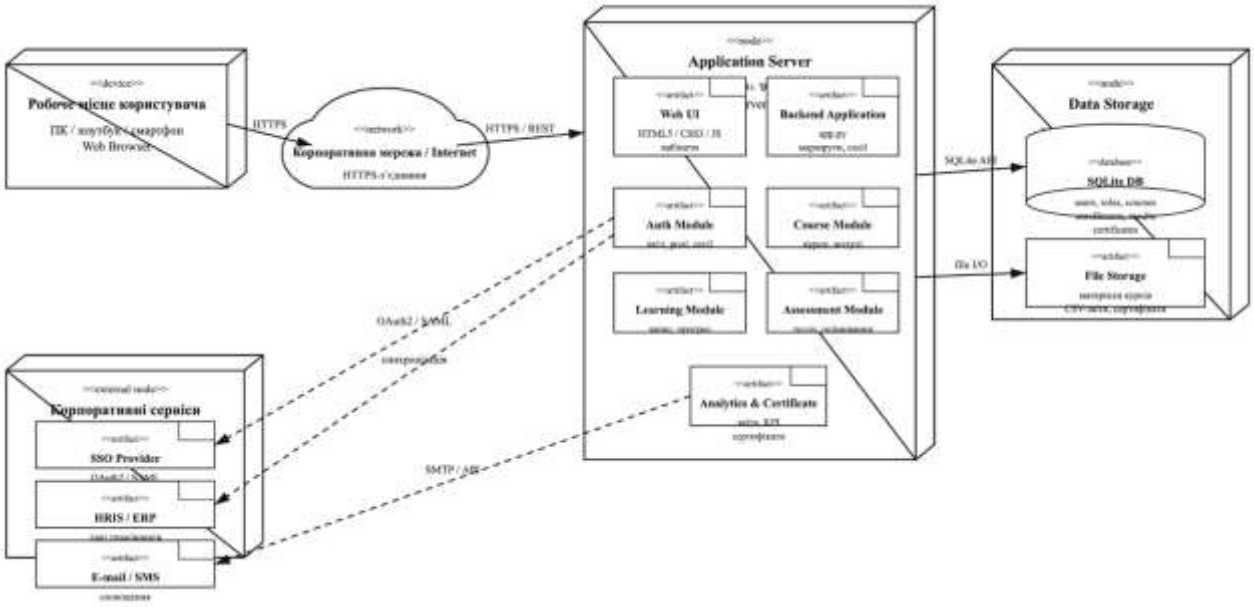

Рисунок 3.8 – Діаграма розгортання системи корпоративного навчання

Відповідно до діаграми, користувачі працюють із системою через робоче місце користувача: персональний комп’ютер, ноутбук або смартфон із веб-браузером. Підключення до серверної частини здійснюється через корпоративну мережу або Internet за протоколом HTTPS. Такий підхід дозволяє використовувати систему без встановлення окремого клієнтського програмного забезпечення на пристрої користувача.

Основним вузлом розгортання є Application Server. На ньому розміщується веб-застосунок, реалізований засобами Python 3.9+, HTML5, CSS3, JavaScript та SQLite. У межах серверного вузла функціонують такі програмні модулі: Web UI, Backend Application, Auth Module, Course Module, Learning Module, Assessment Module та Analytics & Certificate. Кожен модуль відповідає за окрему частину функціональності системи.

Модуль Web UI забезпечує відображення сторінок користувачів, особистих кабінетів, каталогу курсів, сторінок навчання, аналітики та керування системою. Backend Application відповідає за маршрутизацію запитів, обробку форм, роботу із сесіями та виконання основної бізнес-логіки. Auth Module реалізує авторизацію користувачів, визначення ролей і перевірку доступу до сторінок. Course Module забезпечує створення, редагування та перегляд

навчальних курсів і модулів. Learning Module відповідає за запис на курси, відкриття навчальних матеріалів і оновлення прогресу. Assessment Module виконує обробку тестових питань, перевірку відповідей і збереження результатів. Analytics & Certificate формує аналітичні показники, звіти та сертифікати.

Рівень збереження даних представлено вузлом Data Storage. До його складу входить SQLite DB та файлове сховище. У базі SQLite зберігаються користувачі, ролі, курси, модулі, записи на навчання, результати тестування та сертифікати. File Storage використовується для збереження навчальних матеріалів, CSV-звітів і файлів сертифікатів. Для програмного прототипу така структура є достатньою, оскільки вона не потребує окремого серверу бази даних і дозволяє швидко переносити систему між комп'ютерами.

У перспективі система може взаємодіяти із зовнішніми корпоративними сервісами: SSO Provider, HRIS / ERP та E-mail / SMS. SSO Provider може використовуватися для єдиного входу користувачів через OAuth2 або SAML. HRIS / ERP забезпечує синхронізацію даних про працівників, їхні посади та підрозділи. E-mail / SMS використовується для надсилання повідомлень про призначення курсу, результати тестування або формування сертифіката. У межах поточного програмного прототипу ці інтеграції можуть бути реалізовані як підготовлені модулі або імітовані службовими повідомленнями в системі.

Мінімальні вимоги до середовища розгортання наведено в таблиці 3.3.

Таблиця 3.3 – Вимоги до середовища розгортання системи

Компонент	Мінімальні вимоги
Операційна система	Windows 10/11, Linux або macOS
Інтерпретатор	Python 3.9 або новіший
База даних	SQLite
Браузер	Google Chrome, Mozilla Firefox, Microsoft Edge, Safari
Процесор	2 ядра або більше
Оперативна пам'ять	від 4 ГБ
Дисковий простір	від 200 МБ для застосунку, БД і файлів
Мережа	локальна мережа або доступ через HTTP/HTTPS

Для запуску системи необхідно розпакувати інсталяційний пакет, перейти до каталогу веб-застосунку та запустити основний файл серверної частини. Після запуску сервер відкриває локальну адресу, наприклад http://127.0.0.1:8000, за якою користувач може працювати із системою через браузер. Якщо застосунок розгортається на сервері організації, доступ до нього може бути надано іншим користувачам через локальну мережу або доменне ім’я.

Порядок розгортання системи наведено нижче:

- Розпакувати інсталяційний пакет у робочий каталог.
- Перевірити наявність Python 3.9+.
- Запустити файл app.py.
- Дочекатися створення або підключення бази даних SQLite.
- Відкрити веб-застосунок у браузері.
- Виконати вхід під тестовим або створеним обліковим записом.
- Перевірити доступ до курсів, тестів, аналітики та сертифікатів.

Склад інсталяційного пакету наведено в таблиці 3.4.

Таблиця 3.4 – Склад інсталяційного пакету системи корпоративного навчання

Елемент пакету	Призначення
app.py	Основний файл серверної частини веб-застосунку
learning_portal.db	Файл бази даних SQLite
static/	CSS, JavaScript, зображення та інші статичні файли
templates/	HTML-шаблони сторінок системи
uploads/	Каталог для навчальних матеріалів і службових файлів
reports/	Каталог для сформованих CSV-звітів
certificates/	Каталог для файлів сертифікатів
README.txt	Коротка інструкція із запуску системи

Після розгортання користувач може виконати вхід до системи, переглянути каталог курсів, записатися на навчання, пройти модулі, скласти тест і переглянути результати. Автор курсу отримує доступ до сторінок керування курсами й тестами, HR-менеджер - до аналітики та звітів, адміністратор - до службових розділів керування системою.

Запропонована схема розгортання є простою та придатною для демонстрації програмного прототипу. Водночас вона залишає можливість подальшого розширення: перенесення бази даних із SQLite на PostgreSQL або MySQL, налаштування HTTPS-сертифіката, підключення корпоративного SSO, інтеграції з HRIS / ERP та розгортання системи на віддаленому сервері. Таким чином, розроблена система може використовуватися як локальний навчальний веб-застосунок або як основа для подальшого впровадження в корпоративному середовищі.

3.4 Висновки до третього розділу

У третьому розділі виконано реалізацію та тестування системи корпоративного навчання на основі веб-технологій. На цьому етапі було перевірено працездатність основних програмних модулів, визначено порядок тестування системи, проведено функціональну перевірку веб-застосунку та описано особливості його розгортання.

У межах плану тестування було визначено основні модулі, які підлягають перевірці: модуль авторизації, модуль ролей і доступу, модуль керування курсами, модуль навчальних матеріалів, модуль запису на курс, модуль проходження навчання, модуль тестування, модуль сертифікації, модуль аналітики, а також модуль сповіщень і журналу подій. Для кожного модуля визначено очікуваний результат, що дало змогу системно перевірити відповідність реалізованого функціоналу поставленим вимогам.

Під час тестування програмної системи перевірено роботу сторінки входу, панелі керування користувача, каталогу курсів, розділу «Моє навчання»,

сторінки керування курсами, модуля створення тестових питань та аналітичної панелі HR-менеджера. Результати перевірки показали, що система коректно обробляє вхід користувача, визначає його роль, відкриває відповідні сторінки, завантажує дані з бази SQLite, забезпечує запис на курс і відображає актуальні навчальні показники.

Окремо було описано розгортання системи та склад інсталяційного пакету. Визначено, що система може працювати як локальний веб-застосунок або бути розміщена на сервері організації. Основними елементами розгортання є робоче місце користувача, корпоративна мережа або Internet, сервер застосунку, база даних SQLite, файлове сховище та зовнішні корпоративні сервіси. До складу інсталяційного пакету входять основний файл серверної частини, база даних, HTML-шаблони, статичні файли, каталоги для звітів, сертифікатів і навчальних матеріалів.

Проведене тестування підтвердило, що розроблена система виконує основні функції корпоративного навчання: авторизацію користувачів, розмежування доступу за ролями, перегляд і створення курсів, запис на навчання, проходження модулів, підготовку тестів, формування аналітичних даних і роботу з сертифікатами. Критичних помилок, які унеможливлюють використання системи, не виявлено.

Отже, у третьому розділі підтверджено працездатність розробленого веб-застосунку та його відповідність основним функціональним вимогам. Система може використовуватися як програмний прототип корпоративного навчального порталу та є основою для подальшого розширення, зокрема підключення корпоративного SSO, інтеграції з HRIS / ERP, перенесення бази даних на PostgreSQL або MySQL і розгортання в реальному корпоративному середовищі.

ВИСНОВКИ

У кваліфікаційній роботі вирішено практичне завдання проєктування та розроблення системи корпоративного навчання на основі веб-технологій. Розроблена система призначена для автоматизації процесів навчання персоналу, керування навчальними курсами, контролю проходження матеріалів, оцінювання результатів і формування аналітичної інформації для HR-менеджерів та адміністраторів.

У першому розділі проведено аналіз предметної області корпоративного навчання. Визначено, що сучасні організації потребують гнучких веб-орієнтованих засобів для організації навчального процесу, оскільки традиційні підходи не завжди забезпечують оперативне оновлення навчальних матеріалів, контроль результатів і централізоване збереження даних. Було розглянуто основні ролі користувачів системи: співробітник, автор курсу, HR-менеджер та адміністратор. Також виконано аналіз існуючих рішень, зокрема Moodle, Google Classroom, Canvas LMS, Blackboard Learn та Blend-ed Platform. На основі порівняння визначено доцільність розроблення власної веб-системи, орієнтованої на компактну структуру, рольову модель доступу, просте керування курсами та вбудовану аналітику.

У процесі системного аналізу побудовано DFD-модель, BPMN-діаграму процесів, UML-діаграми прецедентів, послідовності та активності. Ці моделі дозволили формалізувати основні сценарії роботи системи: авторизацію користувачів, перегляд каталогу курсів, запис на навчання, проходження модулів, тестування, формування сертифікатів і перегляд аналітичних звітів. Також сформовано функціональні, нефункціональні та безпекові вимоги до програмної системи.

