Міністерство освіти і науки України
Криворізький національний університет
Факультет інформаційних технологій
Кафедра комп’ютерних систем та мереж

Пояснювальна записка
до кваліфікаційної роботи бакалавра
за спеціальністю 123 «Комп’ютерна інженерія»

на тему: Платформа онлайн-курсів з особистим кабінетом та прогресом навчання

Проектував _____ М. О. Лосєв

Керівник роботи _____ Д. І. Кузнєцов

Нормоконтроль _____ Д. І. Кузнєцов

Завідувач кафедри _____ А. І. Купін

Кривий Ріг
2026

Криворізький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних систем та мереж

Ступінь вищої освіти — бакалавр
Спеціальність — 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ
Завідувач кафедри, голова циклової комісії
_____ А. І. Купін
"____" _____ 20__ року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Лосєву Максиму Олексійовичу

1. Тема роботи: Платформа онлайн-курсів з особистим кабінетом та прогресом навчання
керівник роботи: Кузнєцов Денис Іванович
затверджені наказом вищого навчального закладу від "27" січня 2026 року №64с
2. Строк подання студентом роботи: "01" червня 2026 року
3. Вихідні дані до роботи: Матеріали щодо організації дистанційного навчання, принципи побудови вебзастосунків, вимоги до створення особистого кабінету користувача, методи збереження та обробки прогресу навчання. Для реалізації системи передбачено використання HTML5, CSS3, JavaScript, PHP та MySQL.
4. Зміст розрахунково-пояснювальної записки: Аналіз предметної області онлайн-навчання; огляд існуючих платформ онлайн-курсів; формування вимог до платформи; вибір засобів розробки; проєктування архітектури системи та бази даних; розробка особистого кабінету користувача; реалізація механізму відстеження прогресу навчання; тестування платформи.
5. Перелік графічного матеріалу: Структурна схема платформи онлайн-курсів; схема бази даних; алгоритм реєстрації та авторизації користувача; алгоритм відстеження прогресу навчання; скріншоти основних сторінок платформи.

6. Консультанти розділів роботи

Розділ	**Прізвище, ініціали та посада консультанта**	**Підпис, дата завдання видав**	**Підпис, дата завдання прийняв**

7. Дата видачі завдання: “01” лютого 2026 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	**Назва етапів роботи**	**Строк виконання етапів роботи**	**Примітка**
1	Вибір теми та погодження її з керівником	1 тиждень	Виконано
2	Збір і вивчення матеріалів за темою роботи	1 тиждень	Виконано
3	Аналіз предметної області онлайн-навчання	2 тиждень	Виконано
4	Формування вимог до платформи	2 тиждень	Виконано
5	Вибір технологій розробки	3 тиждень	Виконано
6	Проєктування архітектури системи	3 тиждень	Виконано
7	Проєктування структури бази даних	4 тиждень	Виконано
8	Розробка інтерфейсу платформи	5 тиждень	Виконано
9	Реалізація особистого	6 тиждень	Виконано

	кабінету користувача		
10	Реалізація модуля курсів і прогресу навчання	7 тиждень	Виконано
11	Тестування роботи платформи	8 тиждень	Виконано
12	Оформлення пояснювальної записки	9 тиждень	Виконано
13	Підготовка графічних матеріалів і додатків	10 тиждень	Виконано
14	Перевірка роботи керівником та виправлення зауважень	11 тиждень	Виконано
15	Підготовка до захисту	12 тиждень	Виконано

Студент _____ Лосєв М. О.
Керівник роботи _____ Кузнєцов Д. І.

РЕФЕРАТ

Пояснювальна записка складається з 78 сторінок, містить 11 рисунків, 12 таблиць, 1 додаток і список із 20 використаних джерел.

Об’єктом розробки визначено вебплатформу онлайн-курсів, що об’єднує персональний кабінет користувача, доступ до навчальних матеріалів і засоби фіксації результатів проходження навчання.

Метою кваліфікаційної роботи є створення вебплатформи, за допомогою якої користувач може переглядати доступні курси, відкривати навчальні уроки, працювати у власному кабінеті та контролювати поточний стан опрацювання матеріалу. Розроблювана система не обмежується функціями звичайного каталогу сторінок, а забезпечує повний сценарій взаємодії користувача з платформою: від авторизації до перегляду індивідуального результату навчання.

У першому розділі досліджено особливості організації онлайн-навчання у вебсередовищі та розглянуто підходи до побудови функціоналу, що застосовуються в існуючих платформах онлайн-курсів. Другий розділ присвячено проєктуванню системи, зокрема визначенню її архітектури, структури бази даних, логіки роботи особистого кабінету та механізму обліку навчального прогресу. У третьому розділі описано практичну реалізацію основних модулів платформи: реєстрації, авторизації, роботи з курсами, проходження уроків, збереження прогресу та перевірки працездатності розробленого вебзастосунку.

Ключові слова: ОНЛАЙН-КУРС, ВЕБПЛАТФОРМА, ОСОБИСТИЙ КАБІНЕТ, ПРОГРЕС НАВЧАННЯ, БАЗА ДАНИХ, PHP, MYSQL.

ABSTRACT

The bachelor’s qualification paper consists of 78 pages and includes 11 figures, 12 tables, 1 appendix, and 20 references.

The object of development is a web-based platform for online courses that integrates a personal user account, access to educational materials, and tools for recording and displaying learning progress.

The aim of the work is to design and implement a web platform that enables users to browse available courses, open and study lessons, use a personal account, and monitor the current state of course completion. The developed system is not limited to a simple collection of static educational pages; it provides a complete user interaction scenario, from authorization in the platform to viewing individual learning results.

The first section considers the organization of online learning in a web environment and examines functional approaches used in existing online course platforms. The second section is devoted to system design, including the platform architecture, database structure, personal account logic, and the mechanism for tracking learning progress. The third section describes the practical implementation of the main platform modules, including registration, authorization, course management, lesson completion, progress recording, and functional testing of the developed web application.

Keywords: ONLINE COURSE, WEB PLATFORM, USER ACCOUNT, LEARNING PROGRESS, DATABASE, PHP, MYSQL.

ЗМІСТ

					КНУ.РБ.123.26.01.3			
Змн.	Арк.	№ документа	Підпис	Дата				
Розробив		Лосєв			ЗМІСТ	Літера	Аркуш	Аркушів
Перевірив		Кузнєцов						
						КІ-22-2		
Н.контроль		Кузнєцов						
Затвердив		Купін						

ПЕРЕЛІК СКОРОЧЕНЬ

БД — база даних, тобто впорядкована сукупність відомостей, призначена для збереження інформації про користувачів, навчальні курси, уроки та результати їх проходження.

ПЗ — програмне забезпечення, за допомогою якого реалізуються основні функції вебплатформи та забезпечується її робота.

ІС — інформаційна система, що використовується для збирання, оброблення, збереження й подання даних у зручному для користувача вигляді.

HTML — мова гіпертекстової розмітки, яка застосовується для побудови структури вебсторінок і розміщення основних елементів інтерфейсу.

CSS — мова опису стилів, що визначає візуальне оформлення сторінок, зокрема вигляд блоків, кнопок, шрифтів та інших елементів вебінтерфейсу.

JS — мова програмування JavaScript, призначена для створення інтерактивної поведінки вебсторінок і оброблення дій користувача на клієнтській стороні.

PHP — серверна мова програмування, яка використовується для оброблення запитів, роботи з формами, виконання авторизації та взаємодії з базою даних.

SQL — мова структурованих запитів, за допомогою якої здійснюється створення, вибірка, оновлення та видалення даних у реляційній базі.

UI — інтерфейс користувача, через який людина взаємодіє з вебплатформою та виконує основні дії в системі.

ВСТУП

Активне впровадження цифрових технологій суттєво вплинуло на організацію навчального процесу. Якщо раніше здобуття знань переважно залежало від аудиторних занять, сталого розкладу та особистої присутності учасників освітнього процесу, то нині значна частина навчання може здійснюватися у вебсередовищі. Для користувача такий формат є зручним, оскільки навчальні матеріали залишаються доступними незалежно від часу, а швидкість проходження курсу можна адаптувати до власного рівня підготовки, зайнятості та індивідуального темпу роботи.

Платформа онлайн-курсів не повинна розглядатися лише як місце для розміщення текстових матеріалів або відеоуроків. Її призначення полягає не тільки у наданні доступу до навчального контенту, а й в організації послідовного навчального маршруту користувача. Система має показувати, які уроки вже завершено, який обсяг матеріалу ще залишився для опрацювання та який відсоток курсу вже виконано. Через це особистий кабінет і механізм відстеження прогресу є необхідними складовими повноцінної навчальної вебплатформи.

Актуальність теми зумовлена потребою у створенні зрозумілої та зручної вебплатформи, яка поєднує каталог навчальних курсів, персональний кабінет користувача та засоби контролю проходження матеріалу. На рівні окремих функцій така система може здаватися нескладною, однак саме ці елементи визначають якість взаємодії користувача з навчальним сервісом. Якщо людина бачить власний прогрес і може швидко повернутися до потрібного уроку, навчання стає більш упорядкованим і передбачуваним.

Метою кваліфікаційної роботи є розроблення платформи онлайн-навчання, що дає користувачу можливість переглядати доступні курси, проходити уроки через особистий кабінет і контролювати власний навчальний

прогрес. Об’єктом роботи є процес організації онлайн-навчання із застосуванням вебтехнологій.

Предметом роботи є методи, програмні засоби та структурні рішення, які використовуються під час створення вебплатформи онлайн-курсів, особистого кабінету користувача та механізму відстеження прогресу навчання.

Для досягнення поставленої мети необхідно виконати кілька послідовних етапів: дослідити предметну область онлайн-навчання, розглянути наявні платформи онлайн-курсів, визначити вимоги до майбутньої системи, обрати засоби розробки, спроєктувати архітектуру платформи та структуру бази даних, реалізувати основні модулі вебзастосунку й провести тестування його роботи.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Особливості організації онлайн-навчання у вебсередовищі

Онлайн-навчання вже не сприймається лише як допоміжний варіант здобуття знань. За останні роки воно стало повноцінним практичним форматом, у межах якого навчальні матеріали можуть розміщуватися, відкриватися й опрацьовуватися через вебсервіси. Такий підхід знімає жорстку залежність від аудиторії, сталого розкладу та постійної присутності викладача. Користувач отримує можливість самостійно визначати, коли працювати з курсом, скільки часу приділяти окремим темам і в який момент повертатися до вже переглянутих уроків.

Платформу онлайн-курсів не можна зводити до звичайного набору сторінок із текстами або відеоматеріалами. Її основне призначення полягає в тому, щоб організувати для користувача зрозумілий навчальний маршрут. Система повинна показувати, з якого етапу розпочати роботу, де продовжити навчання після перерви та який результат уже зафіксовано. Без таких можливостей користувачеві доводилося б самостійно згадувати, який урок він відкривав останнім і яку частину курсу вже пройшов, що знижує зручність роботи з платформою.

У вебсередовищі навчальний процес формується через взаємодію кількох компонентів: реєстрації користувача, авторизації, особистого кабінету, каталогу курсів, сторінок уроків і механізму збереження прогресу. Кожен із цих елементів має окреме призначення, однак для користувача вони повинні працювати як єдиний послідовний сценарій. Людина входить до системи, відкриває особистий кабінет, обирає курс, переходить до потрібного уроку, завершує його й після цього бачить оновлений результат проходження. Якщо така послідовність відсутня, вебплатформа фактично перетворюється на

простий каталог навчальних матеріалів.

Особистий кабінет у цій структурі виконує роль персонального робочого простору користувача. Через нього відкривається доступ до курсів, відображається стан проходження, показуються завершені уроки та забезпечується швидке повернення до навчання. Тобто кабінет не є лише додатковою сторінкою з даними користувача. Він об’єднує інформацію, потрібну для продовження навчального процесу без зайвих переходів і ручного пошуку потрібного матеріалу.

Механізм відстеження прогресу є необхідною складовою платформи онлайн-курсів. Він дає змогу перетворити перелік уроків на контрольований навчальний процес, у якому користувач бачить не просто набір доступних матеріалів, а конкретний результат власної роботи. Таким результатом може бути кількість завершених уроків або відсоток виконання курсу. Особливо корисною ця функція є для курсів, що містять багато тем і не можуть бути повністю пройдені за один сеанс.

Під час розробки навчальної платформи небажано перевантажувати інтерфейс зайвими елементами. Якщо користувач довго шукає потрібний курс, не розуміє, як перейти до наступного уроку, або не бачить, де позначити матеріал як завершений, система втрачає практичну зручність. Тому під час створення вебзастосунку потрібно враховувати не тільки програмну реалізацію, а й логіку поведінки користувача на сайті. Інтерфейс має послідовно вести його від входу до отримання результату.

У межах цієї роботи онлайн-платформа розглядається як вебзастосунок, що складається з клієнтської частини, серверної логіки та бази даних. Клієнтська частина відповідає за відображення сторінок у браузері та взаємодію користувача з елементами інтерфейсу. Серверна частина обробляє запити, перевіряє авторизаційні дані, отримує інформацію про курси, фіксує завершення уроків і передає користувачу актуальні результати. База даних забезпечує збереження облікових записів, навчальних матеріалів і записів про проходження.

Платформа онлайн-курсів повинна поєднувати технічну працездатність із простою та зрозумілою користувацькою логікою. Для кваліфікаційної роботи така тема є обґрунтованою, оскільки охоплює розробку вебінтерфейсу, організацію серверної обробки, роботу з базою даних, авторизацію користувача, структурування навчальних матеріалів і створення механізму відстеження прогресу навчання.

1.2 Огляд існуючих платформ онлайн-курсів

Перед створенням власної вебплатформи необхідно проаналізувати вже наявні рішення, що використовуються у сфері онлайн-навчання. Такий аналіз дає змогу визначити, які можливості є обов'язковими для навчальної системи, а які належать до розширеного функціоналу і можуть бути реалізовані лише за потреби.

Одним із поширених прикладів платформи онлайн-курсів є Coursera. Цей сервіс надає доступ до навчальних програм від університетів, освітніх організацій і компаній. У платформі передбачено особистий кабінет користувача, поділ курсів на окремі модулі, виконання тестових завдань і отримання сертифікатів після завершення навчання. Для цієї кваліфікаційної роботи найбільший інтерес становить не зміст курсів Coursera, а сама логіка організації навчального процесу. Курс поділяється на послідовні частини, користувач поступово проходить матеріали, а система зберігає інформацію про виконані завдання та загальний стан проходження.

Іншим прикладом є Udemy. Ця платформа орієнтована на широкий каталог курсів, створених різними авторами. У такій моделі особливо помітною є роль сторінки курсу, каталогу навчальних матеріалів і персонального розділу користувача. Користувач може розпочати навчання, зробити перерву, а потім повернутися до того самого курсу без втрати попереднього результату. Для онлайн-навчання така можливість має практичне значення, оскільки користувачі часто проходять матеріали

нерівномірно: один урок може бути переглянутий сьогодні, а наступний — через кілька днів.

Окрему групу рішень представляє Moodle. Ця система широко використовується в закладах освіти для організації дистанційного навчання. Вона підтримує створення курсів, тематичних розділів, завдань, оцінювання, ролі користувачів, повідомлення та інші інструменти для викладачів і студентів. Водночас велика кількість функцій не завжди робить платформу зручнішою для користувача. Для невеликої навчальної вебплатформи доцільніше залишити базові можливості, які безпосередньо пов’язані з темою роботи: реєстрацію, авторизацію, особистий кабінет, перегляд курсів, проходження уроків і відображення прогресу.

Порівняння розглянутих платформ показує, що попри відмінності в масштабі, дизайні та наборі додаткових сервісів, їхня основна логіка є подібною. Користувач створює обліковий запис, отримує доступ до навчальних матеріалів, проходить їх у певній послідовності, а система фіксує результат. Саме ця послідовність є основою для розроблення власної платформи онлайн-курсів.

Порівняльну характеристику існуючих платформ і розроблюваної системи подано в таблиці 1.1.

Платформа	**Основне призначення**	**Особистий кабінет**	**Прогрес навчання**	**Особливості**
Coursera	Проходження курсів від університетів і компаній	Є	Є	Курси поділені на модулі, можуть бути сертифікати
Udemy	Навчання за курсами від різних авторів	Є	Є	Великий каталог курсів, доступ до куплених матеріалів
Moodle	Організація дистанційного навчання у закладах освіти	Є	Є	Багато інструментів для викладачів і студентів
Розроблювана платформа	Проходження онлайн-курсів у спрощеному форматі	Є	Є	Основний акцент зроблено на простоті, кабінеті та прогресі

Таблиця 1.1 – Порівняння існуючих платформ онлайн-курсів

Після аналізу наявних рішень можна визначити основні функції, які необхідно передбачити у розроблюваній платформі:

- реєстрація та авторизація користувача;
- перегляд особистого кабінету;
- доступ до переліку доступних курсів;
- відкриття уроків обраного курсу;
- позначення уроку як завершеного;
- автоматичне обчислення навчального прогресу;
- збереження результатів проходження в базі даних.

