МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«ПРОЕКТУВАННЯ ТА РОЗРОБКА LMS ДЛЯ ПІДТРИМКИ ІНДИВІДУАЛЬНОГО НАВЧАННЯ МАТЕМАТИКИ»**

Виконав: студент 4 курсу, групи 6.1222
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми Комп'ютерні науки
(назва освітньої програми)
В. С. Яценко
(ініціали та прізвище)
Керівник в.о. завідувача кафедри комп'ютерних наук, доцент, к.пед.н. Пшенична О. С.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент доцент кафедри програмної інженерії, к.ф.-м.н. Кривохата А. Г.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти бакалавр

Спеціальність 122 Комп'ютерні науки
(шифр і назва)

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри комп'ютерних наук, к.пед.н., доцент

_____ Пшенична О.С.
(підпис)

" 08 " грудня 2025 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Яценку Владиславу Сергійовичу
(прізвище, ім'я та по-батькові)

1. Тема роботи Проєктування та розробка LMS для підтримки індивідуального навчання математики

керівник роботи Пшенична Олена Станіславівна, к.пед.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 02 » грудня 2025 року № 1741-с

2. Строк подання студентом роботи 04.06.2026

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Аналіз предметної області та постановка задачі.
2. Проєктування LMS для індивідуального навчання.
3. Розробка та тестування LMS.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
презентація

6. Консультанти розділів роботи

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

7. Дата видачі завдання 05.12.2025

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	16.02.2026	
2.	Збір вихідних даних.	20.02.2026	
3.	Обробка методичних та теоретичних	10.03.2026	
	джерел.		
4.	Розробка першого та другого розділу.	26.04.2026	
5.	Розробка третього розділу.	10.05.2026	
6.	Оформлення та нормоконтроль	04.06.2026	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	20.06.2026	

Студент _____ В. С. Яценко
(підпис) (ініціали та прізвище)

Керівник роботи _____ О. С. Пшенична
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г.Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Проєктування та розробка LMS для підтримки індивідуального навчання математики»: 111 с., 25 рис., 19 табл., 44 джерел, 3 додатків.

АДАПТИВНЕ НАВЧАННЯ, БАЗА ДАНИХ, ВЕБЗАСТОСУНОК, ІНДИВІДУАЛЬНЕ НАВЧАННЯ, КОРИСТУВАЦЬКИЙ ІНТЕРФЕЙС, КУРСИ, МАТЕМАТИКА, СИСТЕМА УПРАВЛІННЯ НАВЧАННЯМ, ТЕСТИ, LMS.

Об’єкт дослідження – системи управління навчанням у сфері цифрової освіти.

Предмет дослідження – процес розробки LMS для підтримки індивідуального навчання математики.

Метою роботи є проєктування та розробка LMS для підтримки індивідуального навчання математики, яка забезпечує адаптивну взаємодію між учнем і навчальною системою та формування індивідуальних освітніх траєкторій.

Методи дослідження – аналіз літератури та існуючих LMS, системний аналіз, методи проєктування програмних систем і баз даних, об’єктно-орієнтоване програмування та тестування програмного забезпечення.

У роботі проаналізовано підходи до індивідуального навчання та LMS, спроєктовано архітектуру і базу даних системи. Реалізовано основні модулі: авторизацію користувачів, управління курсами, проходження уроків і тестів, а також елементи індивідуальної навчальної траєкторії. Розроблено користувацький інтерфейс із підтримкою математичних формул. Тестування підтвердило коректну роботу системи та її ефективність для індивідуального навчання математики.

SUMMARY

Bachelor's Qualifying Thesis "Design and development of an LMS for supporting individual mathematics learning": 111 pages, 25 figures, 19 tables, 44 sources, 3 supplements.

ADAPTIVE LEARNING, DATABASE, WEB APPLICATION, INDIVIDUAL LEARNING, USER INTERFACE, COURSES, MATHEMATICS, LEARNING MANAGEMENT SYSTEM, TESTS, LMS.

Object of research – learning management systems in the field of digital education.

Subject of research – the process of developing an LMS to support individualized mathematics learning.

The purpose of the study is to design and develop a Learning Management System (LMS) for supporting individualized mathematics learning, providing adaptive interaction between students and the educational system and enabling the formation of personalized learning pathways.

Research methods include the analysis of scientific literature and existing LMS platforms, systems analysis, software and database design methods, object-oriented programming, and software testing.

The study analyzes approaches to individualized learning and LMS development, and presents the design of the system architecture and database structure. The main modules implemented include user authentication, course management, lesson and test completion, and elements of personalized learning pathways. A user interface with support for mathematical formulas was developed. Testing confirmed the correct operation of the system and its effectiveness for individualized mathematics learning.

ЗМІСТ

Завдання на кваліфікаційну роботу……..2
Реферат 4
Summary 5
Вступ..... 7
1 Аналіз предметної області та постановка задачі 9
1.1 Особливості індивідуального навчання математики 9
1.2 Сучасні LMS та їх можливості 12
1.3 Аналіз існуючих рішень LMS для підтримки індивідуального навчання математики..... 15
1.4 Постановка задачі 21
2 Проєктування LMS для індивідуального навчання..... 24
2.1 Функціональне та інформаційне проєктування..... 24
2.2 Проєктування архітектури системи 27
2.3 Проєктування бази даних..... 34
2.4 Обґрунтування вибору технологій розробки 43
3 Розробка та тестування LMS..... 46
3.1 Реалізація основного функціоналу (курси, завдання, тести, індивідуальні траєкторії)..... 46
3.2 Розробка користувацького інтерфейсу для учнів і викладачів 53
3.3 Тестування системи та оцінка ефективності системи..... 56
3.4 Інструкція користувача..... 59
Висновки 63
Перелік посилань..... 66
Додаток А Лістинг реалізації основного функціоналу 70
Додаток Б Лістинг реалізації користувацького інтерфейсу 85
Додаток В Інтерфейс розробленої LMS..... 105

ВСТУП

Сучасний розвиток освітніх технологій характеризується активним впровадженням цифрових платформ, що забезпечують персоналізацію та підвищують ефективність навчання. Особливо важливим це є у викладанні математики, де учні мають різний рівень підготовки. Попри поширення LMS, таких як Moodle, Google Classroom і Canvas, більшість із них орієнтовані на масове навчання та мають обмежені можливості адаптації до індивідуальних потреб. Це зумовлює необхідність створення спеціалізованих систем, які враховують рівень знань, темп навчання та результати учнів.

Актуальність теми кваліфікаційної роботи зумовлена необхідністю створення сучасної LMS, яка забезпечує підтримку індивідуального навчання математики шляхом адаптації навчального контенту, автоматизації контролю знань та формування персоналізованих навчальних маршрутів. Така система дозволить підвищити якість засвоєння матеріалу, зменшити навантаження на викладача та забезпечити більш ефективну організацію навчального процесу.

Метою роботи є проєктування та розробка LMS для підтримки індивідуального навчання математики, яка забезпечує адаптивну взаємодію між учнем і навчальною системою та формування індивідуальних освітніх траєкторій.

Для досягнення поставленої мети в роботі визначено такі завдання:

- проаналізувати предметну область індивідуального навчання математики та сучасні підходи до використання LMS;
- дослідити існуючі системи управління навчанням і визначити їх переваги та недоліки;
- спроєктувати архітектуру LMS та структуру бази даних;
- реалізувати основні функціональні модулі системи (курси, завдання, тести, індивідуальні траєкторії);

- провести тестування системи та оцінити ефективність її використання в начальному процесі.

Об’єктом дослідження є процес організації навчання математики з використанням інформаційних систем управління навчанням.

Предметом дослідження є методи та програмні засоби проєктування та реалізації LMS для підтримки індивідуального навчання математики.

Історія розвитку систем управління навчанням розпочалася наприкінці XX століття з появою перших електронних освітніх платформ. З часом вони еволюціонували від простих сховищ навчальних матеріалів до інтелектуальних систем, здатних аналізувати успішність учнів і адаптувати навчальний контент до їхніх потреб.

Основними результатами виконання роботи є розроблена архітектура LMS, спроєктована база даних, реалізований програмний продукт, що підтримує функції створення курсів, тестування, виконання завдань та формування індивідуальних навчальних траєкторій. Запропонована система дозволяє автоматизувати процес навчання та забезпечує більш гнучкий підхід до організації освітнього процесу.

Результати роботи можуть бути використані у таких напрямах: використання як основи для подальшого розвитку адаптивних освітніх платформ; інтеграція з існуючими LMS для розширення їх функціональності; застосування у дослідженнях у сфері освітніх технологій та штучного інтелекту.

Тобто розробка LMS для підтримки індивідуального навчання математики є актуальним і перспективним напрямом, що поєднує сучасні досягнення інформаційних технологій та педагогіки з метою підвищення ефективності освітнього процесу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Особливості індивідуального навчання математики

Математика належить до навчальних дисциплін, засвоєння яких значною мірою залежить від індивідуальних особливостей учня, зокрема рівня попередньої підготовки, темпу навчання та здатності до логічного мислення. У зв'язку з цим зростає потреба в організації освітнього процесу, що враховує індивідуальні потреби здобувачів освіти та створює умови для максимально ефективного засвоєння математичних знань. Індивідуальне навчання дозволяє забезпечити гнучку адаптацію змісту, методів і темпу навчання до особливостей кожного учня, що сприяє підвищенню якості освітніх результатів.

Індивідуальне навчання математики є особистісно орієнтованим підходом до організації освітнього процесу, який передбачає врахування попереднього рівня знань, когнітивних особливостей, темпу навчання, інтересів та освітніх цілей кожного учня. Методологічною основою такої форми навчання виступає особистісно орієнтований підхід, який спрямований на розвиток індивідуального потенціалу здобувача освіти та створення умов для його самореалізації. Основна ідея індивідуального навчання полягає в адаптації навчального змісту та методів до конкретного здобувача освіти з метою підвищення ефективності засвоєння математичних знань і формування стійкої навчальної мотивації. Дослідження підтверджують, що індивідуалізоване навчання, яке ґрунтується на результатах оцінювання, забезпечує вищі навчальні досягнення порівняно з традиційними уніфікованими методами та позитивно впливає на успішність учнів різних соціальних груп [17].

Однією з ключових особливостей індивідуального навчання математики є персоналізація освітнього процесу, яка полягає у доборі завдань, методів подання матеріалу та темпу навчання відповідно до потреб конкретного учня. Такий метод дозволяє не лише враховувати індивідуальні відмінності, а й

формувати у здобувачів освіти відчуття залученості до навчального процесу та впевненість у власних математичних здібностях [1]. Безпосередньо з цим пов’язана гнучкість організації навчання, яка проявляється у можливості змінювати темп засвоєння матеріалу та форми роботи: від індивідуальної до роботи в малих групах або фронтального навчання залежно від дидактичної ситуації.

Важливою складовою індивідуального навчання є діагностичний характер оцінювання, коли навчальний процес постійно супроводжується аналізом результатів діяльності учня. Це дозволяє своєчасно виявляти прогалини в знаннях і коригувати навчальну траєкторію, підбираючи оптимальні завдання та рівень складності. Такі методи доводять свою ефективність у підвищенні навчальних досягнень і забезпечують покращення результатів у межах одного навчального року [10]. Оскільки математика має логічно послідовну та ієрархічну структуру, важливим принципом є поступове ускладнення навчального матеріалу, коли учень переходить до нових тем лише після опанування базових понять, що забезпечує міцність знань [16].

Ще однією особливістю індивідуального навчання математики є використання різних способів подання та опрацювання навчального матеріалу. Учням пропонуються альтернативні методи до розв’язання задач, зокрема алгебраїчні, графічні та числові методи, що сприяє розвитку глибшого розуміння математичних понять і формуванню гнучкого мислення [30]. У поєднанні з цим важливу роль відіграє концепція продуктивного ускладнення, яка передбачає залучення учнів до виконання складних завдань із наданням їм необхідної підтримки, що розвиває наполегливість і здатність до самостійного пошуку рішень.