У другому розділі виконано проєктування інформаційного та програмного забезпечення системи. Розроблено логічну модель даних, яка охоплює основні сутності: Role, User, Course, CourseModule, Enrollment, TestResult та Certificate.

Така структура забезпечує збереження інформації про користувачів, ролі, курси, модулі, записи на навчання, результати тестування та сертифікати без дублювання даних.

Також побудовано UML-діаграму класів і кооперації, які деталізують взаємодію об’єктів у процесах запису співробітника на курс, створення навчального курсу, тестування та формування сертифіката. Для подання архітектури програмного забезпечення розроблено діаграму компонентів і діаграму пакетів. У них систему подано як набір взаємопов’язаних модулів: клієнтський інтерфейс, серверна логіка, модулі авторизації, керування курсами, навчання, тестування, аналітики, доступу до даних та інтеграцій.

Для реалізації системи обрано простий і практичний стек технологій: Python 3.9+, SQLite, HTML5, CSS3 та JavaScript. Такий вибір дав змогу реалізувати веб-застосунок без надмірного ускладнення, забезпечити локальне збереження даних і створити зрозумілий інтерфейс для основних ролей користувачів. Додатково розроблено блок-схеми алгоритмів авторизації, запису й проходження курсу, тестування та формування сертифіката.

У третьому розділі виконано реалізацію та тестування програмної системи. Було створено веб-застосунок із базою даних SQLite, авторизацією користувачів, розмежуванням доступу за ролями, каталогом курсів, сторінкою «Моє навчання», модулем керування курсами, модулем тестів, аналітичною панеллю та механізмом роботи із сертифікатами. Інтерфейс системи реалізовано у стриманому корпоративному стилі, що відповідає призначенню системи.

Під час тестування перевірено основні сценарії роботи програмної системи: вхід користувача, перехід між сторінками, перегляд каталогу курсів, запис на курс, відображення активного навчання, створення курсів автором, додавання тестових питань і перегляд аналітики HR-менеджером. Результати тестування підтвердили, що система коректно виконує основні функції, зберігає дані в базі, відображає актуальну інформацію та забезпечує роботу відповідно до ролі користувача.

Також розроблено діаграму розгортання системи та визначено склад інсталяційного пакету. Система може бути розгорнута локально або на сервері організації. До складу пакету входять серверний файл застосунку, база даних, HTML-шаблони, статичні файли, каталоги для навчальних матеріалів, звітів і сертифікатів, а також інструкція із запуску.

Отже, поставлену мету кваліфікаційної роботи досягнуто. Було спроєктовано та реалізовано веб-систему корпоративного навчання, яка забезпечує керування користувачами, курсами, навчальними модулями, записами на навчання, тестуванням, сертифікатами та аналітичними показниками. Розроблена система може використовуватися як програмний прототип корпоративного навчального порталу та бути основою для подальшого розвитку.

Подальше вдосконалення системи може передбачати підключення корпоративного SSO, інтеграцію з HRIS / ERP, перенесення бази даних із SQLite на PostgreSQL або MySQL, додавання адаптивних навчальних траєкторій, розширення аналітики та впровадження автоматичних рекомендацій курсів для працівників.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ДСТУ ISO/IEC/IEEE 12207:2018. Інженерія систем і програмних засобів. Процеси життєвого циклу програмних засобів. Київ : ДП «УкрНДНЦ», 2018.

2. ДСТУ ISO/IEC 25010:2016. Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (SQuaRE). Моделі якості системи та програмних засобів. Київ : ДП «УкрНДНЦ», 2016.

3. ISO/IEC/IEEE 29148:2018. Systems and software engineering — Life cycle processes — Requirements engineering. Geneva : ISO, 2018.

4. ISO/IEC 27001:2022. Information security, cybersecurity and privacy protection — Information security management systems — Requirements. Geneva : ISO, 2022.

5. ISO/IEC 27002:2022. Information security, cybersecurity and privacy protection — Information security controls. Geneva : ISO, 2022.

6. OMG Unified Modeling Language. Version 2.5.1. Object Management Group, 2017. URL: https://www.omg.org/spec/UML/2.5.1

7. OMG Business Process Model and Notation. Version 2.0.2. Object Management Group, 2014. URL: https://www.omg.org/spec/BPMN/2.0.2

8. OWASP Application Security Verification Standard. OWASP Foundation. URL: https://owasp.org/www-project-application-security-verification-standard/

9. OWASP Top 10: The Ten Most Critical Web Application Security Risks. OWASP Foundation. URL: https://owasp.org/www-project-top-ten/

10. Fielding R., Nottingham M., Reschke J. HTTP Semantics. RFC 9110. IETF, 2022. URL: https://www.rfc-editor.org/info/rfc9110

11. Hardt D. The OAuth 2.0 Authorization Framework. RFC 6749. IETF, 2012. URL: https://www.rfc-editor.org/rfc/rfc6749

12. Security Assertion Markup Language (SAML) v2.0. OASIS Open, 2005. URL: https://www.oasis-open.org/standard/saml/

13. Python 3.9 Documentation. Python Software Foundation. URL: https://docs.python.org/3.9/

14. SQLite Documentation. SQLite Consortium. URL: https://www.sqlite.org/docs.html

15. SQLite Foreign Key Support. SQLite Consortium. URL: https://www.sqlite.org/foreignkeys.html

16. HTML: HyperText Markup Language. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/HTML

17. CSS Reference. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/CSS/Reference

18. JavaScript Guide. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide

19. Moodle Developer Documentation. MoodleDocs. URL: https://docs.moodle.org/dev/Main_Page

20. Moodle Web Service API Functions. MoodleDocs. URL: https://docs.moodle.org/dev/Web_service_API_functions

21. Using Web Services. MoodleDocs. URL: https://docs.moodle.org/en/Using_web_services

22. Google Classroom Help. Google for Education. URL: https://support.google.com/edu/classroom/

23. Canvas LMS REST API Documentation. Instructure. URL: https://canvas.instructure.com/doc/api/

24. About Blackboard REST APIs. Anthology Developer Documentation. URL: https://docs.anthology.com/docs/blackboard/rest-apis/start-here

25. Blackboard's REST API Framework. Anthology Developer Documentation. URL: https://docs.anthology.com/docs/blackboard/rest-apis/getting-started/framework

26. Basic Authentication with REST. Blackboard Developer Docs. URL: https://docs.anthology.com/docs/blackboard/rest-apis/getting-started/basic-authentication

27. Long P., Siemens G. Penetrating the Fog: Analytics in Learning and Education. EDUCAUSE Review. 2011. Vol. 46, № 5. P. 31–40. URL: https://er.educause.edu/articles/2011/9/penetrating-the-fog-analytics-in-learning-and-education

28. Siemens G. Learning Analytics: The Emergence of a Discipline. American Behavioral Scientist. 2013. Vol. 57, № 10. P. 1380–1400. DOI: https://doi.org/10.1177/0002764213498851

29. Al-Fraihat D., Joy M., Masa'deh R., Sinclair J. Evaluating E-learning Systems Success: An Empirical Study. Computers in Human Behavior. 2020. Vol. 102. P. 67–86. DOI: https://doi.org/10.1016/j.chb.2019.08.004

30. Aparicio M., Bacao F., Oliveira T. An e-Learning Theoretical Framework. Educational Technology & Society. 2016. Vol. 19, № 1. P. 292–307. URL: https://www.jstor.org/stable/jeductechsoci.19.1.292

31. Picciano A. G. Theories and Frameworks for Online Education: Seeking an Integrated Model. Online Learning. 2017. Vol. 21, № 3. P. 166–190. DOI: https://doi.org/10.24059/olj.v21i3.1225

32. Advanced Distributed Learning Initiative. SCORM 2004 4th Edition Documentation. URL: https://adlnet.gov/projects/scorm-2004-4th-edition/

33. IMS Global Learning Consortium. Learning Tools Interoperability Core Specification. URL: https://www.imsglobal.org/spec/lti/v1p3/

34. W3C. Web Content Accessibility Guidelines (WCAG) 2.2. URL: https://www.w3.org/TR/WCAG22/

ДОДАТОК А

Фрагмент коду створення бази даних системи

```
import sqlite3
from hashlib import sha256
from pathlib import Path

DB_PATH = Path("learning_portal.db")


def get_connection():
    conn = sqlite3.connect(DB_PATH)
    conn.row_factory = sqlite3.Row
    return conn


def hash_password(password: str) -> str:
    return sha256(password.encode("utf-8")).hexdigest()


def init_database():
    conn = get_connection()
    cursor = conn.cursor()

    cursor.execute("""
        CREATE TABLE IF NOT EXISTS roles (
            role_id INTEGER PRIMARY KEY AUTOINCREMENT,
            role_name TEXT NOT NULL UNIQUE,
            description TEXT
        )
    """)

    cursor.execute("""
        CREATE TABLE IF NOT EXISTS users (
            user_id INTEGER PRIMARY KEY AUTOINCREMENT,
            role_id INTEGER NOT NULL,
            full_name TEXT NOT NULL,
            email TEXT NOT NULL UNIQUE,
            password_hash TEXT NOT NULL,
            department TEXT,
            position TEXT,
            status TEXT DEFAULT 'active',
            created_at TEXT DEFAULT CURRENT_TIMESTAMP,
            FOREIGN KEY (role_id) REFERENCES roles(role_id)
        )
    """)

    cursor.execute("""
        CREATE TABLE IF NOT EXISTS courses (
            course_id INTEGER PRIMARY KEY AUTOINCREMENT,
            author_id INTEGER NOT NULL,
            title TEXT NOT NULL,
            description TEXT,
            category TEXT,
            level TEXT,
            duration_hours INTEGER DEFAULT 0,
            status TEXT DEFAULT 'draft',
            created_at TEXT DEFAULT CURRENT_TIMESTAMP,
```

```
        FOREIGN KEY (author_id) REFERENCES users(user_id)
    )
""")

cursor.execute("""
    CREATE TABLE IF NOT EXISTS course_modules (
        module_id INTEGER PRIMARY KEY AUTOINCREMENT,
        course_id INTEGER NOT NULL,
        title TEXT NOT NULL,
        content_type TEXT DEFAULT 'text',
        content_url TEXT,
        content_text TEXT,
        order_number INTEGER DEFAULT 1,
        FOREIGN KEY (course_id) REFERENCES courses(course_id)
    )
""")

cursor.execute("""
    CREATE TABLE IF NOT EXISTS enrollments (
        enrollment_id INTEGER PRIMARY KEY AUTOINCREMENT,
        user_id INTEGER NOT NULL,
        course_id INTEGER NOT NULL,
        assigned_by INTEGER,
        status TEXT DEFAULT 'assigned',
        progress INTEGER DEFAULT 0,
        started_at TEXT DEFAULT CURRENT_TIMESTAMP,
        completed_at TEXT,
        FOREIGN KEY (user_id) REFERENCES users(user_id),
        FOREIGN KEY (course_id) REFERENCES courses(course_id),
        FOREIGN KEY (assigned_by) REFERENCES users(user_id),
        UNIQUE(user_id, course_id)
    )
""")

cursor.execute("""
    CREATE TABLE IF NOT EXISTS test_questions (
        question_id INTEGER PRIMARY KEY AUTOINCREMENT,
        course_id INTEGER NOT NULL,
        question_text TEXT NOT NULL,
        option_a TEXT NOT NULL,
        option_b TEXT NOT NULL,
        option_c TEXT NOT NULL,
        option_d TEXT NOT NULL,
        correct_option TEXT NOT NULL,
        FOREIGN KEY (course_id) REFERENCES courses(course_id)
    )
""")

cursor.execute("""
    CREATE TABLE IF NOT EXISTS test_results (
        result_id INTEGER PRIMARY KEY AUTOINCREMENT,
        enrollment_id INTEGER NOT NULL,
        course_id INTEGER NOT NULL,
        score INTEGER NOT NULL,
        max_score INTEGER NOT NULL,
        passed INTEGER NOT NULL,
        attempt_number INTEGER DEFAULT 1,
        completed_at TEXT DEFAULT CURRENT_TIMESTAMP,
        FOREIGN KEY (enrollment_id) REFERENCES enrollments(enrollment_id),
        FOREIGN KEY (course_id) REFERENCES courses(course_id)
    )
```

```
""")

cursor.execute("""
    CREATE TABLE IF NOT EXISTS certificates (
        certificate_id INTEGER PRIMARY KEY AUTOINCREMENT,
        enrollment_id INTEGER NOT NULL UNIQUE,
        certificate_number TEXT NOT NULL UNIQUE,
        issued_at TEXT DEFAULT CURRENT_TIMESTAMP,
        file_url TEXT,
        status TEXT DEFAULT 'issued',
        FOREIGN KEY (enrollment_id) REFERENCES enrollments(enrollment_id)
    )
""")

cursor.execute("""
    CREATE TABLE IF NOT EXISTS notifications (
        notification_id INTEGER PRIMARY KEY AUTOINCREMENT,
        user_id INTEGER NOT NULL,
        message TEXT NOT NULL,
        is_read INTEGER DEFAULT 0,
        created_at TEXT DEFAULT CURRENT_TIMESTAMP,
        FOREIGN KEY (user_id) REFERENCES users(user_id)
    )
""")

cursor.execute("""
    CREATE TABLE IF NOT EXISTS activity_log (
        log_id INTEGER PRIMARY KEY AUTOINCREMENT,
        user_id INTEGER,
        action TEXT NOT NULL,
        details TEXT,
        created_at TEXT DEFAULT CURRENT_TIMESTAMP,
        FOREIGN KEY (user_id) REFERENCES users(user_id)
    )
""")

roles = [
    ("employee", "Співробітник, який проходить навчання"),
    ("author", "Автор навчальних курсів"),
    ("hr", "HR-менеджер з навчання персоналу"),
    ("admin", "Адміністратор системи")
]

cursor.executemany("""
    INSERT OR IGNORE INTO roles (role_name, description)
    VALUES (?, ?)
""", roles)

demo_users = [
    (1, "Іван Петренко", "employee@corp.local", hash_password("123456"), "Відділ продажів", "Менеджер"),
    (2, "Олена Коваль", "author@corp.local", hash_password("123456"), "Навчальний відділ", "Автор курсів"),
    (3, "Марина Шевченко", "hr@corp.local", hash_password("123456"), "HR", "HR-менеджер"),
    (4, "Адміністратор", "admin@corp.local", hash_password("123456"), "IT", "Системний адміністратор")
]

cursor.executemany("""
    INSERT OR IGNORE INTO users
    (role_id, full_name, email, password_hash, department, position)
    VALUES (?, ?, ?, ?, ?, ?)
""", demo_users)
```

```
    demo_courses = [
        (2, "Основи корпоративної безпеки", "Курс для ознайомлення працівників з правилами інформаційної безпеки.", "Безпека", "Базовий", 4, "active"),
        (2, "Ефективна комунікація в команді", "Навчальний курс для розвитку навичок командної взаємодії.", "Комунікації", "Середній", 6, "active"),
        (2, "Вступ до корпоративної культури", "Курс для нових співробітників компанії.", "Адаптація", "Базовий", 3, "active")
    ]

    cursor.executemany("""
        INSERT OR IGNORE INTO courses
        (author_id, title, description, category, level, duration_hours, status)
        VALUES (?, ?, ?, ?, ?, ?, ?)
    """, demo_courses)

    conn.commit()
    conn.close()


if __name__ == "__main__":
    init_database()
    print("Базу даних створено успішно.")
```