Такий набір функцій є достатнім для навчальної вебплатформи в межах кваліфікаційної роботи. Він не ускладнює систему зайвими модулями, але дає змогу реалізувати повний користувацький сценарій: від входу до платформи до збереження й перегляду результатів навчання.

1.3 Аналіз потреб користувачів та основних сценаріїв роботи системи

Під час проєктування платформи онлайн-курсів потрібно враховувати, що користувач очікує простого й зрозумілого способу роботи із системою. Йому не потрібні складні налаштування або зайві переходи: основне завдання полягає в тому, щоб швидко увійти на платформу, знайти потрібний курс і перейти до навчання. Тому система має розглядатися не лише з позиції розробника, а й з позиції звичайного користувача, який уперше відкриває вебсайт і повинен без додаткових пояснень зрозуміти порядок дій.

У межах цієї кваліфікаційної роботи основна увага приділяється сценарію зареєстрованого користувача. Такий користувач проходить авторизацію, переглядає доступні курси, відкриває навчальні уроки, позначає виконані матеріали та контролює власний прогрес. Саме цей сценарій є основою для побудови структури платформи, оскільки він охоплює всі головні функції системи.

Потреби користувача можна звести до кількох ключових дій. Він має бачити перелік доступних курсів, розуміти, які уроки вже завершено, і мати

можливість без зайвого пошуку продовжити навчання з потрібного місця. Наприклад, якщо користувач пройшов два уроки з п’яти, система повинна зберегти цю інформацію в базі даних і відобразити відповідний відсоток проходження. Завдяки цьому навчальний процес стає більш упорядкованим, а користувач бачить реальний результат своєї роботи.

Типовий сценарій взаємодії користувача з платформою можна подати в такій послідовності:

1. Користувач відкриває головну сторінку платформи.
2. Переходить до сторінки реєстрації або авторизації.
3. Після успішного входу потрапляє до особистого кабінету.
4. Переглядає перелік доступних курсів.
5. Обирає курс для проходження.
6. Відкриває урок і опрацьовує навчальний матеріал.
7. Позначає урок як завершений.
8. Система оновлює дані про прогрес навчання.
9. Користувач бачить, яку частину курсу вже виконано.

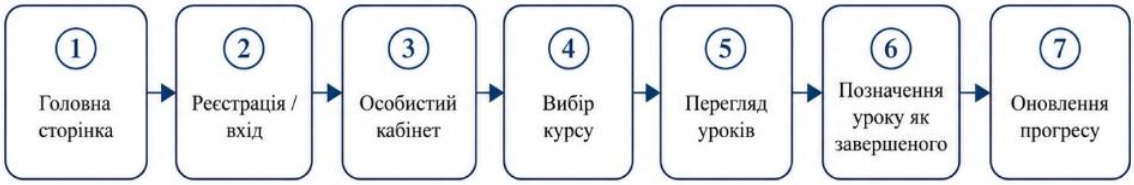

Рисунок 1.1 – Основний сценарій роботи користувача з платформою

На перший погляд, наведений сценарій є досить простим, однак саме він визначає основну логіку роботи платформи. Якщо один із кроків буде незрозумілим або прихованим для користувача, це може ускладнити проходження курсу й знизити зручність роботи із системою. Тому елементи інтерфейсу, що відповідають за вхід, вибір курсу, відкриття уроку та перегляд прогресу, повинні бути помітними й логічно розміщеними.

Навчальні матеріали в межах цієї роботи попередньо підготовлені та зберігаються в базі даних. Це дає змогу зосередити увагу не на створенні великого обсягу контенту, а на реалізації користувацького сценарію: виборі

курсу, перегляді уроків, фіксації завершених матеріалів і відображенні прогресу навчання.

Особистий кабінет у розроблюваній системі виступає центральною сторінкою користувача. У ньому доцільно відображати ім’я користувача, доступні курси, поточний відсоток проходження та посилання для продовження навчання. Такий підхід дозволяє зібрати основну інформацію в одному місці й скоротити кількість зайвих переходів між сторінками.

Проведений аналіз потреб користувача показує, що платформа повинна мати просту структуру, зрозумілий інтерфейс і надійний механізм збереження даних. Основна увага в подальшій розробці приділяється трьом складовим: особистому кабінету, роботі з курсами та механізму відстеження прогресу навчання.

1.4 Функціональні та нефункціональні вимоги до платформи

Після визначення потреб користувача необхідно сформувати вимоги до майбутньої платформи. На цьому етапі встановлюється, які функції повинна виконувати система, які дані вона має обробляти та як повинна реагувати на дії користувача. Чітко визначені вимоги дають змогу уникнути ситуації, коли вебзастосунок складається з окремих сторінок, але не має продуманої логіки роботи.

Функціональні вимоги описують конкретні можливості системи. Для платформи онлайн-курсів вони пов’язані насамперед із роботою користувача, курсами, уроками та механізмом збереження навчального прогресу. Користувач повинен мати можливість створити обліковий запис, увійти до системи, перейти в особистий кабінет, переглянути доступні курси, відкрити потрібний урок і після його завершення побачити оновлений стан проходження.

До основних функціональних вимог розроблюваної платформи належать:

- реєстрація нового користувача;

- авторизація зареєстрованого користувача;
- вихід з облікового запису;
- відкриття особистого кабінету;
- відображення списку доступних курсів;
- перегляд сторінки окремого курсу;
- доступ до навчальних уроків;
- позначення уроку як завершеного;
- збереження інформації про пройдені уроки;
- обчислення прогресу навчання у відсотках;
- відображення прогресу в особистому кабінеті.

Реєстрація та авторизація мають не лише допоміжне значення. Саме ці механізми дозволяють пов’язати результати навчання з конкретним користувачем. Якщо система не визначає, хто саме увійшов до платформи, вона не зможе коректно зберегти інформацію про завершені уроки, відкриті курси та прогрес, який потрібно показати під час наступного входу.

Особистий кабінет виступає центральною сторінкою користувача в межах платформи. У ньому доцільно розміщувати ім’я користувача, перелік доступних або розпочатих курсів, коротку інформацію про навчання та показники проходження. При цьому кабінет не повинен бути перевантажений зайвими блоками. Після входу користувач має одразу бачити головне: свої курси, стан їх проходження та можливість продовжити навчання.

Механізм прогресу навчання повинен працювати за зрозумілою логікою. Після завершення уроку система фіксує відповідний запис у базі даних, а потім порівнює кількість завершених уроків із загальною кількістю уроків у курсі. На основі цього розраховується відсоток проходження. Наприклад, якщо курс містить 10 уроків, а користувач завершив 4 з них, прогрес становить 40 %. Такий підхід є простим для реалізації та зрозумілим для користувача.

Крім функціональних можливостей, потрібно врахувати нефункціональні вимоги. Вони характеризують якість роботи платформи: зручність інтерфейсу, стабільність, безпеку, простоту підтримки та

можливість подальшого розширення. Для навчального вебзастосунку ці вимоги мають велике значення, оскільки навіть правильно реалізовані функції можуть бути незручними, якщо користувачу складно знайти потрібні дії.

До основних нефункціональних вимог можна віднести:

- простий і зрозумілий інтерфейс;
- коректне відображення сторінок у сучасному браузері;
- збереження даних користувачів у базі даних;
- захист особистого кабінету від доступу без авторизації;
- логічну структуру сторінок;
- стабільну роботу основних модулів;
- можливість подальшого розширення платформи;
- простоту локального запуску та тестування.

Узагальнення основних вимог до платформи подано в таблиці 1.2.

Група вимог	**Вимога**	**Пояснення**
Функціональні	Реєстрація користувача	Користувач має створити власний обліковий запис
Функціональні	Авторизація	Система повинна перевіряти користувача перед входом
Функціональні	Особистий кабінет	Користувач повинен бачити свої курси та прогрес
Функціональні	Перегляд курсів	Платформа має показувати список доступних курсів
Функціональні	Проходження уроків	Користувач має відкривати уроки обраного курсу
Функціональні	Прогрес навчання	Система повинна рахувати відсоток завершення курсу
Нефункціональні	Зручність	Інтерфейс має бути простим і зрозумілим
Нефункціональні	Надійність	Дані користувача не повинні втрачатися
Нефункціональні	Безпека	Особистий кабінет має бути доступний тільки після входу

Таблиця 1.2 – Основні вимоги до платформи онлайн-курсів

Наведені вимоги визначають межі подальшого проєктування та допомагають зберегти основну спрямованість роботи. Мета полягає не у створенні надмірно складної системи з великою кількістю додаткових модулів, а в розробленні завершеної та зрозумілої платформи, де користувач може увійти до особистого кабінету, відкрити курс, пройти уроки й побачити власний прогрес.

Разом із цим платформа повинна мати можливість подальшого розвитку. У майбутньому її можна доповнити тестами, сертифікатами, рейтингом користувачів, коментарями до уроків або системою оплати курсів. Проте в межах цієї кваліфікаційної роботи основну увагу зосереджено на базовій логіці онлайн-навчання, роботі особистого кабінету та збереженні результатів користувача.

1.5 Постановка задачі кваліфікаційної роботи

Проведений аналіз предметної області, наявних платформ онлайн-навчання та потреб користувачів дає змогу визначити основну задачу кваліфікаційної роботи. Вона полягає у розробленні вебплатформи онлайн-курсів, що містить особистий кабінет користувача та механізм відстеження прогресу навчання. Розроблювана система має бути зрозумілою для користувача, логічною за структурою та достатньою для демонстрації основних принципів роботи навчального вебзастосунку.

Метою роботи є створення вебплатформи, яка забезпечує можливість реєстрації користувача, входу до особистого кабінету, перегляду доступних онлайн-курсів, проходження уроків і контролю власного навчального прогресу. Для реалізації такого сценарію система повинна поєднувати клієнтську частину, серверну логіку та базу даних. Завдяки цьому платформа не обмежується відображенням окремих сторінок, а забезпечує повний цикл взаємодії користувача з навчальним середовищем.

Для досягнення поставленої мети необхідно виконати такі завдання:

- проаналізувати особливості організації онлайн-навчання та роботи платформ онлайн-курсів;
- розглянути приклади існуючих освітніх вебплатформ;
- визначити основні потреби користувачів;
- сформувати функціональні та нефункціональні вимоги до системи;
- обрати засоби розробки вебплатформи;
- спроєктувати загальну архітектуру системи;
- розробити структуру бази даних;
- передбачити механізм реєстрації та авторизації користувача;
- створити особистий кабінет користувача;
- реалізувати роботу з курсами та уроками;
- розробити механізм обчислення прогресу навчання;
- провести тестування основних функцій платформи.

Об’єктом роботи є процес організації онлайн-навчання за допомогою вебплатформи. Предметом роботи виступають методи, засоби та структурні рішення, що застосовуються під час розробки платформи онлайн-курсів з особистим кабінетом користувача і механізмом контролю прогресу навчання.

Практичне значення роботи полягає у створенні вебзастосунку, який демонструє базові можливості навчальної платформи. Розроблену систему можна використовувати як навчальний приклад або як основу для подальшого розширення. У майбутньому платформу можна доповнити модулями тестування, оцінювання, формування сертифікатів або детальнішою статистикою проходження курсів.

Для реалізації платформи обрано набір технологій HTML5, CSS3, JavaScript, PHP та MySQL. Такий підхід дає змогу розподілити функції системи між кількома складовими. HTML і CSS відповідають за структуру та зовнішнє оформлення сторінок, JavaScript використовується для взаємодії з елементами інтерфейсу, PHP забезпечує обробку запитів користувача, а

MySQL зберігає інформацію про користувачів, курси, уроки та навчальний прогрес.

У результаті виконання кваліфікаційної роботи має бути створена платформа, у якій користувач проходить повний шлях від реєстрації до перегляду власного результату навчання. Саме цей сценарій відображає взаємодію основних елементів вебсистеми: інтерфейс приймає дії користувача, серверна частина обробляє запити, база даних зберігає необхідну інформацію, а особистий кабінет подає результат у зручному для сприйняття вигляді.

Висновки до першого розділу

У першому розділі досліджено предметну область, пов’язану з організацією онлайн-навчання у вебсередовищі. У ході аналізу встановлено, що платформа онлайн-курсів має виконувати не лише функцію доступу до навчальних матеріалів. Вона також повинна допомагати користувачу орієнтуватися в навчальному процесі, бачити послідовність проходження курсу та контролювати власний результат. Через це до важливих складових такої системи належать особистий кабінет, каталог курсів, сторінки уроків і механізм відстеження прогресу.

Під час розгляду існуючих платформ онлайн-навчання визначено, що популярні рішення мають подібну базову логіку. Користувач створює обліковий запис, отримує доступ до курсів, проходить навчальні матеріали у певній послідовності, а система зберігає та відображає результат його роботи. Разом із цим для кваліфікаційної роботи недоцільно відтворювати складну структуру великих освітніх сервісів із великою кількістю додаткових модулів. Доцільніше зосередитися на тих функціях, які безпосередньо відповідають темі роботи та демонструють завершений навчальний сценарій.

У розділі також визначено основні потреби користувача та описано типовий порядок взаємодії з платформою. Користувач повинен мати можливість швидко пройти авторизацію, відкрити особистий кабінет, обрати

потрібний курс, перейти до уроків і побачити відсоток проходження. Основна увага приділяється простому й послідовному сценарію, у якому всі дії є зрозумілими та не потребують зайвих переходів.

На основі проведеного аналізу сформовано функціональні та нефункціональні вимоги до розроблюваної платформи. До основних функцій віднесено реєстрацію, авторизацію, перегляд курсів, відкриття уроків, збереження завершених занять і розрахунок навчального прогресу. Нефункціональні вимоги стосуються зручності інтерфейсу, стабільності роботи системи, захисту особистого кабінету від неавторизованого доступу та можливості подальшого розширення платформи.

Результати першого розділу створюють основу для наступного етапу роботи — проєктування вебплатформи. Надалі необхідно обрати технології розробки, визначити архітектуру системи, спроєктувати структуру бази даних і описати принцип роботи механізму відстеження прогресу навчання.

2. ПРОЄКТУВАННЯ ПЛАТФОРМИ ОНЛАЙН-КУРСІВ З ОСОБИСТИМ КАБІНЕТОМ ТА ПРОГРЕСОМ НАВЧАННЯ

2.1 Вибір засобів і технологій розробки

Перед початком практичної розробки платформи необхідно визначити технології, за допомогою яких буде створено вебзастосунок. Цей етап має не лише формальне значення, оскільки від обраних інструментів залежать структура проєкту, складність реалізації, зручність тестування та можливість подальшого розширення системи.

Для виконання цієї кваліфікаційної роботи обрано набір технологій HTML5, CSS3, JavaScript, PHP та MySQL. Такий вибір є доцільним для створення навчальної вебплатформи, у якій передбачено реєстрацію користувача, авторизацію, особистий кабінет, каталог курсів, сторінки уроків і збереження прогресу навчання. Крім того, ці технології не потребують складного налаштування середовища, що спрощує розробку, локальне тестування та демонстрацію готового проєкту.

HTML5 використовується для побудови структури сторінок платформи. У межах розроблюваної системи за допомогою HTML формуються заголовки, блоки з курсами, форми входу й реєстрації, кнопки керування, посилання, таблиці та сторінки з навчальними матеріалами. Тобто HTML визначає, які елементи будуть розміщені на сторінці та в якій послідовності вони відображатимуться.

CSS3 застосовується для оформлення зовнішнього вигляду вебплатформи. За допомогою стилів задаються кольори, відступи, розміри блоків, вигляд кнопок, шрифтів, карток курсів та інших елементів інтерфейсу. Для навчальної платформи дизайн має бути стриманим і зрозумілим, оскільки головне завдання інтерфейсу полягає не в декоративності, а в тому, щоб

користувач швидко знаходив потрібні дії та міг без ускладнень перейти до навчання.

JavaScript використовується для додавання інтерактивності на клієнтській стороні. З його допомогою можна перевіряти заповнення форм, реагувати на натискання кнопок, показувати повідомлення користувачу або змінювати окремі елементи інтерфейсу без повного перезавантаження сторінки. У межах цієї платформи JavaScript виконує допоміжну роль, оскільки основна логіка обробки даних і збереження результатів реалізується на серверній стороні.

PHP обрано як основну мову серверної розробки. Вона забезпечує обробку запитів користувача, приймання даних із форм, перевірку авторизації, звернення до бази даних і формування сторінок із потрібною інформацією. Наприклад, під час входу користувача до системи саме серверна частина перевіряє введені дані та визначає, чи можна надати доступ до особистого кабінету.

Для збереження інформації використовується система керування базами даних MySQL. У базі даних розміщуються відомості про користувачів, курси, уроки та навчальний прогрес. Завдяки цьому результати проходження не втрачаються після закриття браузера або завершення сеансу роботи. Якщо користувач повертається до платформи пізніше, система може відобразити актуальний стан проходження курсу.

