НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ПОГОДЖЕНО **Декан факультету**	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ** **Завідувач кафедри інформаційних систем і технологій**
_____ Ігор БОЛБОТ	**_____ Михайло ШВИДЕНКО**
«___» _____ 2026 р.	«___» _____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Веб орієнтована автоматизована система підтримки поза університетського навчання»

Спеціальність «122 Комп'ютерні науки»

Освітньо-професійна програма «Комп'ютерні науки»

Гарант освітньої програми д.е.н., професор	_____ (підпис)	**Роман РУДЕНСЬКИЙ**
Керівник бакалаврської кваліфікаційної роботи к.е.н., доцент	_____ (підпис)	**Максим МОКРІЄВ**
Виконав	_____ (підпис)	**Едуард ОПАНАСЕНКО**

Київ-2026

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри інформаційних систем і технологій

к.е.н., доцент _____ **Михайло ШВИДЕНКО**

«___» _____ 2026 р.

ЗАВДАННЯ
ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
ЗДОБУВАЧУ

Опанасенку Едуарду Сергійовичу
(прізвище, ім'я, по батькові)

Спеціальність «122 Комп'ютерні науки»

Освітньо-професійна програма «Комп'ютерні науки»

Тема бакалаврської кваліфікаційної роботи

«Веб орієнтована автоматизована система підтримки поза університетського навчання»

затверджена наказом від «6» січня 2026 р. №46 «С»
Термін подання завершеної роботи на кафедру «25 травня 2026 р.
Вихідні дані до бакалаврської кваліфікаційної роботи (проєкту): відомості про процес навчання як неформальної освіти; перелік вимог до подібних систем; ролі користувачів системи.
Перелік питань, що підлягають дослідженню: аналіз предметної області цифрового навчання; проєктування інформаційної системи для підтримки освітнього позанавчального процесу; розроблення вебсистеми позауніверситетського навчання; тестування та оцінювання роботи системи.

Дата видачі завдання «06» січня 2026 р.

Керівник бакалаврської кваліфікаційної роботи _____ (підпис) **Максим МОКРІЄВ**

Завдання взяв до виконання _____ (підпис) **Едуард ОПАНАСЕНКО**

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....5

ВСТУП.....7

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....9

1.1 Опис предметної області.....9

1.2 Аналіз існуючих рішень.....11

1.3 Моделювання предметної області.....18

1.4 Аналіз вимог розроблюваної системи.....21

1.5 Постановка завдання.....23

1.6 Висновки до першого розділу.....25

2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....26

2.1 Логічна модель даних у вигляді ER-діаграми.....26

2.2 Діаграма класів та кооперації.....27

2.3 Діаграма компонентів.....29

2.4 Діаграма пакетів.....31

2.5 Фізична модель даних.....32

2.6 Висновки до другого розділу.....33

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....34

3.1 Вибір технологій та інструментальних засобів реалізації системи..34

3.2 Алгоритмізація програмних модулів системи.....35

3.4 Реалізація бази даних та інтерфейсних модулів.....39

3.5 Висновки до третього розділу.....40

4 ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ СИСТЕМИ.....42

4.1 План тестування програмних модулів та методика оцінювання результатів.....42

4.2 Тестування веборієнтованої системи підтримки позауніверситетського навчання.....43

4.3 Результати тестування та аналіз ефективності системи.....53

4.4 Розгортання системи та склад інсталяційного пакета.....54

4.5 Висновки до четвертого розділу.....55

ВИСНОВКИ.....57

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....59

ДОДАТОК А.....61

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

1. API - Application Programming Interface, програмний інтерфейс взаємодії клієнтської частини, серверного застосунку та бази даних.
2. CRUD - Create, Read, Update, Delete, базові операції створення, читання, оновлення та видалення записів.
3. CSS - Cascading Style Sheets, мова опису зовнішнього вигляду вебсторінок.
4. DB - Database, база даних програмної системи.
5. ER - Entity Relationship, модель сутностей і зв’язків предметної області.
6. HTTP - HyperText Transfer Protocol, протокол передавання даних між браузером і сервером.
7. HTML - HyperText Markup Language, мова розмітки вебінтерфейсу.
8. JS - JavaScript, мова програмування для клієнтської логіки вебзастосунку.
9. LMS - Learning Management System, система керування навчанням.
10. MVC - Model View Controller, архітектурний підхід до поділу логіки, даних та подання.
11. ORM - Object Relational Mapping, підхід до відображення об’єктів програми у таблиці бази даних.
12. SQL - Structured Query Language, мова структурованих запитів до реляційної бази даних.
13. SQLite - вбудована реляційна система керування базами даних, використана для локального збереження даних.
14. UML - Unified Modeling Language, уніфікована мова моделювання програмних систем.

15. URL - Uniform Resource Locator, уніфікований ідентифікатор вебресурсу.

16. UI - User Interface, користувацький інтерфейс програмного продукту.

17. UX - User Experience, досвід користувача під час взаємодії із системою.

18. БД - база даних, структуроване сховище навчальної, користувацької та службової інформації.

19. ПЗ - програмне забезпечення.

20. ІС - інформаційна система.

ВСТУП

Актуальність теми зумовлена активним поширенням цифрових освітніх сервісів та зростанням потреби у зручних інструментах для організації позауніверситетського навчання. Значна частина слухачів проходить додаткові курси, професійні програми, короткі тренінги та практичні модулі поза межами основної освітньої програми. Водночас процес пошуку курсу, запису на навчання, контролю прогресу, виконання тестів і отримання підтвердження результатів часто залишається фрагментованим.

Позауніверситетське навчання характеризується більшою гнучкістю порівняно з традиційними академічними курсами. Слухач самостійно обирає напрям підготовки, темп проходження матеріалу, набір практичних занять і форму контролю. Через це інформаційна система повинна бути простою для користувача, але достатньо повною з погляду збереження навчальних даних, керування курсами, відстеження прогресу та формування сертифікатів.

Метою кваліфікаційної роботи є дослідження і розробка веборієнтованої автоматизованої системи підтримки позауніверситетського навчання, яка забезпечує реєстрацію користувачів, ведення каталогу курсів, запис слухача на курс, проходження уроків, контроль прогресу, тестування та автоматичне формування результатів навчання.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- проаналізувати предметну область позауніверситетського навчання та визначити потребу в її автоматизації;
- дослідити існуючі освітні вебплатформи, визначити їх переваги й недоліки;
- сформувати основні вимоги до веборієнтованої автоматизованої системи підтримки навчання;
- побудувати UML-діаграми для опису структури та роботи системи;

– розробити ER-модель і фізичну модель бази даних;

– обґрунтувати вибір технологій для реалізації програмного забезпечення;

– реалізувати програмний прототип системи з основними модулями: авторизація, курси, навчальні матеріали, тестування, прогрес і адміністрування;

– провести тестування системи та оцінити коректність роботи її основних функцій.

Об’єктом дослідження є процес організації позауніверситетського навчання з використанням веборієнтованих інформаційних систем. Предметом дослідження є методи, моделі та програмні засоби автоматизації запису на курси, проходження навчальних матеріалів, контролю прогресу, тестування та сертифікації слухачів.

У роботі використано методи системного аналізу, об’єктно-орієнтованого проєктування, UML-моделювання, реляційного моделювання даних, нормалізації бази даних, модульного тестування, функціонального тестування та аналізу якості користувацького інтерфейсу.

Практична цінність роботи полягає у створенні працездатного вебзастосунку, який може використовуватися як навчальний прототип для невеликих освітніх центрів, гуртків, курсів підвищення кваліфікації, тренінгових програм і дистанційних навчальних проєктів. Розроблена система не потребує складного розгортання, використовує вбудовану базу даних SQLite і запускається локально засобами стандартної бібліотеки Python.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

Предметна область веборієнтованої автоматизованої системи підтримки позауніверситетського навчання охоплює процеси організації, супроводу та контролю навчання, яке здійснюється поза межами стандартної університетської освітньої програми. До такого навчання належать короткострокові курси, професійні тренінги, факультативи, тематичні практикуми, мовні програми, підготовчі заняття та онлайн-модулі для самостійного розвитку.

На відміну від класичного навчального процесу, позауніверситетське навчання має гнучку структуру. Один слухач може одночасно проходити кілька курсів, самостійно змінювати темп роботи, повертатися до матеріалів, виконувати підсумкові тести та отримувати сертифікат після успішного завершення. Саме тому система повинна зберігати не лише перелік курсів, а й стан навчання кожного користувача.

У межах розроблюваної системи основними учасниками є гість, слухач, викладач і адміністратор. Гість переглядає доступні курси та створює обліковий запис. Слухач записується на курс, проходить уроки, виконує тестування й отримує результат. Викладач може наповнювати курс навчальними матеріалами, а адміністратор контролює користувачів, курси та загальні параметри системи.

Узагальнену структуру предметної області подано на рис. 1.1. Вона показує, що центральним елементом є курс, який об’єднує навчальні уроки, тестові завдання, записи слухачів, прогрес і сертифікати. Усі ці об’єкти зберігаються у базі даних і використовуються серверною частиною для формування сторінок вебінтерфейсу.

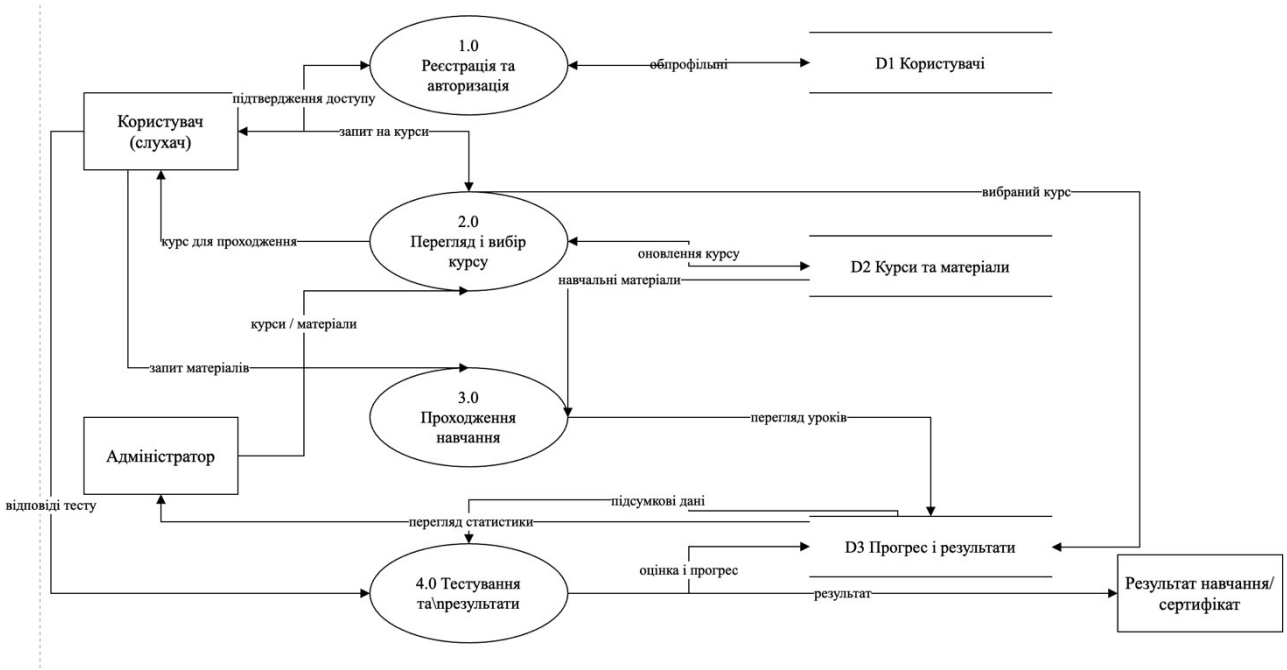

Рис. 1.1 - Узагальнена структура предметної області системи підтримки позауніверситетського навчання

Як видно з рисунка 1.1, система повинна забезпечувати наскрізний цикл роботи з навчальною інформацією. Усі дії користувача мають фіксуватися у базі даних, оскільки саме ці записи використовуються для розрахунку прогресу, перевірки доступу до тестування та підтвердження факту завершення курсу.

Таблиця 1.1

Основні процеси предметної області

Процес	Зміст процесу	Результат
Реєстрація користувача	Створення облікового запису слухача, викладача або адміністратора	Користувач отримує доступ до системи
Перегляд каталогу	Відображення переліку доступних курсів із фільтрацією за категорією та рівнем	Користувач обирає навчальний напрям
Запис на курс	Фіксація зв’язку між користувачем і курсом	Створюється запис у таблиці enrollments
Проходження уроків	Перегляд матеріалів і позначення уроків як	Оновлюється

	виконаних	прогрес навчання

Продовження таблиці 1.1

Підсумковий контроль	Виконання тесту після опрацювання матеріалів	Формується оцінка та статус проходження
Сертифікація	Створення сертифіката після успішного тестування	Слухач отримує підтвердження результату

Таблиця 1.1 демонструє, що система не обмежується зберіганням довідника курсів. Вона повинна підтримувати повний навчальний сценарій, у якому кожен етап логічно пов’язаний із попереднім і наступним. Це дає можливість реалізувати контроль доступу, персональний кабінет і автоматичне формування результатів.

1.2 Аналіз існуючих рішень

Для визначення функціональних можливостей веборієнтованої автоматизованої системи підтримки позауніверситетського навчання доцільно розглянути існуючі програмні рішення, які використовуються для організації онлайн-курсів, розміщення навчальних матеріалів, виконання завдань, тестування, контролю прогресу та адміністрування користувачів. Аналіз аналогів дає змогу визначити набір функцій, які варто реалізувати у власній системі, а також виявити недоліки наявних платформ.

Позауніверситетське навчання відрізняється від класичного університетського процесу більшою гнучкістю, коротшими освітніми програмами, різним рівнем підготовки слухачів і потребою у швидкому доступі до матеріалів. Тому система підтримки такого навчання повинна мати простий вебінтерфейс, каталог курсів, механізм запису на навчання, модулі проходження матеріалів, тестування, контролю результатів і базове

адміністрування. У межах дослідження розглянуто Moodle, Google Classroom, Thinkific, Canvas LMS і Schoology.

Одним із найпоширеніших рішень для організації електронного навчання є Moodle. Це LMS-система, у якій курс може містити навчальні розділи, матеріали, форуми, завдання, тести, журнал оцінок і засоби адміністрування. Загальний вигляд сторінки курсу Moodle наведено на рисунку 1.2.

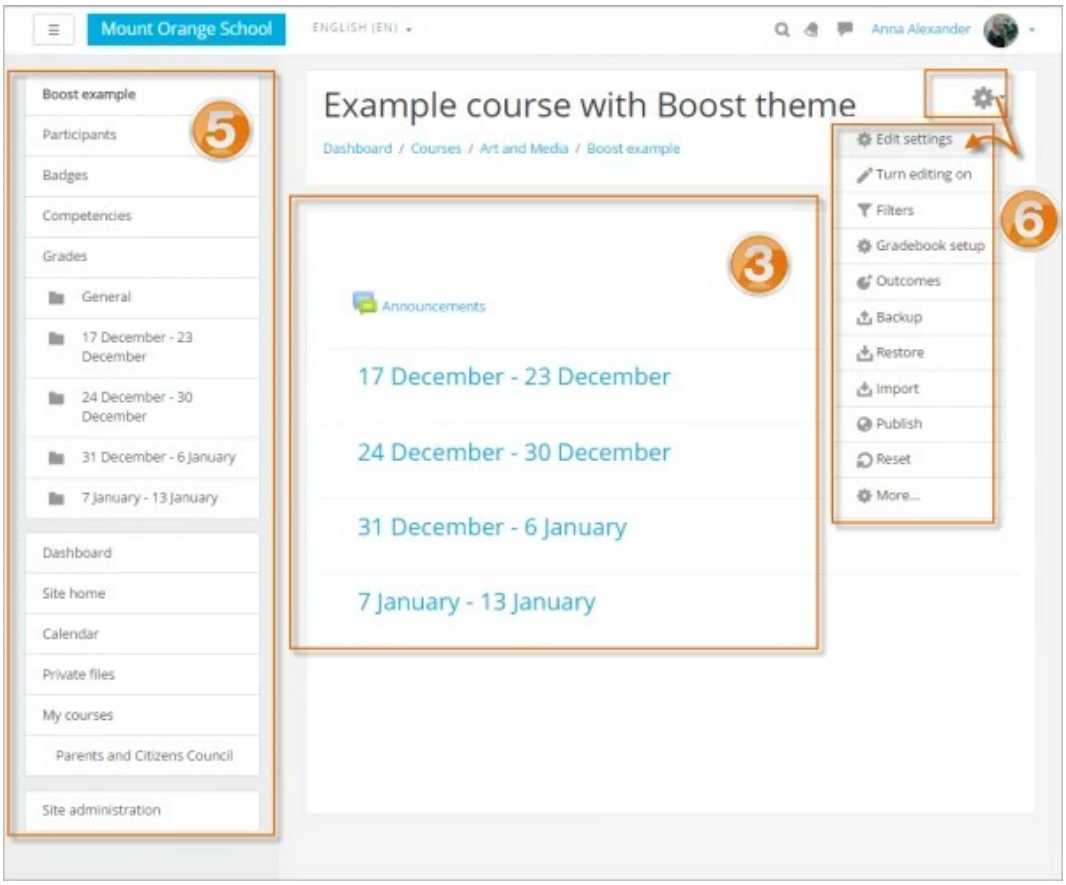

Рис. 1.2 – Інтерфейс сторінки навчального курсу в Moodle

Як видно з рисунка 1.2, Moodle орієнтований на структуроване подання навчального курсу. У лівій частині інтерфейсу розміщується навігація, у центральній частині - тематичні або календарні розділи курсу, а в правій частині можуть бути доступні інструменти керування курсом. Перевагою Moodle є гнучкість, підтримка ролей, можливість додавання різних типів навчальних активностей і контроль оцінювання. Водночас для невеликих позауніверситетських центрів така система може бути надмірно складною,

оскільки потребує налаштування структури, прав доступу, плагінів і параметрів курсу.