ДОДАТОК Б

Фрагмент коду авторизації, ролей і роботи з курсами

```
from http.server import HTTPServer, BaseHTTPRequestHandler
from urllib.parse import parse_qs
from http.cookies import SimpleCookie
import sqlite3
import secrets
from hashlib import sha256

DB_NAME = "learning_portal.db"
sessions = {}


def db():
    conn = sqlite3.connect(DB_NAME)
    conn.row_factory = sqlite3.Row
    return conn


def password_hash(password):
    return sha256(password.encode("utf-8")).hexdigest()


def create_session(user_id):
    token = secrets.token_hex(32)
    sessions[token] = user_id
    return token


def get_user_by_session(headers):
    cookie_header = headers.get("Cookie")
    if not cookie_header:
        return None

    cookie = SimpleCookie(cookie_header)
    token = cookie.get("session")

    if not token:
        return None

    user_id = sessions.get(token.value)
    if not user_id:
        return None

    conn = db()
    user = conn.execute("""
        SELECT users.*, roles.role_name
        FROM users
        JOIN roles ON users.role_id = roles.role_id
        WHERE users.user_id = ?
    """, (user_id,)).fetchone()
    conn.close()

    return user


def log_action(user_id, action, details=""):
```

```
    conn = db()
    conn.execute("""
        INSERT INTO activity_log (user_id, action, details)
        VALUES (?, ?, ?)
    """, (user_id, action, details))
    conn.commit()
    conn.close()


def render_layout(title, user, content):
    user_name = user["full_name"] if user else "Гість"
    role = user["role_name"] if user else ""

    return f"""
    <!DOCTYPE html>
    <html lang="uk">
    <head>
        <meta charset="UTF-8">
        <title>{title}</title>
        <style>
            body {{
                margin: 0;
                font-family: Arial, sans-serif;
                background: #f4f6f8;
                color: #222;
            }}
            .layout {{
                display: grid;
                grid-template-columns: 240px 1fr;
                min-height: 100vh;
            }}
            .sidebar {{
                background: #1f2933;
                color: white;
                padding: 24px;
            }}
            .sidebar h2 {{
                font-size: 18px;
                margin-bottom: 24px;
            }}
            .sidebar a {{
                display: block;
                color: #d9e2ec;
                text-decoration: none;
                padding: 10px 0;
                border-bottom: 1px solid #334150;
            }}
            .main {{
                padding: 28px;
            }}
            .topbar {{
                display: flex;
                justify-content: space-between;
                margin-bottom: 24px;
            }}
            .card {{
                background: white;
                border: 1px solid #d8dee4;
                border-radius: 8px;
                padding: 18px;
                margin-bottom: 16px;
```

```
        }}
        .btn {{
            display: inline-block;
            background: #24527a;
            color: white;
            text-decoration: none;
            border: 0;
            border-radius: 6px;
            padding: 9px 14px;
            cursor: pointer;
        }}
        .btn-light {{
            background: #e5eaf0;
            color: #1f2933;
        }}
        input, select, textarea {{
            width: 100%;
            padding: 9px;
            margin: 6px 0 12px;
            border: 1px solid #c8d0d8;
            border-radius: 5px;
            box-sizing: border-box;
        }}
        table {{
            width: 100%;
            border-collapse: collapse;
            background: white;
        }}
        th, td {{
            border: 1px solid #d8dee4;
            padding: 10px;
            text-align: left;
        }}
        th {{
            background: #edf2f7;
        }}
    </style>
</head>
<body>
    <div class="layout">
        <aside class="sidebar">
            <h2>Навчальний портал</h2>
            <a href="/">Головна</a>
            <a href="/courses">Каталог курсів</a>
            <a href="/my-learning">Моє навчання</a>
            <a href="/manage-courses">Керування курсами</a>
            <a href="/analytics">Аналітика</a>
            <a href="/logout">Вийти</a>
        </aside>
        <main class="main">
            <div class="topbar">
                <div>
                    <h1>{title}</h1>
                </div>
                <div>{user_name} / {role}</div>
            </div>
            {content}
        </main>
    </div>
</body>
</html>
```

```
    """


class LearningHandler(BaseHTTPRequestHandler):
    def send_html(self, html, status=200, cookie=None):
        self.send_response(status)
        self.send_header("Content-Type", "text/html; charset=utf-8")
        if cookie:
            self.send_header("Set-Cookie", cookie)
        self.end_headers()
        self.wfile.write(html.encode("utf-8"))

    def redirect(self, path, cookie=None):
        self.send_response(302)
        self.send_header("Location", path)
        if cookie:
            self.send_header("Set-Cookie", cookie)
        self.end_headers()

    def read_post(self):
        length = int(self.headers.get("Content-Length", 0))
        raw = self.rfile.read(length).decode("utf-8")
        data = parse_qs(raw)
        return {key: value[0] for key, value in data.items()}

    def do_GET(self):
        user = get_user_by_session(self.headers)

        if self.path == "/login":
            self.login_page()
        elif self.path == "/logout":
            self.redirect("/login", "session=; Max-Age=0; Path=/")
        elif not user:
            self.redirect("/login")
        elif self.path == "/":
            self.dashboard(user)
        elif self.path == "/courses":
            self.courses_page(user)
        elif self.path == "/my-learning":
            self.my_learning_page(user)
        elif self.path == "/manage-courses":
            self.manage_courses_page(user)
        elif self.path == "/analytics":
            self.analytics_page(user)
        elif self.path.startswith("/enroll"):
            self.enroll_course(user)
        else:
            self.send_html("Сторінку не знайдено", 404)

    def do_POST(self):
        if self.path == "/login":
            self.login_action()
        elif self.path == "/create-course":
            user = get_user_by_session(self.headers)
            if not user:
                self.redirect("/login")
            else:
                self.create_course(user)
        else:
            self.send_html("Дію не знайдено", 404)
```

```
def login_page(self):
    html = """
    <!DOCTYPE html>
    <html lang="uk">
    <head>
        <meta charset="UTF-8">
        <title>Вхід до системи</title>
        <style>
            body {
                margin: 0;
                font-family: Arial, sans-serif;
                background: #eef2f5;
                display: flex;
                justify-content: center;
                align-items: center;
                height: 100vh;
            }
            .login {
                width: 360px;
                background: white;
                border: 1px solid #d8dee4;
                border-radius: 8px;
                padding: 28px;
            }
            input {
                width: 100%;
                padding: 10px;
                margin-bottom: 14px;
                border: 1px solid #c8d0d8;
                border-radius: 5px;
                box-sizing: border-box;
            }
            button {
                width: 100%;
                padding: 10px;
                background: #24527a;
                color: white;
                border: 0;
                border-radius: 5px;
                cursor: pointer;
            }
        </style>
    </head>
    <body>
        <form class="login" method="POST" action="/login">
            <h2>Система корпоративного навчання</h2>
            <input type="email" name="email" placeholder="E-mail" required>
            <input type="password" name="password" placeholder="Пароль" required>
            <button type="submit">Увійти</button>
            <p>employee@corp.local / 123456</p>
            <p>author@corp.local / 123456</p>
            <p>hr@corp.local / 123456</p>
            <p>admin@corp.local / 123456</p>
        </form>
    </body>
    </html>
    """
    self.send_html(html)

def login_action(self):
    data = self.read_post()
```

```
        email = data.get("email", "")
        password = data.get("password", "")

        conn = db()
        user = conn.execute("""
            SELECT * FROM users
            WHERE email = ? AND password_hash = ? AND status = 'active'
        """, (email, password_hash(password))).fetchone()
        conn.close()

        if not user:
            self.redirect("/login")
            return

        token = create_session(user["user_id"])
        log_action(user["user_id"], "login", "Користувач увійшов до системи")
        self.redirect("/", f"session={token}; Path=/; HttpOnly")

    def dashboard(self, user):
        conn = db()
        courses_count = conn.execute("SELECT COUNT(*) AS c FROM courses").fetchone()["c"]
        enrollments_count = conn.execute("""
            SELECT COUNT(*) AS c FROM enrollments
            WHERE user_id = ?
        """, (user["user_id"],)).fetchone()["c"]
        certificates_count = conn.execute("""
            SELECT COUNT(*) AS c
            FROM certificates
            JOIN enrollments ON certificates.enrollment_id = enrollments.enrollment_id
            WHERE enrollments.user_id = ?
        """, (user["user_id"],)).fetchone()["c"]
        conn.close()

        content = f"""
        <div class="card">
            <h2>Панель керування</h2>
            <p>Доступні курси: {courses_count}</p>
            <p>Мої навчання: {enrollments_count}</p>
            <p>Сертифікати: {certificates_count}</p>
        </div>
        """
        self.send_html(render_layout("Головна", user, content))

    def courses_page(self, user):
        conn = db()
        courses = conn.execute("""
            SELECT courses.*, users.full_name AS author_name
            FROM courses
            JOIN users ON courses.author_id = users.user_id
            WHERE courses.status = 'active'
            ORDER BY courses.created_at DESC
        """).fetchall()
        conn.close()

        rows = ""
        for course in courses:
            rows += f"""
            <div class="card">
                <h3>{course["title"]}</h3>
                <p>{course["description"]}</p>
                <p>Категорія: {course["category"]} | Рівень: {course["level"]} | Тривалість: {course["duration_hours"]}
```

```
год.</p>
            <a class="btn" href="/enroll?id={course["course_id"]}">Записатися</a>
          </div>
          """

        self.send_html(render_layout("Каталог курсів", user, rows))

    def enroll_course(self, user):
        course_id = self.path.split("id=")[-1]

        conn = db()
        conn.execute("""
            INSERT OR IGNORE INTO enrollments
            (user_id, course_id, assigned_by, status, progress)
            VALUES (?, ?, ?, 'in_progress', 0)
        """, (user["user_id"], course_id, user["user_id"]))
        conn.commit()
        conn.close()

        log_action(user["user_id"], "enroll_course", f"Запис на курс ID {course_id}")
        self.redirect("/my-learning")

    def my_learning_page(self, user):
        conn = db()
        items = conn.execute("""
            SELECT enrollments.*, courses.title, courses.category, courses.level
            FROM enrollments
            JOIN courses ON enrollments.course_id = courses.course_id
            WHERE enrollments.user_id = ?
            ORDER BY enrollments.started_at DESC
        """, (user["user_id"],)).fetchall()
        conn.close()

        rows = ""
        for item in items:
            rows += f"""
            <div class="card">
              <h3>{item["title"]}</h3>
              <p>Категорія: {item["category"]} | Рівень: {item["level"]}</p>
              <p>Статус: {item["status"]}</p>
              <p>Прогрес: {item["progress"]}%</p>
              <a class="btn" href="/test?enrollment_id={item["enrollment_id"]}">Перейти до тесту</a>
            </div>
            """

        self.send_html(render_layout("Моє навчання", user, rows))

    def manage_courses_page(self, user):
        if user["role_name"] not in ("author", "admin"):
            self.redirect("/")
            return

        conn = db()
        courses = conn.execute("""
            SELECT * FROM courses
            WHERE author_id = ?
            ORDER BY created_at DESC
        """, (user["user_id"],)).fetchall()
        conn.close()

        rows = ""
```

```
    for course in courses:
        rows += f"""
        <tr>
            <td>{course["title"]}</td>
            <td>{course["category"]}</td>
            <td>{course["level"]}</td>
            <td>{course["status"]}</td>
        </tr>
        """

    content = f"""
    <div class="card">
        <h2>Створення курсу</h2>
        <form method="POST" action="/create-course">
            <input name="title" placeholder="Назва курсу" required>
            <textarea name="description" placeholder="Опис курсу"></textarea>
            <input name="category" placeholder="Категорія" required>
            <select name="level">
                <option value="Базовий">Базовий</option>
                <option value="Середній">Середній</option>
                <option value="Поглиблений">Поглиблений</option>
            </select>
            <input name="duration_hours" type="number" placeholder="Тривалість, год." required>
            <button class="btn" type="submit">Створити курс</button>
        </form>
    </div>
    <table>
        <tr>
            <th>Курс</th>
            <th>Категорія</th>
            <th>Рівень</th>
            <th>Статус</th>
        </tr>
        {rows}
    </table>
    """
    self.send_html(render_layout("Керування курсами", user, content))

def create_course(self, user):
    if user["role_name"] not in ("author", "admin"):
        self.redirect("/")
        return

    data = self.read_post()

    conn = db()
    conn.execute("""
        INSERT INTO courses
        (author_id, title, description, category, level, duration_hours, status)
        VALUES (?, ?, ?, ?, ?, ?, 'active')
    """, (
        user["user_id"],
        data.get("title"),
        data.get("description"),
        data.get("category"),
        data.get("level"),
        data.get("duration_hours")
    ))
    conn.commit()
    conn.close()
```

```
        log_action(user["user_id"], "create_course", data.get("title"))
        self.redirect("/manage-courses")
```