Для локального запуску вебплатформи доцільно використовувати OpenServer або XAMPP. Такі середовища містять вебсервер, PHP та MySQL, тому дають змогу перевіряти роботу сайту на комп’ютері без його розміщення в мережі Інтернет. Це зручно для дипломного проєкту, оскільки дозволяє продемонструвати роботу платформи локально, показати структуру бази даних і пояснити взаємодію основних модулів системи.

Обраний набір технологій також зручний для пояснення архітектури проєкту. Платформа поділяється на кілька зрозумілих частин: HTML і CSS відповідають за структуру та зовнішній вигляд сторінок, JavaScript забезпечує

окремі елементи взаємодії на клієнтській стороні, PHP реалізує серверну логіку, а MySQL відповідає за збереження даних. Такий поділ робить роботу системи послідовною та зрозумілою.

Призначення основних технологій, використаних для розробки платформи, наведено в таблиці 2.1.

Технологія	**Призначення у проєкті**
HTML5	Створення структури вебсторінок
CSS3	Оформлення інтерфейсу користувача
JavaScript	Додавання інтерактивних елементів
PHP	Обробка запитів, авторизація, робота з даними
MySQL	Збереження користувачів, курсів, уроків і прогресу
OpenServer / XAMPP	Локальний запуск і тестування платформи
phpMyAdmin	Перегляд і керування базою даних

Таблиця 2.1 – Засоби розробки платформи онлайн-курсів

Обраний стек технологій є достатнім для реалізації поставленої задачі. Мета роботи полягає не у використанні складного фреймворку, а в розробленні зрозумілої вебплатформи, у якій користувач може зареєструватися, увійти до особистого кабінету, проходити курси та бачити власний прогрес навчання.

2.2 Загальна архітектура платформи

Після вибору технологій необхідно визначити загальну архітектуру майбутньої платформи. Під архітектурою системи розуміється її внутрішня будова, взаємозв’язок основних компонентів і порядок обміну даними між користувачем, серверною частиною та базою даних. Іншими словами, архітектура показує не зовнішній вигляд сайту, а принцип його роботи зсередини.

Розроблювана платформа онлайн-курсів має клієнт-серверну структуру. Користувач взаємодіє з вебінтерфейсом через браузер, а основна обробка даних виконується на сервері. База даних використовується для збереження інформації, необхідної для роботи системи: облікових записів користувачів, відомостей про курси, уроків і записів про проходження навчальних матеріалів.

Загальний принцип роботи платформи можна описати так: користувач відкриває сайт у браузері та виконує певну дію, наприклад реєструється, входить до системи, відкриває курс або позначає урок як завершений. Після цього браузер надсилає запит до серверної частини. PHP-скрипт обробляє отриманий запит, звертається до бази даних MySQL, отримує або змінює потрібну інформацію та повертає користувачу результат у вигляді сторінки або повідомлення.

Основними складовими платформи є:

- клієнтська частина;
- серверна частина;
- база даних;
- модуль авторизації;
- модуль курсів і уроків;
- модуль прогресу навчання.

Клієнтська частина відповідає за все, що користувач бачить у браузері. До неї належать головна сторінка, сторінки реєстрації та входу, особистий кабінет, каталог курсів, сторінка окремого курсу та сторінка уроку. Ці сторінки повинні бути побудовані послідовно, щоб користувач міг без ускладнень перейти від входу до системи до проходження навчального матеріалу.

Серверна частина забезпечує обробку даних і виконання основної логіки платформи. Вона перевіряє правильність введених даних під час авторизації, отримує список курсів, відкриває потрібні уроки, записує завершені заняття та розраховує прогрес навчання. Для користувача ці процеси відображаються як звичайні дії на сайті: натискання кнопки, перехід на сторінку або оновлення

показника прогресу. Насправді за кожною такою дією відбувається обмін даними між браузером, сервером і базою даних.

База даних є основним засобом збереження інформації в системі. Без неї платформа могла б лише відображати сторінки, але не могла б зберігати облікові записи користувачів і результати проходження курсів. Саме база даних дає змогу фіксувати навчальний прогрес окремо для кожного користувача та відновлювати ці дані під час наступного входу до системи.

Модуль авторизації використовується для захисту особистого кабінету та персональних даних користувача. Якщо користувач не виконав вхід до системи, він не повинен мати доступ до сторінок, пов’язаних із його курсами та прогресом. Після успішної авторизації створюється сеанс користувача, за допомогою якого система визначає, хто саме працює з платформою.

Модуль курсів і уроків відповідає за організацію навчальних матеріалів. Курс містить назву, опис і перелік уроків. Урок, своєю чергою, може містити заголовок, текстовий матеріал або посилання на відео. Така структура є достатньо простою, але вона дозволяє впорядкувати навчання й подати матеріали у послідовному вигляді.

Модуль прогресу навчання пов’язує користувача з уроками, які він уже завершив. Коли користувач натискає кнопку завершення уроку, система створює відповідний запис у базі даних. Після цього кількість завершених уроків порівнюється із загальною кількістю уроків у курсі, а результат відображається у відсотках. Саме цей модуль є одним із головних для теми кваліфікаційної роботи, оскільки він забезпечує контроль проходження навчального матеріалу.

Навчальні матеріали платформи зберігаються в базі даних. У межах цієї роботи вони додаються на етапі підготовки системи, тому основний акцент зроблено на користувацькому сценарії, роботі особистого кабінету та механізмі фіксації прогресу.

Загальну архітектуру платформи онлайн-курсів подано на рисунку 2.1.

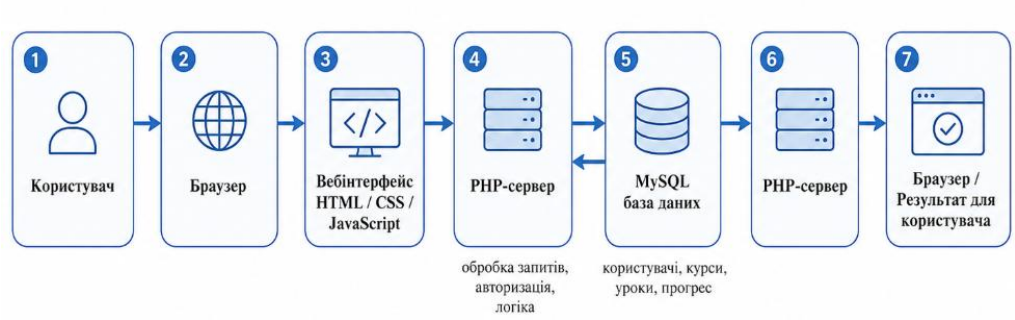

Рисунок 2.1 – Загальна архітектура платформи онлайн-курсів

Схема відображає основний шлях передавання даних у системі. Наприклад, коли користувач відкриває особистий кабінет, браузер надсилає запит на сервер. Серверна частина перевіряє, чи користувач авторизований, отримує з бази даних інформацію про його курси та прогрес, після чого формує сторінку кабінету з актуальними даними.

Під час проєктування важливо зберегти архітектуру достатньо простою та зрозумілою. Тому платформа будується навколо основного сценарію: користувач входить до системи, відкриває курс, проходить уроки та бачить власний прогрес. Така структура дозволяє реалізувати необхідні функції без надмірного ускладнення системи.

2.3 Проєктування структури бази даних

База даних є однією з основних складових платформи онлайн-курсів, оскільки саме вона забезпечує збереження інформації після завершення роботи користувача із системою. Без бази даних неможливо повноцінно реалізувати авторизацію, особистий кабінет і механізм відстеження прогресу навчання, оскільки ці функції потребують постійного зберігання даних.

Для розроблюваної платформи використовується реляційна база даних MySQL. Такий вибір є обґрунтованим, оскільки дані в системі мають чітку структуру та можуть бути подані у вигляді пов’язаних таблиць. Основними сутностями платформи є користувачі, курси, уроки, записи про проходження уроків і початок навчання на курсі.

Основними таблицями бази даних є:

- users — користувачі платформи;
- courses — онлайн-курси;
- lessons — уроки, що належать до відповідних курсів;
- progress — інформація про проходження уроків користувачами;
- course_starts — інформація про початок проходження курсів.

Таблиця users призначена для збереження відомостей про зареєстрованих користувачів. Вона містить ідентифікатор користувача, ім’я, електронну пошту, пароль у захищеному вигляді та дату створення облікового запису. Ці дані використовуються під час авторизації та дають змогу пов’язати навчальний прогрес із конкретним користувачем.

Таблиця courses містить інформацію про навчальні курси. Для кожного курсу зберігається його ідентифікатор, назва, короткий опис і, за потреби, зображення. Саме з цієї таблиці система отримує дані для відображення каталогу курсів і сторінки окремого курсу.

Таблиця lessons пов’язана з таблицею courses, оскільки один курс може містити кілька уроків. Кожен запис у таблиці уроків має посилання на відповідний курс. Завдяки цьому система може показувати користувачу не всі навчальні матеріали одразу, а лише ті уроки, які належать до обраного курсу.

Таблиця progress використовується для фіксації результатів навчання. У ній зберігається інформація про те, який користувач завершив певний урок. Також доцільно зберігати дату завершення уроку. На основі цих записів система визначає, скільки уроків уже пройдено в межах конкретного курсу.

Таблиця course_starts призначена для збереження інформації про початок проходження курсу користувачем. Вона дозволяє відокремити сам факт відкриття або початку курсу від завершення окремих уроків. Це корисно для особистого кабінету, де можна показувати не лише завершені матеріали, а й курси, які користувач уже розпочав.

Структуру бази даних платформи наведено в таблиці 2.2.

Назва таблиці	**Призначення**
users	Збереження даних користувачів
courses	Збереження інформації про курси
lessons	Збереження уроків, що належать до курсів
progress	Збереження інформації про завершені уроки

Таблиця 2.2 – Основні таблиці бази даних

Структуру таблиці users подано в таблиці 2.3.

Поле	**Тип даних**	**Призначення**
id	INT	Унікальний ідентифікатор користувача
name	VARCHAR	Ім’я користувача
email	VARCHAR	Електронна пошта
password	VARCHAR	Пароль користувача у захищеному вигляді
role	VARCHAR	Роль користувача в системі

Таблиця 2.3 – Структура таблиці users

Структуру таблиці courses наведено в таблиці 2.4.

Поле	**Тип даних**	**Призначення**
id	INT	Унікальний ідентифікатор курсу
title	VARCHAR	Назва курсу
description	TEXT	Опис курсу
image	VARCHAR	Назва або шлях до зображення курсу

Таблиця 2.4 – Структура таблиці courses

Структуру таблиці lessons наведено в таблиці 2.5.

Поле	Тип даних	Призначення
id	INT	Унікальний ідентифікатор уроку
course_id	INT	Ідентифікатор курсу
title	VARCHAR	Назва уроку
content	TEXT	Навчальний матеріал уроку
lesson_order	INT	Порядковий номер уроку в курсі

Таблиця 2.5 – Структура таблиці lessons

Структуру таблиці progress наведено в таблиці 2.6.

Поле	Тип даних	Призначення
id	INT	Унікальний ідентифікатор запису
user_id	INT	Ідентифікатор користувача
lesson_id	INT	Ідентифікатор завершеного уроку
completed_at	DATETIME	Дата та час завершення уроку

Таблиця 2.6 – Структура таблиці progress

Зв’язки між таблицями побудовано за логікою навчального процесу. Один курс може містити багато уроків, один користувач може проходити багато уроків, а один урок може бути завершений різними користувачами. Через це таблиця progress фактично виконує роль зв’язку між користувачами та уроками.

Обрана структура є простою, але достатньою для реалізації поставленої задачі. Коли користувач завершує урок, у таблицю progress додається запис з ідентифікатором користувача та ідентифікатором уроку. Для обчислення прогресу система визначає загальну кількість уроків у курсі та порівнює її з кількістю уроків цього курсу, які вже завершив конкретний користувач.

Наприклад, якщо курс складається з 8 уроків, а користувач завершив 2 з них, прогрес становить 25 %. Якщо завершено 4 уроки, показник проходження

дорівнює 50 %. Такий спосіб обчислення є зрозумілим для користувача та зручним для програмної реалізації.

Запропонована структура бази даних також залишає можливість для подальшого розширення платформи. У майбутньому до неї можна додати таблиці для тестів, відповідей користувачів, сертифікатів, коментарів або оплат. У межах цієї кваліфікаційної роботи наведені таблиці охоплюють основну логіку системи: збереження користувачів, курсів, уроків і результатів проходження навчання.

2.4 Проєктування особистого кабінету користувача

Особистий кабінет є однією з основних складових платформи онлайн-курсів. Саме через нього користувач отримує доступ до власного навчального простору, переглядає доступні курси, переходить до уроків і контролює стан проходження матеріалів. Якщо головна сторінка виконує ознайомчу функцію, то особистий кабінет відображає вже персоналізовану частину системи, пов’язану з конкретним користувачем.

Під час проєктування особистого кабінету необхідно уникати перевантаження інтерфейсу. Користувач не повинен витрачати час на пошук основних функцій серед великої кількості другорядних блоків. Після входу до системи він має одразу бачити своє ім’я, перелік курсів і поточний стан навчання. Такий підхід робить платформу зрозумілою навіть для користувачів, які раніше не працювали з подібними сервісами.

У розроблюваній платформі особистий кабінет виконує такі функції:

- відображення імені авторизованого користувача;
- перегляд доступних курсів;
- перехід до сторінки обраного курсу;
- відображення прогресу проходження курсів;
- можливість продовжити навчання;
- вихід з облікового запису.

Після успішної авторизації система повинна визначити, який саме користувач увійшов до платформи. Для цього використовується механізм сеансу. У сеансі зберігається ідентифікатор користувача, за допомогою якого серверна частина отримує з бази даних потрібну інформацію: ім’я користувача, доступні курси та дані про навчальний прогрес. Для користувача цей процес не відображається напряму, оскільки він бачить уже сформовану сторінку особистого кабінету.

Структуру особистого кабінету доцільно поділити на кілька логічних блоків. Перший блок містить короткі дані про користувача. Другий блок відображає список курсів. Третій блок відповідає за показники навчального прогресу. Якщо користувач має доступ до кількох курсів, біля кожного з них можна показувати відсоток проходження. Це дає змогу швидко визначити, який курс уже майже завершено, а який лише розпочато.

Приклад структури особистого кабінету наведено в таблиці 2.7.

Блок кабінету	**Призначення**
Дані користувача	Відображення імені та електронної пошти
Список курсів	Перегляд доступних курсів для навчання
Прогрес навчання	Показ відсотка проходження курсу
Кнопка продовження	Перехід до поточного або обраного уроку
Вихід із системи	Завершення сеансу користувача

Таблиця 2.7 – Структура особистого кабінету користувача

Окремої уваги потребує сторінка курсу, на яку користувач переходить з особистого кабінету. На ній мають відображатися назва курсу, короткий опис, перелік уроків і позначення матеріалів, які вже завершені. Така структура допомагає користувачу не втрачати орієнтацію під час навчання. Наприклад, якщо він уже пройшов перші два уроки, ця інформація повинна бути помітною без додаткових дій.

З погляду логіки роботи особистий кабінет пов’язаний одразу з кількома таблицями бази даних. Дані користувача отримуються з таблиці users, інформація про курси — з таблиці courses, уроки — з таблиці lessons, а

відомості про проходження — з таблиці progress. Завдяки такому зв’язку кабінет формується не як статична сторінка, а як персональний розділ, дані якого залежать від конкретного авторизованого користувача.

Особистий кабінет не потребує складного візуального оформлення. Для навчальної платформи важливішою є логічна структура сторінки: просте меню, зрозумілі картки курсів, кнопки переходу до уроків і наочний показник прогресу. Якщо користувач бачить назву курсу, можливість перейти до навчання та відсоток виконання, цього достатньо для комфортної роботи з базовою версією системи.

У подальшому особистий кабінет можна розширити додатковими можливостями. Зокрема, до нього можна додати історію активності, сертифікати, налаштування профілю, повідомлення або блок рекомендованих курсів. Проте в межах цієї кваліфікаційної роботи основна увага приділяється базовим функціям, необхідним для проходження курсів і перегляду навчального прогресу.

2.5 Розробка алгоритму відстеження прогресу навчання

Механізм відстеження прогресу є однією з основних функцій платформи онлайн-курсів. Він дає користувачу можливість бачити, яку частину курсу вже виконано, а які матеріали ще залишаються для проходження. Без такого механізму система фактично працювала б як звичайний набір навчальних сторінок, де користувачеві потрібно самостійно запам’ятовувати, на якому етапі він зупинився.

У розроблюваній платформі прогрес навчання визначається за кількістю завершених уроків. Кожен курс складається з певної кількості навчальних матеріалів. Коли користувач відкриває урок і натискає кнопку завершення, система фіксує цю дію в таблиці progress. Після цього можна порівняти кількість завершених уроків із загальною кількістю уроків у межах курсу та отримати відсоток проходження.