Ефективність індивідуального навчання значною мірою залежить від системного використання діагностики та формувального оцінювання. Постійний моніторинг навчальних досягнень дозволяє адаптувати зміст і темп навчання відповідно до реальних потреб учня, що робить процес більш керованим і результативним [15]. В умовах сучасної освіти все більшого значення набуває

використання цифрових технологій, які забезпечують автоматизовану перевірку завдань, адаптивне підбирання вправ та аналітику навчальної діяльності, підвищуючи мотивацію та ефективність засвоєння матеріалу [16].

Разом із перевагами індивідуальне навчання має і певні обмеження. Зокрема, це підвищене навантаження на викладача, пов’язане з необхідністю розробки індивідуальних завдань і постійного супроводу кожного учня, а також ризик зниження рівня соціальної взаємодії між учнями у випадку надмірної індивідуалізації. Крім того, ефективність такого методу значною мірою залежить від якості діагностики та правильності побудови навчальної траєкторії, оскільки помилки на цьому етапі можуть призвести до неефективного засвоєння матеріалу [4].

Тобто індивідуальне навчання математики є складним і багатокомпонентним педагогічним процесом, який поєднує персоналізацію, діагностику навчальних досягнень, адаптацію змісту та використання сучасних освітніх технологій. Його ефективність визначається здатністю освітньої системи враховувати індивідуальні особливості учнів та забезпечувати гнучке управління їхньою навчальною траєкторією, що робить цей підхід особливо актуальним у контексті цифровізації освіти.

Власне, у сучасних умовах традиційні методи навчання вже не повною мірою забезпечують необхідний рівень індивідуалізації, оскільки не дозволяють оперативно реагувати на різний темп засвоєння матеріалу та освітні потреби здобувачів освіти. Саме тому все більшої актуальності набуває використання цифрових освітніх платформ і систем управління навчанням (LMS), які дають змогу автоматизувати процеси діагностики знань, підбору навчального контенту та відстеження прогресу учнів.

Отже, інтеграція індивідуального навчання з можливостями LMS є одним із найперспективніших напрямів розвитку сучасної освіти, оскільки дозволяє поєднати педагогічні принципи персоналізації з технологічними засобами їх реалізації. Це створює умови для побудови динамічних навчальних траєкторій,

підвищення мотивації учнів та забезпечення більш стабільних і вимірюваних результатів навчання математики.

1.2 Сучасні LMS та їх можливості

Система управління навчанням (LMS – Learning Management System) є цифровою освітньою платформою, яка використовується для створення навчальних курсів, їх доставки здобувачам освіти, контролю навчального прогресу та управління всім освітнім процесом [4]. Сучасні LMS підтримують персоналізацію освітнього процесу, дозволяючи адаптувати навчальні матеріали та траєкторії навчання до потреб конкретного здобувача освіти. Важливою перевагою таких систем є можливість накопичення та аналізу даних про навчальну активність користувачів, що сприяє підвищенню ефективності навчання та якості освітніх послуг [2].

Варто зазначити, що саме наявність постійного збору даних про навчальну активність є ключовою відмінністю LMS від традиційних форм організації навчання. У звичайному класичному навчанні оцінювання часто є епізодичним, тоді як LMS дозволяє фіксувати кожну взаємодію учня із матеріалом, що створює основу для подальшої персоналізації.

Однією з ключових характеристик сучасних LMS є глибока інтеграція з іншими цифровими системами. Завдяки API та готовим коннекторам вони можуть взаємодіяти з різними сервісами, формуючи єдину освітню екосистему. Зокрема, інтеграція з HR-системами дозволяє автоматично керувати користувачами та їхніми ролями, що усуває необхідність ручного введення даних. Інтеграція з CRM-платформами дає змогу запускати навчальні процеси на основі дій користувачів, а зв’язок із інструментами комунікації, такими як Slack або Microsoft Teams, забезпечує безперервне навчання без відриву від робочого середовища. Також сучасні LMS можуть підключатися до зовнішніх бібліотек навчального контенту, що розширює доступ до освітніх ресурсів [3].

Ще одним важливим аспектом є система управління користувачами та адміністрування, яка включає створення, редагування та видалення облікових записів, а також розподіл ролей і прав доступу. Сучасні платформи підтримують автоматичне зарахування користувачів на курси, що дозволяє значно спростити організацію навчального процесу. Крім того, такі системи забезпечують централізоване управління навчальними ресурсами та можуть інтегруватися з хмарними сервісами для зберігання даних [13].

Важливою тенденцією розвитку LMS є впровадження адаптивного та персоналізованого навчання. Використання алгоритмів штучного інтелекту дозволяє формувати індивідуальні навчальні траєкторії на основі результатів учня. Системи рекомендацій пропонують навчальні модулі відповідно до рівня знань і прогресу, що підвищує ефективність засвоєння матеріалу. Також реалізуються динамічні навчальні маршрути, які дозволяють учням рухатися нелінійно відповідно до власних цілей і можливостей, а компетентнісний підхід забезпечує оцінювання не лише знань, а й практичних навичок [39].

У контексті математики саме адаптивність є найбільш цінною функцією LMS, оскільки навчальний процес має логічну послідовність і кожна нова тема базується на попередній. Тому автоматичне визначення рівня підготовки учня дозволяє уникнути накопичення прогалин у знаннях.

Окрему роль відіграє мікронавчання та сучасні формати подачі контенту. Навчальні матеріали подаються у вигляді невеликих модулів, що полегшує їх засвоєння та дозволяє інтегрувати навчання у повсякденний графік користувача. Використання відео, інтерактивних вправ та коротких навчальних блоків підвищує рівень залученості та сприяє кращому запам’ятовуванню інформації. Сучасні LMS також активно використовують механізми миттєвого зворотного зв’язку та автоматизованого оцінювання.

Учні отримують результати виконання завдань у режимі реального часу, що дозволяє швидко коригувати навчальний процес. Автоматизовані системи перевірки завдань зменшують навантаження на викладача та підвищують об’єктивність оцінювання [13].

Важливою складовою сучасних LMS є аналітика навчальних даних. Платформи не лише фіксують факт проходження матеріалу, але й аналізують ефективність навчання, виявляють прогалини у знаннях та прогнозують успішність учнів. Використання аналітики навчальних даних із елементами прогнозування дозволяє завчасно визначати ризик відставання та вчасно впроваджувати коригувальні заходи. Крім того, аналіз навчальної активності допомагає вдосконалювати навчальні курси та підвищувати їх якість [4].

Окремо слід відзначити елементи гейміфікації та підвищення залученості користувачів. Використання систем балів, рівнів, досягнень і навчальних завдань у форматі «квестів» значно підвищує мотивацію здобувачів освіти. Такі механізми роблять процес навчання більш інтерактивним і сприяють довготривалому утриманню уваги.

Не менш важливим напрямом розвитку є соціальне та колаборативне навчання, яке передбачає взаємодію між учнями та викладачами через форуми, чати та спільні проєкти. Це сприяє кращому засвоєнню матеріалу через обмін досвідом і колективне розв’язання навчальних задач. Також сучасні LMS підтримують постановку навчальних цілей і моніторинг їх досягнення, що підвищує відповідальність і мотивацію учнів [39].

Порівняння сучасних і традиційних LMS показує суттєві відмінності. Якщо традиційні системи базуються на статичних курсах і обмеженій аналітиці, то сучасні LMS орієнтовані на адаптивні траєкторії навчання, розширену інтеграцію, прогнозування результатів і підтримку персоналізованого підходу. Такі системи забезпечують більш гнучке управління навчальним процесом та кращу взаємодію між усіма його учасниками.

Отже, сучасні LMS є комплексними цифровими платформами, що поєднують освітні, аналітичні та технологічні можливості. Вони забезпечують персоналізацію навчання, автоматизацію управління освітнім процесом та підвищення ефективності засвоєння знань, що робить їх ключовим інструментом у розвитку сучасної цифрової освіти та основою для створення адаптивних систем навчання, зокрема в галузі індивідуального навчання математики.

1.3 Аналіз існуючих рішень LMS для підтримки індивідуального навчання математики

Система управління навчанням є центральним компонентом сучасної цифрової освіти, що забезпечує організацію, доставку та відстеження навчального контенту. Станом на 2023 рік у світі налічується понад 73,8 мільйона користувачів LMS-платформ [1], а їхнє застосування у шкільній та університетській освіті стрімко зростає. Проте ключовою проблемою залишається забезпечення індивідуальної траєкторії навчання – особливо у такій когнітивно складній дисципліні, як математика.

Дослідники виділяють два принципово різних підходи до реалізації індивідуального навчання засобами LMS: перший ґрунтується на адаптивних алгоритмах, що підлаштовують складність завдань у режимі реального часу; другий передбачає ручне конфігурування індивідуальних маршрутів викладачем через інструменти обмеження доступу та умовного відображення контенту [34]. Обидва підходи мають свої переваги та суттєві обмеження, розглянуті нижче.

Moodle (Modular Object-Oriented Dynamic Learning Environment) – найбільш поширена у світі відкрита LMS-платформа, яка широко використовується у закладах освіти України (рис. 1.1) [11]. Вона пропонує потужний арсенал засобів для адміністрування курсів, організації групової та індивідуальної взаємодії, а також моніторингу та аналітики освітньої траєкторії кожного студента. Зокрема, Moodle підтримує умовний доступ до ресурсів, яскраво виражені інструменти тестування та плагіни для адаптивного навчання.

Для реалізації адаптивного тестування з математики у Moodle застосовується плагін Adaptive Quiz (CAT – Computer-Adaptive Testing), оновлений у квітні 2024 року [28]. Цей модуль адаптує складність питань до рівня знань студента відповідно до алгоритму Item Response Theory. Проте традиційні методи оцінювання у Moodle мають суттєві обмеження: недостатня персоналізація, слабка реактивність у реальному часі та обмежений індивідуальний супровід учня [27].

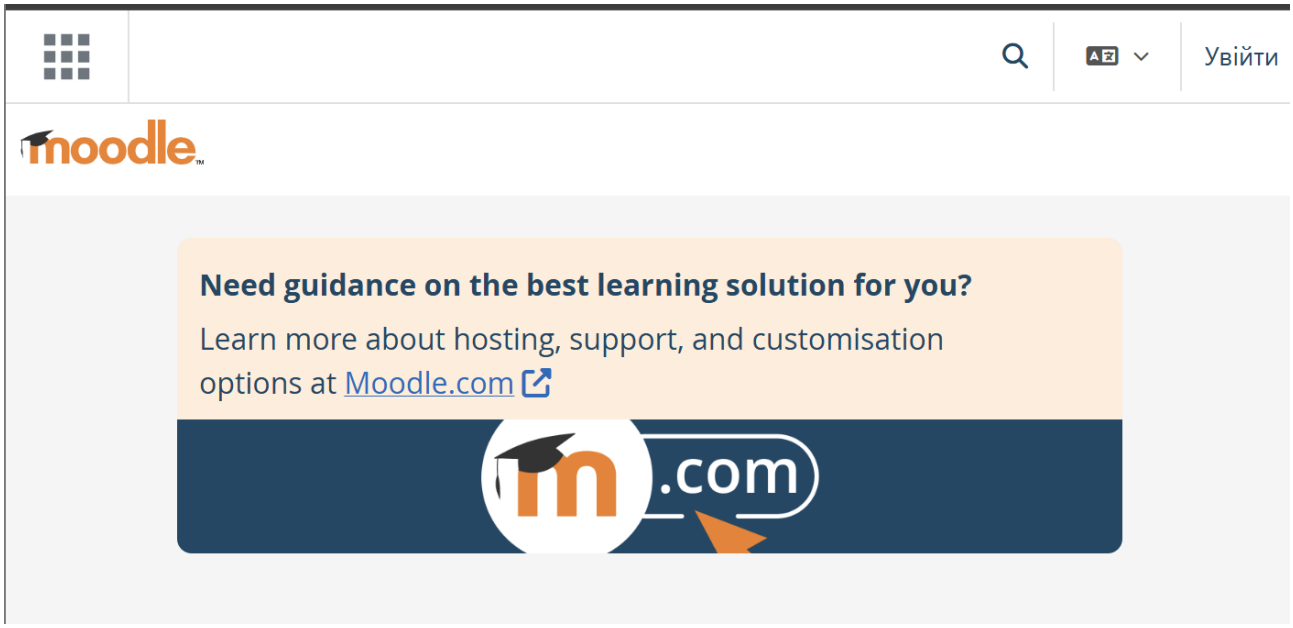


Рисунок 1.1 – Moodle

Дослідження 2024 року, проведені в межах конференції ICACTCE, пропонують вирішення цих обмежень шляхом інтеграції Big Data та методів штучного інтелекту у середовище Moodle. Запропонована методологія передбачає аналіз даних у реальному часі для миттєвого зворотного зв'язку та генерацію персоналізованих питань засобами NLP [27]. Однак така інтеграція вимагає значних технічних ресурсів і кваліфікованого персоналу, що ускладнює її широке впровадження у школах.