ДОДАТОК В

Фрагмент коду тестування, сертифікації та аналітики

```
import sqlite3
import csv
from datetime import datetime
from pathlib import Path

DB_NAME = "learning_portal.db"
REPORT_DIR = Path("reports")
CERT_DIR = Path("certificates")

REPORT_DIR.mkdir(exist_ok=True)
CERT_DIR.mkdir(exist_ok=True)


def db():
    conn = sqlite3.connect(DB_NAME)
    conn.row_factory = sqlite3.Row
    return conn


def calculate_test_result(course_id: int, submitted_answers: dict):
    conn = db()
    questions = conn.execute("""
        SELECT question_id, correct_option
        FROM test_questions
        WHERE course_id = ?
    """, (course_id,)).fetchall()
    conn.close()

    score = 0
    max_score = len(questions)

    for question in questions:
        qid = str(question["question_id"])
        if submitted_answers.get(qid) == question["correct_option"]:
            score += 1

    percent = 0
    if max_score > 0:
        percent = round((score / max_score) * 100)

    passed = 1 if percent >= 60 else 0

    return {
        "score": score,
        "max_score": max_score,
        "percent": percent,
        "passed": passed
    }


def save_test_result(enrollment_id: int, course_id: int, result: dict):
    conn = db()
```

```
    last_attempt = conn.execute("""
        SELECT COALESCE(MAX(attempt_number), 0) + 1 AS attempt_number
        FROM test_results
        WHERE enrollment_id = ?
    """, (enrollment_id,)).fetchone()["attempt_number"]

    conn.execute("""
        INSERT INTO test_results
        (enrollment_id, course_id, score, max_score, passed, attempt_number)
        VALUES (?, ?, ?, ?, ?, ?)
    """, (
        enrollment_id,
        course_id,
        result["score"],
        result["max_score"],
        result["passed"],
        last_attempt
    ))

    if result["passed"]:
        conn.execute("""
            UPDATE enrollments
            SET status = 'completed',
                progress = 100,
                completed_at = CURRENT_TIMESTAMP
            WHERE enrollment_id = ?
        """, (enrollment_id,))

    conn.commit()
    conn.close()


def generate_certificate(enrollment_id: int):
    conn = db()

    enrollment = conn.execute("""
        SELECT enrollments.enrollment_id,
               users.full_name,
               courses.title
        FROM enrollments
        JOIN users ON enrollments.user_id = users.user_id
        JOIN courses ON enrollments.course_id = courses.course_id
        WHERE enrollments.enrollment_id = ?
          AND enrollments.status = 'completed'
    """, (enrollment_id,)).fetchone()

    if not enrollment:
        conn.close()
        return None

    exists = conn.execute("""
        SELECT *
        FROM certificates
        WHERE enrollment_id = ?
    """, (enrollment_id,)).fetchone()

    if exists:
        conn.close()
        return exists["file_url"]

    certificate_number = f"CERT-{datetime.now().strftime('%Y%m%d')}-{enrollment_id:04d}"
```

```
    file_name = f"{certificate_number}.txt"
    file_path = CERT_DIR / file_name

    certificate_text = f"""
    СЕРТИФІКАТ

    Видано: {enrollment["full_name"]}

    Підтверджує успішне завершення навчального курсу:
    {enrollment["title"]}

    Номер сертифіката: {certificate_number}
    Дата видачі: {datetime.now().strftime("%d.%m.%Y")}
    """

    file_path.write_text(certificate_text, encoding="utf-8")

    conn.execute("""
        INSERT INTO certificates
        (enrollment_id, certificate_number, file_url, status)
        VALUES (?, ?, ?, 'issued')
    """, (enrollment_id, certificate_number, str(file_path)))

    conn.commit()
    conn.close()

    return str(file_path)


def get_analytics():
    conn = db()

    users_count = conn.execute("""
        SELECT COUNT(*) AS value
        FROM users
    """).fetchone()["value"]

    courses_count = conn.execute("""
        SELECT COUNT(*) AS value
        FROM courses
    """).fetchone()["value"]

    enrollments_count = conn.execute("""
        SELECT COUNT(*) AS value
        FROM enrollments
    """).fetchone()["value"]

    completed_count = conn.execute("""
        SELECT COUNT(*) AS value
        FROM enrollments
        WHERE status = 'completed'
    """).fetchone()["value"]

    certificates_count = conn.execute("""
        SELECT COUNT(*) AS value
        FROM certificates
    """).fetchone()["value"]

    avg_score = conn.execute("""
        SELECT ROUND(AVG(score * 100.0 / max_score), 1) AS value
        FROM test_results
```

```
        WHERE max_score > 0
    """).fetchone()["value"]

    conn.close()

    return {
        "users_count": users_count,
        "courses_count": courses_count,
        "enrollments_count": enrollments_count,
        "completed_count": completed_count,
        "certificates_count": certificates_count,
        "avg_score": avg_score or 0
    }


def export_learning_report():
    conn = db()

    rows = conn.execute("""
        SELECT users.full_name,
               users.department,
               courses.title AS course_title,
               enrollments.status,
               enrollments.progress,
               enrollments.started_at,
               enrollments.completed_at
        FROM enrollments
        JOIN users ON enrollments.user_id = users.user_id
        JOIN courses ON enrollments.course_id = courses.course_id
        ORDER BY enrollments.started_at DESC
    """).fetchall()

    conn.close()

    file_path = REPORT_DIR / f"learning_report_{datetime.now().strftime('%Y%m%d_%H%M%S')}.csv"

    with open(file_path, "w", encoding="utf-8", newline="") as file:
        writer = csv.writer(file)
        writer.writerow([
            "Працівник",
            "Підрозділ",
            "Курс",
            "Статус",
            "Прогрес",
            "Дата початку",
            "Дата завершення"
        ])

        for row in rows:
            writer.writerow([
                row["full_name"],
                row["department"],
                row["course_title"],
                row["status"],
                row["progress"],
                row["started_at"],
                row["completed_at"] or ""
            ])

    return str(file_path)
```

```
def create_notification(user_id: int, message: str):
    conn = db()
    conn.execute("""
        INSERT INTO notifications (user_id, message)
        VALUES (?, ?)
    """, (user_id, message))
    conn.commit()
    conn.close()


def complete_learning_process(enrollment_id: int, course_id: int, answers: dict, user_id: int):
    result = calculate_test_result(course_id, answers)
    save_test_result(enrollment_id, course_id, result)

    if result["passed"]:
        certificate_path = generate_certificate(enrollment_id)
        create_notification(user_id, "Курс успішно завершено. Сертифікат сформовано.")
        return {
            "status": "passed",
            "score": result["score"],
            "max_score": result["max_score"],
            "certificate": certificate_path
        }

    create_notification(user_id, "Тест не складено. Повторіть матеріал і спробуйте ще раз.")
    return {
        "status": "failed",
        "score": result["score"],
        "max_score": result["max_score"],
        "certificate": None
    }
```