Алгоритм розрахунку прогресу має таку послідовність:

1. Користувач входить до системи.
2. Система визначає ідентифікатор авторизованого користувача.
3. Користувач відкриває обраний курс.
4. Система отримує загальну кількість уроків у цьому курсі.
5. Система перевіряє, скільки уроків цього курсу вже завершив користувач.
6. Кількість завершених уроків ділиться на загальну кількість уроків.
7. Отримане значення множиться на 100.
8. Результат відображається користувачу у вигляді відсотка прогресу.

Формулу розрахунку можна подати так:

Прогрес = (Кількість завершених уроків / Загальна кількість уроків) × 100 %

Наприклад, якщо курс містить 10 уроків, а користувач завершив 3 з них, прогрес становить 30 %. Якщо завершено 7 уроків із 10, показник проходження дорівнює 70 %. Такий спосіб обчислення є простим, але для базової навчальної платформи він повністю відповідає поставленій задачі.

Під час реалізації потрібно передбачити, щоб один і той самий урок не записувався як завершений кілька разів для одного користувача. Для цього перед додаванням нового запису система перевіряє, чи існує в таблиці progress запис із відповідними значеннями user_id та lesson_id. Якщо такий запис уже є, повторне додавання не виконується. Якщо запис відсутній, система створює новий результат проходження.

З технічного погляду алгоритм прогресу поєднує три сутності: користувача, урок і курс. Користувач завершує конкретний урок, урок належить до певного курсу, а прогрес розраховується для всього курсу. Саме цей зв’язок робить платформу персоналізованою, оскільки кожен користувач бачить не загальний стан курсу, а власний результат навчання.

Роботу механізму прогресу можна подати у вигляді такої послідовності:

Користувач відкриває урок → натискає «Завершити урок» → система перевіряє наявність запису в базі даних → додає результат → перераховує прогрес → відображає оновлений відсоток.

Цю послідовність доцільно подати у вигляді блок-схеми.

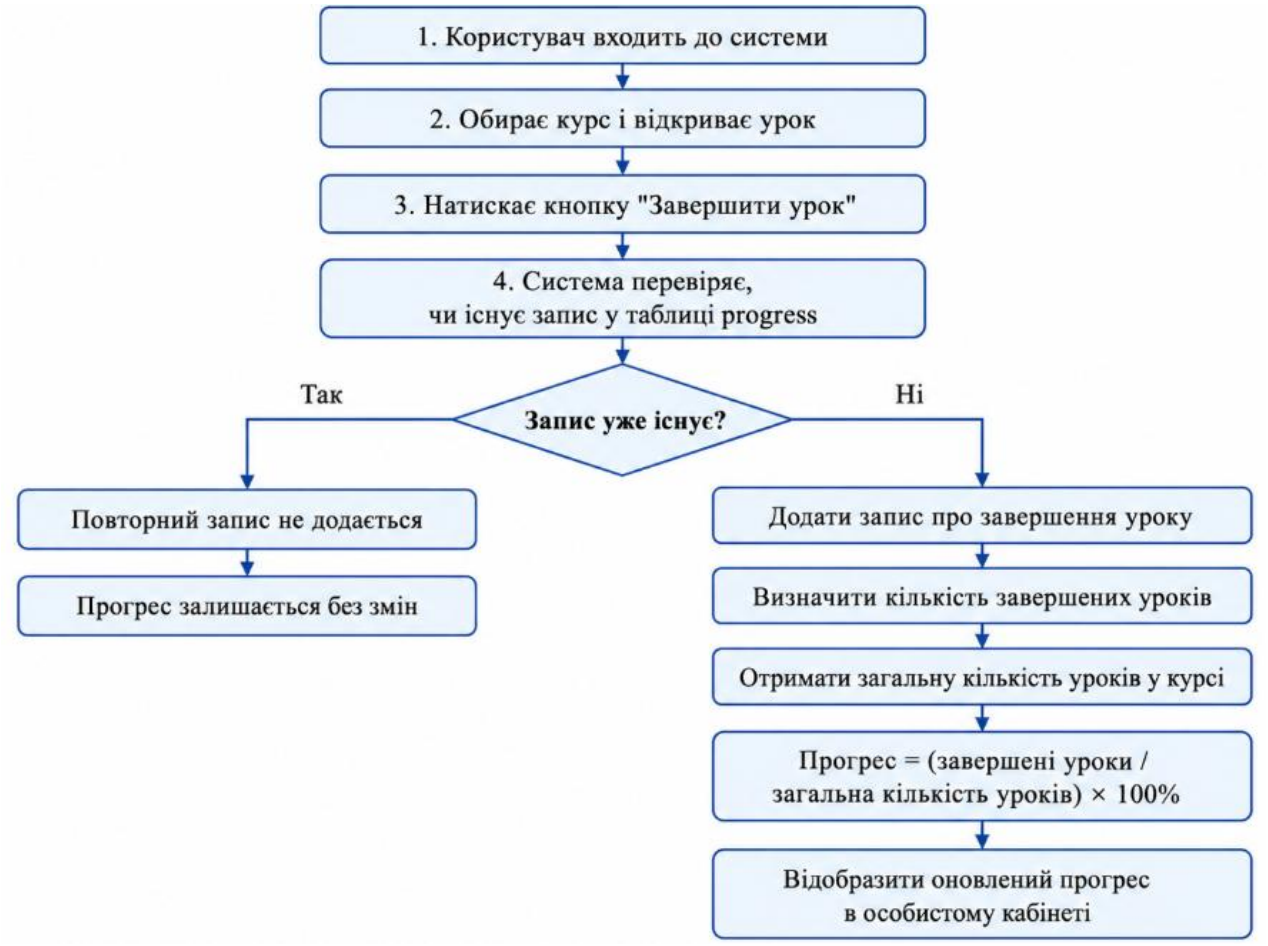

Рисунок 2.2 – Алгоритм відстеження прогресу навчання

На рисунку 2.2 показано порядок дій, за яким система оновлює прогрес користувача під час проходження уроків. Користувач виконує дію на сторінці уроку, після чого серверна частина перевіряє дані, звертається до бази даних і за потреби створює новий запис про завершення уроку. Після цього система повторно обчислює відсоток проходження курсу та відображає оновлений результат в особистому кабінеті або на сторінці курсу.

Такий підхід має кілька переваг. Він зрозумілий для користувача, не потребує складних обчислень і може бути реалізований засобами PHP та MySQL. Крім того, алгоритм легко пояснити під час демонстрації роботи

платформи, оскільки він безпосередньо пов’язаний із діями користувача на сайті.

У майбутньому цей механізм можна розширити. Наприклад, прогрес може враховувати не лише завершені уроки, а й результати тестів, оцінки, кількість спроб або час, витрачений на навчання. Проте в межах цієї кваліфікаційної роботи облік завершених уроків є оптимальним рішенням. Він відповідає темі проєкту, не ускладнює структуру системи та дозволяє показати основний принцип роботи платформи онлайн-курсів.

2.6 Проєктування інтерфейсу користувача

Інтерфейс користувача є частиною платформи, з якою людина взаємодіє безпосередньо. Через сторінки вебзастосунку користувач проходить реєстрацію, виконує вхід до системи, відкриває особистий кабінет, переглядає курси, переходить до уроків і контролює власний навчальний прогрес. Тому під час проєктування інтерфейсу потрібно враховувати не лише його зовнішній вигляд, а й логіку розміщення основних елементів.

Для розроблюваної платформи обрано просту структуру сторінок. Таке рішення є доцільним, оскільки надмірна кількість блоків, кнопок і другорядних елементів може ускладнити роботу користувача. У навчальній системі головними мають бути швидкий доступ до курсів, зрозумілий перехід до уроків і наочне відображення результатів проходження. Якщо користувач після відкриття сайту одразу розуміє, яку дію потрібно виконати далі, інтерфейс виконує своє основне призначення.

До основних сторінок платформи належать:

- головна сторінка;
- сторінка реєстрації;
- сторінка авторизації;
- особистий кабінет;
- сторінка списку курсів;
- сторінка окремого курсу;

- сторінка уроку.

Головна сторінка призначена для короткого ознайомлення користувача з платформою. На ній доцільно розмістити назву системи, стислий опис її можливостей, кнопку переходу до курсів, а також посилання на вхід або реєстрацію. Цю сторінку не потрібно перевантажувати великим обсягом тексту, оскільки її завдання полягає в тому, щоб користувач швидко зрозумів призначення сервісу та міг перейти до подальших дій.

Сторінка реєстрації містить форму для створення облікового запису. Користувач вводить ім’я, електронну пошту та пароль, після чого серверна частина перевіряє отримані дані й додає нового користувача до бази даних. Сторінка авторизації має подібну структуру, однак виконує іншу функцію: перевіряє введені облікові дані та надає доступ до особистого кабінету в разі успішного входу.

Особистий кабінет є центральною сторінкою для зареєстрованого користувача. У ньому мають відображатися ім’я користувача, доступні курси та показники навчального прогресу. Також у системі потрібно передбачити можливість виходу з облікового запису. Ця функція є важливою для завершення сеансу роботи та захисту персонального розділу користувача.

Сторінка списку курсів використовується для перегляду доступних навчальних напрямів. Кожен курс доцільно подавати у вигляді окремої картки, що містить назву, короткий опис і кнопку переходу до детальнішої інформації. Такий формат є зручним, оскільки дозволяє користувачу швидко переглянути доступні варіанти й обрати потрібний курс.

Сторінка окремого курсу містить розширену інформацію про навчальний матеріал. На ній розміщується опис курсу, перелік уроків і показник прогресу. Біля кожного уроку доцільно відображати його стан: завершений або ще не пройдений. Завдяки цьому користувач бачить власний рух у межах курсу без додаткових переходів між сторінками.

Сторінка уроку повинна мати максимально просту структуру. Основну частину сторінки займає навчальний матеріал, а поруч або нижче

розміщуються елементи керування проходженням. Зокрема, передбачається кнопка «Позначити як пройдений» і можливість перейти до наступного уроку. Після зміни статусу серверна частина записує результат у базу даних, а система оновлює показник прогресу.

Навчальні матеріали уроків у межах проєкту підготовлені заздалегідь і зберігаються в базі даних. Такий підхід є достатнім для демонстрації роботи платформи, оскільки основна увага приділяється не створенню великого обсягу контенту, а реалізації користувацького сценарію, проходженню уроків і відображенню результатів навчання.

Структуру сторінок платформи подано в таблиці 2.8.

Сторінка	Призначення
Головна сторінка	Ознайомлення користувача з платформою
Реєстрація	Створення нового облікового запису
Авторизація	Вхід користувача до системи
Особистий кабінет	Перегляд курсів і прогресу навчання
Список курсів	Відображення доступних онлайн-курсів
Сторінка курсу	Перегляд опису курсу та списку уроків
Сторінка уроку	Вивчення матеріалу та завершення уроку
Адміністративна	Керування курсами та уроками

Таблиця 2.8 – Основні сторінки платформи онлайн-курсів

Під час створення інтерфейсу необхідно дотримуватися кількох принципів. По-перше, навігація на основних сторінках повинна бути однаковою, щоб користувач не втрачав орієнтацію під час роботи із системою. По-друге, кнопки й посилання мають мати зрозумілі назви. По-третє, прогрес навчання потрібно показувати достатньо помітно, але без надмірного візуального акценту, щоб він не заважав перегляду навчальних матеріалів.

Для відображення курсів у цій платформі доцільно використовувати карткову структуру. Картка дозволяє компактно подати назву курсу, короткий опис і кнопку переходу. Для показу прогресу можна застосувати текстовий відсоток або просту шкалу. Наприклад, біля курсу може відображатися напис

«Прогрес: 40 %», чого достатньо для швидкого розуміння поточного результату.

Інтерфейс платформи не повинен вимагати від користувача спеціальних технічних знань. Людина має послідовно виконати базові дії: відкрити сайт, зареєструватися або увійти до системи, обрати курс, перейти до уроку та продовжити навчання. Саме така логіка покладена в основу проєктування сторінок розроблюваної вебплатформи.

Висновки до другого розділу

У другому розділі виконано проєктування платформи онлайн-курсів з особистим кабінетом користувача та механізмом відстеження прогресу навчання. Насамперед обґрунтовано вибір технологій для реалізації системи. Для розробки обрано HTML5, CSS3, JavaScript, PHP та MySQL, оскільки цей набір засобів дає змогу створити зрозумілий вебзастосунок із клієнтською частиною, серверною логікою та базою даних.

Також описано загальну архітектуру платформи. Система побудована за клієнт-серверним принципом: користувач взаємодіє з вебінтерфейсом у браузері, серверна частина обробляє запити, а база даних зберігає відомості про користувачів, курси, уроки та результати проходження. Така структура є зручною для розробки, локального тестування й пояснення роботи основних модулів платформи.

Окрему увагу приділено проєктуванню бази даних. Для збереження основної інформації передбачено таблиці users, courses, lessons, progress і course_starts. Вони забезпечують реєстрацію користувачів, роботу з курсами й уроками, фіксацію початку навчання та збереження результатів проходження. Зв’язки між таблицями дозволяють визначати, які уроки завершив конкретний користувач і який відсоток прогресу він має в межах обраного курсу.

У розділі спроєктовано особистий кабінет користувача. Він розглядається як центральна сторінка платформи, де відображаються дані користувача, доступні курси, стан їх проходження та можливість продовжити

навчання. Такий підхід робить взаємодію з платформою більш послідовною, оскільки основні дії зібрані в одному персональному розділі.

Розроблено алгоритм відстеження прогресу навчання. Його логіка полягає в тому, що після завершення уроку система створює або перевіряє відповідний запис у базі даних, визначає кількість завершених уроків і розраховує відсоток проходження курсу. Це рішення є достатньо простим для реалізації, але повністю відповідає поставленій задачі, оскільки дає змогу показувати кожному користувачу його індивідуальний результат.

Крім того, визначено структуру користувацького інтерфейсу. Передбачено головну сторінку, сторінки реєстрації та авторизації, особистий кабінет, список курсів, сторінку окремого курсу та сторінку уроку. Усі ці сторінки формують логічний шлях користувача: від входу на платформу до проходження навчальних матеріалів і перегляду власного прогресу.

Результати другого розділу створюють основу для практичної реалізації системи. Наступним етапом є опис створення вебзастосунку, реалізації основних програмних модулів, підключення до бази даних, роботи з курсами й уроками, а також тестування функціональності розробленої платформи.

3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПЛАТФОРМИ ОНЛАЙН-КУРСІВ

3.1 Створення структури вебзастосунку

Після проєктування архітектури, бази даних та інтерфейсу можна перейти до практичної реалізації платформи. На цьому етапі важливо не просто створити набір окремих сторінок, а об’єднати їх у цілісний вебзастосунок, у якому кожен файл виконує конкретну функцію. Структура проєкту впливає на зручність підтримки системи, додавання нових сторінок і перевірку роботи окремих модулів.

Для розроблюваної платформи використано просту файлову структуру. Вона підходить для навчального проєкту, оскільки не потребує використання складних фреймворків, але водночас дозволяє логічно розділити сторінки, стилі, скрипти та файл підключення до бази даних.

Орієнтовну структуру вебзастосунку подано нижче.

Файл або папка	Призначення
index.php	Головна сторінка платформи
register.php	Сторінка реєстрації користувача
login.php	Сторінка авторизації
logout.php	Вихід користувача з облікового запису
dashboard.php	Особистий кабінет користувача
courses.php	Сторінка зі списком курсів
course.php	Сторінка окремого курсу
lesson.php	Сторінка уроку
complete_lesson.php	Обробка завершення уроку
admin.php	Базова сторінка адміністратора
db.php	Підключення до бази даних
css/style.css	Файл стилів інтерфейсу
js/script.js	Допоміжні JavaScript-функції
images/	Папка для зображень курсів

Таблиця 3.1 – Структура файлів вебзастосунку

Файл index.php є головною сторінкою платформи. З цієї сторінки користувач може перейти до реєстрації, авторизації або перегляду доступних курсів. У межах навчального проєкту головна сторінка не повинна бути надто складною. Її основне призначення полягає в короткому поясненні можливостей платформи та наданні користувачу зрозумілого шляху для початку роботи.

Файли register.php і login.php відповідають за створення облікового запису та вхід до системи. Через ці сторінки користувач передає дані на сервер, після чого PHP-скрипти перевіряють введену інформацію та взаємодіють із базою даних. Якщо реєстрація або авторизація проходить успішно, користувач отримує доступ до особистого кабінету.

Файл dashboard.php реалізує сторінку особистого кабінету. На цій сторінці користувач бачить власне ім’я, доступні курси та показники навчального прогресу. Доступ до особистого кабінету має надаватися лише після авторизації. Якщо користувач не увійшов до системи, його потрібно перенаправити на сторінку входу.

Робота з навчальним контентом реалізована через файли courses.php, course.php та lesson.php. Файл courses.php відповідає за відображення каталогу курсів, course.php відкриває сторінку конкретного курсу з переліком уроків, а lesson.php відображає матеріал окремого уроку. Такий поділ робить структуру системи зрозумілою, оскільки користувач послідовно переходить від загального списку курсів до конкретного навчального матеріалу.