Khan Academy (рис. 1.2) є однією з найвідоміших освітніх платформ, орієнтованих на персоналізоване навчання математики. Платформа грунтується на принципах mastery-based pedagogy – навчання на основі досягнення майстерності – та адаптивних алгоритмах, що дозволяють кожному учню навчатися у власному темпі, заповнювати прогалини у знаннях та прискорювати засвоєння матеріалу [32]. Дослідження ефективності, проведені у 2024 році, охопили близько 350 тисяч учнів 3–8 класів і підтвердили статистично значущий позитивний вплив платформи на математичні результати.

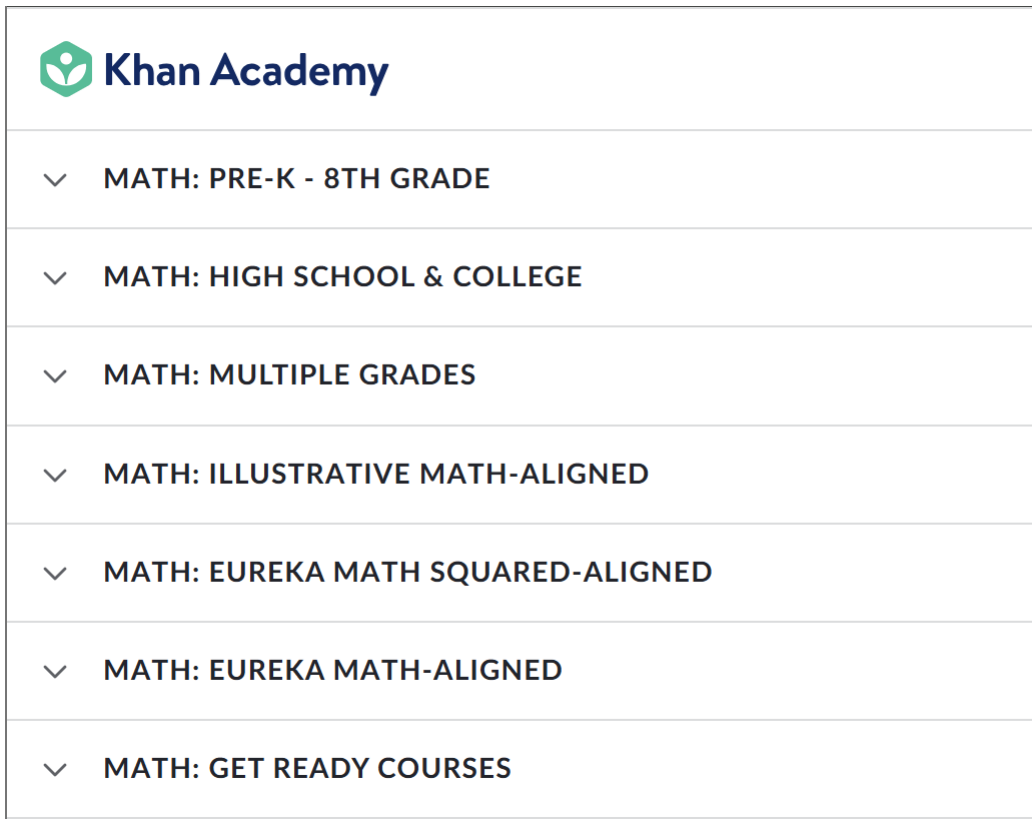


Рисунок 1.2 – Khan Academy

Окреме дослідження, проведене на базі школи Assad Abdala, зафіксувало покращення академічних показників з математики на 66% у класі, що використовував Khan Academy, та на 88% порівняно з контрольним класом [20]. Це свідчить про значний потенціал платформи як інструменту підтримки індивідуального навчання. Крім того, такі інструменти, як Khan Academy та IXL, аналізують відповіді учнів і адаптують складність завдань у реальному часі [37].

Водночас Khan Academy має і недоліки: платформа пропонує фіксований контент, обмежені можливості для вчителів щодо кастомізації курсу, а зміна підходу від студентоцентричних «Місій» до орієнтованих на вчителя завдань знизила гнучкість диференціації [41]. Також платформа не підтримує роботу зі складними математичними виразами у форматі, необхідному для вищої математики.

IXL є найпопулярнішою у світі платформою з платною підпискою для K–12, якою користуються понад 18 мільйонів учнів (рис. 1.3). Платформа охоплює понад 17 000 тем з математики, мов, природничих та соціальних наук [36]. IXL

використовує власний алгоритм SmartScore для адаптивного підбору завдань: у міру того, як Khan Academy поступово відходив від адаптивного навчання, IXL активно розвивав свої алгоритми [41]. Головним обмеженням IXL є комерційна модель – платна підписка робить платформу недоступною для частини учнів – та відносно обмежена кількість завдань творчого та дослідницького характеру.

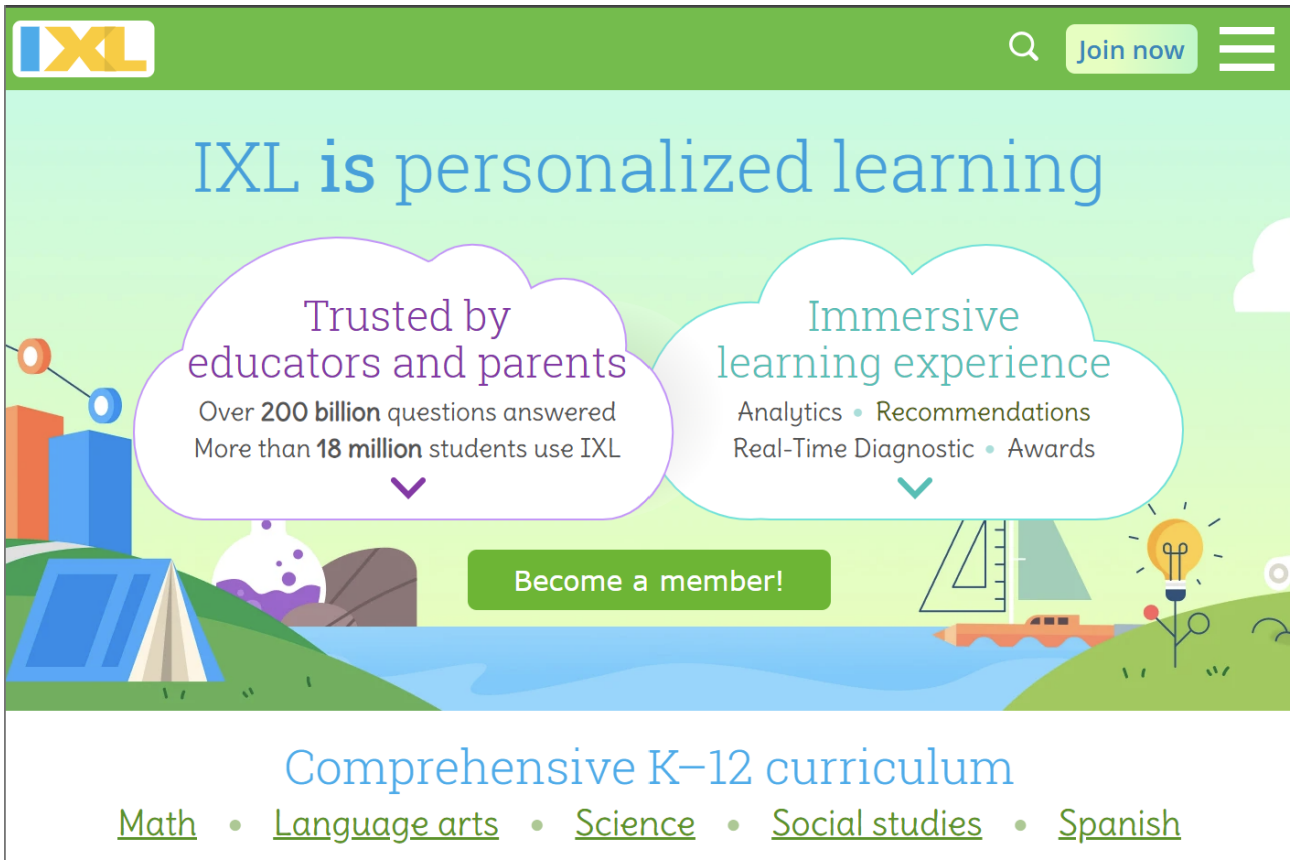


Рисунок 1.3 – IXL

DreamBox Learning (з 2023 року – у складі Discovery Education) орієнтований виключно на математику для рівнів K–8 і є одним із найбільш технологічно просунутих адаптивних рішень завдяки алгоритмам штучного інтелекту, що аналізують не лише правильність відповідей, але й стратегію розв'язання задач (рис. 1.4) [33]. Проте DreamBox має суттєве обмеження: кожне завдання містить значну інтерактивну анімацію, що сповільнює темп навчання – учень може витрачати кілька хвилин на одне питання навіть тоді, коли рівень

знань уже очевидний [41]. Крім того, платформа охоплює лише початкову та середню школу й потребує дорогої ліцензії.