ДОДАТОК Г

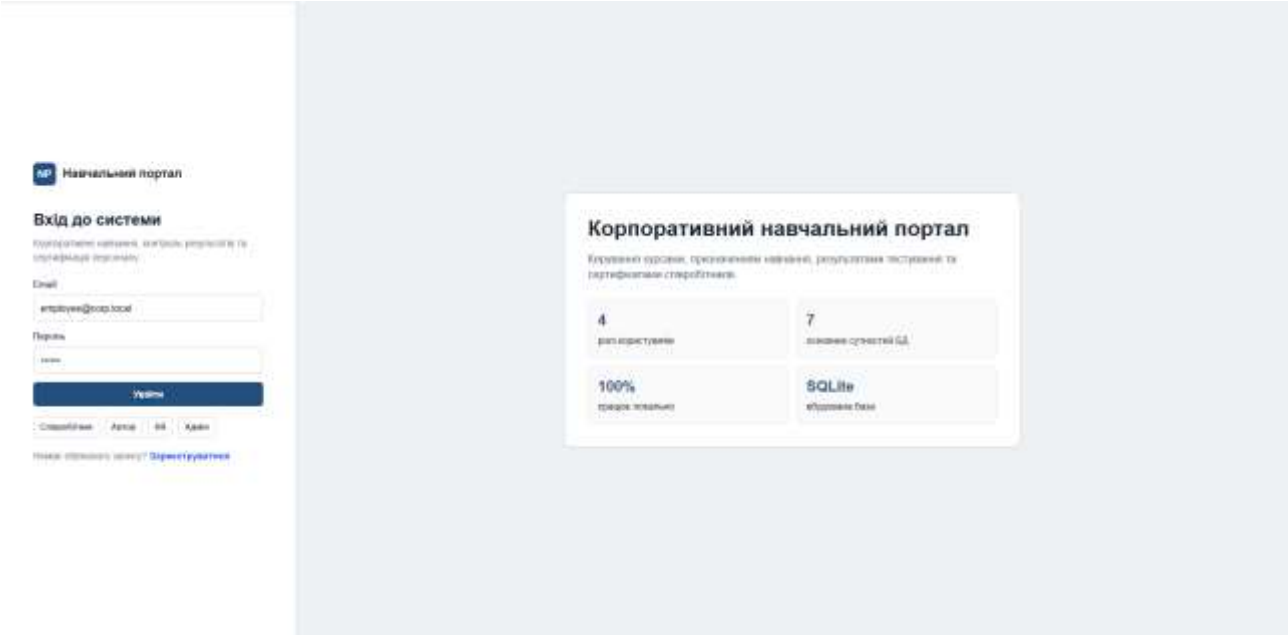

Рисунок.4 – Вхід співробника на портал

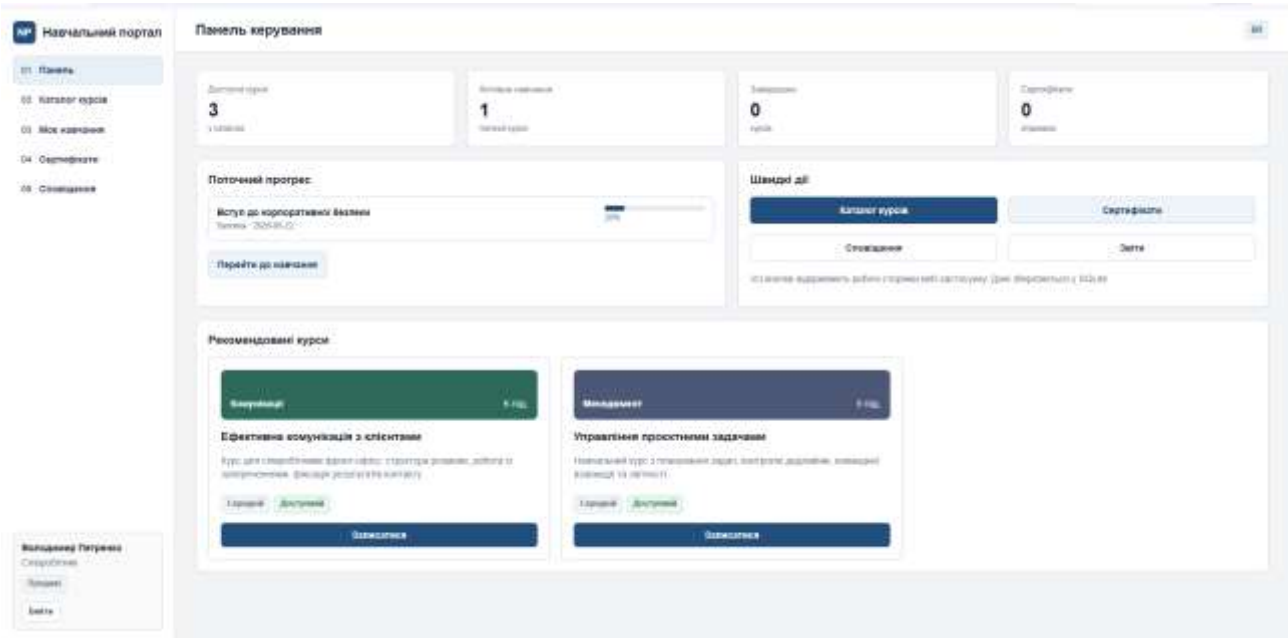

Рисунок.4.1 - Особистий кабінет співробітника

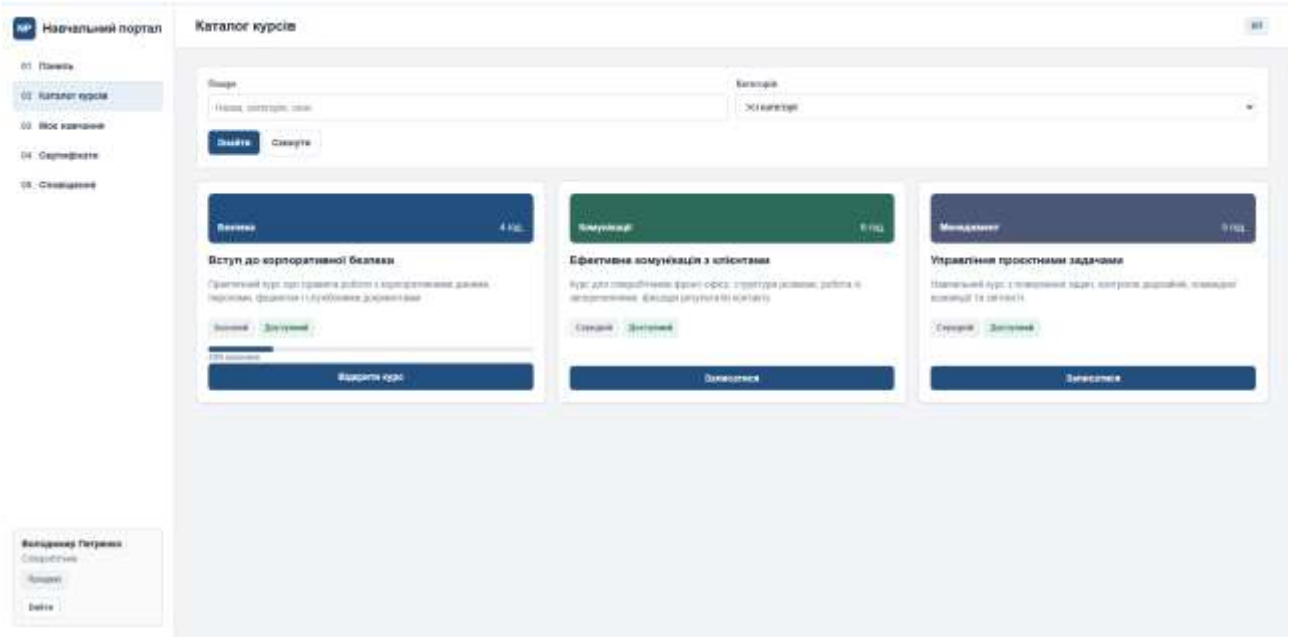

Рисунок.4.2 - Інтерфейс загального каталогу курсів для співробітників

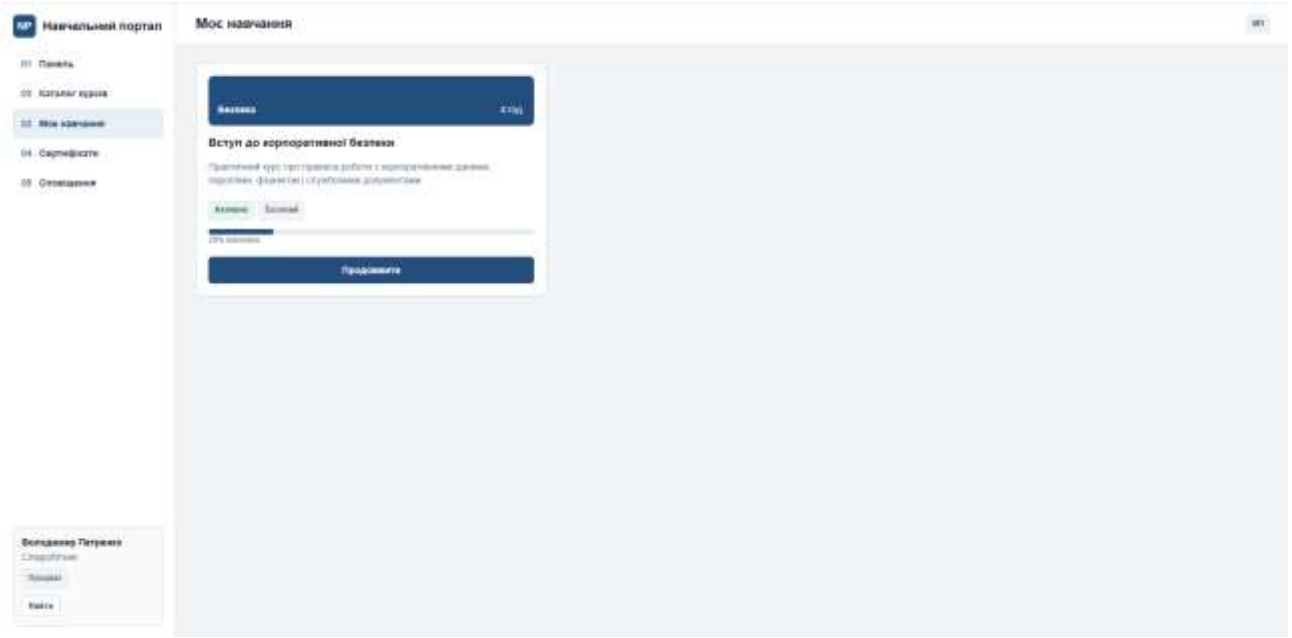

Рисунок.4.3 - Модуль «Моє навчання» в особистому кабінеті співробітника