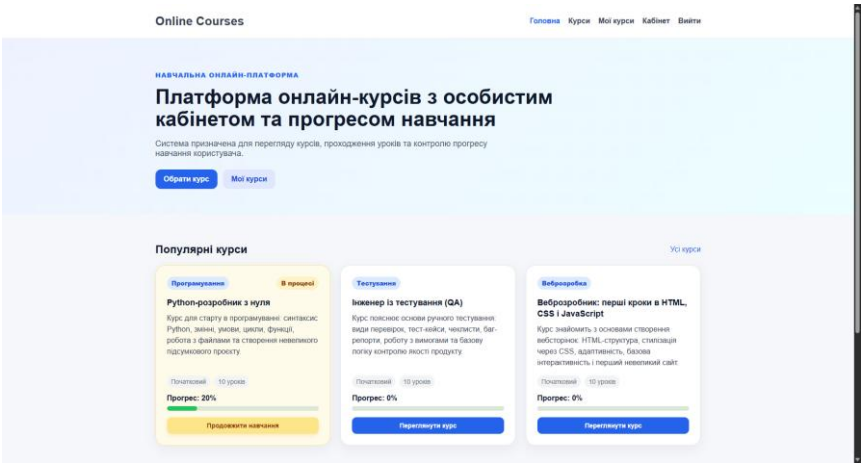

Рисунок 3.1 – Головна сторінка платформи онлайн-курсів

Файл toggle_lesson.php використовується в механізмі оновлення прогресу. Він обробляє запит після натискання кнопки «Позначити як пройдений», перевіряє стан уроку для поточного користувача та оновлює відповідний запис у таблиці progress. Після виконання цієї дії користувач повертається до сторінки уроку або курсу й бачить оновлений результат проходження.

Окремо винесено файл db.php, у якому містяться параметри підключення до бази даних. Таке рішення є зручним, оскільки підключення використовується на різних сторінках платформи. Якщо параметри доступу до бази даних зміняться, достатньо буде відредагувати один файл, а не вносити зміни до кожної сторінки окремо.

Для зовнішнього оформлення використовується файл assets/css/style.css. У ньому задаються стилі для сторінок, форм, кнопок, карток курсів, блоків особистого кабінету та інших елементів інтерфейсу. Файл assets/js/script.js призначений для невеликих інтерактивних дій на клієнтській стороні, зокрема для показу або приховування пароля у формах входу та реєстрації.

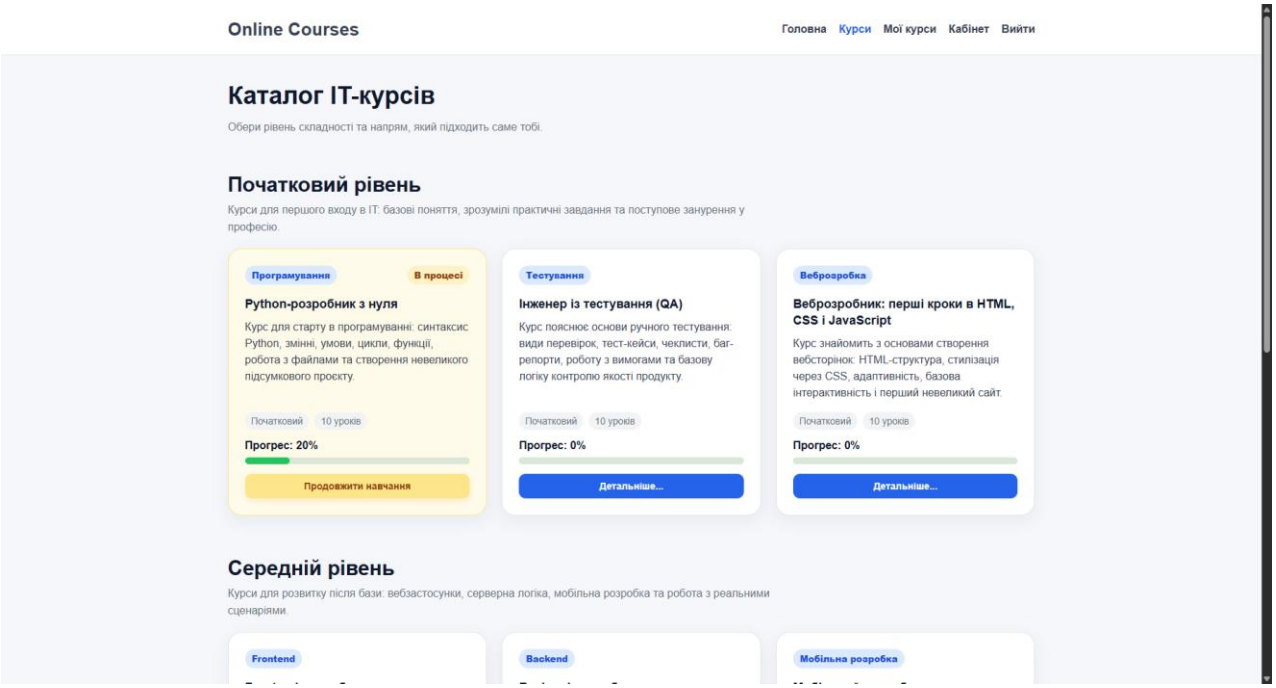

Рисунок 3.2 – Каталог курсів

Обрана структура не є єдиним можливим варіантом організації проєкту. У складніших вебзастосунках можуть використовуватися фреймворки, маршрутизація, шаблонізатори та поділ логіки на окремі модулі. Проте для цієї кваліфікаційної роботи така файлова структура є достатньою. Вона дозволяє наочно показати роботу основних частин платформи: сторінок інтерфейсу, серверної обробки, підключення до бази даних і механізму збереження прогресу навчання.

3.2 Реалізація підключення до бази даних

Для роботи платформи необхідно забезпечити зв’язок між PHP-скриптами та базою даних MySQL. Саме через це підключення система отримує й обробляє інформацію про користувачів, курси, уроки та навчальний прогрес. Якщо з’єднання з базою даних налаштовано неправильно, більшість функцій вебзастосунку не зможе працювати коректно.

У проєкті підключення до бази даних винесено в окремий файл db.php. Такий підхід є зручним, оскільки різні сторінки платформи можуть використовувати один і той самий файл через інструкцію include або require. Завдяки цьому код підключення не дублюється на кожній сторінці, а структура проєкту залишається більш упорядкованою.

Приклад підключення до бази даних наведено нижче.

```php
<?php
$host = "localhost";
$user = "root";
$password = "";
$database = "online_courses";
$conn = mysqli_connect($host, $user, $password, $database);
if (!$conn) {
    die("Помилка підключення до бази даних: " . mysqli_connect_error());
}
mysqli_set_charset($conn, "utf8");
?>
```

У наведеному фрагменті задаються основні параметри з’єднання: адреса сервера, ім’я користувача, пароль і назва бази даних. Для локального середовища зазвичай використовується сервер localhost, користувач root і порожній пароль. На реальному хостингу ці значення можуть відрізнятися, однак загальний принцип підключення залишається таким самим.

Змінна $conn зберігає результат встановлення з’єднання з базою даних. Якщо підключення не вдалося виконати, система повинна повідомити про помилку, щоб розробник міг швидко визначити причину проблеми. Також у файлі підключення доцільно встановити кодування utf8, оскільки платформа працює з українським текстом. Це забезпечує коректне збереження й відображення назв курсів, уроків, описів і повідомлень користувачу.

Після створення файлу db.php його можна підключати до інших сторінок платформи. Наприклад:

```
<?php
      require_once "db.php";
?>
```

Такий рядок дає сторінці доступ до з’єднання з базою даних. Після цього через змінну $conn можна виконувати SQL-запити: додавати нових користувачів, отримувати список курсів, перевіряти дані авторизації, відкривати уроки або оновлювати прогрес навчання.

Підключення до бази даних є невеликим за обсягом, але важливим елементом проєкту. Без нього платформа не зможе працювати як персоналізована система. Саме база даних дає змогу зберігати облікові записи, пов’язувати дії з конкретним користувачем і відображати йому власні курси та результати проходження навчання.

3.3 Реалізація реєстрації та авторизації користувача

Реєстрація та авторизація є базовими функціями платформи онлайн-курсів, оскільки саме вони дають змогу пов’язати дії в системі з конкретним користувачем. Без облікового запису платформа не зможе зберігати завершені

уроки, відображати персональний кабінет або відновлювати стан навчання під час повторного входу.

Під час реєстрації користувач вводить ім’я, електронну пошту та пароль. Після надсилання форми серверна частина приймає ці дані, виконує необхідну перевірку та додає нового користувача до таблиці users. Електронна пошта повинна бути унікальною, оскільки вона використовується як один з основних ідентифікаторів для входу до системи.

Приклад HTML-форми реєстрації наведено нижче.

```
<form action="register.php" method="post">
<label>Ім’я користувача</label>
<input type="text" name="name" required>

<label>Електронна пошта</label>
<input type="email" name="email" required>

<label>Пароль</label>
<input type="password" name="password" required>

<button type="submit">Зареєструватися</button>
</form>
```

Форма передає дані методом POST. Такий спосіб є доцільним для реєстрації, оскільки пароль та інші особисті дані не повинні передаватися через адресний рядок браузера.

Обробка реєстрації на серверній стороні має такий вигляд.

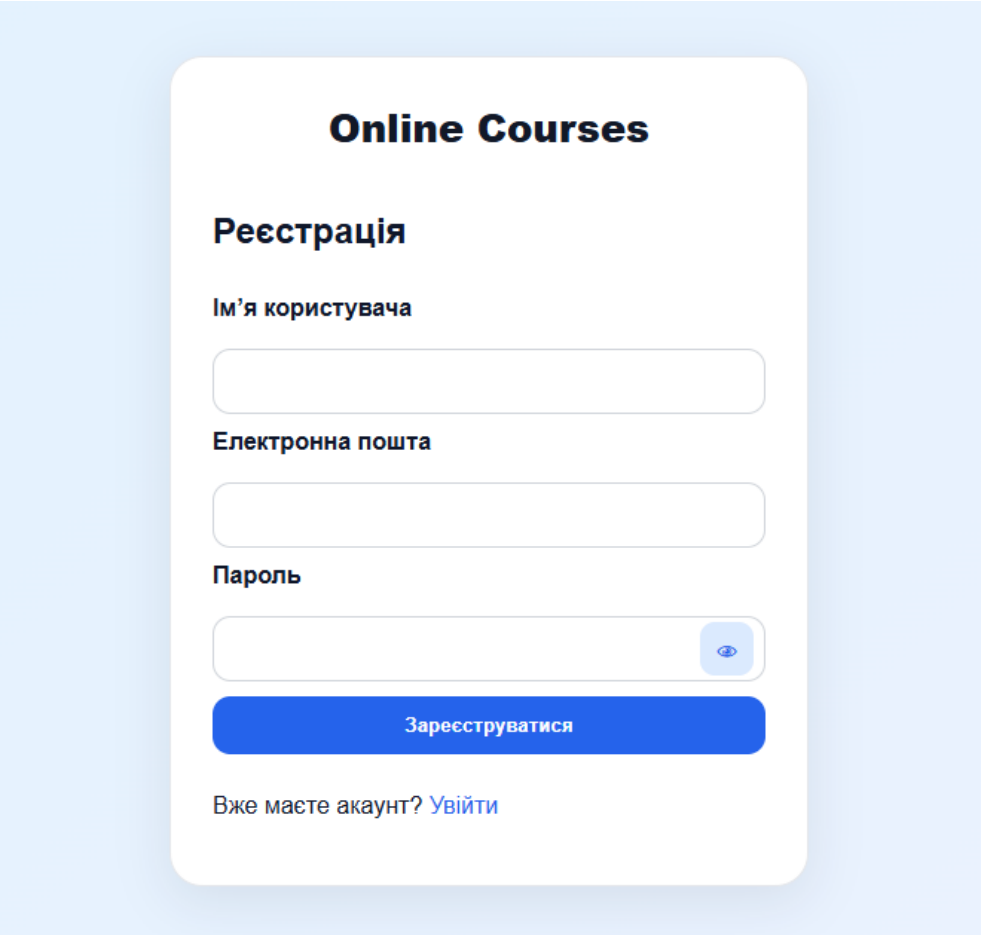

Рисунок 3.3 – Сторінка реєстрації користувача

```
<?php
require_once "db.php";

$name = $_POST["name"];
$email = $_POST["email"];
$password = password_hash($_POST["password"], PASSWORD_DEFAULT);

$query = "INSERT INTO users (name, email, password)
VALUES ('$name', '$email', '$password')";

if (mysqli_query($conn, $query)) {
header("Location: login.php");
} else {
echo "Помилка реєстрації користувача";
}
?>
```

У наведеному фрагменті пароль перед збереженням хешується, тому в базі даних він не зберігається у відкритому вигляді. Для цього використовується функція password_hash. Такий підхід є важливим навіть для

навчального проєкту, оскільки демонструє правильну роботу з обліковими даними користувача.

Авторизація працює за схожим принципом, однак її завдання полягає не в додаванні нового запису, а в пошуку наявного користувача в базі даних. Якщо користувач із введеною електронною поштою існує, система перевіряє відповідність пароля. У разі успішної перевірки створюється сеанс користувача, після чого він отримує доступ до особистого кабінету.

Приклад перевірки авторизації наведено нижче.

```
<?php
session_start();
require_once "db.php";

$email = $_POST["email"];
$password = $_POST["password"];

$query = "SELECT * FROM users WHERE email = '$email'";
$result = mysqli_query($conn, $query);
$user = mysqli_fetch_assoc($result);

if ($user && password_verify($password, $user["password"])) {
    $_SESSION["user_id"] = $user["id"];
    $_SESSION["user_name"] = $user["name"];
    header("Location: dashboard.php");
} else {
    echo "Неправильна електронна пошта або пароль";
}
?>
```

Сеанс потрібен для того, щоб система могла визначати, хто саме працює з платформою в поточний момент. Після успішного входу ідентифікатор користувача зберігається в $_SESSION["user_id"]. Надалі це значення використовується на інших сторінках, зокрема в особистому кабінеті, на сторінках курсів і під час оновлення навчального прогресу.

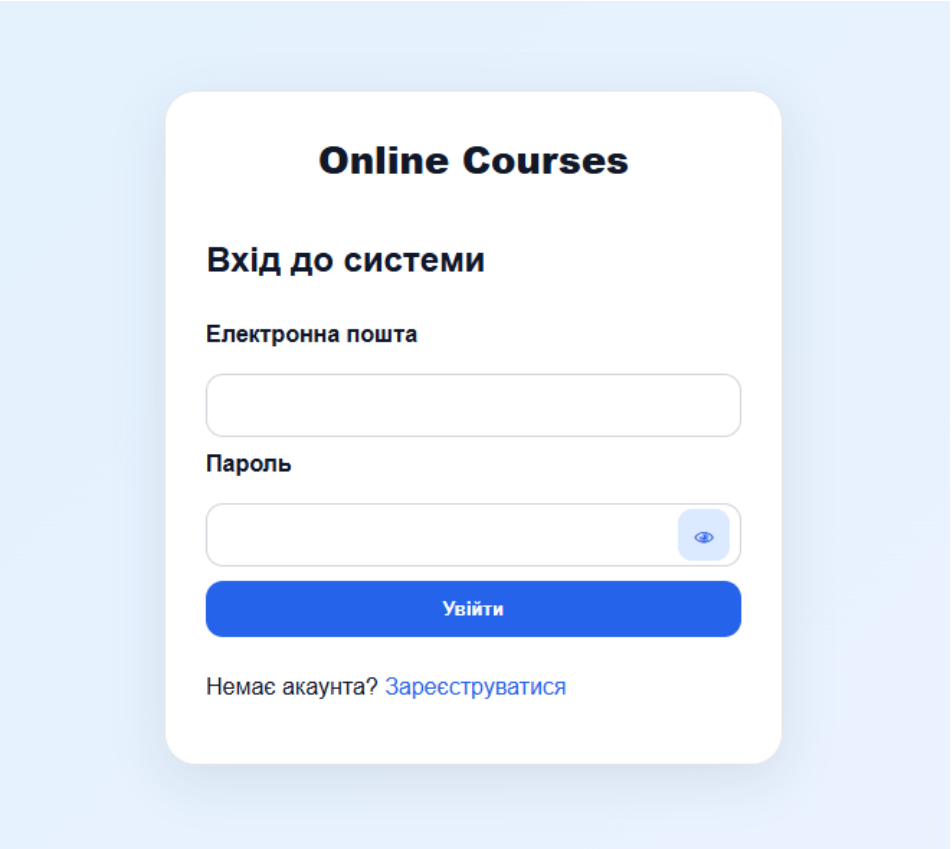

Рисунок 3.4 – Сторінка авторизації користувача

Для виходу з облікового запису використовується файл logout.php, який завершує сеанс користувача.

```
<?php
session_start();
session_destroy();
header("Location: index.php");
exit;
?>
```

Після виконання цього коду користувач повертається на головну сторінку або сторінку входу, а доступ до персонального розділу припиняється. Це проста, але необхідна частина роботи з обліковим записом, оскільки вона дозволяє коректно завершити сеанс.