Google Classroom є простішим рішенням, орієнтованим на швидке створення навчального класу, розміщення завдань і обмін матеріалами між викладачем та слухачами. Типовий вигляд панелі курсів Google Classroom подано на рисунку 1.3.

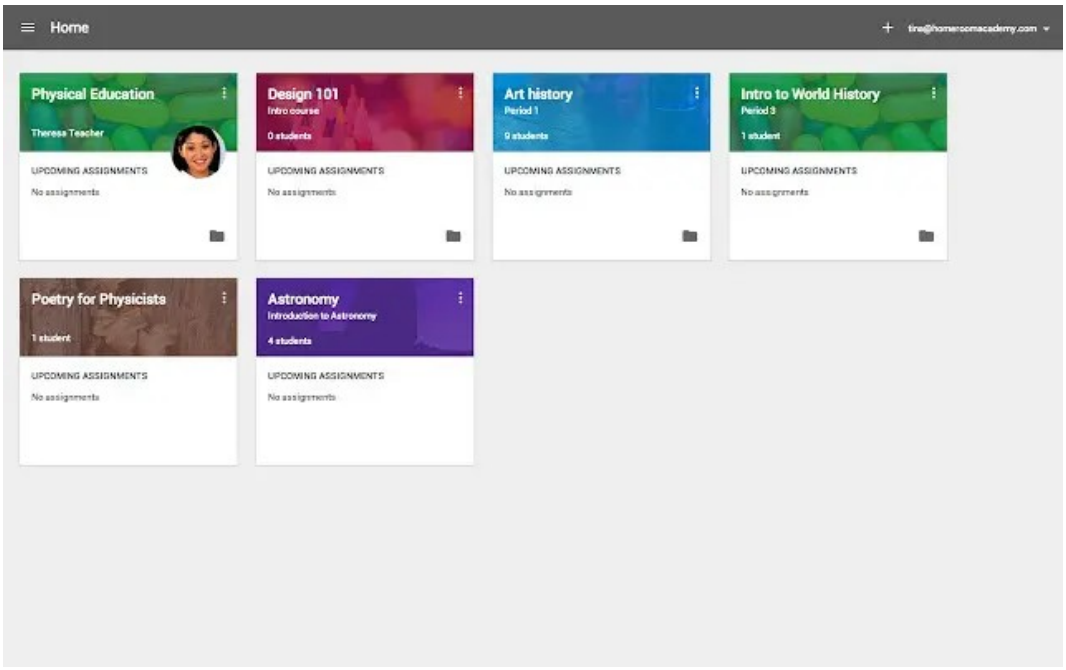

Рис. 1.3 – Інтерфейс головної сторінки курсів у Google Classroom

З рисунка 1.3 видно, що Google Classroom використовує карткову модель подання навчальних класів. Кожна картка містить назву курсу, викладача, кількість учасників або інформацію про найближчі завдання. Такий підхід є зручним для користувачів, оскільки дозволяє швидко перейти до потрібного курсу. Сильними сторонами Google Classroom є простота, інтеграція з Google Drive, Docs, Forms і Calendar, а також низький поріг входу для викладача. Недоліком є обмеженість функцій для відкритого каталогу курсів, сертифікації, розширеної аналітики та гнучкого адміністрування позауніверситетських програм.

Платформа Thinkific орієнтована на створення та публікацію онлайн-курсів, зокрема для тренерів, навчальних центрів і авторів комерційних освітніх продуктів. Приклад інтерфейсу проходження курсу в Thinkific наведено на рисунку 1.4.

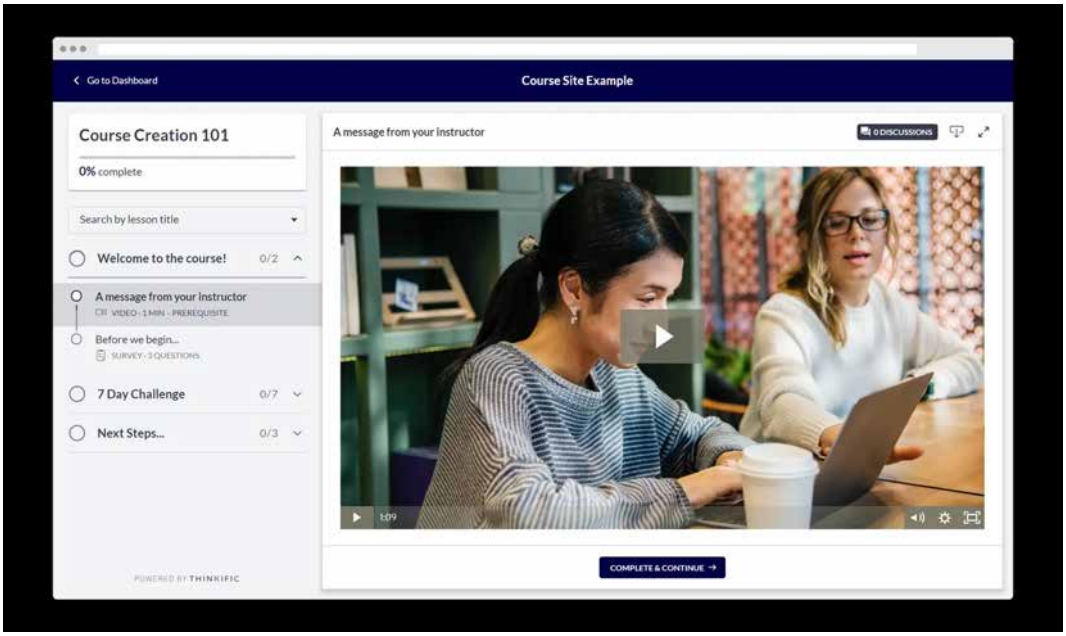

Рис. 1.4 – Інтерфейс проходження онлайн-курсу в Thinkific

На рисунку 1.4 показано, що в Thinkific навчальний курс складається з послідовних занять, відеоматеріалів, текстових блоків, опитувань і додаткових елементів. У лівій частині інтерфейсу відображається структура курсу та прогрес проходження. Така організація є зручною для самостійного навчання, оскільки користувач бачить, які модулі вже пройдено, а які ще потрібно виконати. Для розроблюваної системи доцільно використати подібний підхід до відображення прогресу, але без надмірної комерційної складової та складної тарифної моделі.

Canvas LMS є сучасною системою керування навчанням, яка широко використовується в освітніх закладах і підтримує курси, завдання, календар, повідомлення, обговорення та оцінювання. Загальний вигляд панелі курсів Canvas LMS наведено на рисунку 1.5.

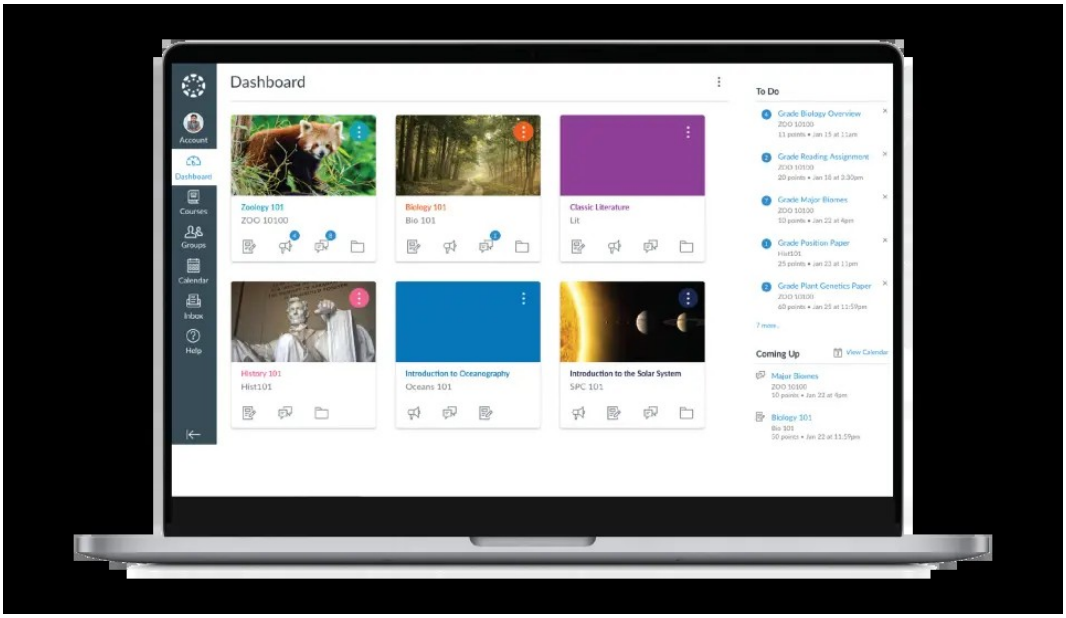

Рис. 1.5 – Панель навчальних курсів у Canvas LMS

Інтерфейс, показаний на рисунку 1.5, побудований навколо інформаційної панелі, де користувач бачить доступні курси, найближчі завдання та навчальний календар. Перевагою Canvas є зручне поєднання курсової структури, оцінювання і сповіщень. Для системи позауніверситетського навчання корисним є сам принцип виведення важливої інформації на головну панель, оскільки слухач повинен швидко бачити активні курси, стан виконання завдань і результати тестування. Водночас Canvas, як і Moodle, потребує налаштування та адміністрування, що може бути зайвим для малого навчального проєкту.

Schoology є комплексною LMS-платформою, яка поєднує навчальні матеріали, журнал оцінок, аналітику, нагадування, групи та комунікаційні інструменти. Приклад сторінки навчальних матеріалів у Schoology подано на рисунку 1.6.

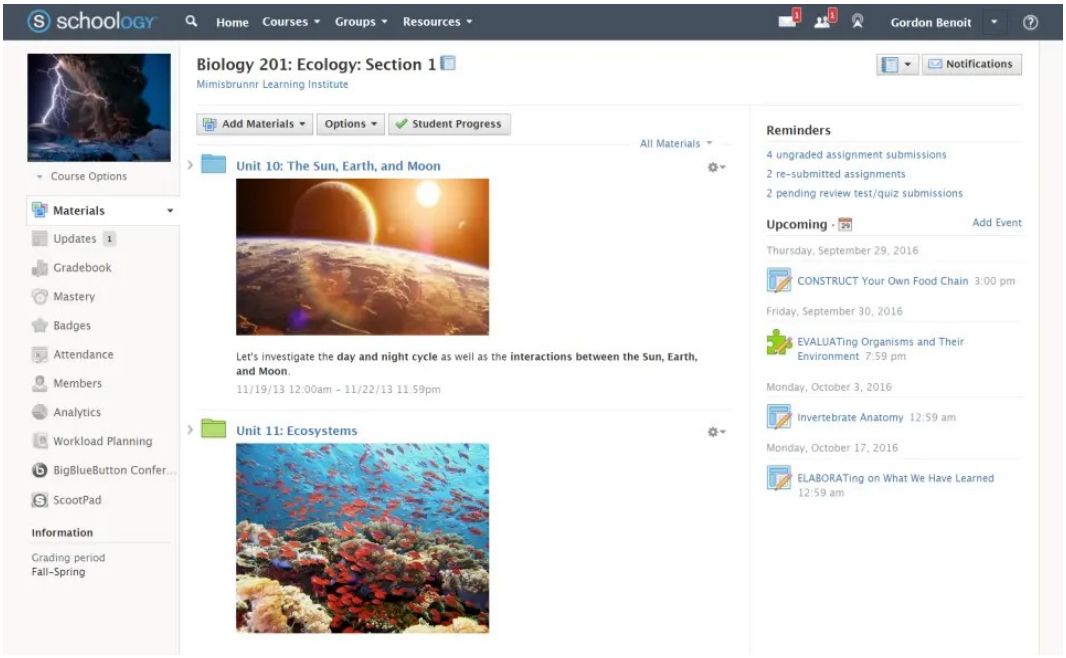

Рис. 1.6 – Інтерфейс навчальних матеріалів у Schoology

Як видно з рисунка 1.6, Schoology має розгорнуту структуру курсу: окремі розділи, матеріали, календар подій, нагадування та блоки керування. Це зручно для систематичного навчання, але для невеликих позауніверситетських курсів може виглядати складно. Під час розроблення власної системи варто зберегти логіку поділу матеріалів за темами, але зробити її простішою: курс має містити уроки, завдання, тести й показник прогресу без надмірної кількості другорядних інструментів.

Порівняльну характеристику розглянутих рішень наведено в таблиці 1.2. У таблиці враховано призначення кожної платформи, її сильні сторони, недоліки та можливість використання окремих підходів під час розроблення власної системи.

Таблиця 1.2

Порівняльний аналіз існуючих рішень для підтримки онлайн-навчання

Рішення	Основне призначення	Переваги	Недоліки
Moodle	Повноцінна LMS для створення	Гнучка структура курсу, ролі	Складне первинне налаштування,

	курсів, матеріалів, завдань і тестів.	користувачів, журнал оцінок, підтримка активностей і ресурсів.	перевантажений інтерфейс для невеликих навчальних центрів.

Продовження таблиці 1.2

Google Classroom	Організація навчальних класів, завдань, матеріалів і комунікації.	Простий інтерфейс, швидкий запуск курсу, інтеграція з Google Drive, Docs, Forms і Calendar.	Обмежена аналітика, слабка підтримка відкритого каталогу курсів і сертифікації.
Thinkific	Створення та публікація онлайн-курсів для самостійного навчання.	Зручний конструктор курсів, підтримка відео, тексту, тестів, обговорень і поетапного доступу.	Орієнтація на комерційні курси, залежність від зовнішньої платформи та тарифних обмежень.
Canvas LMS	Керування курсами, оцінюванням, обговореннями та навчальними календарями.	Сучасна панель курсів, зручне відстеження завдань, підтримка оцінювання та повідомлень.	Потребує адміністрування, у безкоштовному режимі має обмеження за ресурсами та підтримкою.
Schoology	Комплексна LMS для матеріалів, оцінок, аналітики, груп і навчальної взаємодії.	Зручна структура матеріалів курсу, gradebook, аналітика, нагадування та спільна робота.	Для невеликих позауніверситетських проєктів може бути функціонально надмірною.
Розроблювана система	Веборієнтована автоматизована система підтримки позауніверситетського навчання.	Простий каталог курсів, авторизація, запис на курс, навчальні матеріали, тестування, прогрес, сертифікати та	Функціональність обмежується межами розробленого прототипу та потребує подальшого масштабування.

		адміністрування.	

Дані таблиці 1.2 показують, що більшість існуючих платформ має розвинені засоби організації онлайн-навчання, однак не всі вони однаково придатні для підтримки позауніверситетських курсів. Moodle, Canvas LMS і Schoology мають потужний функціонал, але потребують складнішого адміністрування. Google Classroom є простим і зрозумілим, проте не забезпечує повноцінного каталогу курсів, сертифікації та глибшого контролю прогресу. Thinkific добре підходить для побудови онлайн-курсів, але орієнтований переважно на комерційну модель розміщення освітнього контенту.

Отже, аналіз існуючих рішень підтвердив доцільність розроблення власної веборієнтованої автоматизованої системи підтримки позауніверситетського навчання. Розроблювана система повинна поєднувати простоту Google Classroom, структурованість Moodle, зручне проходження курсу як у Thinkific, інформаційну панель як у Canvas LMS і тематичну організацію матеріалів як у Schoology. Водночас вона має бути менш перевантаженою, орієнтованою на невеликі навчальні центри, гуртки, тренінги, короткострокові курси та індивідуальну освітню траєкторію слухача.

.