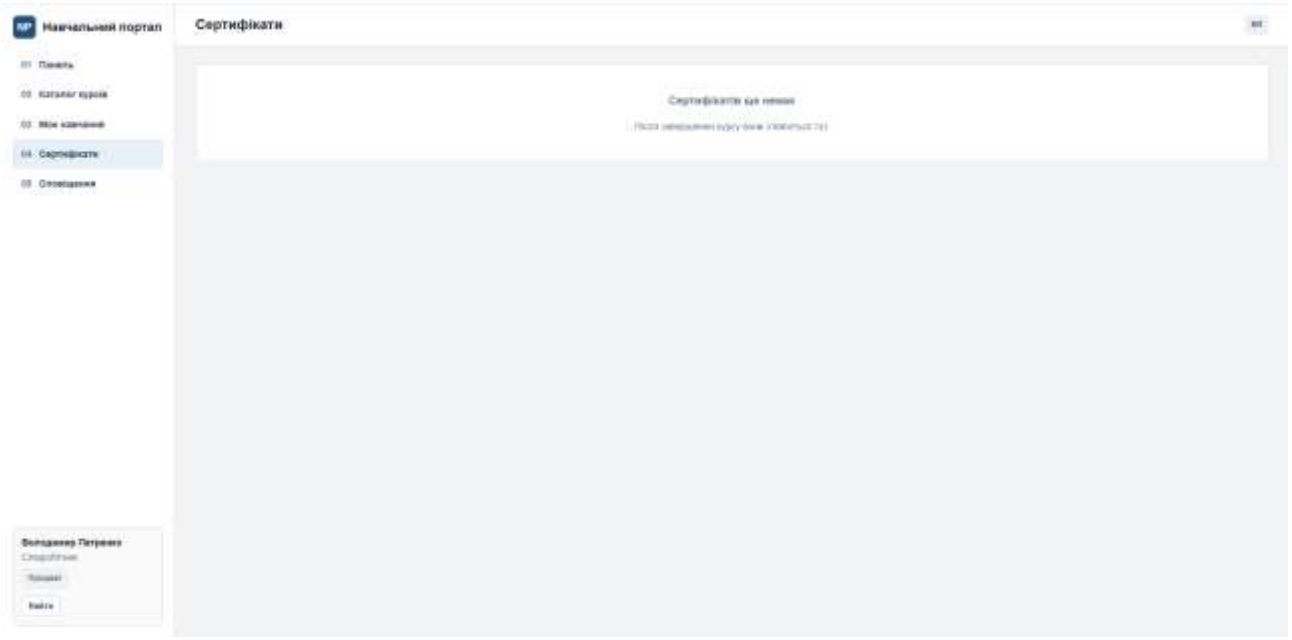

Рисунок.4.4 - Розділ «Мої сертифікати» в особистому кабінеті співробітника

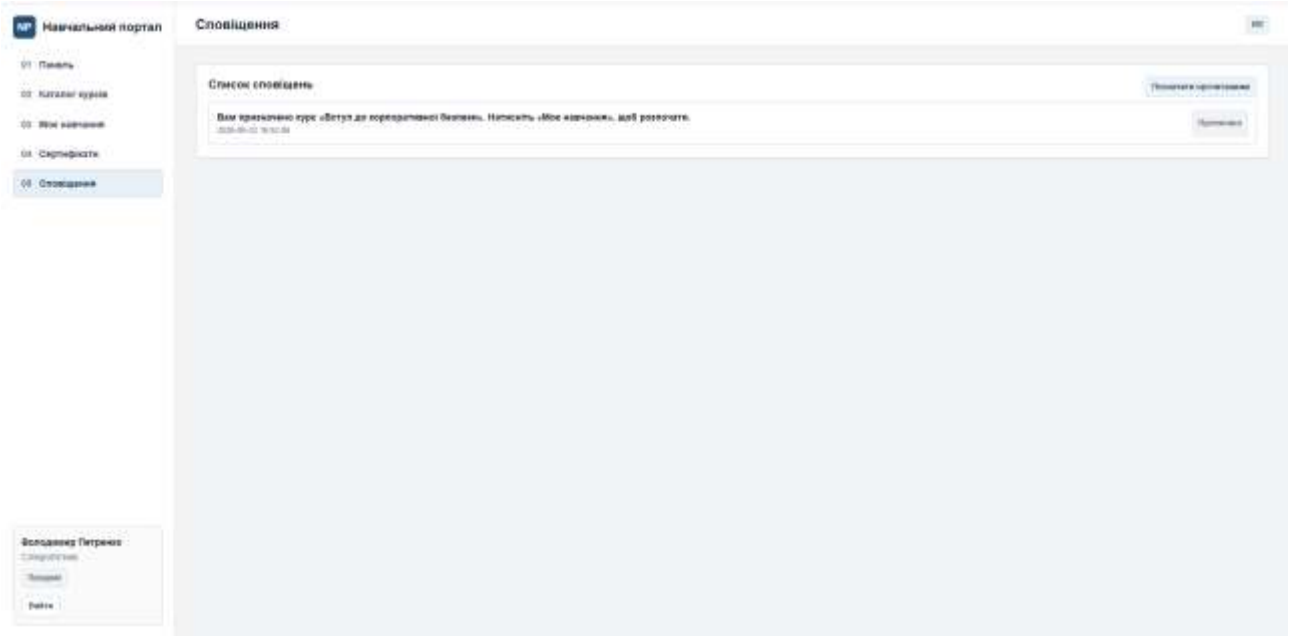

Рисунок.4.5 - Модуль «Сповіщення» в особистому кабінеті співробітника

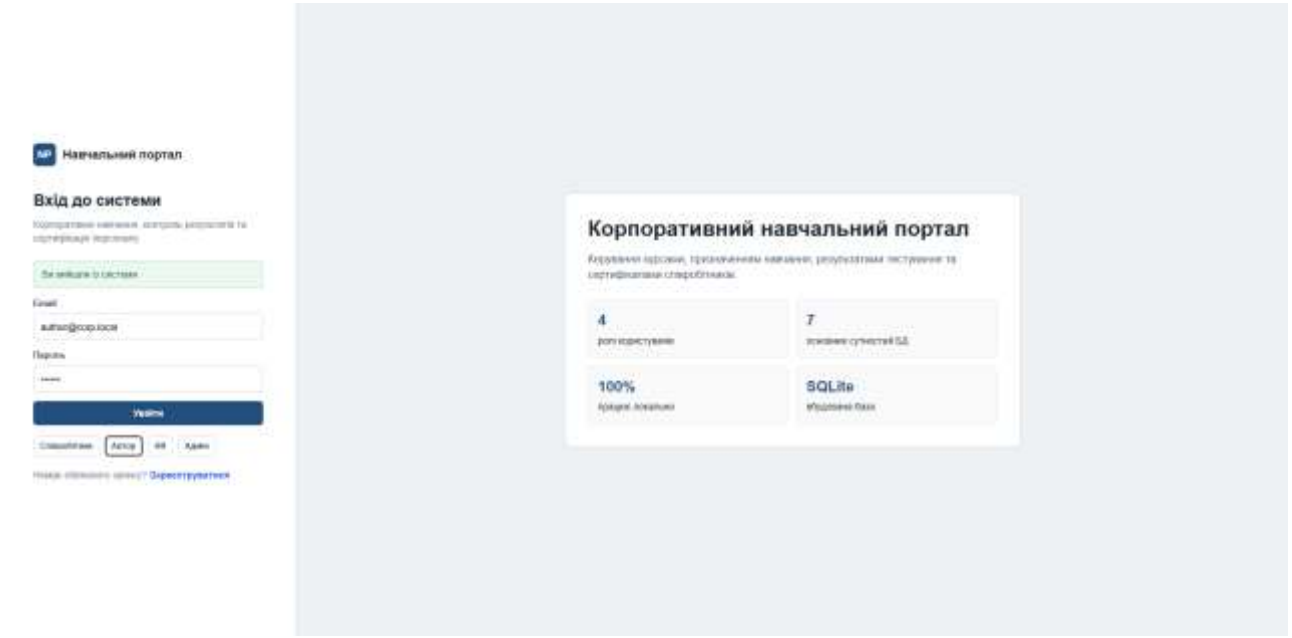

Рисунок.4.6 – Вхід до системи користувача Автор

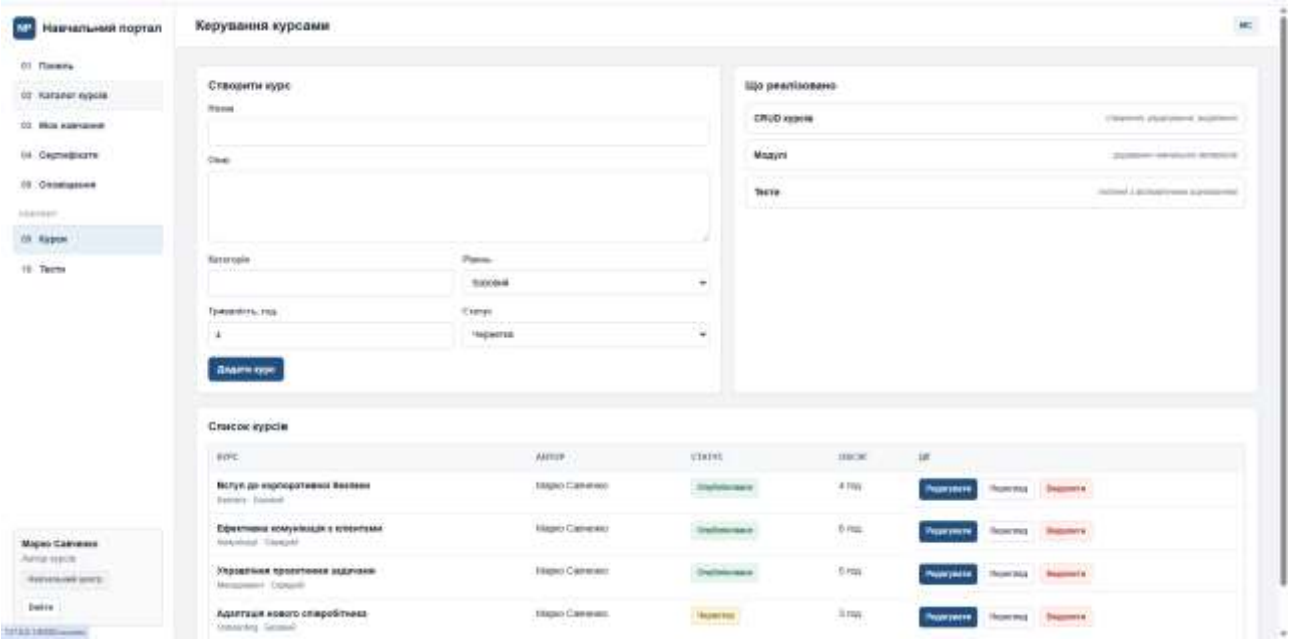