Також у системі потрібно реалізувати перевірку авторизації для сторінок, які мають бути доступні лише зареєстрованому користувачу.

Наприклад, на початку файлу dashboard.php можна додати перевірку наявності активного сеансу.

Приклад такої перевірки наведено нижче.

```
<?php
session_start();

if (!isset($_SESSION["user_id"])) {
    header("Location: login.php");
    exit;
}
?>
```

Якщо користувач не авторизований, система перенаправляє його на сторінку входу. Такий механізм запобігає несанкціонованому доступу до особистого кабінету у випадках, коли користувач намагається відкрити сторінку напряму без попередньої авторизації.

Реалізація реєстрації та авторизації створює основу для подальшої роботи платформи. Після цього можна переходити до формування особистого кабінету, відображення курсів і збереження прогресу навчання. Усі ці функції використовують дані авторизованого користувача, тому без модуля входу вони не могли б працювати як персоналізована система.

3.4 Реалізація особистого кабінету та списку курсів

Після успішної авторизації користувач переходить до особистого кабінету. Це одна з основних сторінок платформи, оскільки саме в цьому розділі відображаються навчальні матеріали, доступні користувачу, та загальний стан проходження курсів. Якщо сторінка входу лише надає доступ до системи, то особистий кабінет уже формує персональну частину платформи для конкретного користувача.

Перед відображенням особистого кабінету необхідно перевірити, чи користувач авторизований. Якщо активний сеанс відсутній, сторінка не повинна відкриватися. Така перевірка потрібна для захисту персональних

даних і результатів навчання від перегляду без входу до системи. Після підтвердження авторизації система отримує ідентифікатор користувача із сеансу та використовує його для завантаження потрібної інформації.

Приклад початкової перевірки доступу до особистого кабінету наведено нижче.

```
<?php
session_start();
require_once "db.php";

if (!isset($_SESSION["user_id"])) {
    header("Location: login.php");
    exit;
}

$user_id = $_SESSION["user_id"];
$user_name = $_SESSION["user_name"];
?>
```

У цьому фрагменті використовується змінна $_SESSION["user_id"]. Якщо вона відсутня, користувач перенаправляється на сторінку входу. Якщо ж сеанс існує, система продовжує виконання сторінки та може відображати дані особистого кабінету.

В особистому кабінеті доцільно розмістити привітання, список доступних курсів і показники прогресу за кожним курсом. Для отримання курсів із бази даних виконується SQL-запит до таблиці courses. Після цього отримані записи можна вивести на сторінку у вигляді карток або списку.

```
<?php
$query = "SELECT * FROM courses";
$result = mysqli_query($conn, $query);
?>
```

Після завантаження даних кожен курс відображається окремим блоком. У ньому розміщуються назва курсу, короткий опис і кнопка переходу до сторінки курсу. Такий спосіб подання є зручним для користувача, оскільки він

одразу бачить доступні навчальні матеріали та може швидко перейти до потрібного курсу.

```
<div class="courses-list">
<?php while ($course = mysqli_fetch_assoc($result)): ?>
<div class="course-card">
<h3><?php echo $course["title"]; ?></h3>
<p><?php echo $course["description"]; ?></p>
<a href="course.php?id=<?php echo $course["id"]; ?>">
Перейти до курсу
</a>
</div>
<?php endwhile; ?>
</div>
```

У коді для виведення курсів використовується цикл while, який послідовно обробляє всі записи, отримані з бази даних. Для кожного курсу створюється окремий елемент інтерфейсу з назвою, описом і посиланням на сторінку конкретного курсу. Завдяки цьому список курсів формується динамічно, а не задається вручну в HTML-коді

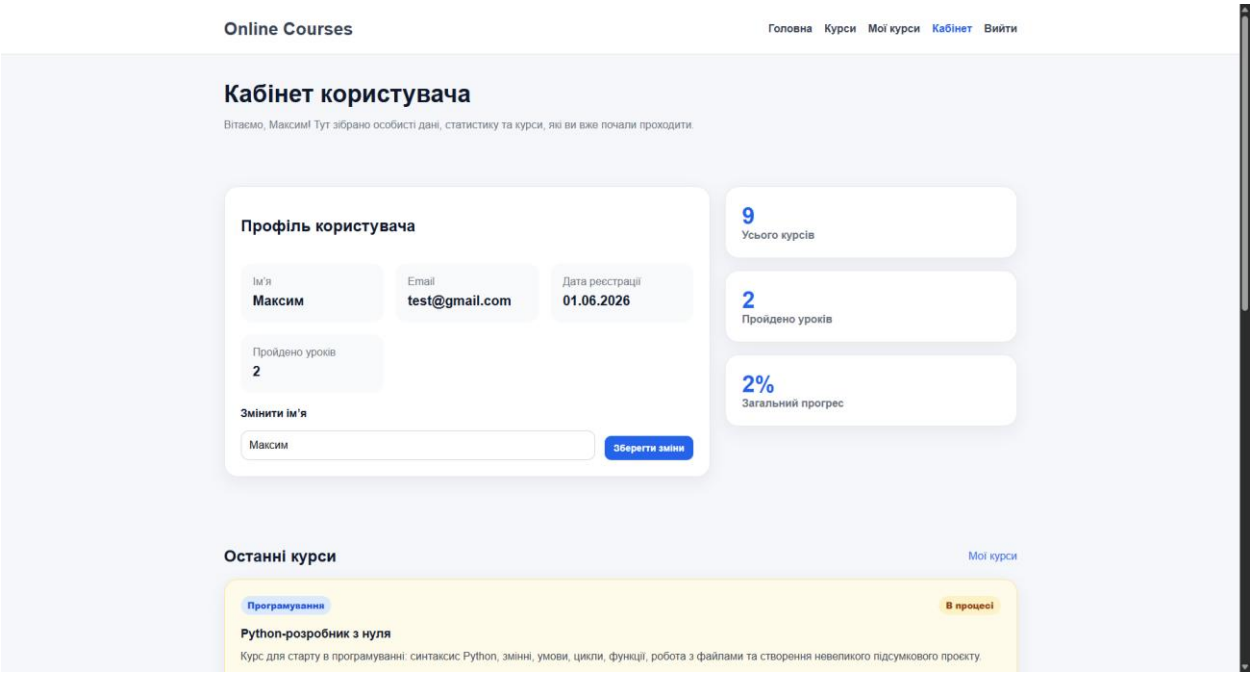

Рисунок 3.5 – Особистий кабінет користувача

На сторінці course.php користувач бачить уже не загальний перелік курсів, а матеріали конкретного обраного курсу. Для цього сторінка отримує

ідентифікатор курсу через параметр id в адресному рядку. Далі система виконує запит до таблиці lessons і вибирає лише ті уроки, які належать до відповідного курсу.

```
<?php
$course_id = $_GET["id"];

$query = "SELECT * FROM lessons
          WHERE course_id = $course_id
          ORDER BY lesson_order ASC";

$lessons = mysqli_query($conn, $query);
?>
```

Сортування за полем lesson_order потрібне для правильного відображення навчальних матеріалів. Для освітньої платформи порядок уроків має значення, оскільки користувач повинен проходити матеріал послідовно: від першої теми до останньої. Якщо не враховувати порядок уроків, структура курсу може стати незрозумілою.

```
<ul class="lessons-list">
    <?php while ($lesson = mysqli_fetch_assoc($lessons)): ?>
        <li>
            <a href="lesson.php?id=<?php echo $lesson["id"]; ?>">
                <?php echo $lesson["title"]; ?>
            </a>
        </li>
    <?php endwhile; ?>
</ul>
```

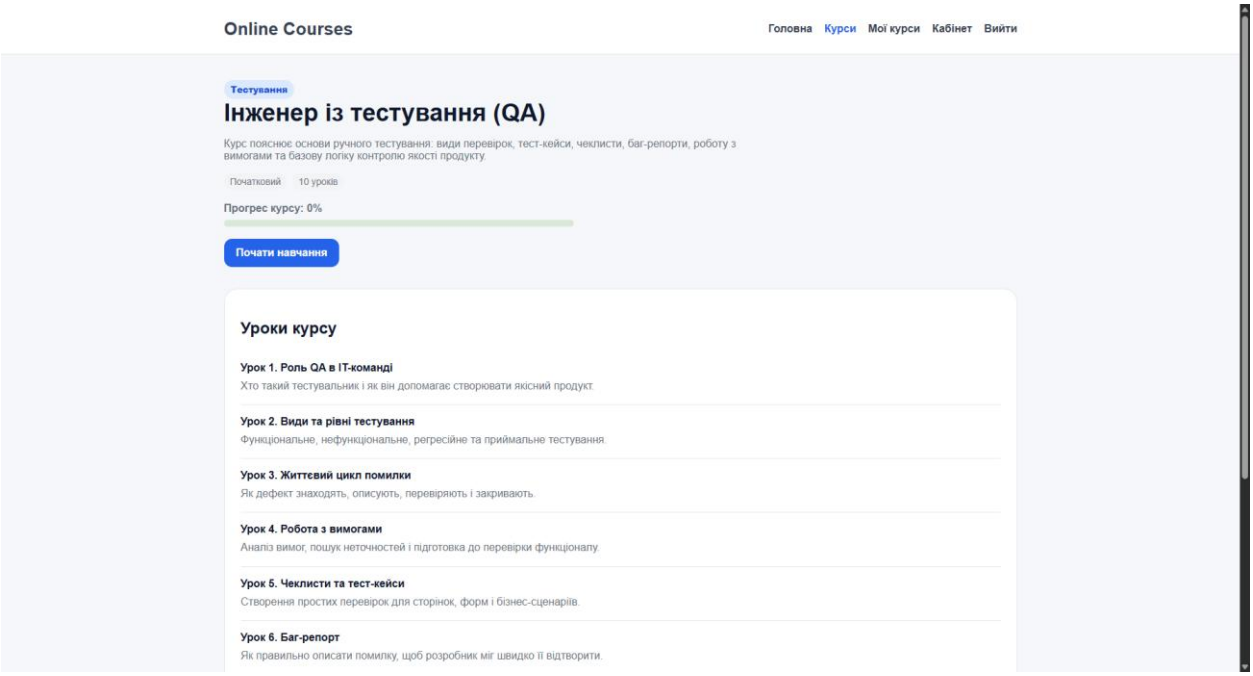

Рисунок 3.6 – Сторінка окремого курсу

Після отримання уроків система виводить їх у вигляді списку. Біля кожного уроку розміщується кнопка або посилання для переходу до навчального матеріалу. Така реалізація забезпечує зрозумілий маршрут роботи з платформою: особистий кабінет → курс → урок. Ця схема є простою, але достатньою для базової навчальної системи.

Під час реалізації особистого кабінету також необхідно передбачити кнопку виходу з облікового запису. Вона може вести на файл logout.php, який завершує поточний сеанс користувача. Без цього елемента робота з персональним розділом була б неповною, оскільки користувач не мав би зручного способу завершити взаємодію із системою.

У результаті особистий кабінет виконує роль центральної сторінки платформи. Через нього користувач отримує доступ до курсів, переходить до уроків і надалі переглядає власний навчальний прогрес. Саме тому реалізація цієї сторінки є важливою частиною практичної розробки вебзастосунку.

3.5 Реалізація проходження уроків і прогресу навчання

Після створення особистого кабінету та сторінки курсу необхідно реалізувати проходження уроків. Саме на цьому етапі платформа починає працювати як навчальна система, а не лише як каталог матеріалів. Користувач відкриває урок, ознайомлюється з його змістом і після завершення натискає кнопку, яка змінює стан уроку та впливає на показник прогресу.

Сторінка lesson.php отримує ідентифікатор уроку через параметр id в адресному рядку. Після цього система звертається до бази даних і завантажує назву та вміст потрібного уроку. Приклад відповідного запиту наведено нижче.

```
<?php
require_once "db.php";

$lesson_id = $_GET["id"];

$query = "SELECT * FROM lessons WHERE id = $lesson_id";
$result = mysqli_query($conn, $query);
$lesson = mysqli_fetch_assoc($result);
?>
```

Після отримання даних урок відображається на сторінці. Основний блок містить заголовок уроку, навчальний матеріал і кнопку для зміни статусу проходження.

```
<div class="lesson-page">
    <h2><?php echo $lesson["title"]; ?></h2>

    <div class="lesson-content">
        <?php echo $lesson["content"]; ?>
    </div>

   <form action="toggle_lesson.php" method="post">
        <input type="hidden" name="lesson_id"
               value="<?php echo $lesson["id"]; ?>">
        <button type="submit">Завершити урок</button>
    </form>
</div>
```

У формі використовується приховане поле lesson_id, через яке до файлу toggle_lesson.php передається ідентифікатор уроку. Для користувача цей елемент не відображається: він бачить лише кнопку зміни статусу. Водночас саме приховане поле дає серверній частині змогу визначити, який урок потрібно позначити як пройдений або, за потреби, повернути до непройденого стану.

Файл toggle_lesson.php відповідає за оновлення прогресу в базі даних. Перед зміною статусу система повинна перевірити, чи був цей урок уже позначений як завершений поточним користувачем. Якщо запис існує, його можна видалити. Якщо запису немає, система додає новий рядок до таблиці progress. Такий підхід запобігає дублюванню даних і не дозволяє одному уроку впливати на прогрес кілька разів.

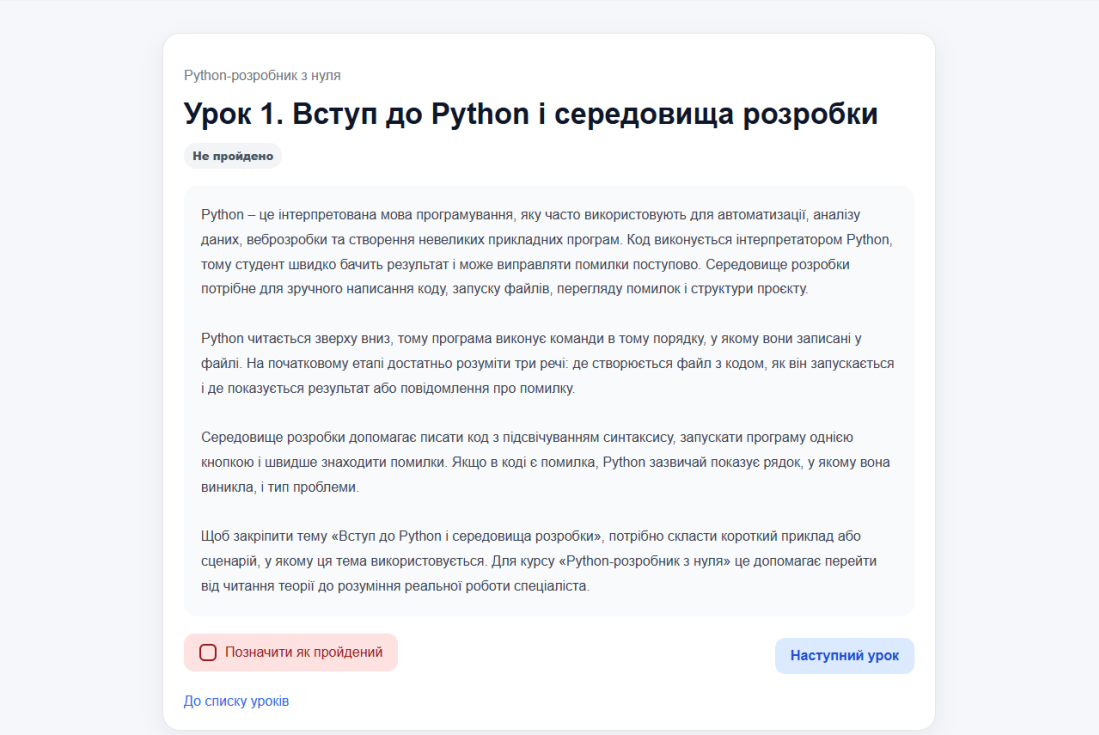

Рисунок 3.7 – Сторінка уроку

```
<?php
session_start();
require_once "db.php";

if (!isset($_SESSION["user_id"])) {
```

```
    header("Location: login.php");
    exit;
}

$user_id = $_SESSION["user_id"];
$lesson_id = $_POST["lesson_id"];

$check = "SELECT * FROM progress
          WHERE user_id = $user_id AND lesson_id = $lesson_id";

$result = mysqli_query($conn, $check);

if (mysqli_num_rows($result) == 0) {
    $insert = "INSERT INTO progress (user_id, lesson_id, completed_at)
               VALUES ($user_id, $lesson_id, NOW())";
    mysqli_query($conn, $insert);
}

header("Location: dashboard.php");
exit;
?>
```

У цьому фрагменті коду спочатку перевіряється авторизація користувача. Потім система отримує ідентифікатор користувача та уроку. Далі виконується запит до таблиці progress, щоб перевірити наявність запису. Якщо такого запису немає, він додається до бази даних.