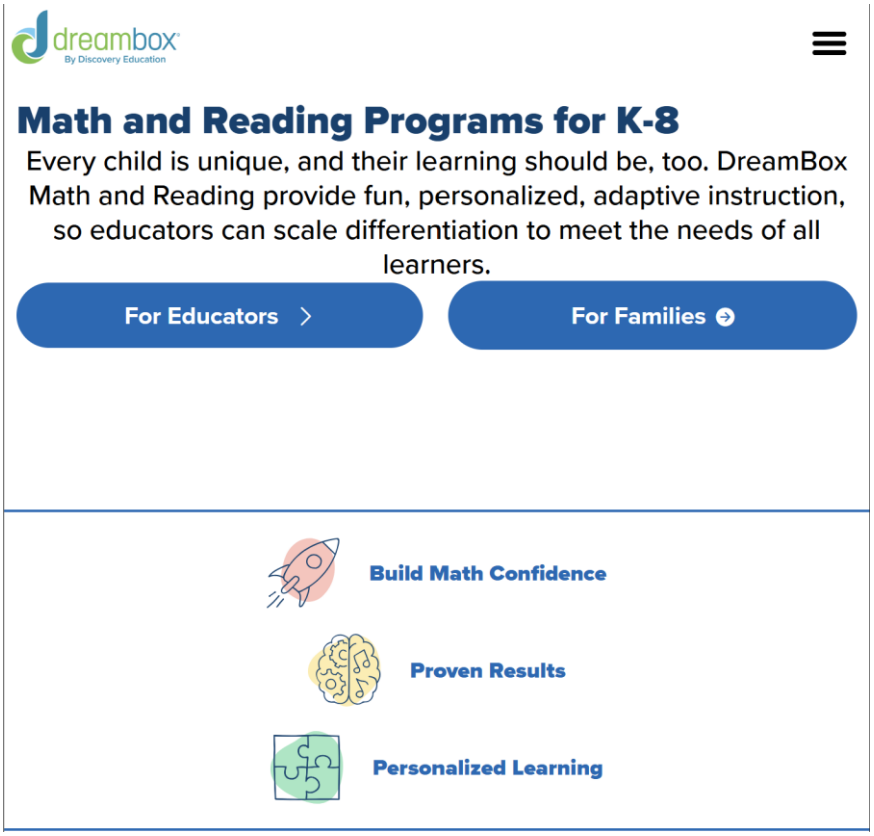


Рисунок 1.4 – DreamBox Learning

У результаті дослідження було сформовано порівняльну характеристику кожної платформи (табл. 1.1). Для більшої наочності та об’єктивності порівняння до аналізу були відібрані ключові критерії, які безпосередньо впливають на ефективність підтримки індивідуального навчання математики, зокрема рівень адаптивності системи, наявність математичної спрямованості контенту, відкритість платформи (open source) та основні функціональні обмеження.

У межах такого підходу порівняння дозволяє не лише оцінити загальні можливості LMS, але й визначити їхню придатність саме для організації персоналізованого навчання. Наприклад, платформи, що мають високий рівень адаптивності, але закриту архітектуру, обмежують можливості кастомізації, тоді як open-source рішення забезпечують більшу гнучкість, але часто потребують додаткового налаштування для реалізації адаптивних механізмів.

Таблиця 1.1 – Порівняльний аналіз LMS-платформ за ключовими критеріями підтримки індивідуального навчання математики

Платформа	Адаптивність	Фокус на математиці	Open Source	Основні обмеження
Moodle	Частково (плагіни)	Загальна	Так	Потребує плагінів; слабка вбудована адаптивність
Khan Academy	Так (mastery-based)	Математика та природничі	Ні	Обмежені інструменти для вчителів; фіксований контент
IXL	Так (SmartScore)	Математика та мови	Ні	Платна підписка; менше творчих завдань
DreamBox	Так (ШІ)	Математика K-8	Ні	Тільки K-8; дорога ліцензія; повільний темп

Проведений порівняльний аналіз дозволяє зробити висновок, що жодна з розглянутих платформ не є повністю універсальною для реалізації індивідуального навчання математики. Частина систем демонструє високий рівень адаптивності, однак має обмеження у гнучкості налаштування або доступності функціоналу для викладача. Інші, навпаки, є відкритими та гнучкими, але потребують значних доопрацювань для реалізації персоналізованих навчальних траєкторій.

Огляд у вигляді scoping review, що охопив 69 досліджень (2012–2024 рр.), встановив: найпоширенішим індикатором для активації адаптивної доставки контенту є попередні тести знань, а найбільш використовуваними адаптивними платформами залишаються McGraw-Hill Connect LearnSmart та Moodle [34]. Це підкреслює домінуючу роль Moodle у формальній освіті попри її відносно слабку вбудовану адаптивність.

Попри різноманітність існуючих LMS-рішень, аналіз виявляє низку спільних обмежень, що перешкоджають ефективній підтримці індивідуального навчання математики:

– недостатня глибина адаптивності: більшість платформ адаптують лише складність завдань, але не враховують стиль навчання, темп засвоєння чи емоційний стан учня;

- слабка підтримка математичної нотації: введення формул, рівнянь і графіків залишається технічно складним у більшості LMS;
- обмежена аналітика для вчителя: наявні інструменти звітності часто надають дані про успішність, але не дають рекомендацій щодо коригування навчальної траєкторії;
- комерційні бар'єри: найбільш технологічно досконалі рішення є платними і недоступні для широкого кола шкіл та учнів, особливо в умовах обмеженого фінансування;
- вікові обмеження: спеціалізовані математичні платформи (DreamBox, Khan Academy) орієнтовані переважно на K–8, залишаючи поза увагою старшу школу та вищу освіту.

Отже, жодна з існуючих платформ не забезпечує комплексного вирішення задачі індивідуального навчання математики для різних вікових категорій і рівнів підготовки. Це обумовлює актуальність розробки або адаптації LMS з розширеними механізмами персоналізації, орієнтованої на специфіку математичної дисципліни.

1.4 Постановка задачі

Сучасний розвиток цифрових освітніх технологій та систем управління навчанням (LMS) створює передумови для переходу від традиційної моделі навчання до персоналізованих, адаптивних освітніх середовищ. Водночас аналіз існуючих рішень показує, що більшість популярних LMS орієнтовані на масове навчання і недостатньо враховують індивідуальні особливості здобувачів освіти, особливо у такій складній дисципліні, як математика. Це зумовлює необхідність розробки спеціалізованої системи, яка б підтримувала індивідуальні навчальні траєкторії, забезпечувала адаптацію навчального контенту та автоматизувала процес оцінювання знань.

У межах даної кваліфікаційної роботи необхідно спроєктувати та реалізувати LMS для підтримки індивідуального навчання математики, яка забезпечує ефективну організацію навчального процесу, гнучке управління навчальними матеріалами та можливість персоналізації навчання для кожного користувача.

Основними функціональними вимогами до системи є забезпечення можливості створення та управління навчальними курсами, додавання теоретичних матеріалів і практичних завдань різного рівня складності, а також формування тестових завдань для контролю знань. Система повинна підтримувати реєстрацію та авторизацію користувачів із розмежуванням ролей (учень, викладач, адміністратор), а також зберігати та обробляти дані про навчальні результати користувачів.

Окрему увагу слід приділити реалізації механізму індивідуальних навчальних траєкторій, який дозволяє адаптувати послідовність вивчення матеріалу відповідно до рівня знань і результатів виконаних завдань. Такий механізм має базуватися на результатах тестування та поточному прогресі користувача, забезпечуючи поступове ускладнення навчального матеріалу або повернення до попередніх тем у разі необхідності.

Нефункціональні вимоги до системи включають забезпечення зручного та інтуїтивно зрозумілого інтерфейсу, високої продуктивності, масштабованості та надійності зберігання даних. Також система повинна бути безпечною, забезпечуючи захист персональних даних користувачів та контроль доступу до навчальних матеріалів відповідно до ролей.

З технічної точки зору система має бути побудована на основі клієнт-серверної архітектури з використанням сучасних вебтехнологій, що дозволяє забезпечити доступ до LMS через браузер без необхідності встановлення додаткового програмного забезпечення. База даних повинна забезпечувати цілісність, нормалізацію та ефективне зберігання навчальних даних, включаючи інформацію про користувачів, курси, завдання, результати тестування та навчальні траєкторії.

Отже, постановка задачі полягає у створенні веборієнтованої LMS для індивідуального навчання математики, яка поєднує функції управління навчальним контентом, адаптивного навчання та автоматизованого контролю знань, що забезпечує підвищення ефективності освітнього процесу та підтримку персоналізованого підходу до кожного здобувача освіти.

2 ПРОЄКТУВАННЯ LMS ДЛЯ ІНДИВІДУАЛЬНОГО НАВЧАННЯ

2.1 Функціональне та інформаційне проєктування

Проєктована LMS для індивідуального навчання математики орієнтована на забезпечення повноцінного освітнього процесу для трьох категорій користувачів: учнів, викладачів та адміністраторів. Система повинна підтримувати як базові навчальні функції (доступ до курсів, уроків і тестів), так і розширені механізми персоналізації навчання через відстеження прогресу та аналіз результатів.

Функціональні вимоги до системи визначають основні можливості платформи, пов’язані з автентифікацією користувачів, управлінням курсами, організацією навчального контенту та тестуванням. Особлива увага приділяється підтримці повного циклу навчання – від реєстрації користувача до отримання результатів і аналітики його навчальної діяльності.

Нижче наведено узагальнені функціональні вимоги до системи (табл. 2.1).

Таблиця 2.1 – Функціональні вимоги до системи

ID	Категорія	Вимога	Пріоритет
FR-01	Автентифікація	Реєстрація нових користувачів з валідацією даних	Високий
FR-02	Автентифікація	Вхід в систему через email і пароль	Високий
FR-03	Безпека	Використання JWT для збереження сесій	Високий
FR-04	Курси	Перегляд каталогу курсів	Високий
FR-05	Курси	Запис на курс та відстеження прогресу	Високий
FR-06	Курси	Створення та редагування курсів викладачем	Високий
FR-07	Уроки	Перегляд навчальних матеріалів з підтримкою LaTeX	Високий
FR-08	Уроки	Завершення уроку та фіксація прогресу	Середній
FR-09	Тестування	Проходження тестів різних типів	Високий
FR-10	Тестування	Автоматична перевірка відповідей	Високий
FR-11	Тестування	Перегляд результатів з поясненнями	Середній
FR-12	Аналітика	Перегляд прогресу учня	Середній
FR-13	Аналітика	Аналіз успішності учнів для викладача	Середній
FR-14	Адміністрування	Управління користувачами системи	Високий
FR-15	Адміністрування	Модерація курсів і контенту	Високий

Нефункціональні вимоги (табл. 2.2) визначають якісні характеристики системи, зокрема продуктивність, безпеку, масштабованість і надійність. Вони забезпечують стабільну роботу платформи при одночасному використанні великою кількістю користувачів.

Таблиця 2.2 – Нефункціональні вимоги до системи

ID	Категорія	Вимога	Метрика
NFR-01	Продуктивність	Час завантаження сторінки	< 3 сек
NFR-02	Продуктивність	Час відповіді API	< 500 мс
NFR-03	Безпека	Хешування паролів (bcrypt)	обов'язково
NFR-04	Безпека	Захист сесій через HTTP-only cookies	обов'язково
NFR-05	Доступність	Адаптивний інтерфейс	mobile + desktop
NFR-06	Масштабованість	Підтримка користувачів	> 100 одночасно
NFR-07	Надійність	Доступність системи	99.5%

Діаграма варіантів використання (рис. 2.1) відображає взаємодію трьох основних ролей системи з функціональними можливостями LMS: учень взаємодіє з навчальним контентом і тестами, викладач – з курсами та аналітикою, адміністратор – з управлінням системою та користувачами.