1.3 Моделювання предметної області

Моделювання предметної області виконано за допомогою UML-діаграм, які описують систему з різних точок зору. Діаграма прецедентів визначає основні ролі та функції. Діаграма послідовності показує порядок обміну повідомленнями під час запису на курс і проходження матеріалу. Діаграма активності відображає загальний алгоритм роботи слухача із системою.

Основними акторами системи є гість, слухач, викладач та адміністратор. Гість може переглядати каталог і зареєструватися. Слухач виконує основні навчальні дії: записується на курс, переглядає уроки, проходить тест і отримує

сертифікат. Викладач відповідає за навчальні матеріали. Адміністратор контролює облікові записи, курси та структуру довідників.

На рис. 1.7 наведено зміст діаграми прецедентів. Її використання дозволяє зафіксувати межі системи та не перевантажувати програмну реалізацію другорядними функціями. Основними прецедентами визначено реєстрацію, авторизацію, перегляд каталогу, запис на курс, проходження уроків, тестування, формування сертифіката та адміністрування курсів.

Рис. 1.7 - Діаграма прецедентів веборієнтованої системи підтримки позауніверситетського навчання

Після визначення прецедентів побудовано діаграму послідовності, зміст якої подано на рис. 1.8. Вона деталізує сценарій запису слухача на курс. Користувач надсилає запит через вебінтерфейс, сервер перевіряє авторизацію, звертається до бази даних, створює запис у таблиці enrollments і повертає оновлену сторінку кабінету.

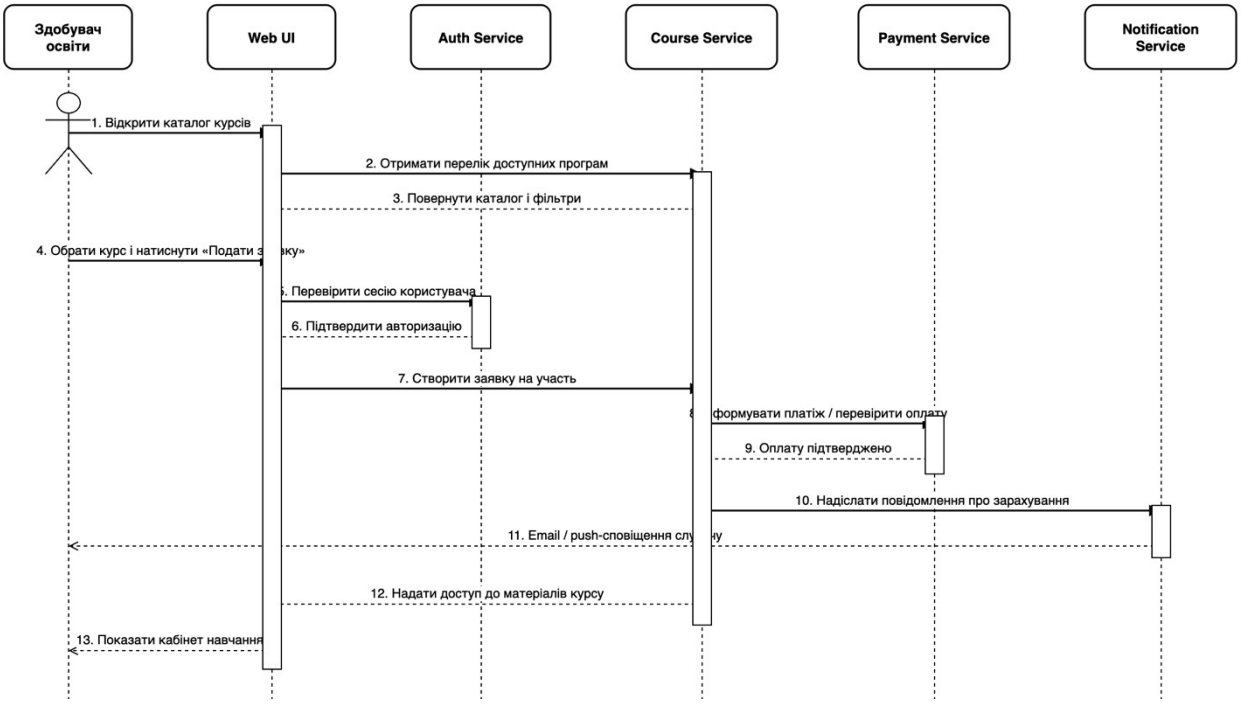

Рис. 1.8 - Діаграма послідовності процесу запису слухача на курс

Діаграма активності, наведена на рис. 1.5, описує загальний маршрут слухача від входу в систему до отримання сертифіката. Важливо, що проходження тесту стає доступним тільки після опрацювання навчального матеріалу або після достатнього рівня прогресу, що підвищує послідовність навчання.

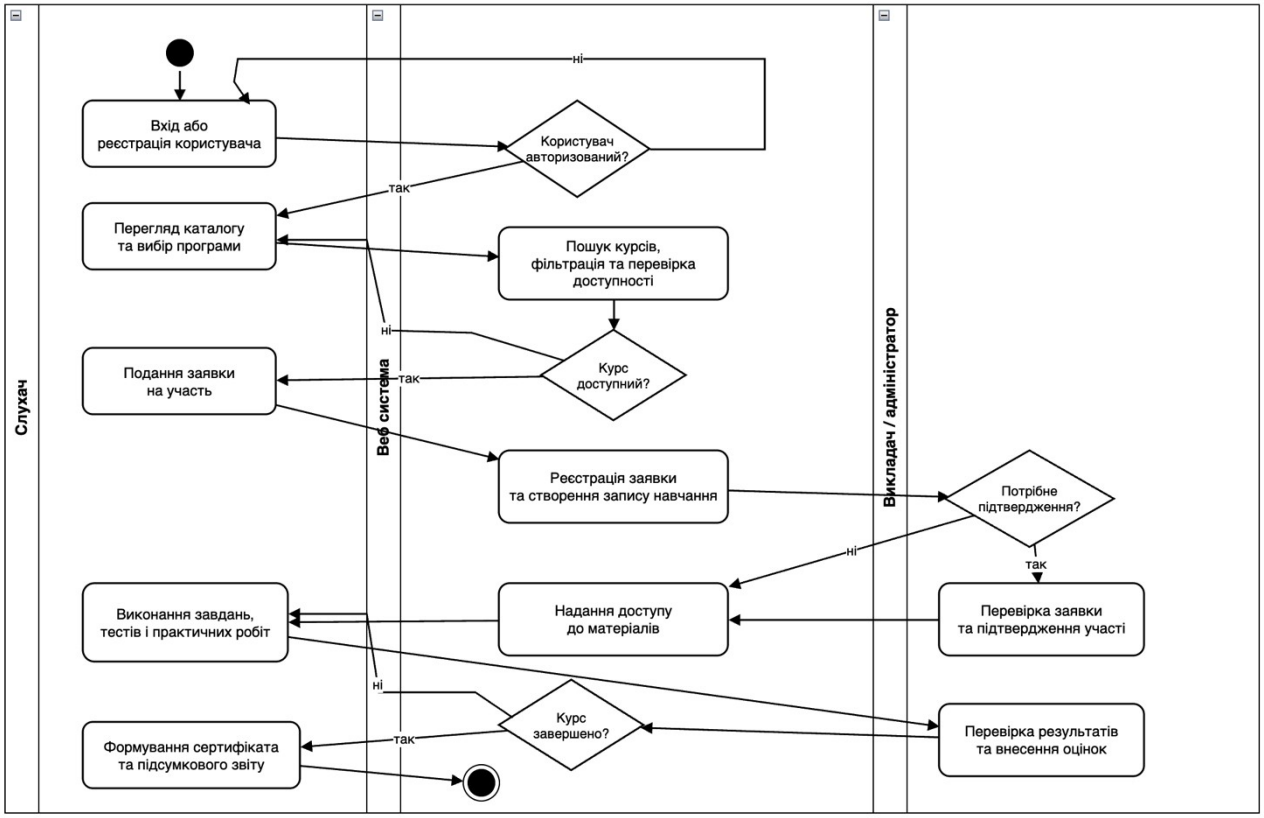

Рис. 1.9 - Діаграма активності проходження курсу слухачем

Сформовані діаграми забезпечують основу для подальшого проєктування класів, компонентів, пакетів і бази даних. Вони також допомагають узгодити текстову постановку задачі з реальною програмною реалізацією.

1.4 Аналіз вимог розроблюваної системи

Вимоги до системи сформовано з урахуванням потреб трьох основних груп користувачів: слухачів, викладачів і адміністраторів. Слухачеві потрібні прості засоби пошуку курсу, запису, проходження уроків і контролю результатів. Викладачеві потрібні засоби створення й наповнення курсу. Адміністратор має керувати структурою системи, користувачами та базовими довідниками.

Функціональні вимоги визначають дії, які система повинна виконувати безпосередньо. Їх наведено у таблиці 1.3. При формуванні вимог було враховано, що система реалізована як компактний вебзастосунок, тому

пріоритет надано критично важливим процесам, які демонструють повний навчальний цикл.

Таблиця 1.3

Функціональні вимоги розроблюваної системи

№	Вимога	Опис	Результат
1	Реєстрація та авторизація	Створення облікового запису та вхід за email і паролем	Користувач отримує доступ до ролі
2	Каталог курсів	Перегляд опублікованих курсів із пошуком і фільтрацією	Слухач знаходить потрібний курс
3	Запис на курс	Створення зв’язку між користувачем і курсом	Курс з’являється у кабінеті
4	Проходження уроків	Перегляд матеріалів і фіксація виконання	Оновлюється прогрес навчання
5	Підсумкове тестування	Виконання тесту з автоматичним підрахунком результату	Слухач отримує оцінку
6	Сертифікація	Формування сертифіката після успішного тесту	Підтверджується завершення курсу
7	Адміністрування	Створення та видалення курсів, контроль користувачів	Підтримується актуальність даних

Нефункціональні вимоги описують якість роботи системи, її надійність, зручність, безпеку і придатність до розгортання. Вони наведені в таблиці 1.4 і є критеріями для подальшого тестування.

Таблиця 1.4

Нефункціональні вимоги системи

Категорія	Вимога	Пояснення
Продуктивність	Відповідь сторінки не повинна перевищувати 1-2 с у локальному середовищі	Система використовує легку SQLite БД і стандартний HTTP-сервер

Продовження таблиці 1.4

Надійність	Некоректні дії не повинні призводити до аварійного завершення	Потрібні перевірки форм і повідомлення про помилки
Безпека	Паролі не зберігаються відкритим текстом	Використовується хешування паролів і сесійна ідентифікація
Зручність	Основні дії мають виконуватися без складного навчання користувача	Інтерфейс побудовано у стилі dashboard
Переносимість	Програма повинна запускатися без складної інсталяції	Достатньо Python 3 і запуску main.py
Супроводжуваність	Код має бути структурований за функціями обробки запитів	Це спрощує подальше розширення системи

Сформовані вимоги дають змогу перейти до постановки завдання та проєктування інформаційного забезпечення. Вони визначають межі програмної реалізації та критерії, за якими оцінюватиметься готовий прототип.

1.5 Постановка завдання

Завданням кваліфікаційної роботи є розробка веборієнтованої автоматизованої системи підтримки позауніверситетського навчання, яка

дозволяє організувати повний цикл роботи з навчальним курсом: від реєстрації слухача до проходження тестування та отримання сертифіката.

Вхідними даними системи є облікові записи користувачів, параметри курсів, навчальні уроки, тестові питання, записи про зарахування, позначки виконання уроків і результати тестових спроб. Вихідними даними є персональний кабінет слухача, перелік доступних курсів, розрахований прогрес, результат тестування та сертифікат про успішне завершення курсу.

У процесі розроблення необхідно реалізувати базу даних із таблицями users, courses, lessons, enrollments, lesson_progress, tests, test_attempts і certificates. Ці таблиці повинні бути пов’язані зовнішніми ключами, що забезпечує цілісність даних і коректне видалення пов’язаних записів.

Програмна реалізація повинна містити серверну частину, маршрути обробки HTTP-запитів, HTML-сторінки, CSS-оформлення, механізм роботи із сесіями, форми реєстрації, авторизації, запису на курс, проходження уроків і тестування. Для демонстраційного запуску система має використовувати локальний сервер і автоматично створювати базу даних після першого старту.

Таблиця 1.5

Вхідні та вихідні дані системи

Група даних	Приклад	Призначення
Вхідні дані	Email, пароль, роль користувача	Створення облікового запису
Вхідні дані	Назва курсу, категорія, рівень, опис	Формування каталогу курсів
Вхідні дані	Текст уроку, позиція уроку	Наповнення структури навчального матеріалу
Проміжні дані	Запис enrollment, позначки lesson_progress	Розрахунок прогресу слухача
Вихідні дані	Оцінка test_attempt, статус passed	Фіксація результату тестування

Вихідні дані	Номер і дата сертифіката	Підтвердження успішного завершення курсу

Отже, постановка завдання визначає практичний напрям розроблення: створення компактної, але повноцінної вебсистеми, яка демонструє основні процеси позауніверситетського навчання та може бути використана як основа для подальшого розвитку.

1.6 Висновки до першого розділу

У першому розділі виконано системний аналіз предметної області підтримки позауніверситетського навчання. Визначено, що така система повинна забезпечувати не лише перегляд навчальних матеріалів, а й запис на курси, фіксацію прогресу, тестування та підтвердження результатів.

Проведено порівняльний аналіз існуючих рішень, зокрема Moodle, Google Classroom, Coursera, Udemy та Prometheus. Встановлено, що для поставленої задачі доцільно створити легку автономну систему, яка поєднує каталог курсів, персональний кабінет, прогрес і сертифікацію без складного адміністрування.

За допомогою UML-моделювання уточнено ролі користувачів, ключові сценарії та загальну логіку проходження курсу. Сформовано функціональні, нефункціональні та технічні вимоги, які стали основою для подальшого проєктування інформаційного та програмного забезпечення.

2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Логічна модель даних у вигляді ER-діаграми

Логічна модель даних визначає структуру інформаційного забезпечення системи та показує, які сутності необхідно зберігати для підтримки навчального процесу. У розроблюваній системі центральними сутностями є користувач, курс, урок, запис на курс, прогрес уроку, тест, тестова спроба та сертифікат.

ER-діаграма подана на рис. 2.1. Вона відображає зв’язки між основними таблицями. Один користувач може мати багато записів на курси, один курс містить багато уроків і тестових питань, а кожний запис про прогрес пов’язує конкретного користувача з конкретним уроком.

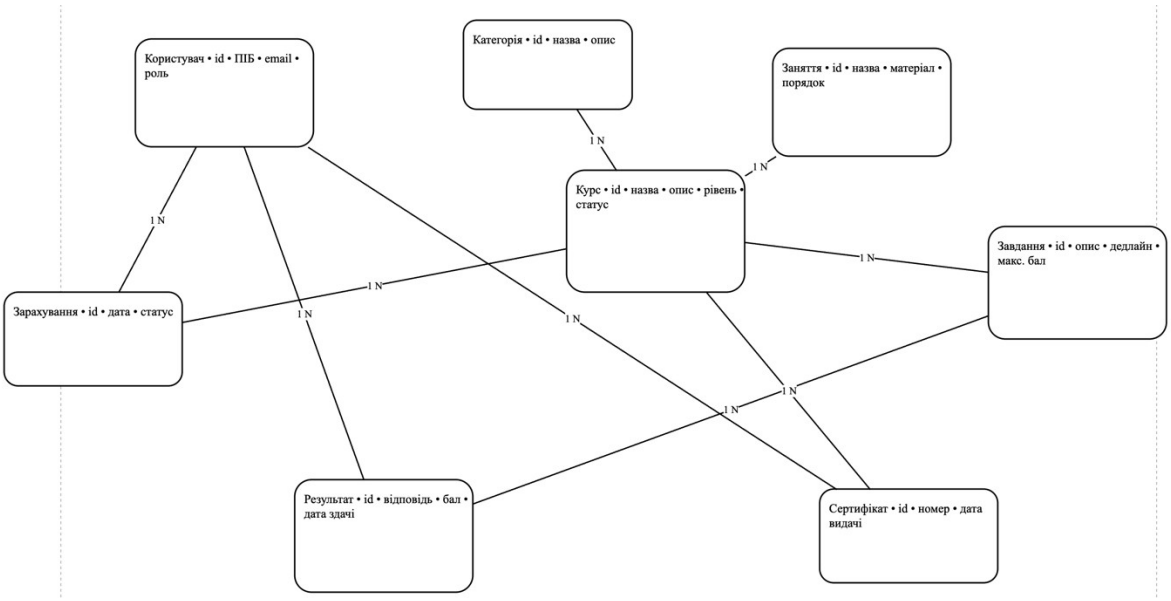

Рис. 2.1 - ER-діаграма логічної моделі даних системи підтримки позауніверситетського навчання

Структура, наведена на рисунку 2.1, забезпечує нормалізоване збереження даних. Навчальні матеріали не дублюються для кожного слухача, оскільки прогрес зберігається окремо у таблиці lesson_progress. Аналогічно

тестові завдання належать до курсу, а спроби тестування зберігають лише результат конкретного користувача.