Після завершення уроку користувач повертається до особистого кабінету. Там система повинна показати оновлений прогрес. Для розрахунку прогресу потрібно знати дві величини: загальну кількість уроків у курсі та кількість уроків, завершених користувачем.

```
<?php
$course_id = $course["id"];

$total_query = "SELECT COUNT(*) AS total
                FROM lessons
                WHERE course_id = $course_id";
```

```
$total_result = mysqli_query($conn, $total_query);
$total_lessons = mysqli_fetch_assoc($total_result)["total"];

$completed_query = "SELECT COUNT(*) AS completed
                    FROM progress p
                    JOIN lessons l ON p.lesson_id = l.id
                    WHERE p.user_id = $user_id
                    AND l.course_id = $course_id";

$completed_result = mysqli_query($conn, $completed_query);
$completed_lessons = mysqli_fetch_assoc($completed_result)["completed"];

if ($total_lessons > 0) {
    $progress = round(($completed_lessons / $total_lessons) * 100);
} else {
    $progress = 0;
}
?>
```

Цей код рахує прогрес за простою формулою: кількість завершених уроків ділиться на загальну кількість уроків і множиться на 100. Функція round використовується для округлення результату до цілого числа. Наприклад, якщо користувач завершив 2 уроки з 5, система покаже 40 %.

Після розрахунку прогрес можна відобразити в особистому кабінеті біля кожного курсу.

```
<div class="progress-block">
    <p>Прогрес навчання: <?php echo $progress; ?>%</p>

    <div class="progress-line">
        <div class="progress-fill"
             style="width: <?php echo $progress; ?>%;">
        </div>
    </div>
</div>
```

У цьому фрагменті прогрес показується двома способами: числом у відсотках і графічною шкалою. Такий варіант зручний для користувача, бо він одразу бачить результат проходження курсу.

Для оформлення шкали прогресу використовуються CSS-стилі.

```
.progress-line {
    width: 100%;
    height: 12px;
    background: #e5e5e5;
    border-radius: 10px;
    overflow: hidden;
}

.progress-fill {
    height: 100%;
    background: #4a90e2;
    border-radius: 10px;
}
```

Завдяки такому оформленню прогрес виглядає наочно. Якщо користувач пройшов половину курсу, заповнюється половина шкали. Якщо курс завершено повністю, шкала заповнюється на 100 %.

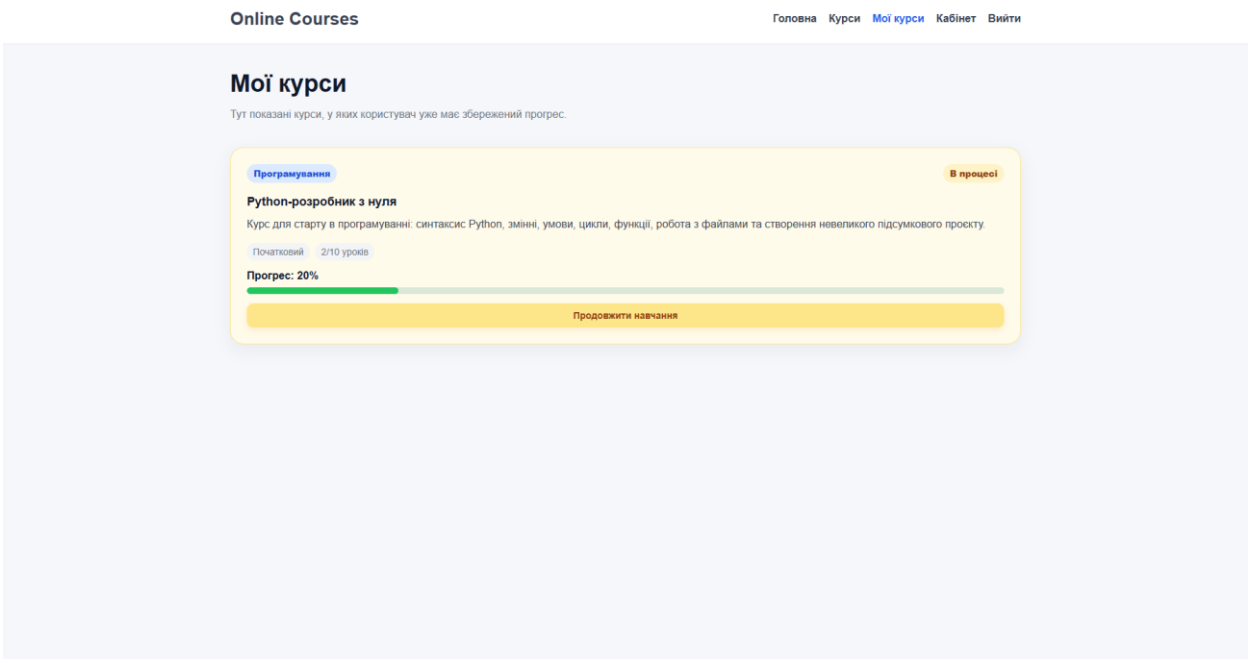

Рисунок 3.8 – Сторінка «Мої курси» з відображенням прогресу

Реалізований механізм проходження уроків і прогресу є достатнім для базової платформи онлайн-курсів. Він не перевантажує систему, але дозволяє

показати головну ідею роботи: користувач навчається, система зберігає результат, а особистий кабінет показує поточний стан проходження.

3.6 Тестування роботи платформи

Після реалізації основних функцій платформи необхідно перевірити її працездатність. Тестування дає змогу встановити, чи правильно виконується основний сценарій роботи користувача: створення облікового запису, авторизація, відкриття особистого кабінету, перехід до курсу, завершення уроку та перегляд оновленого прогресу навчання. Без такої перевірки неможливо підтвердити, що вебзастосунок працює як цілісна система.

У межах цієї кваліфікаційної роботи тестування виконувалося вручну. Такий підхід є достатнім для навчального вебпроєкту, оскільки платформа має обмежену кількість основних функцій і зрозумілий користувацький сценарій. Перевірка проводилася шляхом послідовного виконання дій у браузері з подальшим порівнянням фактичного результату з очікуваним.

Основні результати перевірки роботи платформи наведено в таблиці 3.2.

№	Функція	Дія користувача	Очікуваний результат	Фактичний результат
1	Реєстрація	Користувач заповнює форму реєстрації та натискає кнопку створення облікового запису	Дані користувача зберігаються в базі даних	Виконано
2	Авторизація	Користувач вводить електронну пошту та пароль	Відкривається особистий кабінет	Виконано
3	Захист кабінету	Неавторизований користувач намагається відкрити dashboard.php	Система перенаправляє на сторінку входу	Виконано
4	Перегляд курсів	Користувач відкриває особистий кабінет	Відображається список доступних курсів	Виконано
5	Відкриття курсу	Користувач натискає кнопку переходу до курсу	Відкривається сторінка курсу зі списком уроків	Виконано
6	Перегляд уроку	Користувач натискає на назву уроку	Відкривається сторінка з навчальним матеріалом	Виконано
7	Завершення уроку	Користувач натискає кнопку «Завершити урок»	У базі даних створюється запис про завершення уроку	Виконано
8	Повторне завершення уроку	Користувач повторно натискає кнопку завершення того самого уроку	Повторний запис не додається	Виконано
9	Розрахунок прогресу	Користувач завершує частину уроків курсу	У кабінеті відображається правильний відсоток проходження	Виконано
10	Вихід із системи	Користувач натискає кнопку виходу	Сеанс завершується, відкривається головна сторінка	Виконано

Таблиця 3.2 – Результати тестування платформи онлайн-курсів

Як видно з таблиці 3.2, ключові функції платформи працюють коректно. Користувач може пройти повний шлях від реєстрації до перегляду власного навчального прогресу. Це є важливим результатом, оскільки саме такий сценарій покладено в основу розроблюваної системи.

Окремо перевірялася робота механізму відстеження прогресу. Для цього було створено тестовий курс із кількома уроками. Після завершення одного уроку система оновлювала відсоток проходження. Якщо користувач завершував наступний урок, значення прогресу збільшувалося відповідно до кількості пройдених матеріалів. Повторне натискання кнопки завершення для вже пройденого уроку не створювало дубльований запис у базі даних, тому показник прогресу не спотворювався.

Також було перевірено роботу авторизації. Якщо користувач вводив правильні облікові дані, система відкривала особистий кабінет. У разі введення неправильних даних доступ до персональної частини платформи не надавався. Це підтверджує, що сторінки, пов’язані з особистим кабінетом і навчальним прогресом, залежать від успішної перевірки користувача.

Під час тестування не було виявлено критичних помилок, які перешкоджали б виконанню основного сценарію роботи платформи. Водночас система має можливості для подальшого вдосконалення. У майбутньому до неї можна додати перевірку складності пароля, відновлення доступу через електронну пошту, тести після уроків або детальнішу статистику навчання.

Отже, результати тестування показали, що реалізованого функціоналу достатньо для роботи базової платформи онлайн-курсів. Система забезпечує реєстрацію й авторизацію користувача, доступ до особистого кабінету, перегляд курсів і уроків, збереження завершених матеріалів та коректне відображення прогресу навчання.

3.7 Аналіз результатів розробки

У результаті виконання практичної частини кваліфікаційної роботи створено вебплатформу онлайн-курсів з особистим кабінетом користувача та

механізмом відстеження прогресу навчання. Розроблена система дає змогу користувачу зареєструватися, увійти до облікового запису, переглядати доступні курси, відкривати уроки та фіксувати їх проходження. Після завершення уроків прогрес автоматично оновлюється й відображається у відсотках.

Реалізована платформа має просту та логічну структуру, що є перевагою для навчального проєкту. Користувачеві не потрібно виконувати складні дії або довго шукати потрібні сторінки. Основний сценарій роботи побудовано послідовно: вхід до системи, перехід в особистий кабінет, вибір курсу, перегляд уроку, позначення уроку як завершеного та оновлення прогресу.

З технічного погляду платформа складається з клієнтської частини, серверних PHP-файлів і бази даних MySQL. Клієнтська частина відповідає за відображення сторінок, форм, кнопок і блоків інтерфейсу. Серверна частина обробляє дії користувача, перевіряє авторизацію, отримує необхідні дані та оновлює результати проходження. База даних зберігає інформацію про користувачів, курси, уроки та навчальний прогрес.

Розроблена система не є статичним сайтом, оскільки вона підтримує персоналізацію. Після входу користувач бачить власний кабінет, доступні курси та індивідуальний результат проходження. Саме ця особливість відрізняє платформу онлайн-курсів від звичайного набору навчальних сторінок.

До основних результатів розробки можна віднести:

- створення структури вебзастосунку;
- реалізацію підключення до бази даних;
- створення механізмів реєстрації та авторизації користувача;
- реалізацію особистого кабінету;
- відображення списку доступних курсів;
- відкриття уроків обраного курсу;
- збереження інформації про завершені уроки;
- розрахунок і відображення прогресу навчання;

- тестування основних функцій платформи.

Розроблена платформа може бути використана як основа для подальшого розвитку. У майбутньому до неї можна додати систему тестування, оцінювання результатів, формування сертифікатів, коментарі до уроків, завантаження навчальних матеріалів або детальнішу статистику проходження курсів. Водночас навіть у базовому вигляді система демонструє головну ідею проєкту: користувач проходить онлайн-курс, а платформа зберігає та відображає його навчальний прогрес.

Фрагменти програмного коду основних модулів платформи наведено у додатку А.

Висновки до третього розділу

У третьому розділі описано практичну реалізацію платформи онлайн-курсів з особистим кабінетом користувача та механізмом відстеження прогресу навчання. У межах цього етапу визначено структуру вебзастосунку, розглянуто призначення основних файлів і показано логіку взаємодії між сторінками системи.

Окрему увагу приділено підключенню до бази даних MySQL. База даних забезпечує збереження інформації про користувачів, курси, уроки та результати проходження навчальних матеріалів. Завдяки цьому розроблена платформа працює не як статичний набір сторінок, а як вебзастосунок, що обробляє персональні дані користувача та відображає індивідуальний результат навчання.

Також розглянуто реалізацію реєстрації та авторизації користувача. Ці функції забезпечують створення облікового запису, вхід до системи та захист особистого кабінету від неавторизованого доступу. Після успішної авторизації користувач отримує доступ до власного навчального простору, у якому може переглядати курси, відкривати уроки й продовжувати навчання.

У розділі описано реалізацію особистого кабінету, списку курсів, сторінки окремого курсу та сторінки уроку. Для збереження результатів

проходження використовується таблиця progress, у якій фіксуються завершені уроки. На основі цих даних система обчислює відсоток проходження курсу та відображає його у зручному для користувача вигляді.

Проведене тестування показало, що основні функції платформи працюють коректно. Користувач може зареєструватися, увійти до системи, відкрити курс, перейти до уроку, позначити його як завершений і побачити оновлений прогрес. Отже, практична частина роботи відповідає поставленій задачі та демонструє роботу основних модулів платформи онлайн-курсів.

ВИСНОВКИ

Кваліфікаційна робота присвячена розробці платформи онлайн-навчання, що поєднує каталог курсів, особистий кабінет користувача та механізм контролю проходження уроків. Обрана тема є актуальною, оскільки онлайн-навчання широко використовується для самостійного здобуття знань, підвищення кваліфікації та організації дистанційного освітнього процесу. При цьому для користувача важливим є не лише доступ до навчальних матеріалів, а й можливість бачити власний результат проходження курсу.

У першому розділі проаналізовано предметну область онлайн-навчання. Було розглянуто особливості роботи вебплатформ, призначених для проходження курсів, а також наведено приклади існуючих рішень. На основі проведеного аналізу визначено основні потреби користувачів і сформовано вимоги до розроблюваної системи. Особливу увагу приділено реєстрації, авторизації, роботі особистого кабінету, перегляду курсів, проходженню уроків і відстеженню навчального прогресу.

У другому розділі виконано проєктування платформи. Обґрунтовано вибір технологій розробки: HTML5, CSS3, JavaScript, PHP та MySQL. Такий набір засобів дозволяє створити зрозумілий вебзастосунок із клієнтською частиною, серверною логікою та базою даних. Також було описано загальну архітектуру системи, структуру бази даних, логіку роботи особистого кабінету, алгоритм відстеження прогресу та основні сторінки користувацького інтерфейсу.

У третьому розділі розглянуто практичну реалізацію платформи. Було описано структуру вебзастосунку, підключення до бази даних, реалізацію реєстрації та авторизації користувача, створення особистого кабінету, відображення курсів і уроків, а також механізм збереження завершених занять. Для відстеження прогресу використано підхід, за якого система порівнює

кількість завершених уроків із загальною кількістю уроків у курсі та відображає результат у відсотках.

Під час тестування перевірено основні функції платформи: створення облікового запису, вхід користувача, захист особистого кабінету, перегляд курсів, відкриття уроків, збереження завершеного уроку та оновлення прогресу навчання. Результати перевірки показали, що основний сценарій роботи користувача виконується коректно.

Практичним результатом роботи стала вебплатформа, яка демонструє базову логіку онлайн-навчання. Користувач може зареєструватися, увійти до системи, перейти до особистого кабінету, обрати курс, переглянути уроки та побачити власний прогрес. Це підтверджує досягнення поставленої мети кваліфікаційної роботи.

Розроблена платформа має можливості для подальшого розвитку. У майбутньому її можна доповнити тестами після уроків, системою оцінювання, сертифікатами, коментарями до навчальних матеріалів і детальною статистикою проходження курсів. Водночас навіть у базовому вигляді система виконує головне завдання: організовує доступ до онлайн-курсів і дає користувачу можливість контролювати власний навчальний прогрес.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методичні вказівки до виконання кваліфікаційної роботи бакалавра за спеціальністю 123 «Комп'ютерна інженерія» усіх форм навчання / уклад.: А. І. Купін, В. А. Чубаров, І. О. Музика. Кривий Ріг : Криворізький національний університет, 2019.
2. MDN Web Docs. HTML: HyperText Markup Language. URL: https://developer.mozilla.org/en-US/docs/Web/HTML
3. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. Київ : ДП «УкрНДНЦ», 2016.
4. PHP Documentation. PHP Manual. URL: https://www.php.net/manual/en/
5. Міністерство освіти і науки України. Дистанційне навчання. URL: https://mon.gov.ua/osvita-2/pozashkilna-osvita/distantsiyne-navchannya
6. MDN Web Docs. CSS: Cascading Style Sheets. URL: https://developer.mozilla.org/en-US/docs/Web/CSS
7. Всеукраїнська школа онлайн. URL: https://lms.e-school.net.ua/
8. MySQL Documentation. MySQL Reference Manual. URL: https://dev.mysql.com/doc/refman/8.4/en/
9. Дія.Освіта. Національна едьютейнмент освітня платформа. URL: https://osvita.diia.gov.ua/
10. MDN Web Docs. JavaScript. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript
11. Prometheus. Найбільша платформа онлайн-курсів в Україні. URL: https://prometheus.org.ua/
12. PHP Documentation. password_hash. URL: https://www.php.net/manual/en/function.password-hash.php

13. Міністерство освіти і науки України. Всеукраїнська школа онлайн. URL: https://mon.gov.ua/tag/vseukrainska-shkola-onlayn
14. PHP Documentation. Sessions. URL: https://www.php.net/manual/en/book.session.php
15. СТОУ-01-07. Вимоги до оформлення студентських текстових і графічних матеріалів. Кривий Ріг : Криворізький технічний університет, 2007.
16. PHP Documentation. MySQLi. URL: https://www.php.net/manual/en/book.mysqli.php
17. MoodleDocs. Course completion. URL: https://docs.moodle.org/en/Course_completion
18. OWASP Cheat Sheet Series. Authentication Cheat Sheet. URL: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html
19. Coursera. Online Courses and Professional Certificates. URL: https://www.coursera.org/
20. OWASP Cheat Sheet Series. SQL Injection Prevention Cheat Sheet. URL: https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html

Додаток А

Фрагменти програмного коду платформи онлайн-курсів

У додатку А наведено фрагменти програмного коду, які відображають основні частини розробленої платформи онлайн-курсів. Зокрема, подано приклади підключення до бази даних, обробки реєстрації та авторизації користувача, отримання курсів і уроків, зміни статусу проходження уроку, розрахунку навчального прогресу та створення таблиць бази даних.