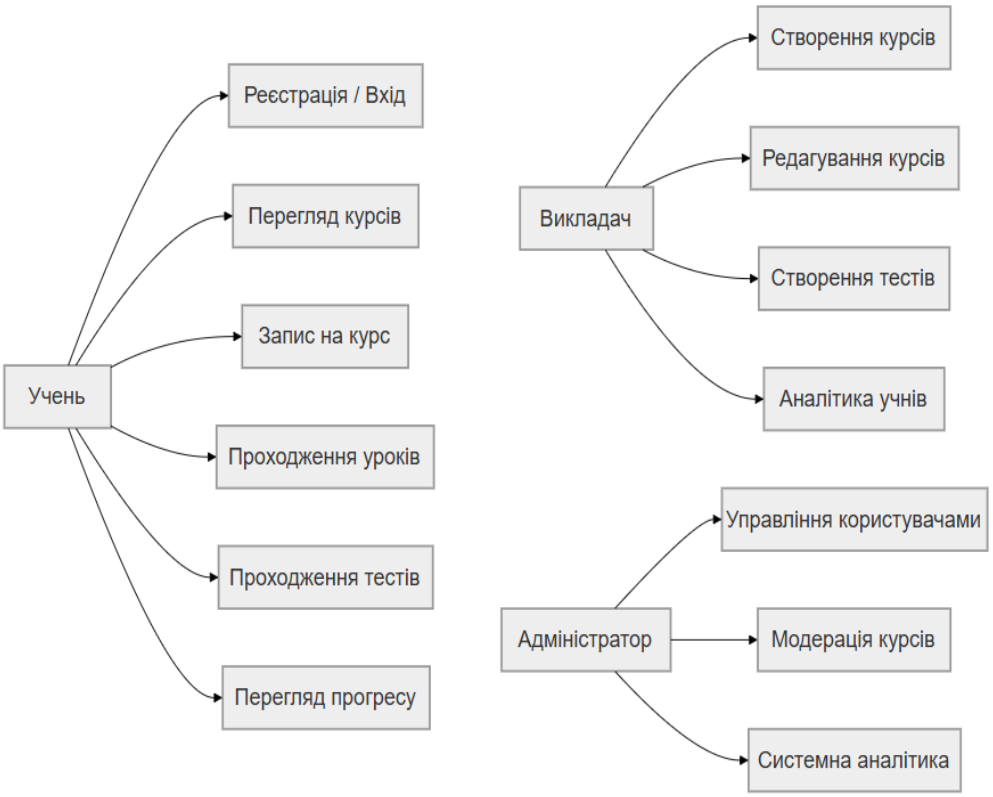


Рисунок 2.1 – Діаграма варіантів використання (Use Case)

Діаграма відображає основні функціональні можливості системи LMS для трьох типів користувачів. Учень взаємодіє з системою через навчальний процес (курси, уроки, тести, прогрес). Викладач відповідає за створення та управління навчальним контентом, а також аналіз результатів студентів. Адміністратор забезпечує загальне управління системою, користувачами та модерацією контенту.

Одним із ключових сценаріїв роботи системи є проходження тестів учнем (табл. 2.3), оскільки саме цей процес дозволяє оцінити рівень засвоєння матеріалу та формувати індивідуальну траєкторію навчання.

Таблиця 2.3 – UC6: Проходження тестів

Елемент	Опис
Назва	Проходження тестів
Актор	Учень
Передумови	Учень авторизований та записаний на курс
Основний сценарій	1. Учень відкриває тест.2. Система відображає питання.3. Учень відповідає на питання.4. Учень завершує тест.5. Система перевіряє відповіді.6. Відображаються результати з поясненнями.
Альтернативний сценарій	3а. Учень може пропустити питання.4а. Учень може повернутися до попередніх питань.
Постумови	Результати тесту збережені в базі даних і використовуються для формування прогресу учня.

Основним актором є учень, який після авторизації та зарахування на курс проходить тестування для оцінки рівня засвоєння навчального матеріалу. Основний сценарій передбачає відкриття тесту, послідовне виконання завдань, завершення тестування та автоматичну перевірку відповідей із відображенням результатів і пояснень. Альтернативні сценарії забезпечують гнучкість проходження тесту, дозволяючи пропускати питання або повертатися до попередніх. Постумови полягають у збереженні результатів у базі даних, що дає можливість надалі використовувати їх для аналізу прогресу учня та формування індивідуальної навчальної траєкторії.

Інформаційна структура системи базується на реляційній моделі даних, яка забезпечує цілісність, нормалізацію та ефективне зберігання навчальної інформації. Основними сутностями є користувачі, курси, модулі, уроки, тести та

результати тестування, що дозволяє відобразити повний цикл навчального процесу.

Користувачі системи зберігаються в таблиці users, де визначається їх роль у системі (учень, викладач або адміністратор). Курси (courses) пов’язані з викладачами та містять навчальний контент, який структурований у модулі (modules) та уроки (lessons). Кожен урок може містити тестові завдання (quizzes), які складаються з питань (quiz_questions) та варіантів відповідей (quiz_options).

Результати проходження тестів зберігаються в таблицях quiz_attempts та quiz_answers, що дозволяє детально аналізувати навчальну діяльність кожного учня. Додатково використовується таблиця user_progress, яка відображає загальний прогрес проходження уроків, та enrollments, що фіксує записи користувачів на курси.

Таким чином, інформаційна модель системи забезпечує повний зв’язок між користувачем, навчальним контентом і результатами навчання, що є основою для реалізації індивідуальних навчальних траєкторій та аналітики ефективності навчального процесу.

Сформовано основні функціональні та нефункціональні вимоги до LMS-системи, визначено ролі користувачів і ключові сценарії їх взаємодії з платформою. Описано інформаційну модель системи, яка охоплює дані про користувачів, курси, уроки, тести та результати навчання. Це створює основу для подальшої реалізації системи та забезпечує узгоджену структуру її роботи.

2.2 Проєктування архітектури системи

Архітектура системи LMS для індивідуального навчання математики побудована за трирівневою моделлю, яка забезпечує розділення відповідальностей між клієнтською частиною, серверною логікою та рівнем зберігання даних. Додатково використовується адаптований MVC-підхід, характерний для сучасних вебзастосунків на основі Next.js.

Рівні архітектури (рис. 2.2).

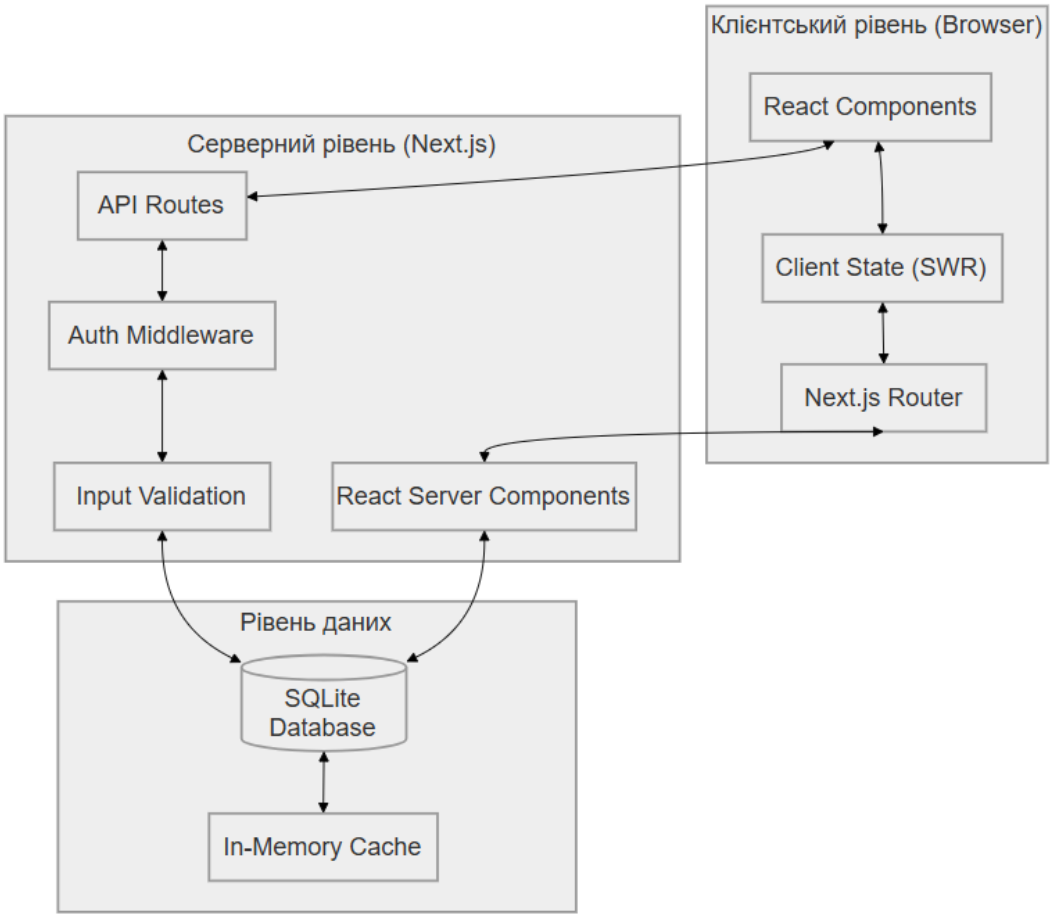


Рисунок 2.2 – Рівні архітектури

Клієнтський рівень (Client Layer) – відповідає за інтерфейс користувача, реалізований через React-компоненти. Забезпечує взаємодію учня, викладача та адміністратора з системою.

Серверний рівень (Server Layer / Next.js API Routes) – реалізує бізнес-логіку, обробку запитів, авторизацію, роботу з тестами, курсами та аналітикою.

Рівень даних (Data Layer) – представлений SQLite базою даних, що забезпечує зберігання структурованої інформації про користувачів, курси, уроки та результати тестів.

Клієнтський рівень взаємодіє із серверним через REST API запити. Серверний рівень обробляє запити, виконує перевірку доступу через middleware (JWT), після чого звертається до бази даних. Отримані дані повертаються клієнту

у форматі JSON. Для підвищення продуктивності використовується кешування стану на клієнті (SWR) та серверні React-компоненти (RSC).

Діаграма демонструє багаторівневу структуру системи, де клієнтський браузер взаємодіє з React-компонентами та маршрутизацією Next.js. Далі запити передаються до API Routes, які виконують бізнес-логіку, включаючи автентифікацію, валідацію даних та обробку навчального контенту. Серверний рівень взаємодіє з SQLite базою даних, а також використовує in-memory кеш для прискорення доступу до часто використовуваних даних.

Компонентна архітектура системи побудована за принципом модульності, що дозволяє ізолювати логічні частини застосунку та забезпечити їх повторне використання.

Основні групи компонентів:

Сервісний рівень (Services Layer) – містить бізнес-логіку (Auth Service, Courses Service, Quizzes Service).

API Layer – набір REST-ендпоінтів для взаємодії з клієнтом.

UI Components – повторно використовувані інтерфейсні елементи (форми, кнопки, картки курсів).

Math Components – спеціалізовані компоненти для відображення математичних формул (LaTeX renderer, formula editor).

Сторінки (Pages Layer) – реалізація основних сценаріїв користувача (dashboard, courses, quizzes, admin panel).

Користувацькі сторінки використовують UI-компоненти, які звертаються до API Layer. API Layer, у свою чергу, викликає сервісний рівень, що інкапсулює бізнес-логіку. Дані зберігаються в базі даних та передаються назад через API у вигляді структурованих відповідей. Окремо виділено математичний модуль, який забезпечує рендеринг та редагування формул.

Автентифікація реалізована на основі JWT-токенів із використанням HTTP-only cookies, що підвищує рівень безпеки (рис. 2.3).

Опис процесу:

- користувач вводить email і пароль у формі входу;

- запит надходить до API маршруту /api/auth/login;
- сервер перевіряє наявність користувача в базі даних;
- виконується перевірка пароля за допомогою bcrypt;
- у разі успіху генерується JWT токен;
- токен зберігається в HTTP-only cookie;
- користувач перенаправляється на dashboard;
- у випадку помилки (невірні дані) система повертає статус 401 Unauthorized та повідомлення про помилку.