Таблиця 2.1

Основні сутності логічної моделі даних

Сутність	Ключові атрибути	Призначення
User	id, full_name, email, password_hash, role	Зберігання облікових записів і ролей
Course	id, title, category, level, description, duration_hours	Опис навчального курсу
Lesson	id, course_id, title, content, position	Структура навчальних матеріалів
Enrollment	id, user_id, course_id, status, enrolled_at	Фіксація запису слухача на курс
LessonProgress	user_id, lesson_id, is_done, done_at	Контроль виконання уроків
Test	id, course_id, question, options, correct_option	Підсумкові питання курсу
TestAttempt	id, user_id, course_id, score, total, passed	Результат проходження тесту
Certificate	id, user_id, course_id, certificate_no, issued_at	Підтвердження завершення курсу

Таблиця 2.1 узагальнює сутності, які реалізують базову інформаційну модель. Така структура є достатньою для демонстраційного прототипу і водночас може бути розширена додатковими таблицями для коментарів, платежів, розкладу занять або завантаження файлів.

2.2 Діаграма класів та кооперації

Діаграма класів описує програмну модель системи з позиції об’єктно-орієнтованого проєктування. У спрощеній структурі виділено класи User,

Course, Lesson, Enrollment, Test, Certificate, AuthService, CourseService, ProgressService і Database. Такий поділ дозволяє логічно відокремити дані предметної області від сервісів, які виконують операції над ними.

На рис. 2.2 подано діаграму класів. Вона відображає, що користувач взаємодіє з курсами через запис Enrollment, прогрес формується на основі уроків, а сертифікат залежить від успішного проходження тесту.

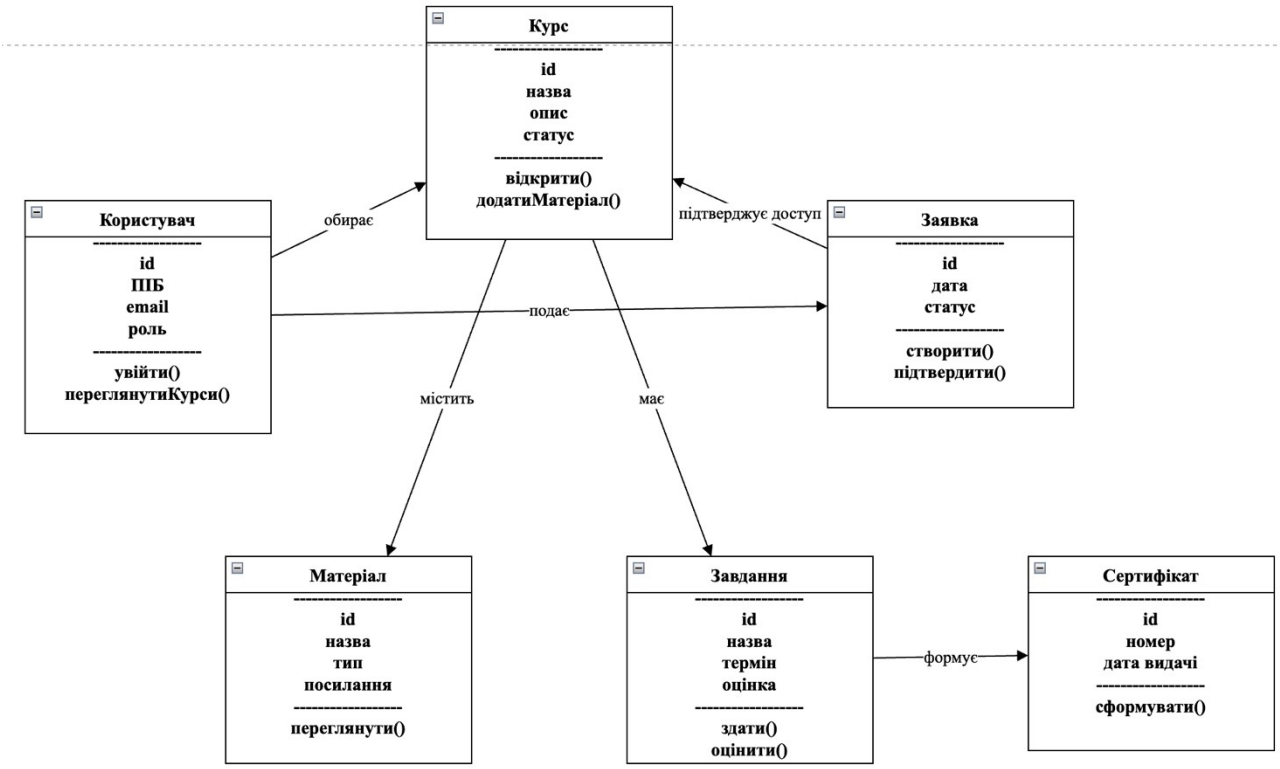

Рис. 2.2 - Діаграма класів веборієнтованої системи підтримки позауніверситетського навчання

Для деталізації взаємодій побудовано три діаграми кооперації. Перша кооперація описує реєстрацію й авторизацію користувача, друга - запис на курс і створення навчального кабінету, третя - проходження тестування та формування сертифіката.

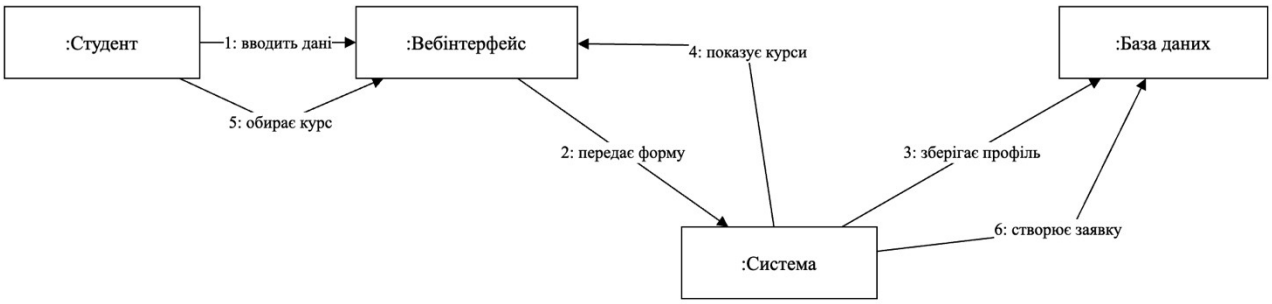

Рис. 2.3 - Кооперація «Реєстрація та авторизація користувача»

Кооперація, показана на рис. 2.3, демонструє, що процес входу не повинен безпосередньо звертатися до навчальних модулів. Спочатку система встановлює особу користувача, визначає його роль і тільки після цього надає доступ до кабінету або адміністративної панелі.

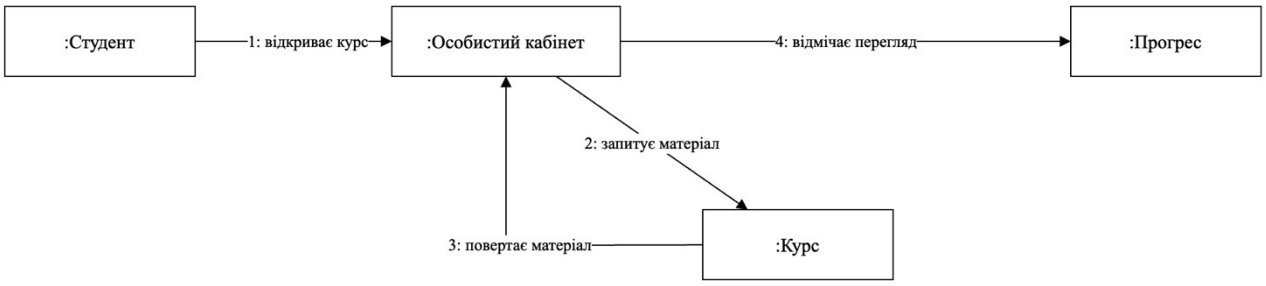

Рис. 2.4 - Кооперація «Запис слухача на курс»

Кооперація, наведена на рис. 2.4, пов’язує каталог курсів із персональним кабінетом. Після запису користувача на курс система створює відповідний запис у базі даних, а кабінет відображає статус проходження та доступні уроки.

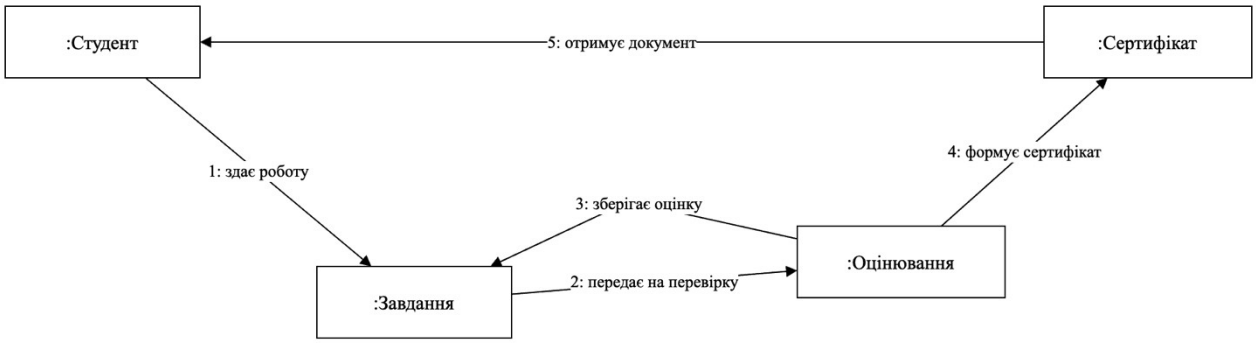

Рис. 2.5 - Кооперація «Тестування та сертифікація»

Рисунок 2.5 показує, що сертифікат не створюється вручну, а формується як результат успішного тестування. Це зменшує ймовірність помилок і

забезпечує зв’язок між фактичним навчальним результатом і документом, який підтверджує завершення курсу.

2.3 Діаграма компонентів

Діаграма компонентів відображає фізичну та логічну структуру програмної системи. У розроблюваному прототипі виділено вебінтерфейс, HTTP-сервер, модуль автентифікації, модуль курсів, модуль прогресу, модуль тестування, модуль сертифікації та базу даних SQLite.

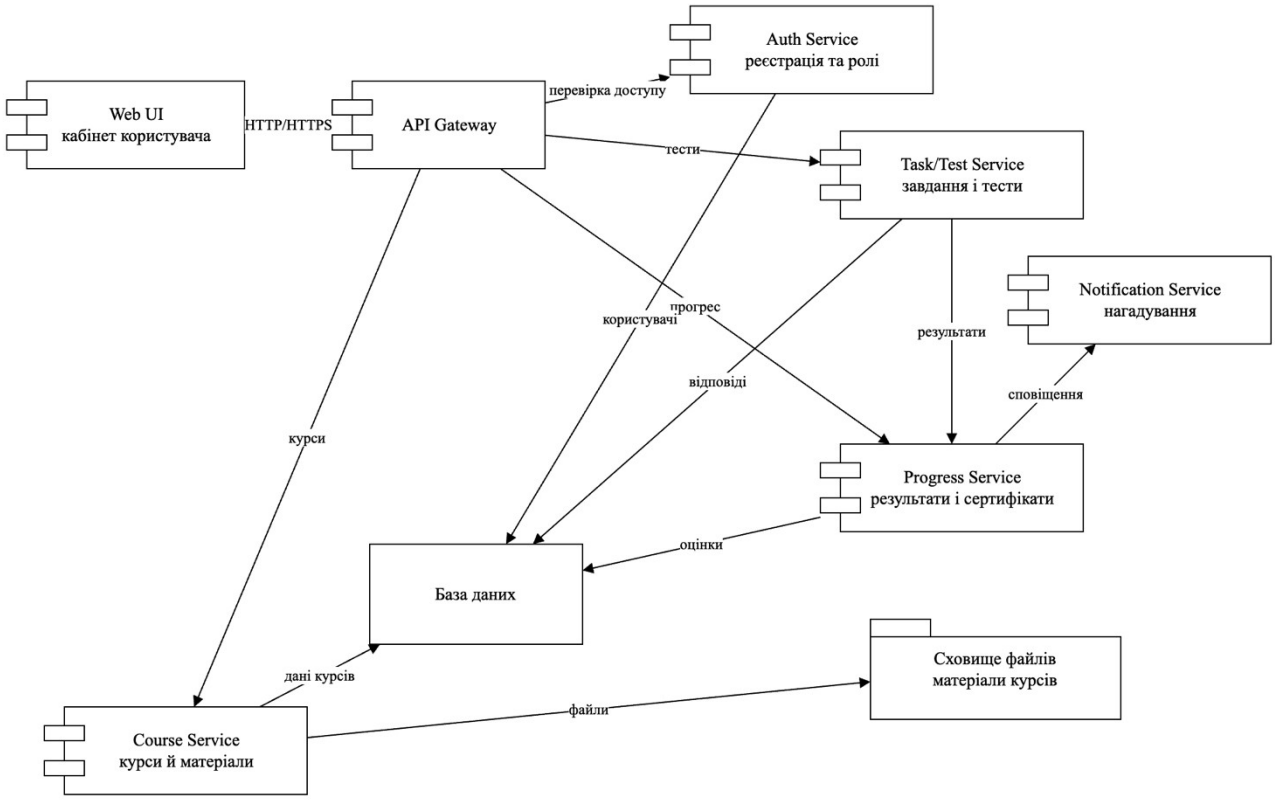

Рис. 2.6 - Діаграма компонентів програмної системи

Як видно з рисунка 2.6, серверна частина є центральним координатором усіх дій. Вона приймає запити з браузера, перевіряє сесію користувача, виконує потрібну бізнес-операцію та повертає HTML-відповідь. Такий підхід спрощує реалізацію і добре підходить для локального або навчального прототипу.

Таблиця 2.2

Призначення компонентів системи

Компонент	Призначення	Взаємодія
Web UI	Відображення сторінок, форм, карток курсів і кабінету	Надсилає HTTP-запити серверу
HTTP Server	Обробка маршрутів і формування відповідей	Керує всіма модулями системи
Auth Module	Реєстрація, вхід, вихід, перевірка ролей	Працює з users і сесіями
Course Module	Каталог, пошук, створення курсів	Працює з courses і lessons
Progress Module	Фіксація виконаних уроків	Працює з lesson_progress
Test Module	Перевірка тестів і підрахунок балів	Працює з tests і test_attempts
Certificate Module	Створення сертифікатів	Працює з certificates
SQLite DB	Збереження всіх даних	Надає дані для модулів

Таблиця 2.2 деталізує функціональне призначення компонентів. Компонентна структура є простою, але вона дозволяє розвивати систему: наприклад, винести базу даних на окремий сервер, додати REST API або замінити HTML-відповіді на клієнтський SPA-застосунок.

2.4 Діаграма пакетів

Діаграма пакетів використовується для відображення організації програмного коду. Для даної системи доцільно виділити пакети interface, auth, courses, learning, testing, certificates, database та utilities. У межах навчального прототипу ці пакети можуть бути реалізовані в одному файлі або групі файлів, але логічне розділення допомагає підтримувати структуру програми.

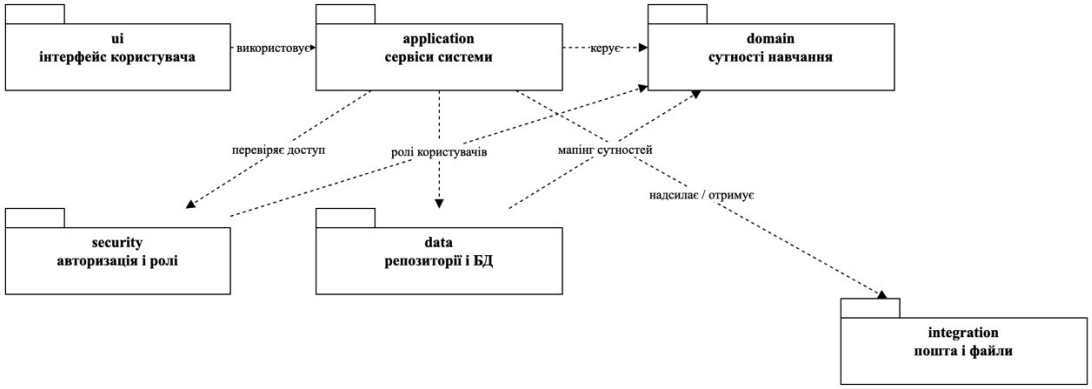

Рис. 2.7 - Діаграма пакетів веборієнтованої системи підтримки позауніверситетського навчання

Рисунок 2.7 демонструє залежності між пакетами. Інтерфейсний пакет не повинен напряму змінювати дані без виклику відповідних сервісів. Пакет database є нижнім рівнем, який забезпечує виконання SQL-запитів, а пакети предметної логіки використовують його для збереження результатів.