Рисунок.4.7 – Керування курсами. Автор може створювати курси

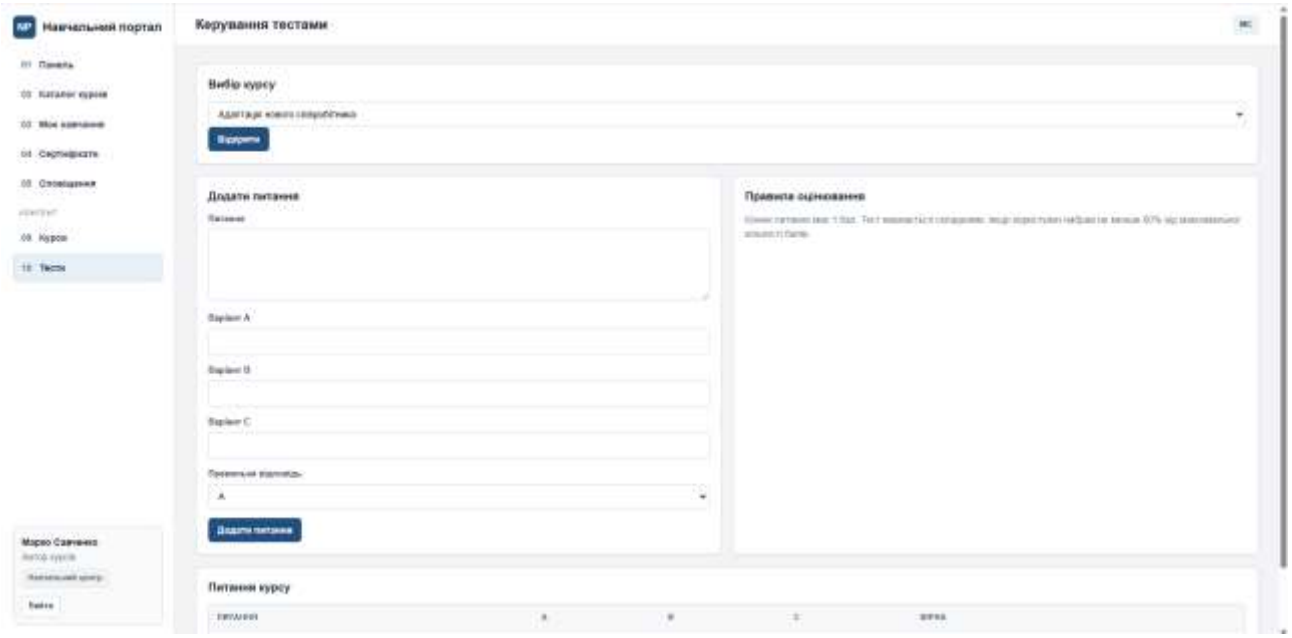

Рисунок.4.8 - Керування тестами. Автор може створювати тести

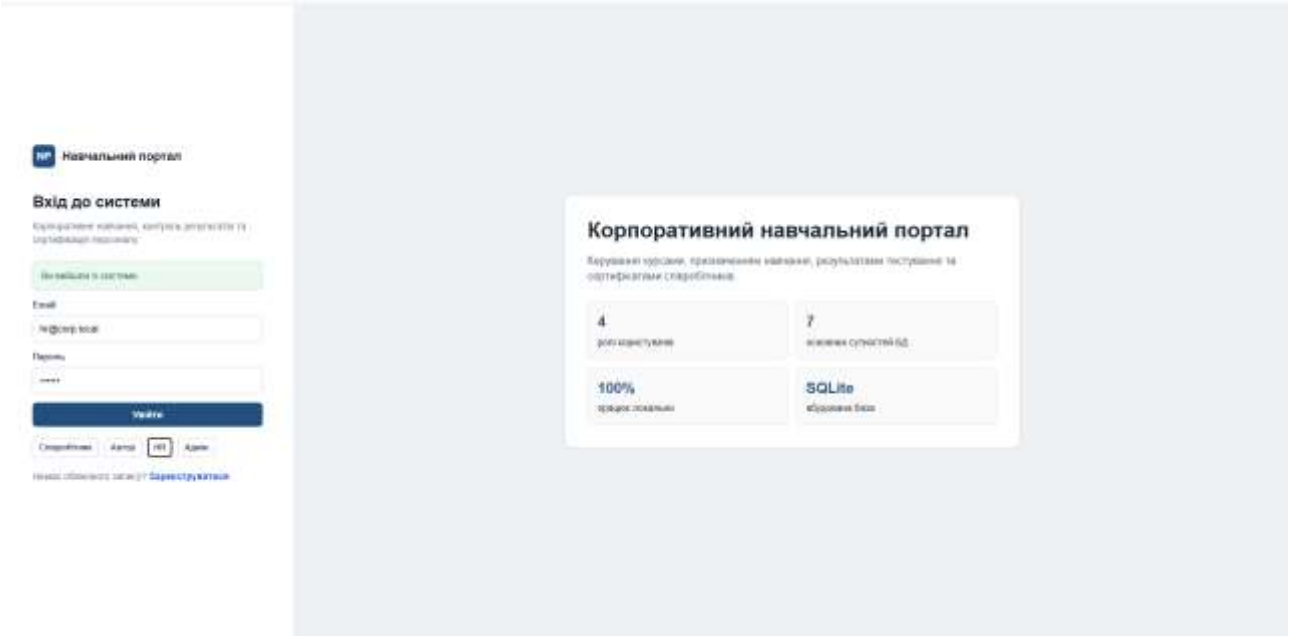

Рисунок.4.9 - хід до системи користувача HR

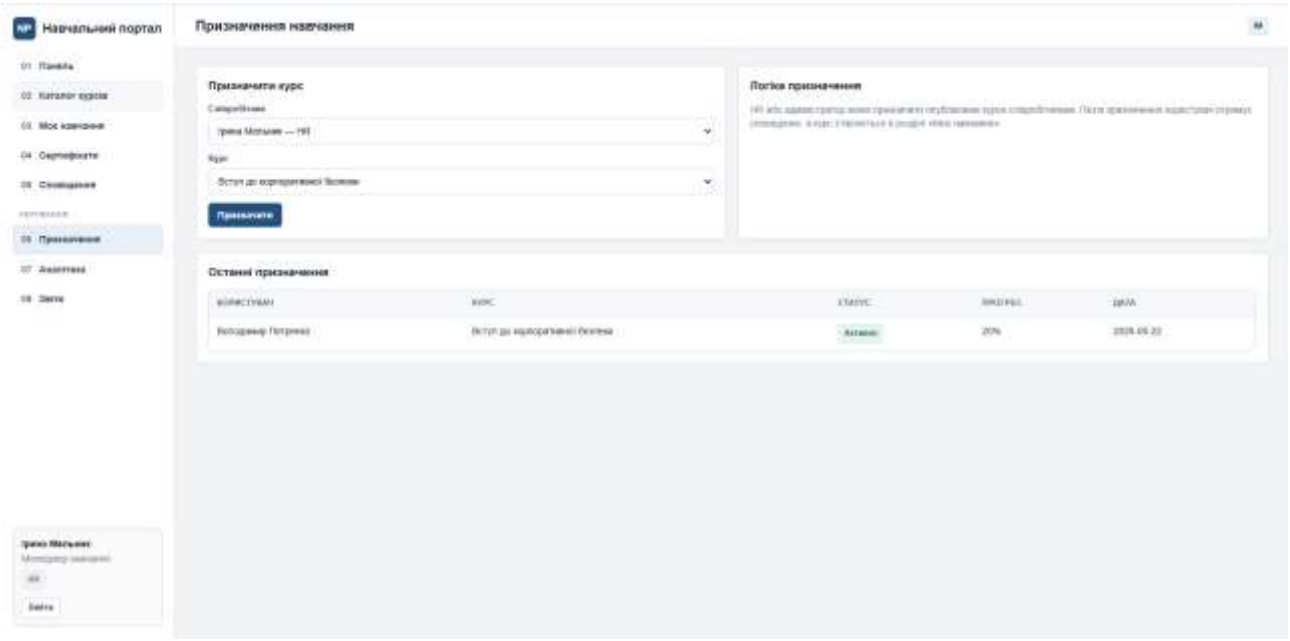

Рисунок.4.10 - Призначення навчання. HR призначує співробітникам які курси вони повинні пройти