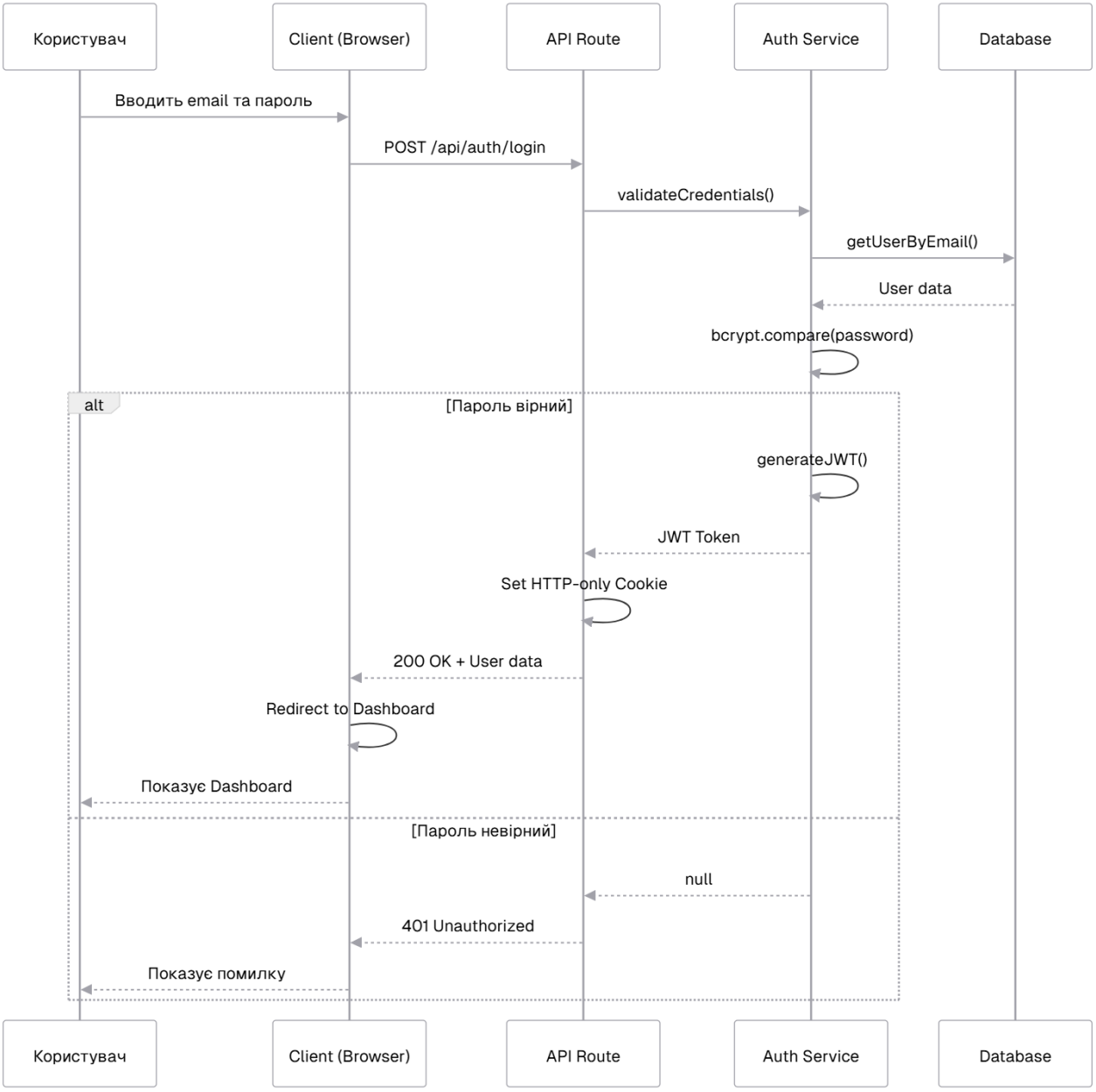


Рисунок 2.3 – Процес автентифікації користувачів

Діаграма послідовності відображає взаємодію між користувачем, клієнтським браузером, API маршрутом, сервісом автентифікації та базою даних. Показано два сценарії: успішний вхід і відмову в доступі. Успішний сценарій завершується видачею JWT токена та перенаправленням користувача, тоді як помилковий сценарій повертає відповідь з кодом 401.

Процес тестування реалізує повний цикл взаємодії учня з системою оцінювання знань (рис. 2.4).

Опис процесу:

- учень відкриває тест через інтерфейс курсу;
- клієнт надсилає запит GET /api/quizzes/{id};
- сервер повертає тест із питаннями та варіантами відповідей;
- учень відповідає на запитання;
- після завершення тесту надсилається POST /api/quizzes/{id}/attempt;
- сервер перевіряє відповіді та обчислює результат;
- результат зберігається у таблицях quiz_attempts та quiz_answers;
- користувач отримує результати з поясненнями.

Діаграма відображає циклічну взаємодію між учнем і системою під час проходження тесту. У процесі роботи для кожного питання послідовно виконується його відображення, після чого система фіксує відповідь користувача. Після завершення тестування сервер ініціює автоматичну перевірку відповідей, виконує підрахунок балів і формує підсумковий результат, який повертається користувачу у зручному для сприйняття вигляді.

Водночас діаграма дозволяє простежити загальну архітектурну організацію системи, яка побудована за принципом поділу на окремі логічні рівні. Такий підхід забезпечує чітке розмежування відповідальностей і підвищує гнучкість усієї LMS.

Зокрема, можна виділити основні підсистеми. Frontend Layer відповідає за користувацький інтерфейс та взаємодію з API, забезпечуючи зручну роботу учня з навчальним контентом. Backend Layer реалізує основну бізнес-логіку системи

та обробку запитів, координуючи роботу всіх компонентів. Окремо виділено Auth Service, який забезпечує процеси автентифікації та авторизації користувачів. API Routes виступають проміжним шаром, через який здійснюється доступ до функцій курсів, уроків, тестів та адміністративних можливостей. Усі дані системи зберігаються в Database Layer, який забезпечує надійне та структуроване збереження інформації.

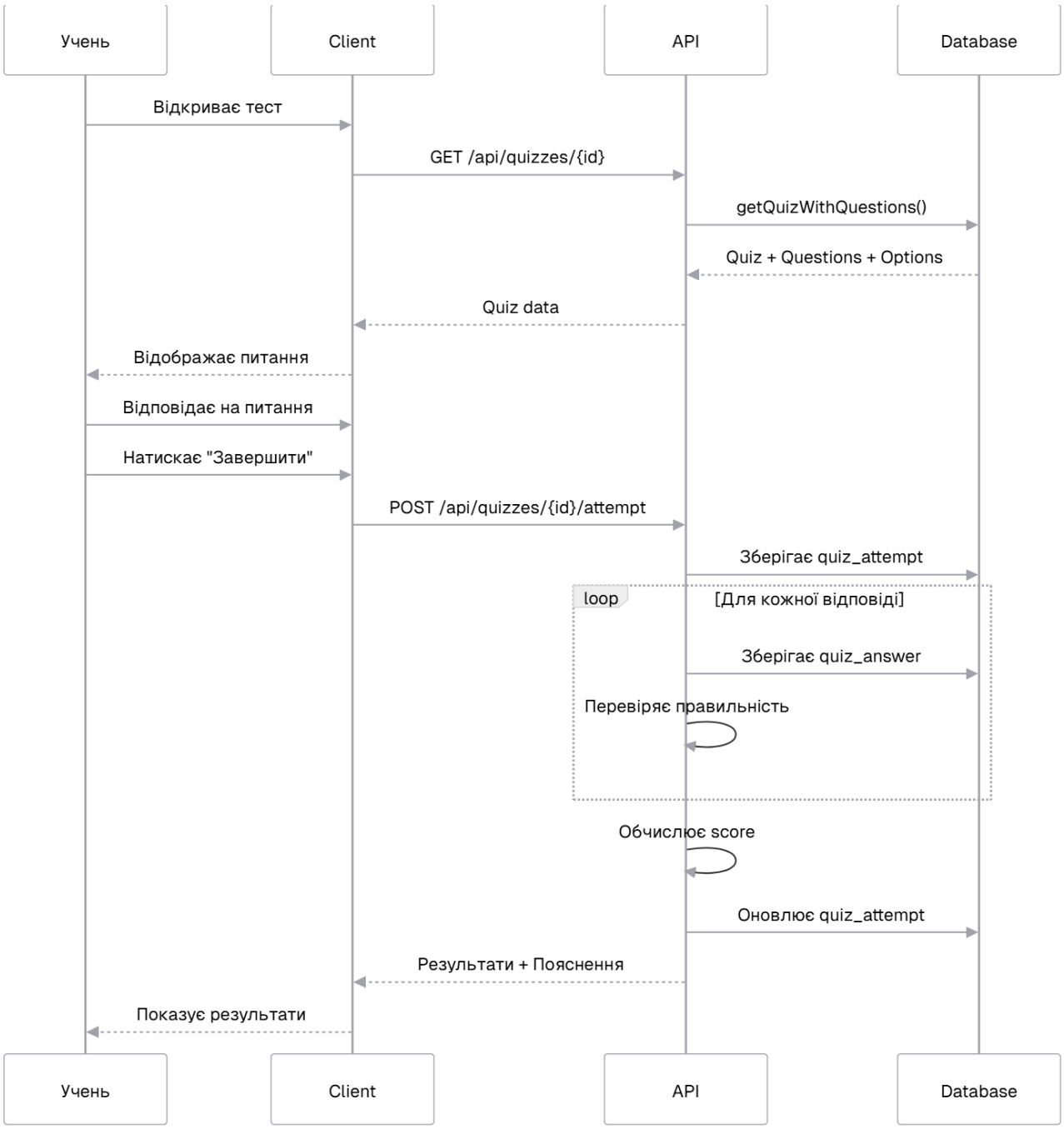


Рисунок 2.4 – Процес тестування

Діаграма компонентів системи (рис. 2.5) деталізує взаємодію між цими модулями. Frontend-компоненти взаємодіють із API Routes, які, у свою чергу, передають запити до відповідних сервісів бізнес-логіки. Сервіси опрацьовують отримані дані та взаємодіють із базою даних, забезпечуючи повний цикл обробки запитів. Окрему роль відіграють модулі автентифікації, а також математичні компоненти, зокрема підтримка LaTeX та редактор формул, які інтегровані безпосередньо в навчальний процес.