2.5 Фізична модель даних

Фізична модель даних реалізована засобами SQLite. Вибір SQLite обґрунтований тим, що система має демонстраційний і навчальний характер, не потребує окремого серверного СКБД та може запускатися на будь-якому комп’ютері з Python. Файл бази даних створюється автоматично після першого запуску програми.

У фізичній моделі використано первинні ключі INTEGER PRIMARY KEY AUTOINCREMENT, зовнішні ключі з каскадним видаленням, унікальні обмеження для email користувача та унікальності запису слухача на курс. Для прискорення пошуку передбачено індекси за email, курсом і користувачем.

Таблиця 2.3

Таблиці фізичної моделі даних

Таблиця	Ключові поля	Обмеження
users	id, email, password_hash,	UNIQUE(email), role IN

	role	student/teacher/admin
courses	id, title, category, level	is_published, created_at
lessons	id, course_id, position	FOREIGN KEY course_id, ON DELETE CASCADE
enrollments	user_id, course_id, status	UNIQUE(user_id, course_id)
lesson_progress	user_id, lesson_id, is_done	UNIQUE(user_id, lesson_id)
tests	course_id, question, correct_option	correct_option IN A/B/C/D
test_attempts	user_id, course_id, score, total, passed	Збереження історії спроб
certificates	user_id, course_id, certificate_no	UNIQUE(certificate_no)

Фізична модель відповідає логічній ER-структурі та забезпечує контроль цілісності на рівні бази даних. Це особливо важливо для записів прогресу і сертифікатів, оскільки вони повинні залишатися узгодженими з користувачем і курсом.

2.6 Висновки до другого розділу

У другому розділі виконано проєктування інформаційного та програмного забезпечення системи. Побудовано логічну ER-модель, яка охоплює користувачів, курси, уроки, записи на курси, прогрес, тести, результати тестування і сертифікати.

Розроблено діаграму класів і три кооперації, що деталізують процеси реєстрації, запису на курс і сертифікації. Також визначено компонентну структуру системи та пакетну організацію програмного коду. Окремо сформовано фізичну модель даних SQLite із ключами, обмеженнями та зв’язками.

Отримані проєктні рішення забезпечують узгоджену основу для практичної реалізації системи, опису архітектури, алгоритмізації програмних модулів і подальшого тестування.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір технологій та інструментальних засобів реалізації системи

Для реалізації веборієнтованої автоматизованої системи підтримки позауніверситетського навчання обрано технологічний стек, який дає змогу створити працездатний прототип без складної інсталяції зовнішніх залежностей. Основою серверної частини є Python 3 та стандартний модуль http.server, що дозволяє швидко запустити локальний HTTP-сервер.

Для збереження даних використано SQLite. Такий вибір є доцільним для навчальної системи, оскільки база даних розміщується в одному файлі, не потребує окремого сервера та підтримує SQL-запити, зовнішні ключі, індекси і транзакції. Це дозволяє реалізувати повноцінну структуру даних без ускладнення розгортання.

Клієнтська частина побудована на HTML і CSS із використанням серверної генерації сторінок. Інтерфейс має темний dashboard-дизайн із боковою навігацією, картками курсів, інформаційними блоками та формами. Такий підхід забезпечує зрозумілу навігацію для слухача й адміністратора.

Загальну характеристику обраних технологій наведено в таблиці 3.1. Вибір інструментів відповідає вимогам простоти запуску, наочності, достатньої функціональності та можливості демонстрації повного навчального циклу.

Таблиця 3.1

Технології реалізації програмної системи

Технологія	Призначення	Обґрунтування вибору
Python 3	Серверна логіка та обробка запитів	Простота, наявність стандартних модулів, швидке прототипування

Продовження таблиці 3.1

http.server	Локальний HTTP-сервер	Дозволяє запускати систему без сторонніх фреймворків
SQLite	Файлова реляційна база даних	Не потребує окремого сервера, підтримує SQL і зовнішні ключі
HTML	Структура вебсторінок	Стандартна мова розмітки для браузера
CSS	Оформлення інтерфейсу	Дає змогу реалізувати темний dashboard-дизайн
JavaScript	Допоміжна клієнтська логіка	Може використовуватися для покращення інтерактивності
ZIP-пакет	Поширення програми	Зручно передавати повний проєкт із README та файлами запуску

Як показано у таблиці 3.1, система реалізована з урахуванням мінімальних вимог до середовища запуску. Це робить її придатною для демонстрації на звичайному персональному комп’ютері без налаштування Docker, PostgreSQL або окремого вебсервера.

3.2 Алгоритмізація програмних модулів системи

Алгоритмізація програмних модулів забезпечує формалізований опис логіки роботи системи. У межах роботи розглянуто три ключові алгоритми: авторизація користувача, вибір і запис на курс, проходження курсу з тестуванням та формуванням сертифіката.

Алгоритм авторизації починається з введення email і пароля. Сервер отримує дані форми, знаходить користувача в базі даних, обчислює хеш введеного пароля та порівнює його зі збереженим значенням. Якщо перевірка успішна, створюється сесія, і користувач перенаправляється до кабінету.

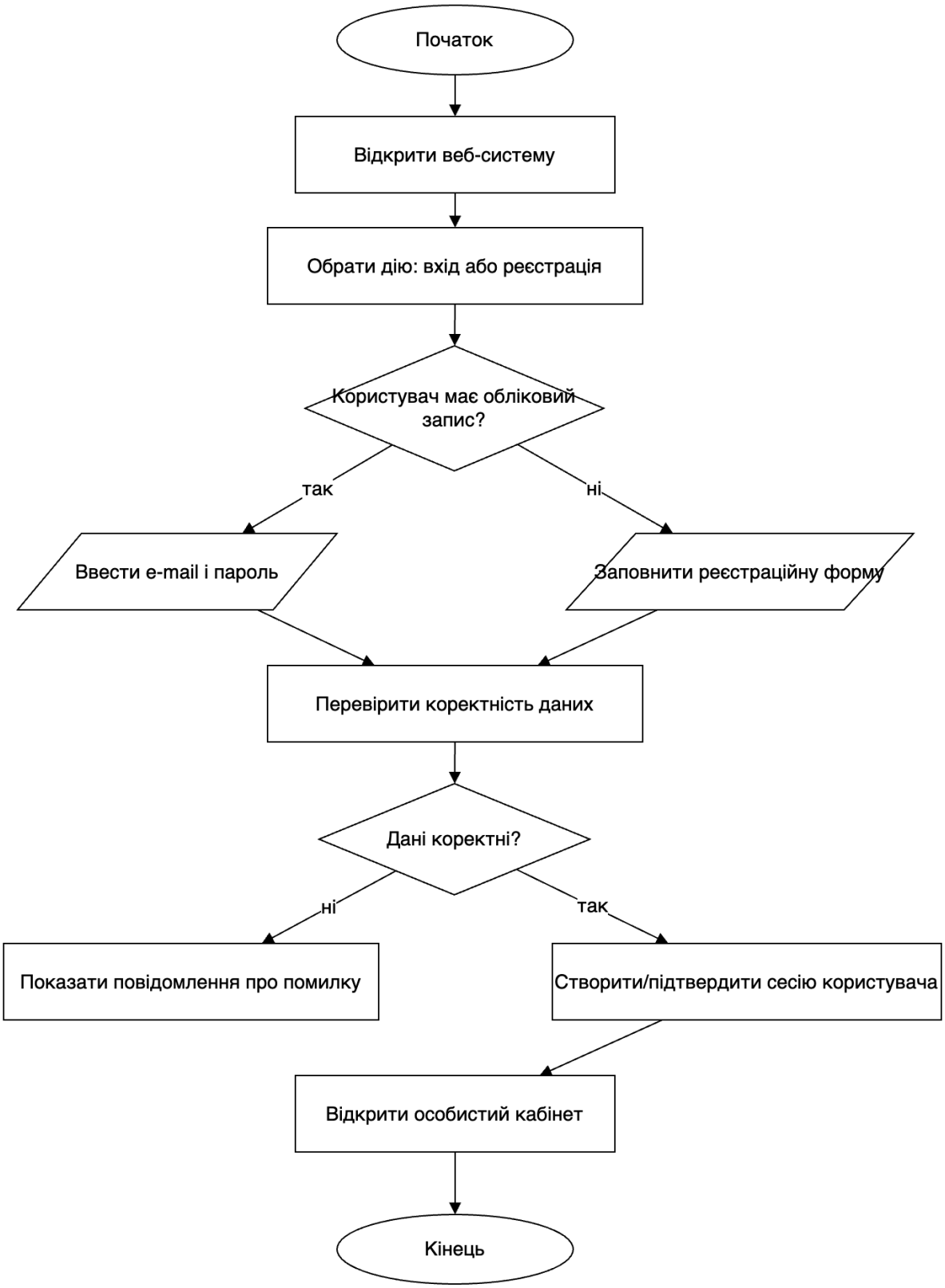

Рис. 3.1 - Блок-схема алгоритму авторизації користувача

Алгоритм запису на курс передбачає перевірку авторизації, наявності курсу та відсутності попереднього запису. Якщо всі умови виконано,

створюється запис у таблиці enrollments. Блок-схему цього алгоритму подано на рис. 3.2.

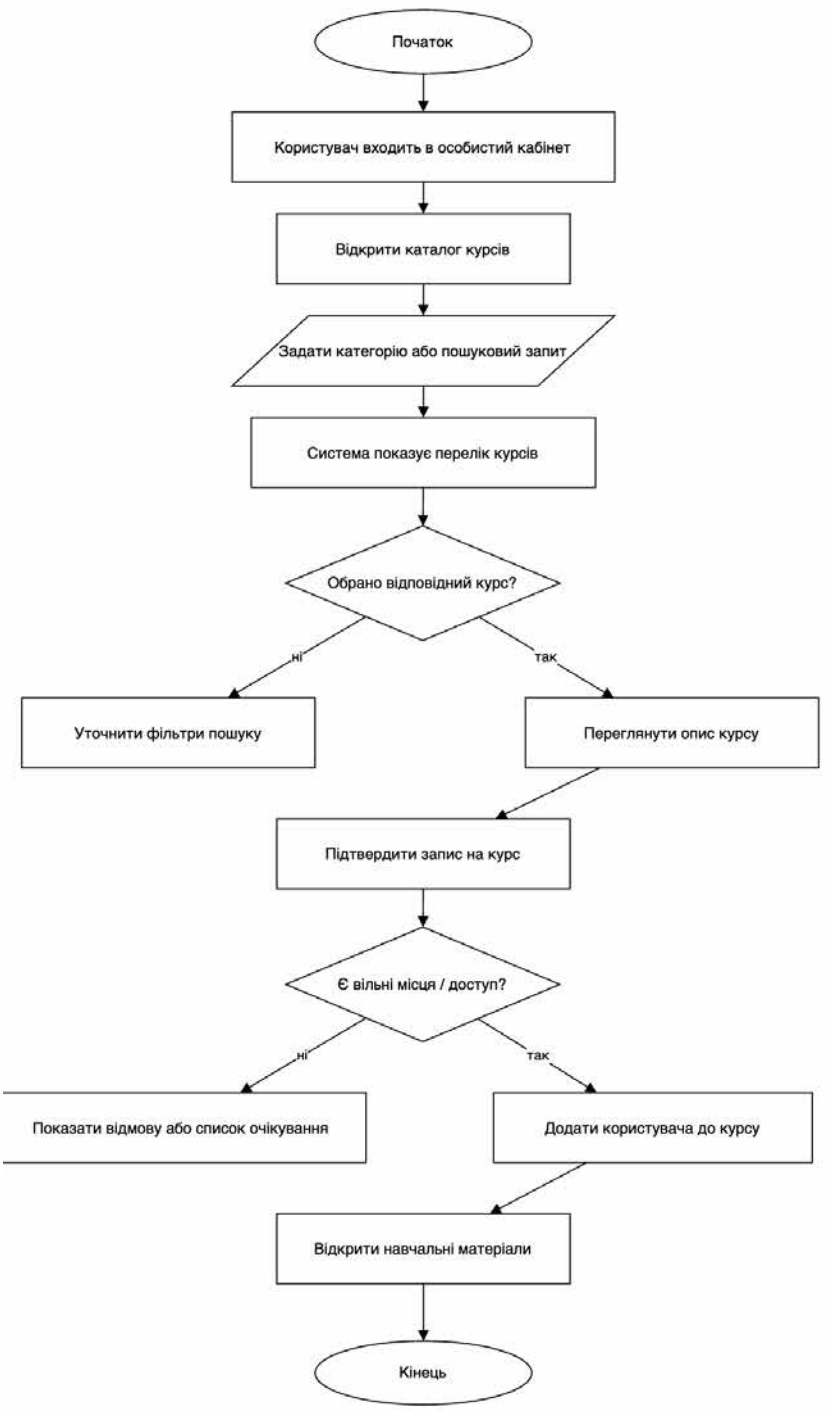

Рис. 3.2 - Блок-схема алгоритму вибору та запису на курс

Третій алгоритм описує проходження курсу. Слухач відкриває уроки, система фіксує виконання, розраховує прогрес і після достатнього опрацювання

матеріалів дозволяє пройти тест. Якщо тест складено успішно, створюється сертифікат.

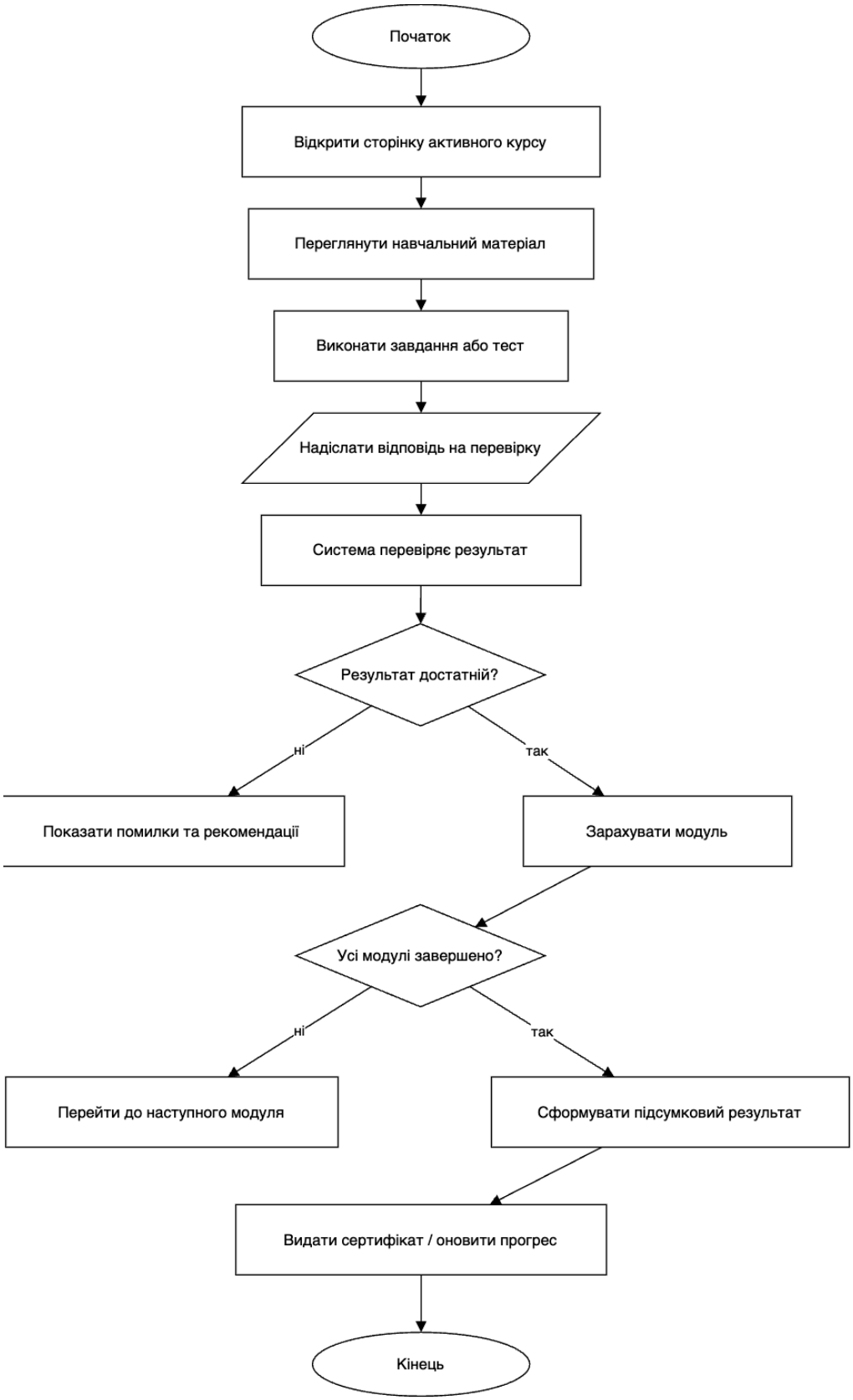

Рис. 3.3 - Блок-схема алгоритму проходження курсу, тестування і сертифікації

Таблиця 3.3

Алгоритмічні модулі системи

Модуль	Вхідні дані	Основна дія	Результат
Авторизація	email, password	Перевірка облікових даних	Активна сесія користувача
Каталог	пошуковий запит, фільтри	Вибірка курсів із БД	Список доступних курсів
Запис	user_id, course_id	Створення enrollment	Курс додано до кабінету
Прогрес	user_id, lesson_id	Оновлення lesson_progress	Змінено відсоток проходження
Тестування	відповіді користувача	Підрахунок правильних відповідей	Створено test_attempt
Сертифікація	успішний test_attempt	Генерація номера сертифіката	Створено certificate

Таблиця 3.3 показує, що кожний алгоритм має чітко визначені вхідні дані та результат. Це спрощує тестування, оскільки для кожного модуля можна підготувати окремі контрольні сценарії.