А.1 Фрагмент підключення до бази даних

```
<?php
// Підключення до бази даних MySQL.
$host = "localhost";
$user = "root";
$password = "";
$database = "online_courses";

$conn = mysqli_connect($host, $user, $password, $database);
if (!$conn) {
    die("Помилка підключення до бази даних: " . mysqli_connect_error());
}
mysqli_set_charset($conn, "utf8mb4");

mysqli_query($conn, "CREATE TABLE IF NOT EXISTS course_starts (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    course_id INT NOT NULL,
    started_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    UNIQUE KEY unique_user_course (user_id, course_id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci");
?>
```

А.2 Фрагмент обробки реєстрації користувача

```
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $name = trim($_POST["name"]);
    $email = trim($_POST["email"]);
    $password = $_POST["password"];
    if ($name === "" || $email === "" || $password === "") {
        $error = "Заповніть усі поля";
    } elseif (strlen($password) < 6) {
        $error = "Пароль повинен містити не менше 6 символів";
    } else {
        $check = mysqli_prepare($conn, "SELECT id FROM users WHERE email = ?");
        mysqli_stmt_bind_param($check, "s", $email);
        mysqli_stmt_execute($check);
        $result = mysqli_stmt_get_result($check);
        if (mysqli_num_rows($result) > 0) {
            $error = "Користувач з такою електронною поштою вже існує";
        } else {
            $hash = password_hash($password, PASSWORD_DEFAULT);
            $stmt = mysqli_prepare($conn, "INSERT INTO users (name, email,
password, created_at) VALUES (?, ?, ?, NOW())");
            mysqli_stmt_bind_param($stmt, "sss", $name, $email, $hash);
            if (mysqli_stmt_execute($stmt)) {
```

```
                $_SESSION['flash_success'] = "Користувача успішно зареєстровано.
Увійдіть до системи.";
                header("Location: login.php"); exit;
            } else { $error = "Помилка реєстрації"; }
        }
    }
}
```

А.3 Фрагмент обробки авторизації користувача

```
if ($_SERVER["REQUEST_METHOD"] === "POST") {
    $email = trim($_POST["email"]);
    $password = $_POST["password"];
    $stmt = mysqli_prepare($conn, "SELECT * FROM users WHERE email = ?");
    mysqli_stmt_bind_param($stmt, "s", $email);
    mysqli_stmt_execute($stmt);
    $result = mysqli_stmt_get_result($stmt);
    $user = mysqli_fetch_assoc($result);
    if ($user && password_verify($password, $user["password"])) {
        $_SESSION["user_id"] = $user["id"];
        $_SESSION["user_name"] = $user["name"];
        header("Location: dashboard.php"); exit;
    } else { $error = "Невірний email або пароль"; }
}
```

А.4 Фрагмент отримання курсів з бази даних

```
// Отримання курсів з бази даних
$sql = "SELECT * FROM courses ORDER BY FIELD(level, 'Початковий', 'Середній',
'Просунутий'), id ASC";
$result = mysqli_query($conn, $sql);

// Групування курсів за рівнями складності
$groups = ['Початковий' => [], 'Середній' => [], 'Просунутий' => []];
while ($course = mysqli_fetch_assoc($result)) {
    $groups[$course['level']][] = $course;
}
```

А.5 Фрагмент виведення каталогу курсів

```
<!-- Виведення курсів окремими блоками за рівнями складності -->
<?php foreach ($groups as $level => $coursesList): ?>
        <section class="level-section">
            <div class="section-head level-head">
                <div>
                    <h2><?php echo e($levelTitles[$level]); ?></h2>
                    <p><?php echo e($levelDescriptions[$level]); ?></p>
                </div>
            </div>
            <?php if (count($coursesList) === 0): ?>
                <p class="muted">За цим рівнем курси не знайдено.</p>
            <?php else: ?>
                <div class="cards-grid three">
                    <?php foreach ($coursesList as $course): ?>
                        <?php
                            // Розрахунок прогресу та стану картки курсу
```

```
                            $stats = getCourseStats($conn, $user_id,
$course['id']);
                            $button = getCourseButtonForCoursesPage($conn,
$user_id, $course['id']);
                            $state = getCourseCardState($conn, $user_id,
$course['id']);
                        ?>
                        <article class="course-card <?php echo e($state['class']);
?>">
                            <div class="course-card-top">
                                <div class="category-row">
                                    <span class="category"><?php echo
e($course['category']); ?></span>
                                    <?php if ($state['label']): ?><span
class="<?php echo e($state['label_class']); ?>"><?php echo e($state['label']);
?></span>
                                </div>
                                <h3><?php echo e($course['title']); ?></h3>
                                <p class="course-description"><?php echo
e($course['description']); ?></p>
                            </div>
                            <div class="course-card-bottom">
                                <div class="meta">
                                    <span><?php echo e($course['level']); ?></span>
                                    <span><?php echo $stats['total']; ?>
уроків</span>
                                </div>
                                <?php if ($user_id): ?>
                                    <div class="progress-block">
                                        <p>Прогрес: <?php echo $stats['percent'];
?>%</p>
                                        <div class="progress-line"><div
class="progress-fill" style="width: <?php echo $stats['percent']; ?>%;"></
...
```

A.6 Фрагмент завантаження даних уроку

```
// Отримання ідентифікатора уроку з адреси сторінки
$lesson_id = (int)$_GET['id'];
$stmt = mysqli_prepare($conn, "SELECT l.*, c.title AS course_title FROM lessons l
JOIN courses c ON l.course_id = c.id WHERE l.id = ?");
mysqli_stmt_bind_param($stmt, "i", $lesson_id);
mysqli_stmt_execute($stmt);
$lesson = mysqli_fetch_assoc(mysqli_stmt_get_result($stmt));
if (!$lesson) { header("Location: courses.php"); exit; }

$user_id = $_SESSION['user_id'];
// Фіксація, що користувач почав проходити курс
markCourseStarted($conn, $user_id, (int)$lesson['course_id']);

// Перевірка, чи вже позначений урок як пройдений
$check = mysqli_prepare($conn, "SELECT id FROM progress WHERE user_id = ? AND
lesson_id = ?");
mysqli_stmt_bind_param($check, "ii", $user_id, $lesson_id);
mysqli_stmt_execute($check);
$is_done = mysqli_num_rows(mysqli_stmt_get_result($check)) > 0;

// Пошук попереднього уроку
```

А.7 Фрагмент сторінки уроку

```
<!-- Основний блок уроку -->
    <article class="lesson-page">
        <p class="muted"><?php echo e($lesson['course_title']); ?></p>
        <h1>Урок <?php echo (int)$lesson['lesson_order']; ?>. <?php echo
e($lesson['title']); ?></h1>
        <span class="badge <?php echo $is_done ? 'done' : 'todo'; ?>"><?php echo
$is_done ? 'Пройдено' : 'Не пройдено'; ?></span>

        <!-- Теоретичний текст уроку -->
        <div class="lesson-content"><?php echo nl2br(e($lesson['content']));
?></div>

        <!-- Кнопки зміни статусу уроку та переходу далі -->
        <div class="lesson-toolbar">
            <form action="toggle_lesson.php" method="post" class="lesson-status-
form">
                <input type="hidden" name="lesson_id" value="<?php echo
$lesson['id']; ?>">
                <input type="hidden" name="course_id" value="<?php echo
$lesson['course_id']; ?>">
                <input type="hidden" name="return_to" value="lesson.php?id=<?php
echo $lesson['id']; ?>">
                <label class="check-button <?php echo $is_done ? 'checked' : '';
?>">
                    <input type="checkbox" name="completed" value="1" <?php echo
$is_done ? 'checked' : ''; ?> onchange="this.form.submit()">
                    <span class="fake-check"><?php echo $is_done ? '✓' : '';
?></span>
                    <span>Позначити як пройдений</span>
                </label>
            </form>

            <?php if ($next_id): ?>
                <a class="btn next-lesson" href="lesson.php?id=<?php echo $next_id;
?>">Наступний урок</a>
            <?php else: ?>
                <a class="btn next-lesson" href="course.php?id=<?php echo
$lesson['course_id']; ?>">Завершити курс</a>
            <?php endif; ?>
        </div>

        <div class="lesson-nav-links">
            <?php if ($prev_id): ?><a class="back-link" href="lesson.php?id=<?php
echo $prev_id; ?>">← Попередній урок</a><?php endif; ?>
            <a class="back-link" href="course.php?id=<?php echo
$lesson['course_id']; ?>">До списку уроків</a>
        </div>
...
```

А.8 Фрагмент розрахунку прогресу навчання

```
function getCourseStats($conn, $user_id, $course_id) {
    $total_stmt = mysqli_prepare($conn, "SELECT COUNT(*) AS total FROM lessons
WHERE course_id = ?");
    mysqli_stmt_bind_param($total_stmt, "i", $course_id);
    mysqli_stmt_execute($total_stmt);
    $total = (int)mysqli_fetch_assoc(mysqli_stmt_get_result($total_stmt))['total'];
```

```
    $completed = 0;
    if ($user_id) {
        $completed_stmt = mysqli_prepare($conn, "SELECT COUNT(*) AS completed FROM
progress p JOIN lessons l ON p.lesson_id = l.id WHERE p.user_id = ? AND l.cou
        mysqli_stmt_bind_param($completed_stmt, "ii", $user_id, $course_id);
        mysqli_stmt_execute($completed_stmt);
        $completed =
(int)mysqli_fetch_assoc(mysqli_stmt_get_result($completed_stmt))['completed'];
    }

    $percent = $total > 0 ? round(($completed / $total) * 100) : 0;
    return ['total' => $total, 'completed' => $completed, 'percent' => $percent];
}
```

А.9 Фрагмент пошуку наступного уроку

```
function getNextLessonId($conn, $user_id, $course_id = null) {
    if ($course_id) {
        $stmt = mysqli_prepare($conn, "SELECT l.id FROM lessons l WHERE l.course_id
= ? AND l.id NOT IN (SELECT lesson_id FROM progress WHERE user_id = ?) ORDER
        mysqli_stmt_bind_param($stmt, "ii", $course_id, $user_id);
    } else {
        $stmt = mysqli_prepare($conn, "SELECT l.id FROM lessons l JOIN courses c ON
l.course_id = c.id WHERE l.id NOT IN (SELECT lesson_id FROM progress WHERE u
        mysqli_stmt_bind_param($stmt, "i", $user_id);
    }
    mysqli_stmt_execute($stmt);
    $row = mysqli_fetch_assoc(mysqli_stmt_get_result($stmt));
    return $row ? (int)$row['id'] : null;
}
```

А.10 Фрагмент фіксації початку проходження курсу

```
function markCourseStarted($conn, $user_id, $course_id) {
    $stmt = mysqli_prepare($conn, "INSERT IGNORE INTO course_starts (user_id,
course_id, started_at) VALUES (?, ?, NOW())");
    mysqli_stmt_bind_param($stmt, "ii", $user_id, $course_id);
    mysqli_stmt_execute($stmt);
}

function getCourseButtonForCoursesPage($conn, $user_id, $course_id) {
    if (!$user_id) {
        return ['href' => 'course.php?id=' . $course_id, 'label' =>
'Детальніше...', 'class' => ''];
    }

    $stats = getCourseStats($conn, $user_id, $course_id);
    if ($stats['percent'] >= 100) {
        return ['href' => 'course.php?id=' . $course_id, 'label' => 'Переглянути
курс', 'class' => 'success'];
    }

    if (isCourseStarted($conn, $user_id, $course_id)) {
        $nextLessonId = getNextLessonId($conn, $user_id, $course_id);
        return ['href' => $nextLessonId ? 'lesson.php?id=' . $nextLessonId :
'course.php?id=' . $course_id, 'label' => 'Продовжити навчання', 'class' => 'warnin
    }
```

```
    return ['href' => 'course.php?id=' . $course_id, 'label' => 'Детальніше...',
'class' => ''];
}
```

А.11 Фрагмент зміни статусу проходження уроку

```
<?php
// Позначення уроку як пройденого або скасування позначки.
session_start();
require_once "db.php";
require_once "functions.php";
if (!isset($_SESSION['user_id'])) { header("Location: login.php"); exit; }

$user_id = $_SESSION['user_id'];
$lesson_id = (int)($_POST['lesson_id'] ?? 0);
$course_id = (int)($_POST['course_id'] ?? 0);
$return_to = $_POST['return_to'] ?? ('course.php?id=' . $course_id);

if (!$lesson_id || !$course_id) { header("Location: courses.php"); exit; }
markCourseStarted($conn, $user_id, $course_id);

// Якщо галочка встановлена — додаємо прогрес, якщо знята — видаляємо
if (isset($_POST['completed'])) {
    $check = mysqli_prepare($conn, "SELECT id FROM progress WHERE user_id = ? AND
lesson_id = ?");
    mysqli_stmt_bind_param($check, "ii", $user_id, $lesson_id);
    mysqli_stmt_execute($check);
    $result = mysqli_stmt_get_result($check);

    if (mysqli_num_rows($result) == 0) {
        $insert = mysqli_prepare($conn, "INSERT INTO progress (user_id, lesson_id,
completed_at) VALUES (?, ?, NOW())");
        mysqli_stmt_bind_param($insert, "ii", $user_id, $lesson_id);
        mysqli_stmt_execute($insert);
    }
} else {
    $delete = mysqli_prepare($conn, "DELETE FROM progress WHERE user_id = ? AND
lesson_id = ?");
    mysqli_stmt_bind_param($delete, "ii", $user_id, $lesson_id);
    mysqli_stmt_execute($delete);
}

header("Location: " . $return_to);
exit;
?>
```

А.12 Фрагмент статистики особистого кабінету

```
// Загальна статистика для особистого кабінету
$total_courses = (int)mysqli_fetch_assoc(mysqli_query($conn, "SELECT COUNT(*) AS
total FROM courses"))['total'];
$total_lessons = (int)mysqli_fetch_assoc(mysqli_query($conn, "SELECT COUNT(*) AS
total FROM lessons"))['total'];
$completed_stmt = mysqli_prepare($conn, "SELECT COUNT(*) AS completed FROM progress
WHERE user_id = ?");
mysqli_stmt_bind_param($completed_stmt, "i", $user_id);
mysqli_stmt_execute($completed_stmt);
$completed_lessons =
(int)mysqli_fetch_assoc(mysqli_stmt_get_result($completed_stmt))['completed'];
```

```
$general_progress = $total_lessons > 0 ? round(($completed_lessons /
$total_lessons) * 100) : 0;
$next_id = getNextLessonId($conn, $user_id);

// Останні курси, у яких користувач має прогрес
```

А.13 Фрагмент структури таблиць бази даних

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(150) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE courses (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT NOT NULL,
    category VARCHAR(100) NOT NULL,
    level VARCHAR(50) NOT NULL,
    duration VARCHAR(100) NOT NULL,
    is_popular TINYINT(1) NOT NULL DEFAULT 0,
    image VARCHAR(255),
    created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE lessons (
    id INT AUTO_INCREMENT PRIMARY KEY,
    course_id INT NOT NULL,
    title VARCHAR(255) NOT NULL,
    short_description VARCHAR(255) NOT NULL,
    content TEXT NOT NULL,
    lesson_order INT NOT NULL,
    FOREIGN KEY (course_id) REFERENCES courses(id) ON DELETE CASCADE
);

CREATE TABLE progress (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    lesson_id INT NOT NULL,
    completed_at DATETIME NOT NULL,
    UNIQUE KEY unique_progress (user_id, lesson_id),
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE,
    FOREIGN KEY (lesson_id) REFERENCES lessons(id) ON DELETE CASCADE
);

CREATE TABLE course_starts (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    course_id INT NOT NULL,
    started_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    UNIQUE KEY unique_user_course (user_id, course_id),
    FOREIGN KEY (user_id) REFERENCES users(id) ON DELETE CASCADE,
    FOREIGN KEY (course_id) REFERENCES courses(id) ON DELETE CASCADE
);
```