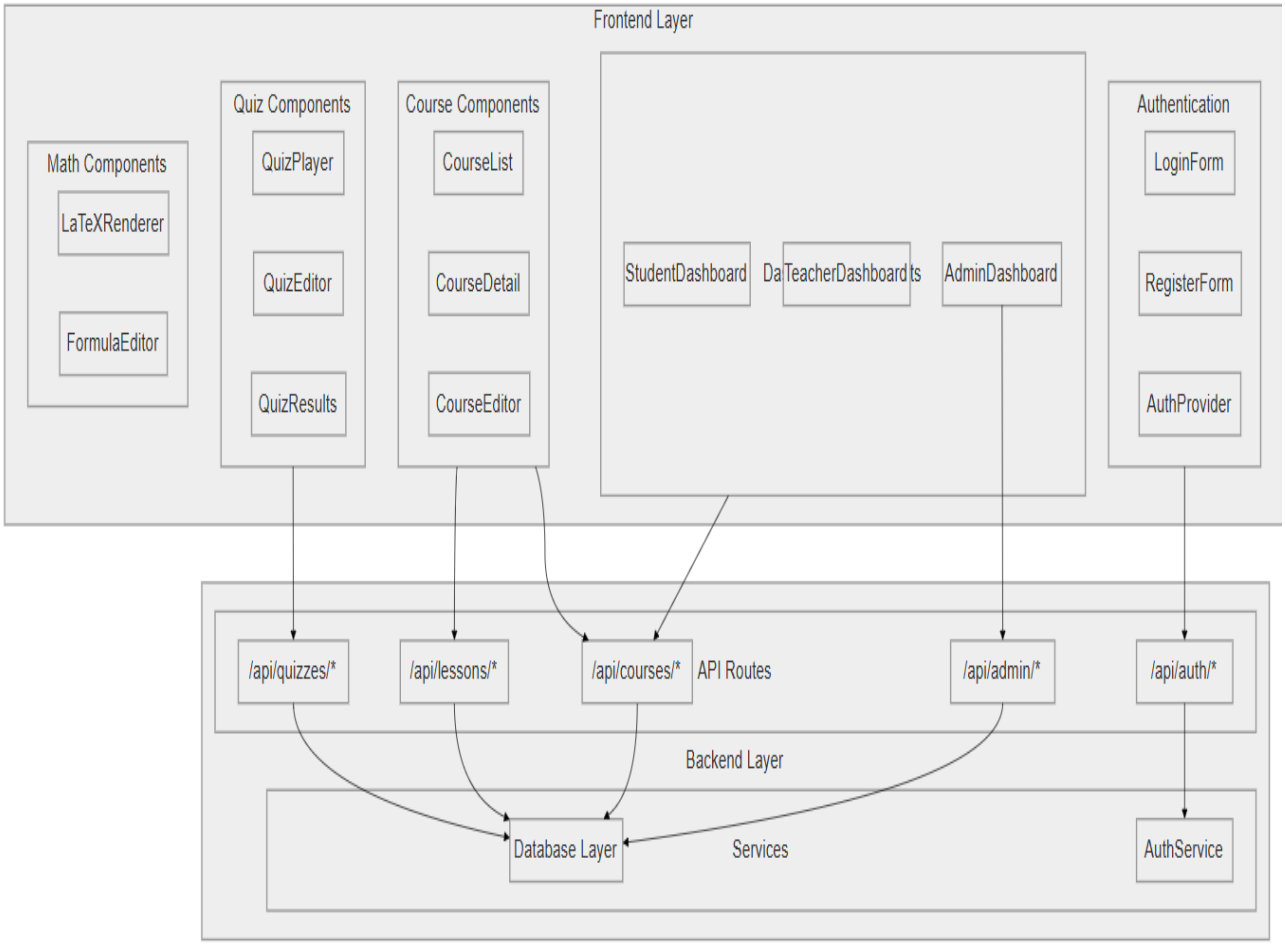


Рисунок 2.5 – Діаграма компонентів системи

Структура проєкту організована відповідно до принципів модульності та масштабованості, що дозволяє легко розширювати функціональність системи без суттєвих змін базової архітектури. Каталог app/ містить основну логіку Next.js, включаючи сторінки, маршрути та layout-компоненти. У директорії

components/ розміщені повторно використовувані UI-компоненти, що забезпечує уніфікацію інтерфейсу. Папка lib/ містить бізнес-логіку, роботу з базою даних та допоміжні утиліти, тоді як public/ використовується для зберігання статичних ресурсів. Додатково варто відзначити ключові особливості архітектурного підходу. Система передбачає розподіл функціональності за ролями користувачів (student, teacher, admin), що дозволяє чітко розмежувати права доступу та сценарії використання. Логіка API ізольована в окремому шарі, що спрощує супровід і масштабування системи. Повторне використання компонентів забезпечує зменшення дублювання коду, а загальна структура дозволяє безперешкодно розширювати функціональність без порушення існуючої архітектури.

Отже, визначено основні вимоги до LMS-системи та спроєктовано її ключові функціональні модулі для учнів, викладачів і адміністраторів. Розроблено структуровану модель даних, яка охоплює курси, уроки, тести та результати навчання. У результаті сформовано трирівневу архітектуру системи, що забезпечує чіткий поділ відповідальностей між інтерфейсом, бізнес-логікою та рівнем даних, підвищуючи масштабованість, підтримуваність і загальну ефективність платформи.

2.3 Проєктування бази даних

Концептуальна модель бази даних LMS-системи для навчання математики відображає основні сутності предметної області та зв’язки між ними. Вона включає користувачів системи, навчальні курси, модулі, уроки, тести та допоміжні сутності для відстеження навчального процесу (прогрес, спроби тестування, запис на курси та сповіщення).

Основною ідеєю моделі є забезпечення повного циклу навчання: від реєстрації користувача до проходження уроків і тестів із фіксацією результатів та прогресу.

Користувачі системи можуть мати різні ролі (учень, викладач, адміністратор), що визначає їхні права доступу та функціональні можливості. Викладачі створюють курси, а учні проходять навчання та тестування.

Таблиця users (табл. 2.4) є базовою сутністю системи та зберігає інформацію про всіх користувачів LMS. Вона забезпечує автентифікацію, авторизацію та розмежування доступу через поле role. Поле email використовується як унікальний логін, а password_hash гарантує безпечне зберігання паролів. Також таблиця підтримує контроль активності акаунтів та аудит змін через часові мітки.

Таблиця 2.4 – Users

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний ідентифікатор користувача
email	TEXT	UNIQUE NOT NULL	Електронна пошта користувача
password_hash	TEXT	NOT NULL	Хеш пароля (bcrypt)
name	TEXT	NOT NULL	Повне ім’я користувача
role	TEXT	CHECK(role IN (‘student’,’teacher’,’admin’))	Роль користувача
is_active	INTEGER	DEFAULT 1	Статус активності акаунта
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата реєстрації
updated_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата оновлення

Таблиця courses (табл. 2.5) зберігає інформацію про навчальні курси. Вона пов’язана з викладачем через teacher_id, що визначає автора курсу. Поле status дозволяє керувати життєвим циклом курсу (чернетка, опублікований або архівований). Також передбачено рівень складності, що дозволяє адаптувати навчальний контент під різні категорії учнів.

Таблиця 2.5 – Courses

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний ідентифікатор курсу
title	TEXT	NOT NULL	Назва курсу
description	TEXT	NULL	Опис курсу
teacher_id	INTEGER	FOREIGN KEY REFERENCES users(id)	Автор курсу
status	TEXT	CHECK(status IN (‘draft’,’published’,’archived’))	Статус курсу
difficulty	TEXT	CHECK(difficulty IN (‘beginner’,’intermediate’,’advanced’))	Рівень складності
image_url	TEXT	NULL	Зображення курсу
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата створення
updated_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата оновлення

Таблиця modules (табл. 2.6) забезпечує структуризацію курсу на логічні тематичні блоки та дозволяє реалізувати послідовну організацію навчального процесу. Поле id є унікальним ідентифікатором модуля, а course_id встановлює зв’язок із відповідним курсом. Поля title і description містять назву та опис модуля, що допомагає структурувати навчальний матеріал, тоді як order_index визначає порядок проходження модулів і формує навчальну траєкторію.

Таблиця 2.6 – Modules

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний ідентифікатор модуля
course_id	INTEGER	FOREIGN KEY REFERENCES courses(id)	Курс
title	TEXT	NOT NULL	Назва модуля
description	TEXT	NULL	Опис модуля
order_index	INTEGER	NOT NULL	Порядок відображення
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата створення

Таблиця lessons (табл. 2.7) містить безпосередній навчальний контент і є основним носієм теоретичних матеріалів у системі. Поле id є унікальним ідентифікатором уроку, а module_id визначає його належність до конкретного модуля. Поле title задає назву уроку, а content містить теоретичний матеріал із можливістю використання математичних формул у форматі LaTeX, що є важливим для математичної предметної області. Поле video_url дозволяє

додавати мультимедійний контент, duration_minutes задає орієнтовну тривалість уроку, а order_index визначає порядок проходження матеріалу, забезпечуючи логічну послідовність навчання.

Таблиця 2.7 – Lessons

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний урок
module_id	INTEGER	FOREIGN KEY REFERENCES modules(id)	Модуль
title	TEXT	NOT NULL	Назва уроку
content	TEXT	NULL	Теоретичний матеріал (LaTeX)
video_url	TEXT	NULL	Відео
duration_minutes	INTEGER	NULL	Тривалість
order_index	INTEGER	NOT NULL	Порядок
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата створення

Таблиця quizzes (табл. 2.8) відповідає за організацію тестового контролю знань і дозволяє оцінювати рівень засвоєння матеріалу. Поле id є унікальним ідентифікатором тесту, а module_id пов’язує тест із відповідним навчальним модулем. Поля title і description визначають назву та опис тесту, тоді як passing_score задає мінімальний прохідний бал для його успішного виконання. Поле time_limit встановлює часові обмеження, що забезпечує контроль умов проходження тестування, а created_at фіксує дату створення тесту.

Таблиця 2.8 – Quizzes

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний тест
module_id	INTEGER	FOREIGN KEY REFERENCES modules(id)	Модуль
title	TEXT	NOT NULL	Назва тесту
description	TEXT	NULL	Опис
passing_score	INTEGER	DEFAULT 60	Прохідний бал
time_limit	INTEGER	NULL	Ліміт часу
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата створення

Таблиця quiz_questions (табл. 2.9) містить структуру тестових запитань і є основою системи оцінювання знань. Поле id є унікальним ідентифікатором запитання, а quiz_id встановлює зв’язок із відповідним тестом. Поле question_text містить текст запитання, question_type визначає тип відповіді (single або multiple), що дозволяє реалізувати різні формати тестових завдань. Поле points задає вагу запитання в загальній оцінці, забезпечуючи диференційований підхід до оцінювання складності, explanation використовується для пояснення правильної відповіді після завершення тесту, що має навчальну функцію, а order_index визначає порядок відображення запитань.

Таблиця 2.9 – Quiz_questions

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Питання
quiz_id	INTEGER	FOREIGN KEY REFERENCES quizzes(id)	Тест
question_text	TEXT	NOT NULL	Текст питання
question_type	TEXT	CHECK(question_type IN (‘single’,’multiple’))	Тип питання
points	INTEGER	DEFAULT 1	Бали
explanation	TEXT	NULL	Пояснення
order_index	INTEGER	NOT NULL	Порядок

Таблиця quiz_options (табл. 2.10) містить варіанти відповідей до тестових запитань і є складовою тестового модуля LMS. Вона забезпечує підтримку одиночного та множинного вибору, що дозволяє формувати різні типи тестових завдань. Поле id є унікальним ідентифікатором варіанта відповіді (первинний ключ), question_id — зовнішній ключ, який пов’язує варіант із відповідним запитанням у таблиці quiz_questions. Поле option_text містить текст відповіді, що відображається користувачу. Поле is_correct визначає правильність варіанта (0 або 1) і використовується для автоматичної перевірки відповідей, а order_index задає порядок відображення варіантів у межах питання.

Таблиця 2.10 – Quiz_options

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Варіант
question_id	INTEGER	FOREIGN KEY REFERENCES quiz_questions(id)	Питання
option_text	TEXT	NOT NULL	Текст варіанту
is_correct	INTEGER	DEFAULT 0	Правильна відповідь
order_index	INTEGER	NOT NULL	Порядок

Таблиця quiz_attempts (табл. 2.11) фіксує всі спроби проходження тестів і використовується для аналізу навчальної активності користувачів. Поле id є унікальним ідентифікатором спроби, quiz_id пов’язує її з конкретним тестом, а user_id — із користувачем. Поля score, max_score та percentage зберігають результат проходження тесту та дозволяють оцінити рівень успішності. Поле passed фіксує факт проходження тесту відповідно до прохідного порогу. Поля started_at і completed_at відображають часові характеристики проходження та дозволяють аналізувати динаміку виконання тестів.

Таблиця 2.11 – Quiz_attempts

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Спроба
quiz_id	INTEGER	FOREIGN KEY REFERENCES quizzes(id)	Тест
user_id	INTEGER	FOREIGN KEY REFERENCES users(id)	Користувач
score	INTEGER	NULL	Бал
max_score	INTEGER	NULL	Максимум
percentage	REAL	NULL	Відсоток
passed	INTEGER	NULL	Результат
started_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Початок
completed_at	TEXT	NULL	Завершення

Таблиця enrollments (табл. 2.12) відображає зарахування користувачів на курси та є зв’язуючою між учнями і курсами. Поле id є унікальним записом зарахування, user_id і course_id формують зв’язок між користувачем і курсом. Поле enrolled_at фіксує дату початку навчання, completed_at — його завершення. Поле progress відображає загальний прогрес користувача у курсі та

використовується для відстеження навчальної динаміки і формування індивідуальної траєкторії.