3.4 Реалізація бази даних та інтерфейсних модулів

Реалізація бази даних виконується автоматично під час першого запуску програми. Якщо файл бази даних відсутній, застосунок створює необхідну структуру таблиць, вмикає підтримку зовнішніх ключів і додає демонстраційні облікові записи адміністратора, слухача та викладача.

Демонстраційні акаунти потрібні для швидкої перевірки ролей без ручного створення користувачів. Адміністратор має доступ до панелі керування курсами, слухач може проходити навчання, а викладач використовується для демонстрації рольової моделі.

Інтерфейс реалізовано у вигляді темної dashboard-структури. Ліва панель навігації містить основні переходи, центральна область відображає картки курсів, статистику, форми та навчальні матеріали. Такий дизайн відрізняється від стандартного світлого шаблону та робить застосунок візуально завершеним.

Таблиця 3.4

Демонстраційні облікові записи

Роль	Email	Пароль	Призначення
Адміністратор	admin@example.com	admin123	Керування курсами та перегляд системних даних
Слухач	student@example.com	student123	Перевірка проходження курсу, уроків і тесту
Викладач	teacher@example.com	teacher123	Демонстрація окремої ролі для наповнення курсів

Реалізована програма має самодостатню структуру: основний файл main.py містить серверну логіку, обробку маршрутів, функції доступу до бази даних і HTML-шаблони. Файл README.md описує запуск, демонстраційні акаунти й основні можливості системи. Для зручності передбачено стартові файли для Windows та macOS/Linux.

3.5 Висновки до третього розділу

У третьому розділі обґрунтовано вибір технологій реалізації системи. Для серверної частини використано Python 3 і стандартний HTTP-сервер, для

збереження даних - SQLite, для побудови інтерфейсу - HTML і CSS. Такий стек забезпечує простий запуск і не потребує встановлення складних залежностей.

Спроєктовано архітектуру вебзастосунку, структуру сторінок, маршрути, алгоритми авторизації, запису на курс, проходження уроків, тестування та сертифікації. Реалізація охоплює повний навчальний цикл і відповідає вимогам, визначеним у першому та другому розділах.

Отриманий програмний прототип є основою для подальшого тестування, оцінювання ефективності, перевірки інтерфейсу та формування інсталяційного пакета.

4 ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ СИСТЕМИ

4.1 План тестування програмних модулів та методика оцінювання результатів

Тестування веборієнтованої автоматизованої системи підтримки позауніверситетського навчання спрямоване на перевірку коректності роботи основних модулів, стабільності бази даних, правильності рольового доступу, зручності інтерфейсу та працездатності процесу розгортання.

План тестування охоплює функціональні, інтеграційні, інтерфейсні та інсталяційні перевірки. Функціональні тести підтверджують виконання окремих дій. Інтеграційні тести перевіряють взаємодію між модулями. Інтерфейсні тести оцінюють зрозумілість сторінок і коректність форм. Інсталяційні тести перевіряють запуск програми в локальному середовищі.

Таблиця 4.1

План тестування програмних модулів

№	Модуль	Перевірка	Очікуваний результат
1	Авторизація	Вхід із правильними та неправильними даними	Коректний вхід або повідомлення про помилку
2	Реєстрація	Створення нового слухача	Новий користувач зберігається у БД
3	Каталог курсів	Пошук і фільтрація курсів	Відображається релевантний список
4	Запис на курс	Натискання кнопки запису	Створюється enrollment
5	Уроки	Позначення	Оновлюється

		уроку виконаним	lesson_progress

6	Тестування	Надсилання відповідей	Підраховується бал і статус passed
7	Сертифікат	Успішне проходження тесту	Створюється сертифікат
8	Адмінпанель	Створення та видалення курсу	Зміни відображаються у каталозі
9	Розгортання	Запуск main.py	Сервер запускається і показує URL
10	Порт сервера	Запуск при зайнятому 8000 порту	Система автоматично переходить на 8001/8002

Таблиця 4.1 визначає основні перевірки, які повинні бути виконані для підтвердження працездатності системи. Особливу увагу приділено сценарію зайнятого порту, оскільки під час практичного запуску попередній процес сервера може залишатися активним.

4.2 Тестування веборієнтованої системи підтримки позауніверситетського навчання

Тестування веборієнтованої автоматизованої системи підтримки позауніверситетського навчання виконувалося з метою перевірки працездатності основних функціональних модулів, коректності взаємодії інтерфейсу з базою даних, правильності оброблення користувацьких дій та зручності проходження навчального процесу. Основна увага приділялася перевірці сценаріїв, які безпосередньо використовує слухач: реєстрація, вхід до системи, перегляд каталогу курсів, запис на курс, перегляд навчального

кабінету, проходження матеріалів, оновлення прогресу та відображення результатів.

Тестування проводилося у браузері після запуску локального вебсервера. Як сховище даних використовувалася база SQLite, у якій зберігалися облікові записи користувачів, перелік курсів, записи на курси, навчальні матеріали, результати проходження уроків і дані про прогрес. Для перевірки роботи системи використовувалися демонстраційні облікові записи адміністратора, студента та викладача, а також створювався новий обліковий запис слухача через форму реєстрації.

На початковому етапі перевірено коректність завантаження головної сторінки системи. Головна сторінка містить назву платформи Extra Learning, навігаційне меню, кнопки переходу до входу та реєстрації, блок із демонстраційними акаунтами, статистичні показники та перелік основних можливостей системи. Відображення головної сторінки наведено на рисунку 4.1.

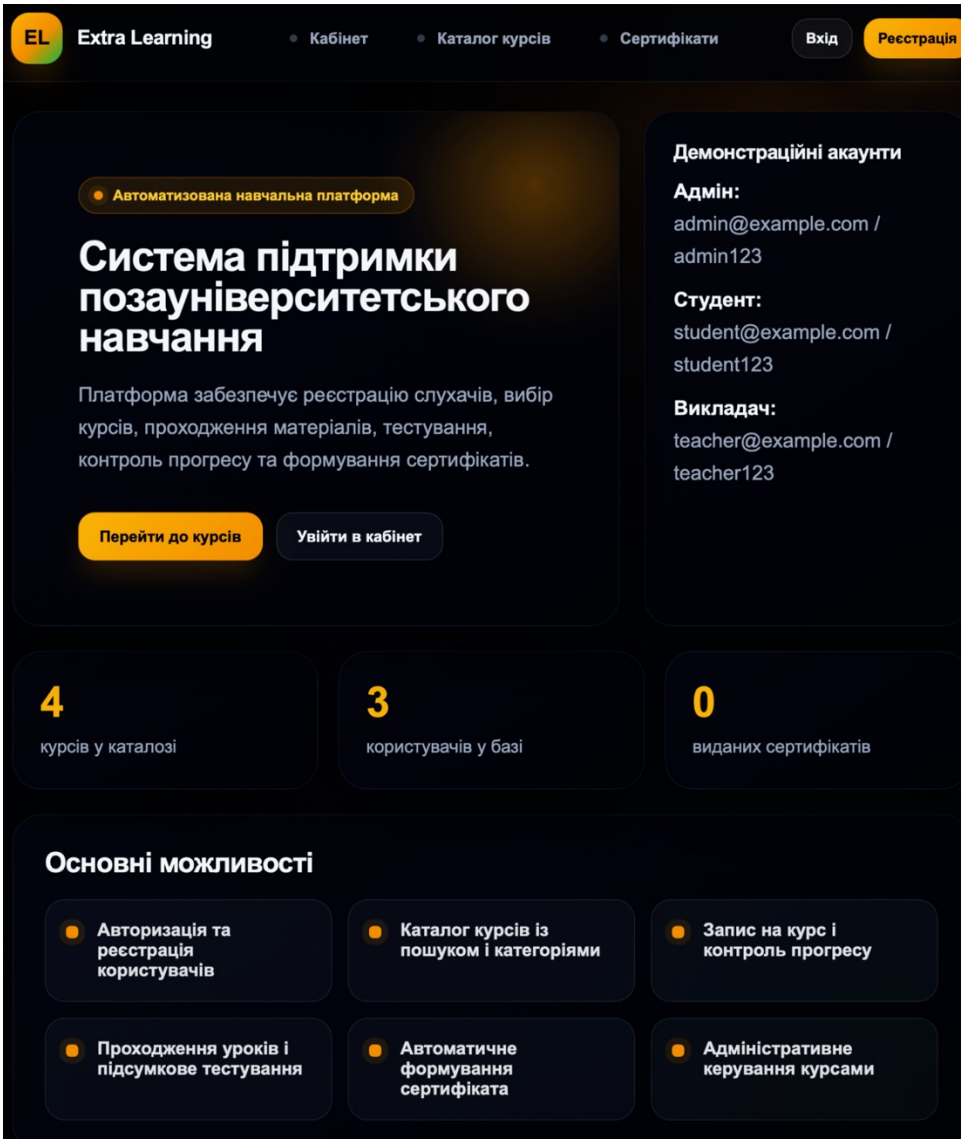

Рис. 4.1 – Головна сторінка веборієнтованої системи підтримки позауніверситетського навчання

Як видно з рисунка 4.1, інтерфейс системи має зрозумілу структуру: у верхній частині розташовано навігаційне меню, у центральній частині подано короткий опис призначення платформи, а нижче наведено основні функції. Під час тестування перевірено, що кнопки «Перейти до курсів», «Увійти в кабінет», «Вхід» і «Реєстрація» коректно відкривають відповідні сторінки. Також перевірено відображення статистичних показників: кількості курсів, користувачів і виданих сертифікатів. Дані відображаються без помилок, що підтверджує правильне зчитування інформації з бази даних.

Наступним етапом стало тестування модуля реєстрації слухача. Форма реєстрації містить поля для введення ПІБ, адреси електронної пошти та пароля. Вигляд сторінки реєстрації подано на рисунку 4.2.

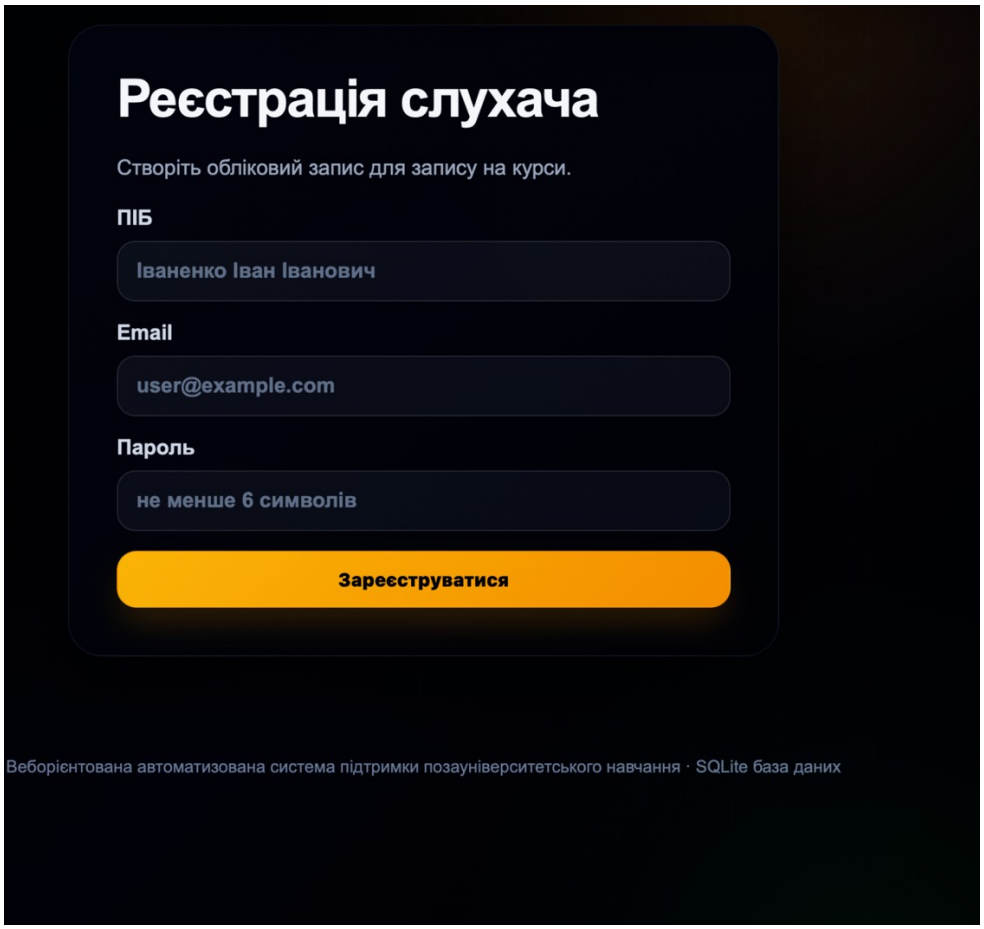

Рис. 4.2 – Форма реєстрації слухача

Під час тестування форми реєстрації перевірено введення коректних і некоректних даних. У разі заповнення всіх полів система створює новий обліковий запис користувача з роллю слухача та зберігає його в базі даних. Також перевірено обмеження на мінімальну довжину пароля та наявність обов’язкових полів. Якщо користувач залишає поле порожнім або вводить некоректні дані, система не повинна створювати неповний запис у базі. За результатами перевірки встановлено, що форма реєстрації працює коректно, а створений користувач може бути використаний для подальшого входу до системи.

Після реєстрації перевірено модуль авторизації. Сторінка входу містить поля для введення email і пароля, а також посилання для переходу до створення нового облікового запису. Сторінку входу до системи наведено на рисунку 4.3.

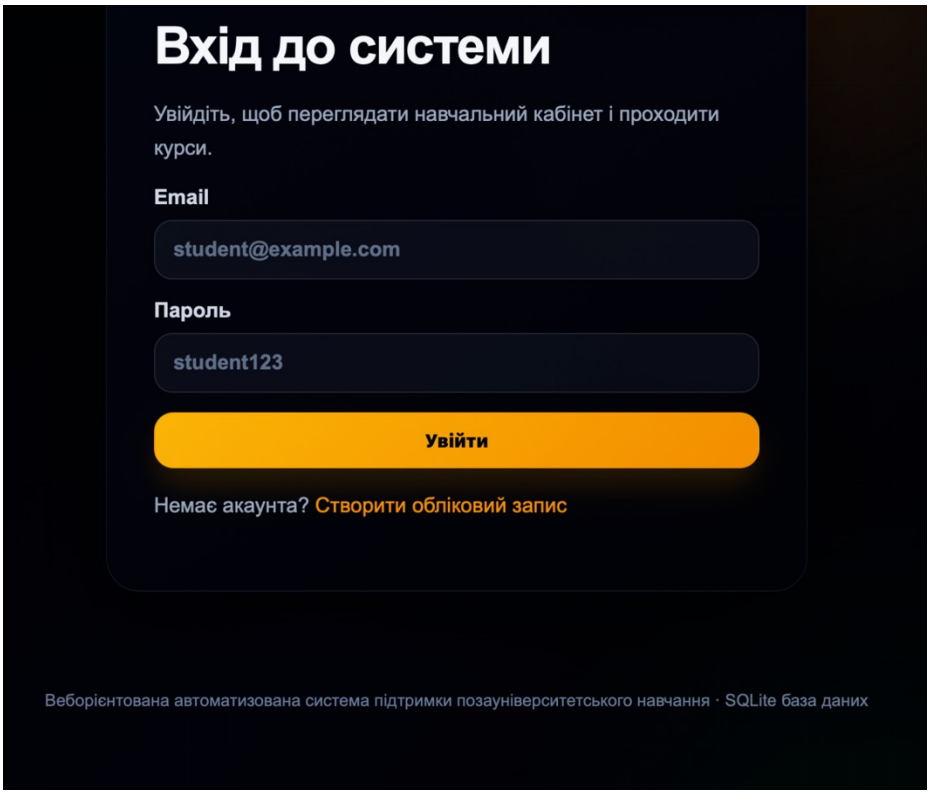

Рис. 4.3 – Форма входу до системи

Тестування авторизації виконувалося з використанням демонстраційного облікового запису студента. Під час введення правильних даних користувач успішно переходить до навчального кабінету. У разі введення неправильного пароля або неіснуючої електронної пошти доступ не надається. Це підтверджує, що система перевіряє облікові дані перед відкриттям персонального кабінету. Також перевірено, що після входу користувач отримує доступ саме до функцій слухача, а не до адміністративних можливостей.