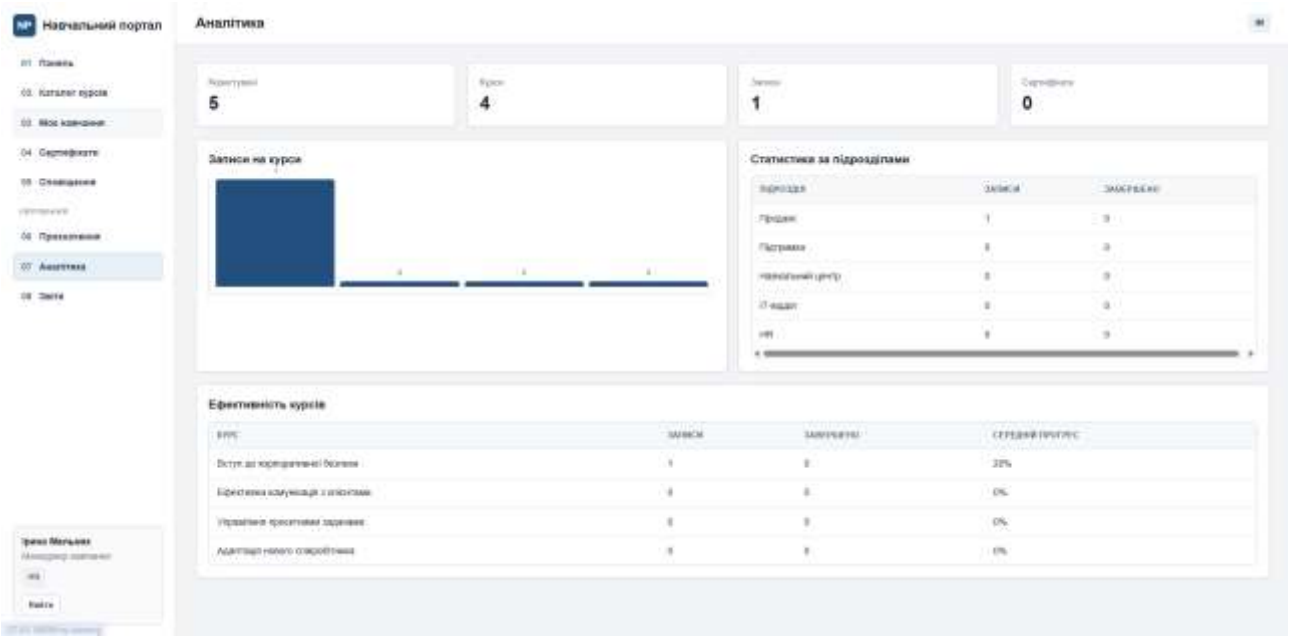

Рисунок.4.11 – Аналітика. HR слідкує за прогресом проходження курсів

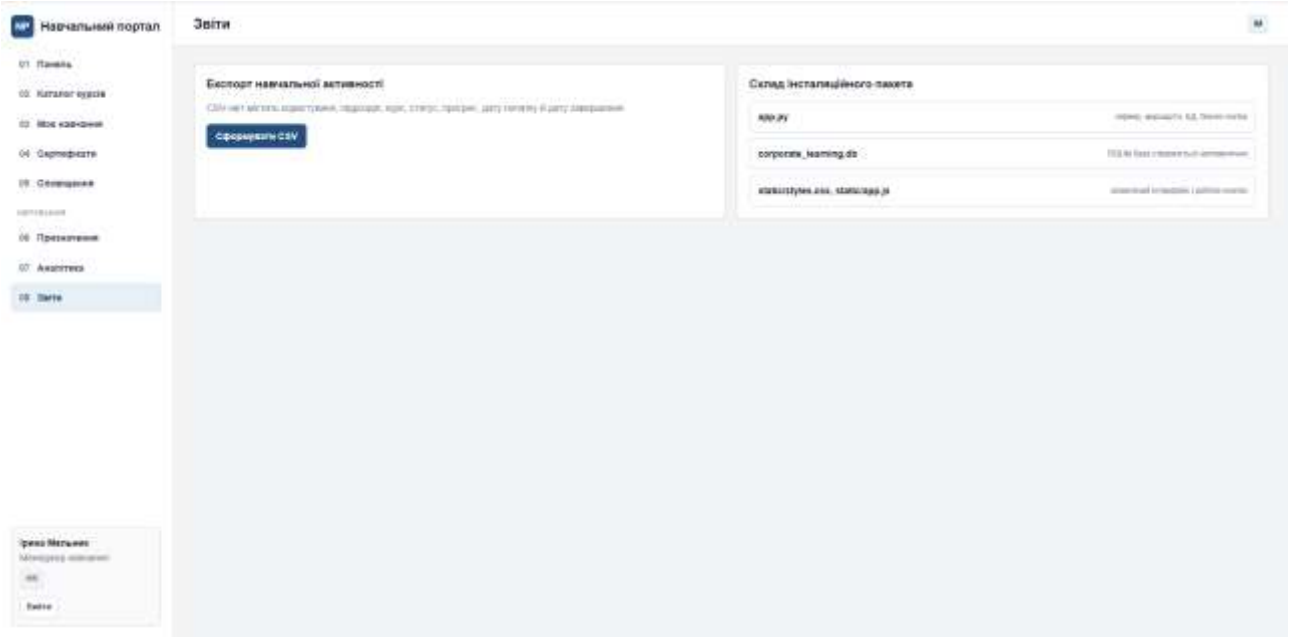

Рисунок.4.12 – Звіт. У HR є можливіть зробити звіт у форматі CSV

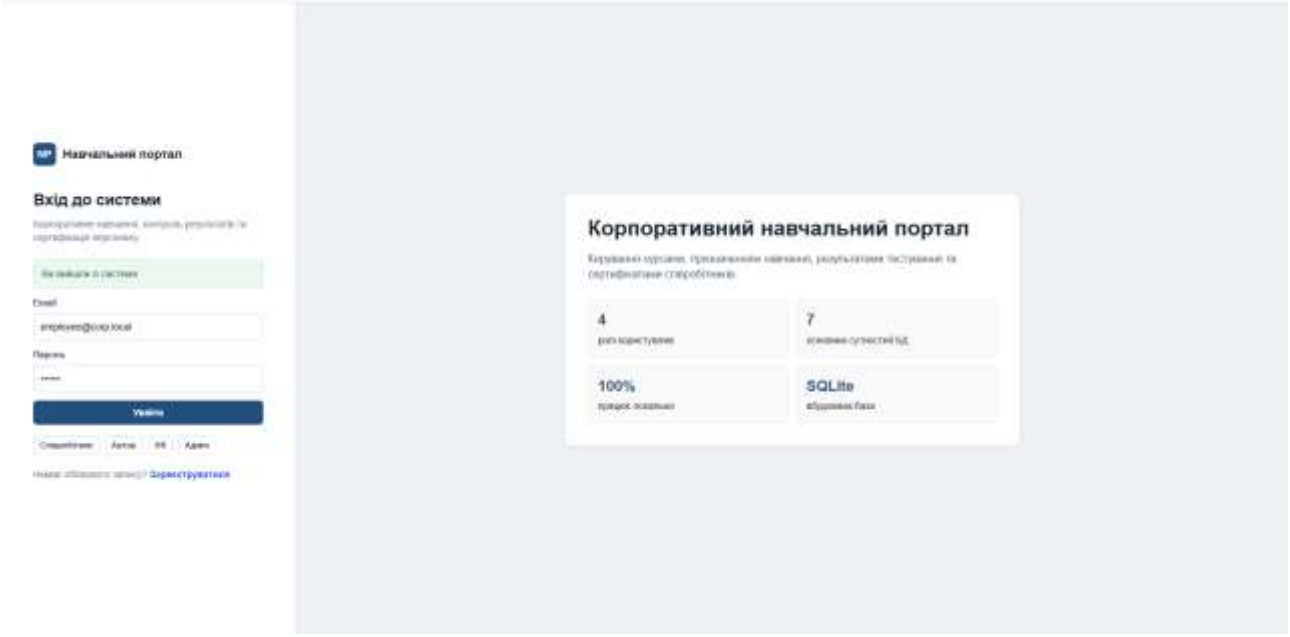

Рисунок.4.13 – Вхід в кабінет адміністратора

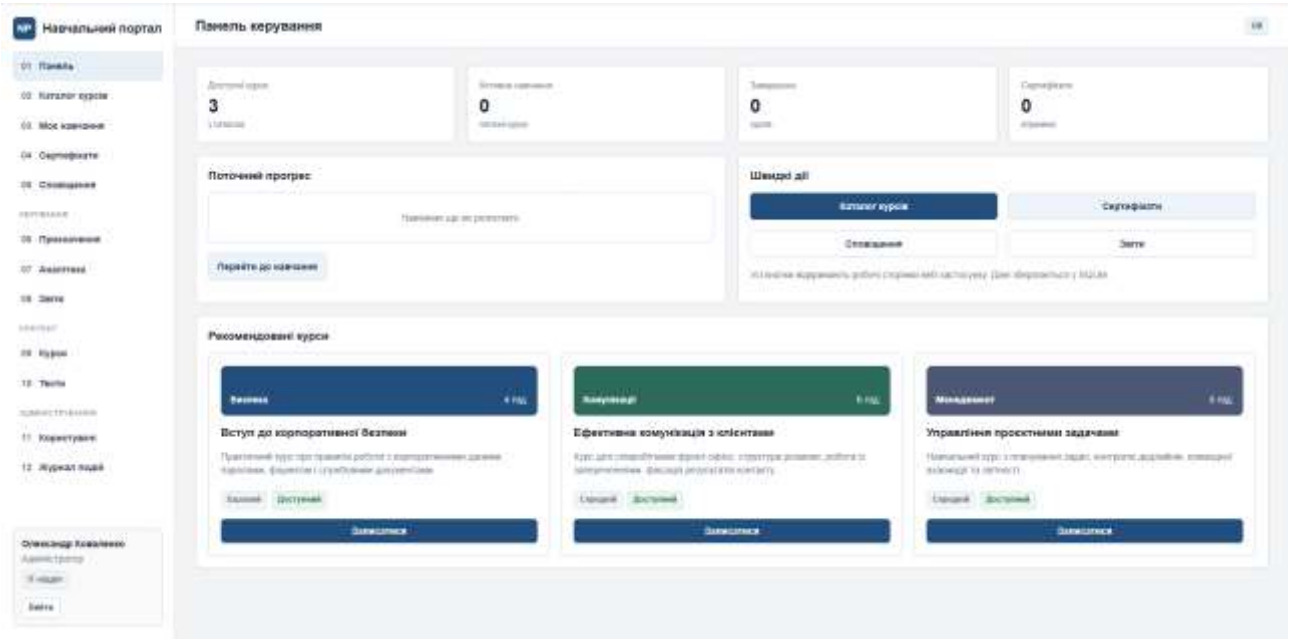

Рисунок.4.14. - Панель адміністратора: інтерфейс повного керування системою та користувачами

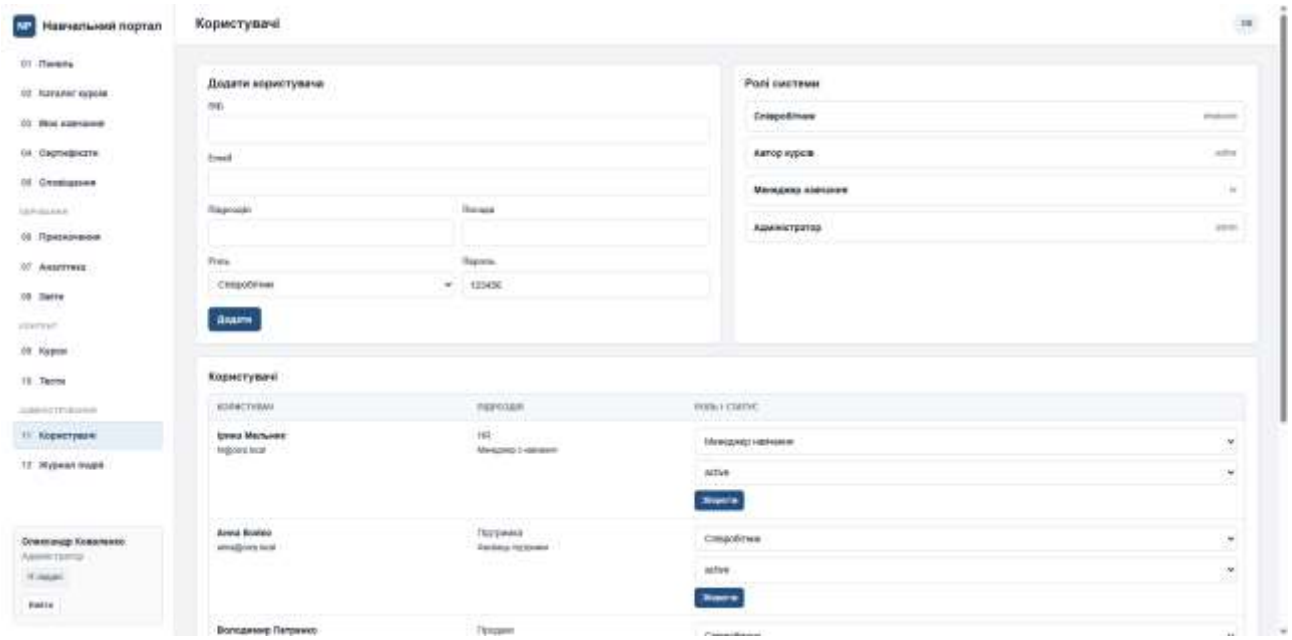

Рисунок.4.15 – Панель адміністратора: реєстр користувачів та інтерфейс додавання нових співробітників

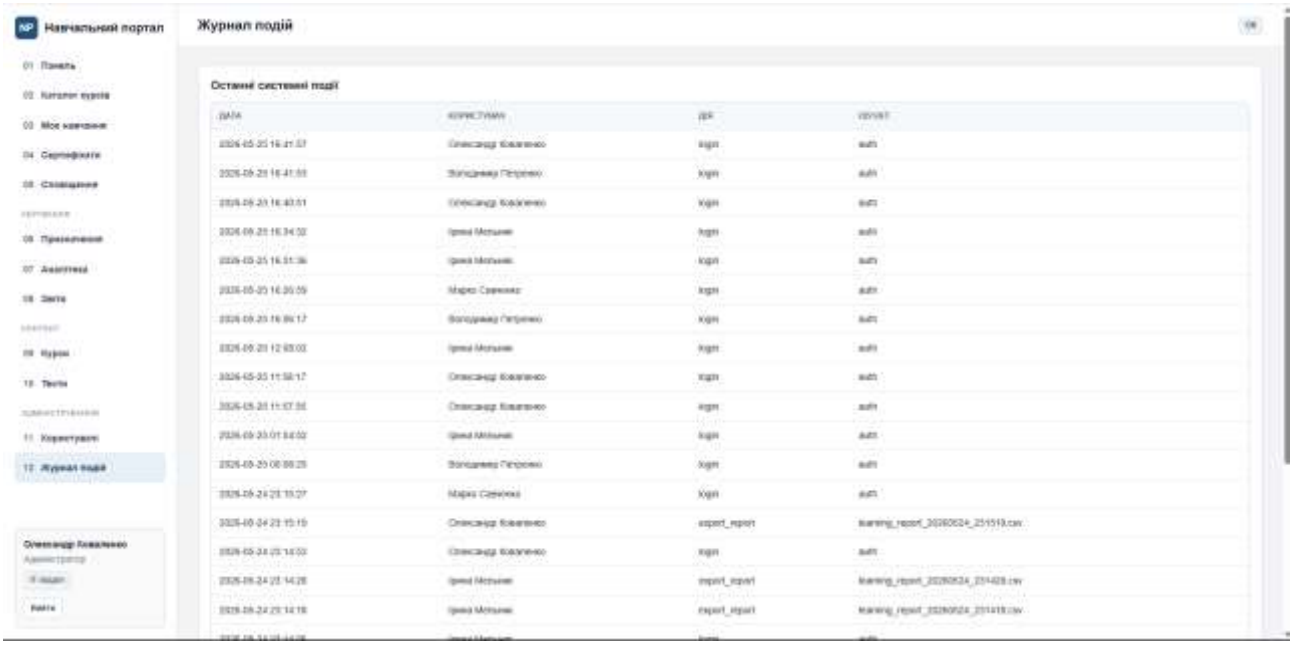

Рисунок.4.16 – Панель адміністратора: інтерфейс модуля «Журнал подій» для моніторингу активності

ДОДАТОК Ґ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Декларація про академічну доброчесність та використання ШІ

ДЕКЛАРАЦІЯ ЗДОБУВАЧА

Я, Бортник Володимир Володимирович, здобувач(ка) НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота (проєкт) на тему

«Система корпоративного навчання на основі веб-технологій» є результатом моєї особистої праці.

2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.

3. Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):

- [ ] інструменти генеративного ШІ не використовувалися.
- [ ] інструменти ШІ (вказати назву: ChatGPT, Gemini, Claude, DeepL, _____інші (вказати назву)) були використані для:
  - [ ] перекладу іншомовних джерел;
  - [ ] стилістичного редагування та перевірки граматики;
  - [ ] допомоги у написанні/відлагодженні програмного коду;
  - [ ] інше: _____.

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

«__» _____ 2026р. _____ Бортник Володимир
(підпис) **(Ім'я та ПРІЗВИЩЕ)**