Далі виконано тестування каталогу курсів. Каталог є одним із головних модулів системи, оскільки через нього слухач обирає доступні навчальні програми. У каталозі відображаються картки курсів із назвою, категорією, рівнем складності, коротким описом, тривалістю та кнопкою перегляду

детальної інформації. Приклад сторінки каталогу курсів наведено на рисунку 4.4.

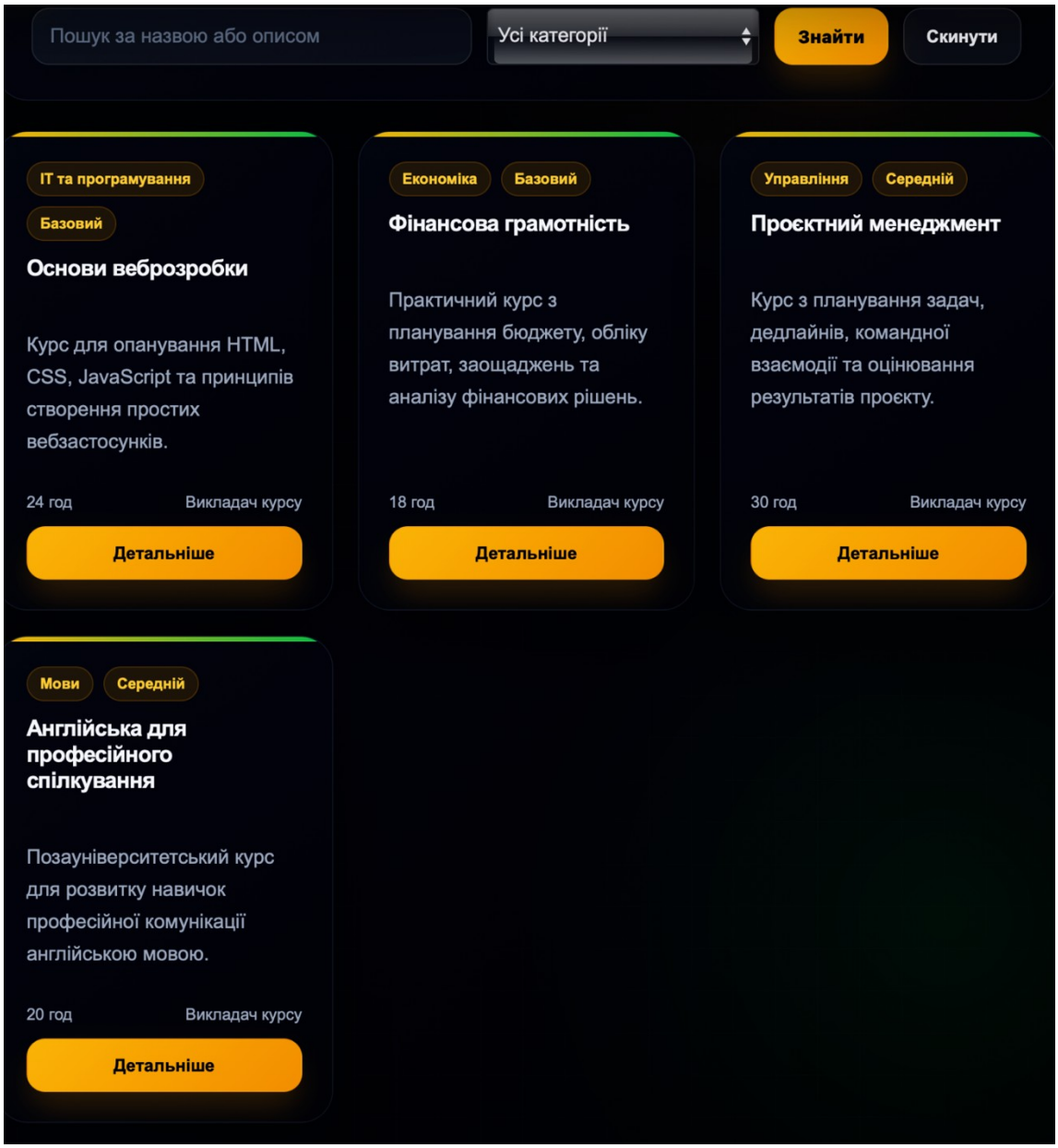

Рис. 4.4 – Каталог курсів із пошуком та фільтрацією

Як показано на рисунку 4.4, каталог містить курси «Основи веброзробки», «Фінансова грамотність», «Проєктний менеджмент» та «Англійська для професійного спілкування». Під час тестування перевірено роботу пошуку за назвою або описом, а також фільтрацію за категоріями. Після введення пошукового запиту система залишає на сторінці лише ті курси, які відповідають введеним умовам. Кнопка «Скинути» повертає повний перелік

курсів. За результатами перевірки встановлено, що каталог курсів працює стабільно, а фільтри не порушують структуру відображення карток.

Після перевірки каталогу протестовано навчальний кабінет користувача. До запису на курс кабінет відображає повідомлення про те, що слухач ще не записаний на навчальні програми. Також на сторінці міститься блок останніх результатів тестування та пояснення логіки проходження курсу. Вигляд кабінету користувача до запису на курс наведено на рисунку 4.5.

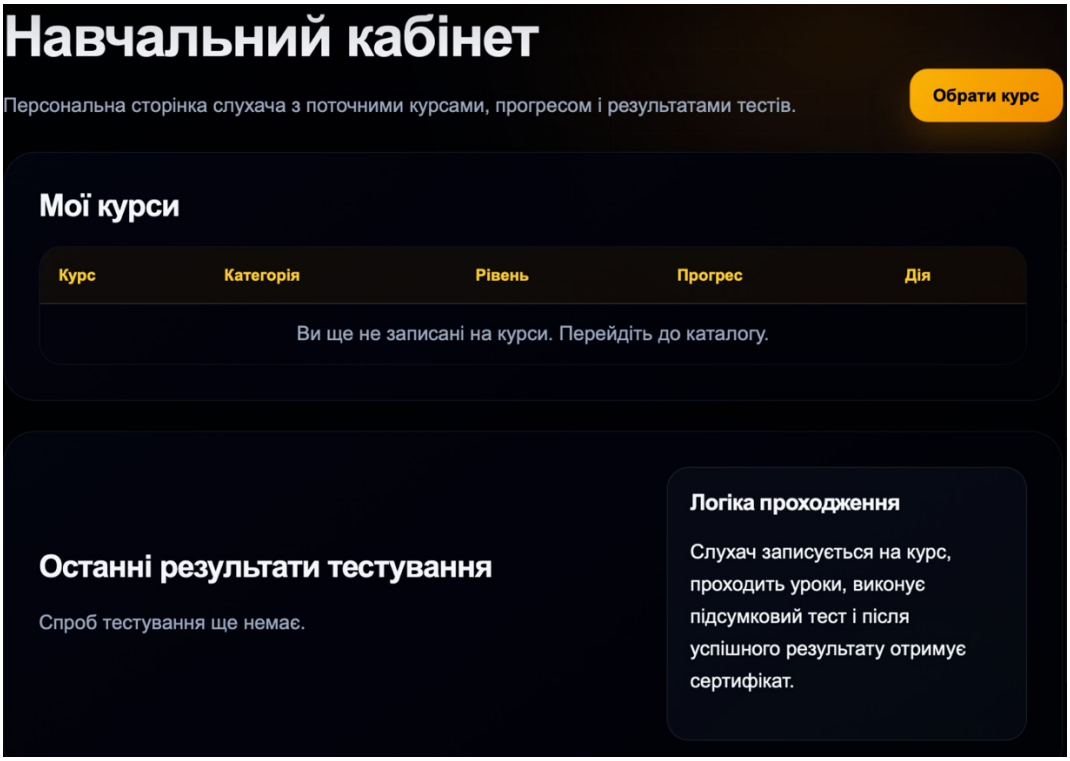

Рис. 4.5 – Навчальний кабінет користувача до запису на курс

З рисунка 4.5 видно, що система коректно визначає відсутність активних курсів у користувача. Це важливо для перевірки зв’язку між таблицями користувачів, курсів і записів на навчання. Якщо у базі даних немає запису про вибраний курс, у кабінеті не відображаються зайві або помилкові дані. Кнопка «Обрати курс» забезпечує швидкий перехід до каталогу, що спрощує подальшу роботу слухача із системою.

Наступним етапом стало тестування сторінки окремого курсу. Для перевірки було обрано курс «Фінансова грамотність». На сторінці курсу відображається поточний прогрес, перелік навчальних матеріалів і блок

підсумкового контролю. Початковий стан сторінки курсу наведено на рисунку 4.6.

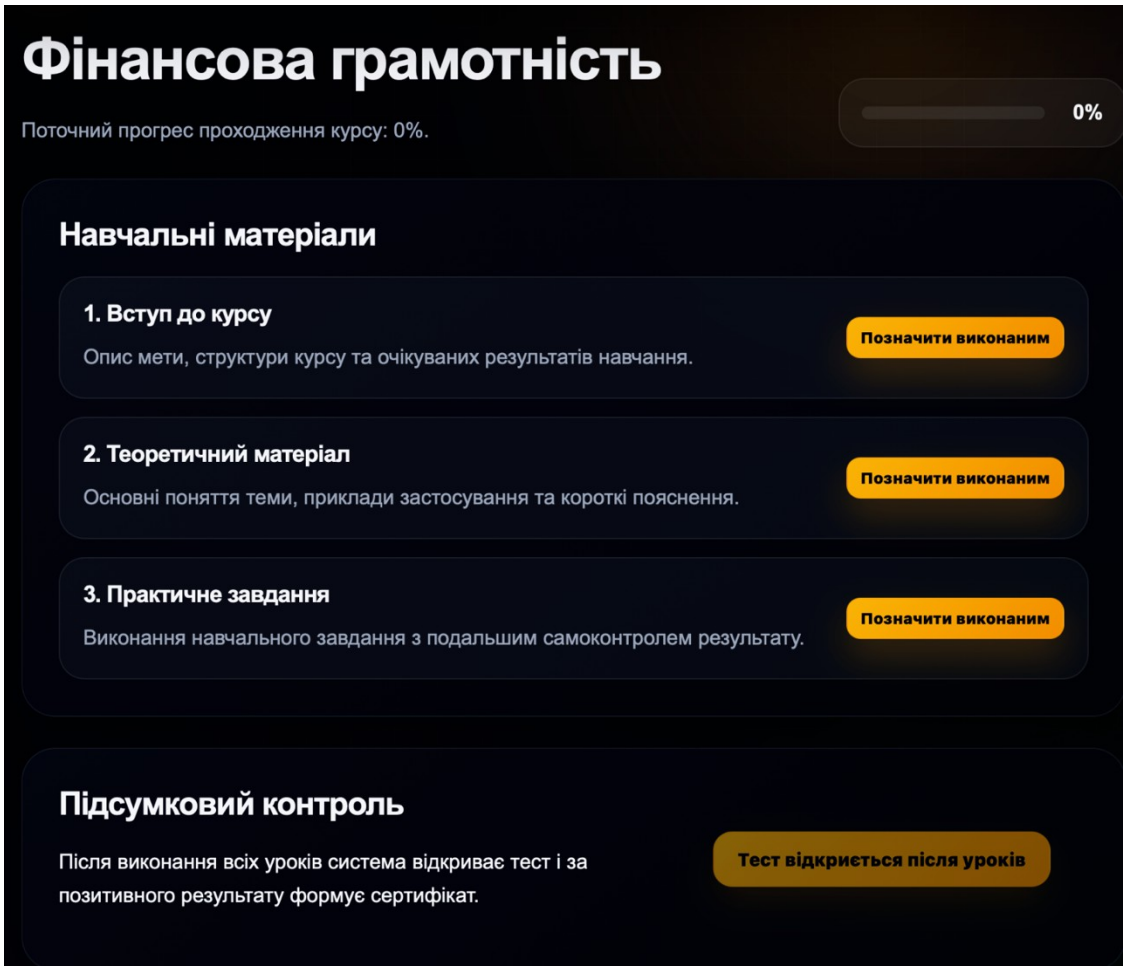

Рис. 4.6 – Сторінка курсу з навчальними матеріалами

На рисунку 4.6 показано, що курс складається з трьох навчальних етапів: вступу до курсу, теоретичного матеріалу та практичного завдання. Для кожного матеріалу доступна кнопка «Позначити виконаним». Підсумковий тест на цьому етапі заблокований, оскільки користувач ще не завершив усі навчальні матеріали. Така логіка є правильною, оскільки контроль знань має відкриватися лише після проходження основної частини курсу.

Після натискання кнопки «Позначити виконаним» для першого навчального матеріалу система оновлює стан проходження курсу. Прогрес змінюється з 0% до 33%, а виконаний етап отримує відповідну позначку. Оновлений стан сторінки курсу наведено на рисунку 4.7.

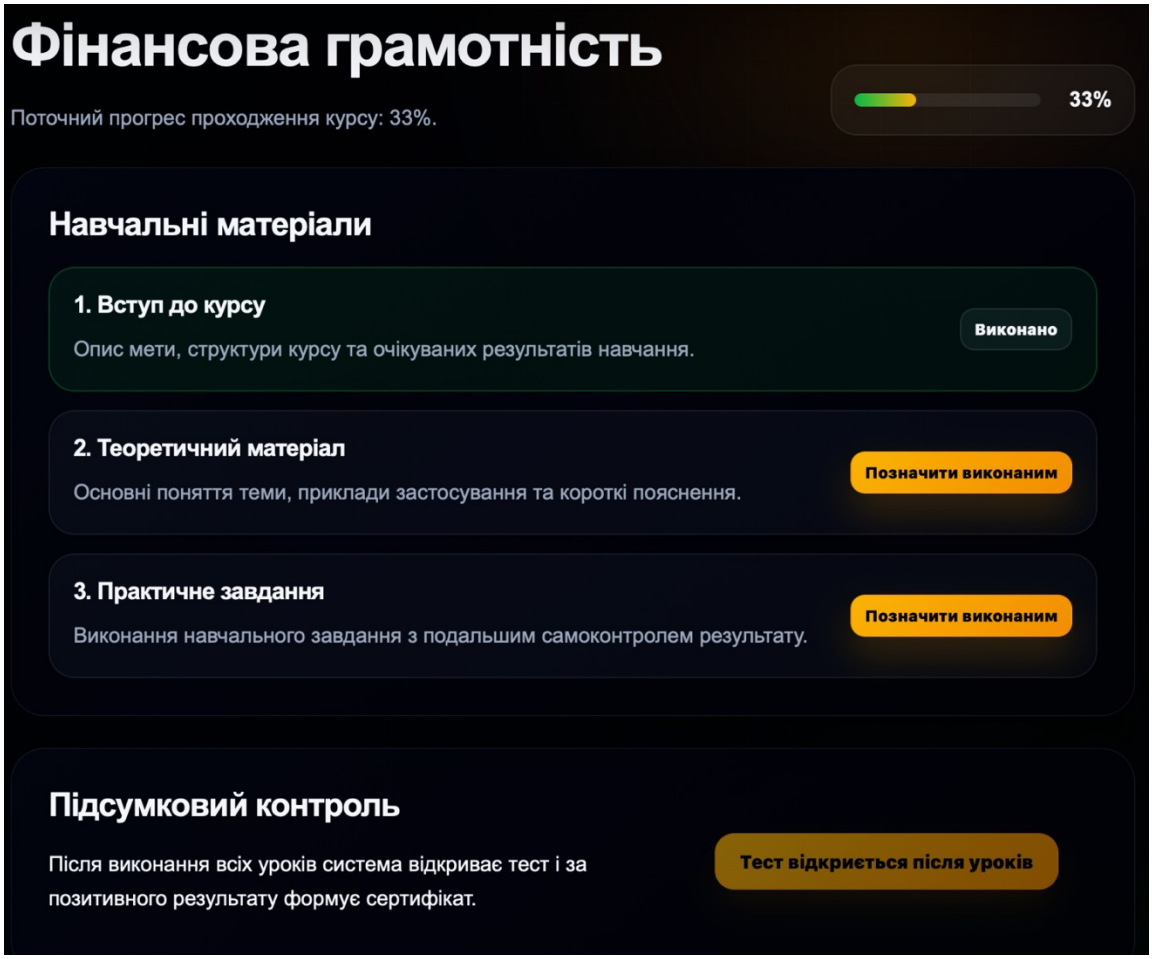

Рис. 4.7 – Оновлення прогресу після виконання першого навчального матеріалу

Як видно з рисунка 4.7, після виконання першого етапу прогрес-бар оновлюється автоматично. Це підтверджує коректну роботу модуля обліку прогресу та зв’язку між інтерфейсом і базою даних. Виконаний матеріал більше не відображається як активне завдання, а позначається статусом «Виконано». При цьому підсумковий тест залишається недоступним, оскільки користувач ще не завершив усі матеріали курсу.

Після запису на курс та виконання першого навчального матеріалу перевірено оновлення інформації в особистому кабінеті. У кабінеті користувача з’являється активний курс, його категорія, рівень, поточний прогрес і кнопка для продовження навчання. Результат оновлення кабінету наведено на рисунку 4.8.

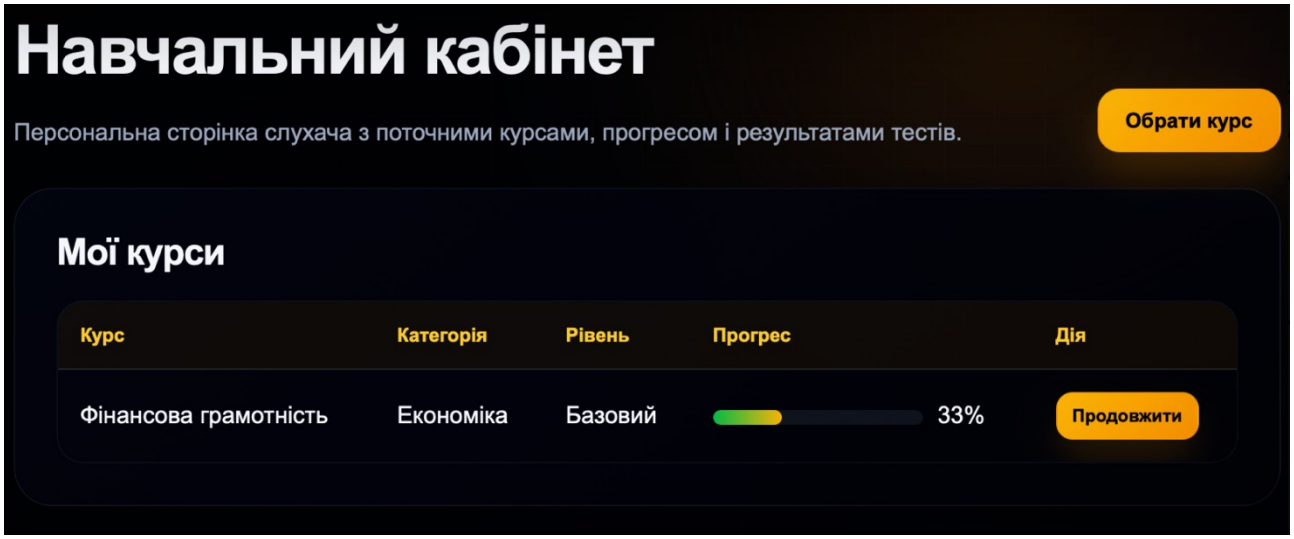

Рис. 4.8 – Відображення активного курсу в навчальному кабінеті

З рисунка 4.8 видно, що система правильно відображає курс «Фінансова грамотність» у списку поточних курсів слухача. Значення прогресу відповідає стану проходження, який був сформований на сторінці курсу. Кнопка «Продовжити» дає змогу повернутися до навчальних матеріалів. Це підтверджує, що зміни, виконані на сторінці курсу, зберігаються в базі даних і правильно відображаються в інших модулях системи.

Для узагальнення результатів тестування основних функцій сформовано таблицю 4.2.

Таблиця 4.2

Результати тестування веборієнтованої системи підтримки позауніверситетського навчання

№	Тестовий сценарій	Дії користувача	Очікуваний результат	Фактичний результат	Статус
1	Завантаження головної сторінки	Відкрити адресу системи у браузері	Відображення головної сторінки, меню, статистики та кнопок переходу	Головна сторінка відкрилася коректно	Успішно
2	Перехід до реєстрації	Натиснути кнопку «Реєстрація»	Відкриття форми створення облікового запису	Форма реєстрації відкрилася	Успішно
3	Реєстрація слухача	Заповнити ПІБ, email і пароль	Створення нового користувача в базі даних	Обліковий запис створено	Успішно
4	Авторизація користувача	Ввести email і пароль	Перехід до навчального кабінету	Вхід виконано коректно	Успішно

Продовження таблиці 4.2

5	Перегляд каталогу курсів	Відкрити сторінку каталогу	Відображення переліку доступних курсів	Курси відображаються картками	Успішно
6	Пошук і фільтрація курсів	Ввести запит або вибрати категорію	Відображення курсів за умовами пошуку	Фільтрація працює коректно	Успішно
7	Перегляд кабінету без курсів	Увійти в кабінет до запису на курс	Повідомлення про відсутність активних курсів	Повідомлення відображено	Успішно
8	Запис на курс	Обрати курс у каталозі	Курс додається до кабінету користувача	Курс з’явився в кабінеті	Успішно
9	Виконання навчального матеріалу	Натиснути «Позначити виконаним»	Зміна статусу матеріалу та оновлення прогресу	Прогрес змінено до 33%	Успішно
10	Перевірка блокування тесту	Відкрити курс без завершення всіх матеріалів	Підсумковий тест недоступний	Тест залишається заблокованим	Успішно
11	Оновлення кабінету після прогресу	Повернутися до навчального кабінету	Відображення активного курсу з поточним прогресом	Курс і прогрес відображаються правильно	Успішно

Результати, наведені в таблиці 4.2, підтверджують, що основні функціональні модулі системи працюють відповідно до очікуваної логіки. Головна сторінка відкривається без помилок, навігаційні елементи виконують перехід на потрібні сторінки, реєстрація та авторизація забезпечують доступ користувача до персонального кабінету, каталог курсів підтримує перегляд і фільтрацію навчальних програм, а модуль проходження курсу коректно оновлює прогрес.

Окремо перевірено збереження даних у базі SQLite. Після реєстрації нового користувача створюється відповідний запис у таблиці користувачів. Після вибору курсу формується запис про навчання слухача. Після виконання матеріалу оновлюється прогрес проходження курсу. Це підтверджує коректну роботу інформаційної бази та зв’язків між основними сутностями системи.

Також під час тестування оцінено зручність інтерфейсу. Темний візуальний стиль, контрастні кнопки, карткова структура курсів і наявність прогрес-барів роблять роботу з системою зрозумілою для користувача. Основні дії не потребують складного налаштування: слухач може зареєструватися, увійти, обрати курс і розпочати навчання за кілька послідовних кроків.

За результатами тестування критичних помилок у роботі системи не виявлено. Основні сценарії виконуються стабільно, дані між сторінками передаються коректно, а інтерфейс відображає актуальний стан навчання користувача. Розроблена веборієнтована система може використовуватися як програмний прототип для підтримки позауніверситетського навчання з подальшим розширенням функцій тестування, сертифікації, аналітики та адміністративного керування курсами.

4.3 Результати тестування та аналіз ефективності системи

За результатами тестування встановлено, що розроблена система виконує основні функції, визначені на етапі аналізу вимог. Вона забезпечує реєстрацію та авторизацію користувачів, перегляд каталогу курсів, запис слухача, відображення кабінету, проходження уроків, тестування, формування сертифіката та базове адміністрування.

Ефективність системи полягає у скороченні кількості ручних дій під час супроводу позауніверситетського навчання. Без автоматизації адміністратор повинен окремо вести список слухачів, перелік курсів, прогрес виконання, результати тестів і сертифікати. У розробленій системі ці дані пов’язані в єдиній базі.

Швидкодія системи у локальному середовищі є достатньою для навчального прототипу. Оскільки застосунок використовує SQLite і серверну генерацію HTML, сторінки відкриваються без помітних затримок, а основні запити виконуються за частки секунди для демонстраційного набору даних.

Зручність інтерфейсу забезпечується темною dashboard-структурою, видимими кнопками дій, логічним поділом сторінок і зрозумілими назвами розділів. Слухач може швидко знайти курс, перейти до кабінету та продовжити навчання без додаткових інструкцій.

Таблиця 4.3

Оцінювання ефективності розробленої системи

Критерій	До автоматизації	Після впровадження системи
Облік слухачів	Ведеться вручну в таблицях або документах	Зберігається у таблиці users
Запис на курс	Потребує окремого списку або заявки	Виконується через кнопку запису
Контроль прогресу	Потребує ручної перевірки матеріалів	Розраховується за lesson_progress
Тестування	Перевіряється вручну	Оцінка формується автоматично
Сертифікація	Номер і дані створюються вручну	Сертифікат формується після успішного тесту
Розгортання	Потребує налаштування серверів	Достатньо запуску main.py

Аналіз таблиці 4.3 показує, що система значно впорядковує роботу з навчальним процесом. Навіть у межах локального прототипу вона забезпечує інтегроване збереження даних, прозорість проходження курсів і автоматизацію результатів.

4.4 Розгортання системи та склад інсталяційного пакета

Розгортання системи виконується локально. Користувач розпаковує архів, відкриває папку проєкту в терміналі та запускає команду python3 main.py. Після

запуску в терміналі відображається адреса локального сервера, яку потрібно відкрити у браузері.

Інсталяційний пакет містить основний файл main.py, файл README.md з інструкцією запуску, стартові скрипти для Windows та macOS/Linux. База даних створюється автоматично під час першого старту, тому користувачеві не потрібно виконувати SQL-скрипти вручну.

Склад інсталяційного пакета наведено у таблиці 4.4. Така структура є достатньою для демонстрації, захисту роботи та подальшого розвитку прототипу.

Таблиця 4.4

Склад інсталяційного пакета

Файл / каталог	Призначення
main.py	Основний програмний файл із сервером, маршрутами, базою даних і HTML-шаблонами
README.md	Інструкція запуску, опис функціоналу та демонстраційних акаунтів
start_mac_linux.sh	Скрипт запуску для macOS та Linux
start_windows.bat	Скрипт запуску для Windows
database/extra_learning.db	Файл бази даних, що створюється автоматично після першого запуску
static/style.css	Файл стилів, якщо інтерфейс винесено в окремий каталог у подальшій версії

Під час тестування розгортання встановлено, що найчастішою проблемою може бути зайнятий порт 8000. Для усунення цієї ситуації реалізовано пошук вільного порту. Завдяки цьому користувач не повинен вручну завершувати попередній процес або змінювати код програми.

4.5 Висновки до четвертого розділу

У четвертому розділі проведено тестування веборієнтованої автоматизованої системи підтримки позауніверситетського навчання. Перевірено авторизацію, реєстрацію, каталог курсів, запис на курс, проходження уроків, тестування, сертифікацію, адміністрування та розгортання.

Результати тестування підтвердили, що система виконує основні функціональні вимоги та забезпечує повний цикл навчального процесу. Окремо перевірено роботу бази даних, унікальні обмеження, оновлення прогресу та автоматичне створення сертифіката після успішного тестування.

Розроблений прототип є працездатним, простим у запуску та придатним для демонстрації як навчальна вебсистема. Надалі систему можна розширити підтримкою відеолекцій, завантаженням файлів, розкладом занять, REST API, адаптивним тестуванням і розгортанням на сервері.

ВИСНОВКИ

У кваліфікаційній роботі виконано розробку веборієнтованої автоматизованої системи підтримки позауніверситетського навчання. Розроблена система забезпечує реєстрацію користувачів, авторизацію за ролями, перегляд каталогу курсів, запис слухача на курс, проходження навчальних матеріалів, контроль прогресу, підсумкове тестування та формування сертифіката.

У першому розділі проведено аналіз предметної області та існуючих рішень. Встановлено, що наявні платформи мають значну функціональність, але для невеликого автономного навчального середовища часто є надмірно складними або недостатньо адаптованими. Це обґрунтувало потребу у створенні компактної системи, орієнтованої на базові процеси позауніверситетського навчання.

У другому розділі спроєктовано інформаційне та програмне забезпечення системи. Побудовано ER-модель, діаграму класів, діаграми кооперацій, компонентів і пакетів. Сформовано фізичну модель бази даних SQLite з таблицями users, courses, lessons, enrollments, lesson_progress, tests, test_attempts і certificates.

У третьому розділі обґрунтовано вибір технологій і реалізовано програмний прототип. Для серверної частини використано Python 3, для збереження даних - SQLite, для інтерфейсу - HTML і CSS. Програма запускається локально командою python3 main.py і не потребує встановлення сторонніх бібліотек.

У четвертому розділі проведено тестування системи. Перевірено роботу авторизації, каталогу, запису на курс, уроків, прогресу, тестування, сертифікації, адміністративної панелі та механізму автоматичного вибору

вільного порту. Результати тестування підтвердили працездатність системи та відповідність основним вимогам.

Практична цінність роботи полягає у створенні готового до запуску навчального вебзастосунку, який може бути використаний як основа для малих освітніх центрів, факультативних програм, гуртків, тренінгів і демонстраційних курсів. Подальший розвиток системи може передбачати інтеграцію відеоматеріалів, платіжних модулів, повідомлень, розширеної аналітики та хмарного розгортання.

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. Моделі якості системи та програмних засобів. Київ : ДП «УкрНДНЦ», 2016.

3. ISO/IEC/IEEE 12207:2017. Systems and software engineering. Software life cycle processes. Geneva : ISO, 2017. URL: https://www.iso.org/standard/63712.html.

4. ISO/IEC 25010:2011. Systems and software engineering. Systems and software Quality Requirements and Evaluation. System and software quality models. Geneva : ISO, 2011.

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

6. Sommerville I. Software Engineering. 10th ed. Boston : Pearson, 2016. 816 p.

7. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner’s Approach. 9th ed. New York : McGraw-Hill Education, 2020. 704 p.

8. Fowler M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. 3rd ed. Boston : Addison-Wesley, 2003. 208 p.

9. Silberschatz A., Korth H. F., Sudarshan S. Database System Concepts. 7th ed. New York : McGraw-Hill Education, 2019. 1376 p.

10. Elmasri R., Navathe S. B. Fundamentals of Database Systems. 7th ed. Boston : Pearson, 2016. 1272 p.

11. Python Software Foundation. Python 3 Documentation. URL: https://docs.python.org/3/.

12. Python Software Foundation. http.server - HTTP servers. Python Documentation. URL: https://docs.python.org/3/library/http.server.html.

13. Python Software Foundation. sqlite3 - DB-API 2.0 interface for SQLite databases. Python Documentation. URL: https://docs.python.org/3/library/sqlite3.html.

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

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

16. MDN Web Docs. CSS: Cascading Style Sheets. URL: https://developer.mozilla.org/en-US/docs/Web/CSS.

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

18. W3C. HTML Standard. URL: https://html.spec.whatwg.org/.

19. OWASP Foundation. OWASP Web Security Testing Guide. URL: https://owasp.org/www-project-web-security-testing-guide/.

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

21. Moodle Documentation. Moodle Docs. URL: https://docs.moodle.org/.

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

23. Coursera. Online Courses and Credentials from Top Educators. URL: https://www.coursera.org/.

24. Udemy. Online Courses. URL: https://www.udemy.com/.

25. Prometheus. Українська платформа масових відкритих онлайн-курсів. URL: https://prometheus.org.ua/.

26. Welling L., Thomson L. PHP and MySQL Web Development. 5th ed. Boston : Addison-Wesley, 2017. 688 p.

27. Duckett J. HTML and CSS: Design and Build Websites. Indianapolis : Wiley, 2011. 490 p.

28. Duckett J. JavaScript and JQuery: Interactive Front-End Web Development. Indianapolis : Wiley, 2014. 640 p.

29. Nielsen J. Usability Engineering. San Francisco : Morgan Kaufmann, 1993. 362 p.
30. Krug S. Don’t Make Me Think, Revisited: A Common Sense Approach to Web Usability. 3rd ed. Berkeley : New Riders, 2014. 216 p.

ДОДАТОК И

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

Декларація про академічну доброчесність та використання ШІ

ДЕКЛАРАЦІЯ ЗДОБУВАЧА

Я, Опанасенко Едуард Сергійович, здобувач(ка) НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота (проєкт) на тему «Веб орієнтована автоматизована система підтримки поза університетського навчання» 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.

«___» _____ 2026 р. _____ **Едуард ОПАНАСЕНКО**
(підпис)