Таблиця 2.12 – Enrollments

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Запис
user_id	INTEGER	FOREIGN KEY REFERENCES users(id)	Учень
course_id	INTEGER	FOREIGN KEY REFERENCES courses(id)	Курс
enrolled_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата
completed_at	TEXT	NULL	Завершення
progress	REAL	DEFAULT 0	Прогрес

Таблиця user_progress (табл. 2.13) призначена для детального відстеження виконання уроків. Поле id є унікальним ідентифікатором запису, user_id визначає користувача, а lesson_id — конкретний урок. Поле completed фіксує факт проходження уроку (0 або 1), а completed_at — дату завершення, що дозволяє аналізувати послідовність і темп навчання.

Таблиця 2.13 – User_progress

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Запис
user_id	INTEGER	FOREIGN KEY REFERENCES users(id)	Користувач
lesson_id	INTEGER	FOREIGN KEY REFERENCES lessons(id)	Урок
completed	INTEGER	DEFAULT 0	Статус
completed_at	TEXT	NULL	Дата

Таблиця notifications (табл. 2.14) реалізує систему сповіщень і забезпечує взаємодію системи з користувачами. Поле id є унікальним ідентифікатором повідомлення, user_id визначає отримувача. Поля title і message містять заголовок і текст сповіщення, type використовується для класифікації подій. Поле is_read фіксує факт прочитання, а created_at — час створення повідомлення.

Таблиця 2.14 – Notification

Поле	Тип	Обмеження	Опис
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Сповіщення
user_id	INTEGER	FOREIGN KEY REFERENCES users(id)	Отримувач
title	TEXT	NOT NULL	Заголовок
message	TEXT	NOT NULL	Повідомлення
type	TEXT	NULL	Тип
is_read	INTEGER	DEFAULT 0	Прочитано
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Дата

ER-діаграма (рис. 2.6) відображає цілісну структуру бази даних LMS та взаємозв'язки між основними сутностями системи, де користувачі пов'язані з курсами через механізм зарахування, курси складаються з модулів і уроків, тестування реалізується через запитання та варіанти відповідей, а результати навчання фіксуються через спроби тестів і прогрес виконання навчальних матеріалів.

Індекси використовуються для оптимізації швидкості виконання запитів, особливо при авторизації, фільтрації курсів та отриманні прогресу користувачів (табл. 2.15). Зокрема, індексування поля email у таблиці users забезпечує швидку авторизацію, індекси за ролями та статусами курсів покращують фільтрацію даних, а складені індекси для прогресу та спроб тестів дозволяють ефективно обробляти навчальну аналітику навіть при великій кількості користувачів.

Таблиця 2.15 – Індекси

Таблиця	Індекс	Поля	Призначення
users	idx_users_email	email	Авторизація
users	idx_users_role	role	Фільтрація ролей
courses	idx_courses_teacher	teacher_id	Курси викладача
courses	idx_courses_status	status	Публікація
enrollments	idx_enroll_user	user_id	Курси користувача
enrollments	idx_enroll_course	course_id	Учні курсу
quiz_attempts	idx_attempts_user	user_id	Історія тестів
user_progress	idx_progress_user_lesson	user_id, lesson_id	Прогрес

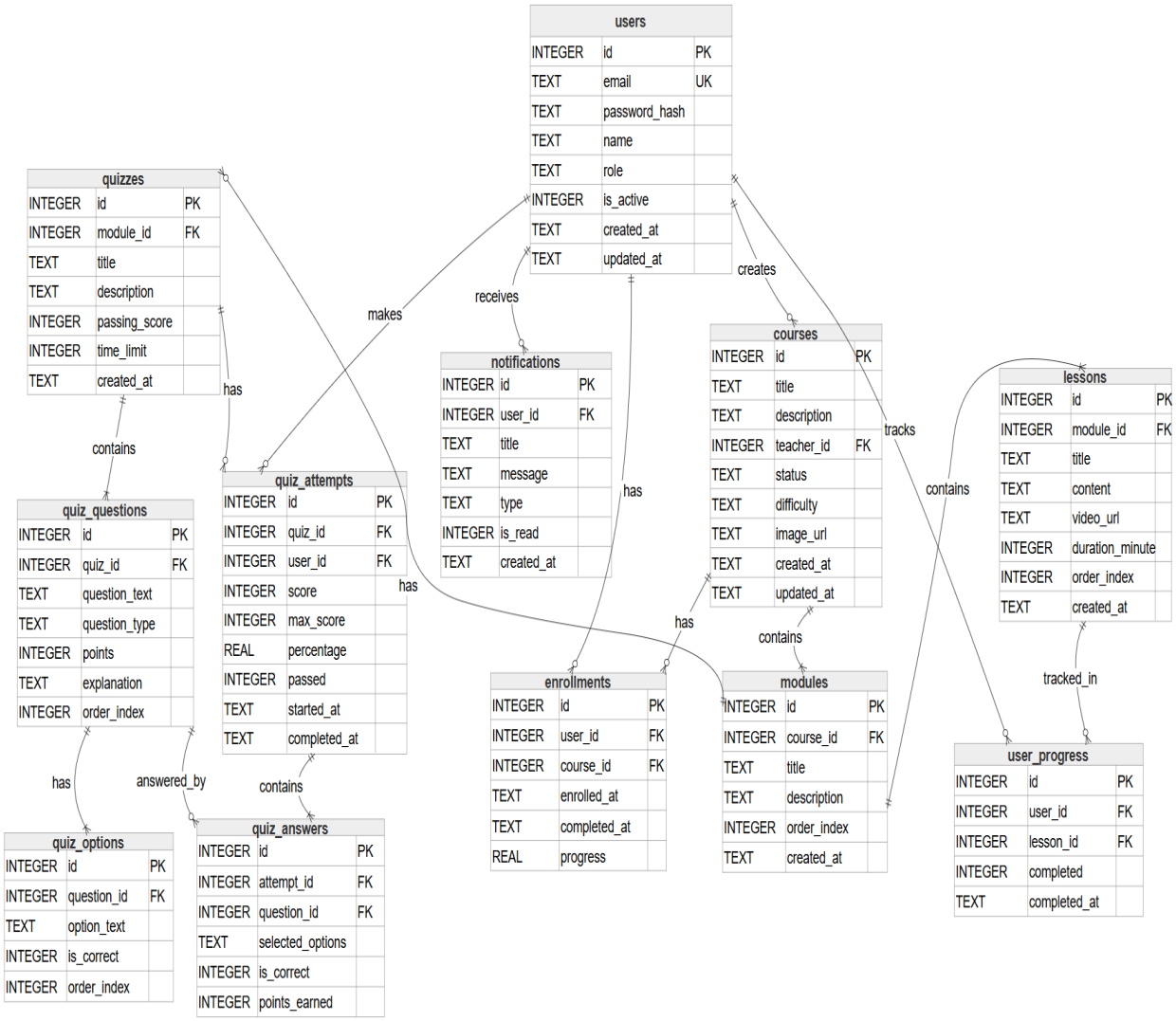


Рисунок 2.6 – ER-діаграма бази даних

Отже, спроєктована база даних забезпечує повне збереження та обробку навчальної інформації LMS-системи, включаючи користувачів, курси, навчальні матеріали, тестування та результати навчання. Використання реляційної моделі з чітко визначеними зв’язками та обмеженнями гарантує цілісність даних і коректність їх взаємодії. Запроваджені індекси підвищують продуктивність системи та забезпечують швидкий доступ до основних даних.

2.4 Обґрунтування вибору технологій розробки

Вибір технологічного стеку для розробки LMS-системи здійснювався з урахуванням вимог до продуктивності, безпеки, масштабованості, швидкості розробки та підтримки складного математичного контенту. Оскільки система орієнтована на освітній процес із використанням інтерактивних елементів і LaTeX-формул, ключовим критерієм також стала можливість коректного та швидкого рендерингу математичних виразів у вебсередовищі.

Для реалізації клієнтської частини розглядалися сучасні JavaScript-фреймворки: Next.js [31], React (CRA) [35], Vue.js [44] та Angular [7].

Create React App є базовим рішенням для React-додатків, однак має суттєві обмеження: відсутність вбудованого SSR, застарілу архітектуру та потребу в додаткових налаштуваннях для SEO та оптимізації продуктивності. Vue.js характеризується низьким порогом входу та зручністю, проте має меншу гнучкість у великих масштабованих системах та менш потужну екосистему для enterprise-рішень. Angular є повноцінним фреймворком, але має складну архітектуру, високу криву навчання та надлишкову складність для середніх освітніх платформ.

Натомість Next.js забезпечує інтегровану підтримку SSR та SSG, що критично важливо для SEO-оптимізації навчального контенту, а також дозволяє значно підвищити продуктивність завдяки server components та оптимізації рендерингу. Додатковою перевагою є вбудована маршрутизація, оптимізація зображень і тісна інтеграція з React-екосистемою.

Як backend-рішення використовується Next.js API Routes [42], що дозволяє створювати повноцінний fullstack-додаток без необхідності окремого серверного фреймворку. Альтернативами могли бути Express.js або NestJS. Express є дуже гнучким, але вимагає самостійного налаштування архітектури, безпеки та структури проєкту [14]. NestJS, хоча й є більш структурованим, додає додаткову складність і оверхед, що не є критично необхідним для MVP-версії системи [25].

Використання Next.js API Routes дозволяє уніфікувати frontend і backend в одному середовищі, зменшити складність розгортання та пришвидшити розробку.

Для зберігання даних розглядалися SQLite [40], PostgreSQL [33], MySQL [29] та MongoDB [26]. PostgreSQL є потужною enterprise-реляційною базою даних із високою масштабованістю, але потребує окремого сервера та складнішого налаштування. MySQL має подібні характеристики, але поступається PostgreSQL у розширених можливостях роботи з даними. MongoDB забезпечує гнучку NoSQL-модель, однак не гарантує повної ACID-сумісності, що критично для навчальних записів, прогресу та результатів тестування.

SQLite було обрано як оптимальне рішення для MVP, оскільки вона не потребує серверної інфраструктури, легко інтегрується в застосунок, підтримує SQL-запити та забезпечує ACID-транзакції. Це дозволяє швидко запускати систему та мінімізувати витрати на інфраструктуру з можливістю подальшої міграції на PostgreSQL при масштабуванні.

Серед UI-рішень розглядалися Material UI [21], Ant Design [8], Chakra UI [9] та shadcn/ui [38]. Material UI та Ant Design пропонують готові компоненти, однак мають великий розмір бандлу та обмежену гнучкість кастомізації. Chakra UI є більш гнучкою, але все ще додає значну кількість runtime-залежностей.

shadcn/ui обрано як оптимальне рішення, оскільки воно не є класичною бібліотекою залежностей, а надає компоненти, які копіюються безпосередньо в проєкт. Це забезпечує повний контроль над UI, мінімальний розмір бандлу, високу продуктивність і глибоку інтеграцію з Tailwind CSS та Radix UI, що особливо важливо для адаптивного навчального інтерфейсу.

Для рендерингу математичних виразів розглядалися KaTeX [19], MathJax [22] та MathLive [23]. MathJax має найширшу підтримку LaTeX, але є повільнішим і має великий розмір. MathLive орієнтований більше на інтерактивне редагування формул, що не є основною вимогою системи.

KaTeX було обрано як найбільш оптимальне рішення завдяки високій швидкості рендерингу, підтримці серверного відображення та достатньому

набору математичних символів для освітніх потреб. Це дозволяє забезпечити швидке завантаження сторінок із великою кількістю формул.

Для тестування застосунку розглядався набір інструментів Jest [18], Mocha [24] та Vitest [43]. Mocha є гнучким, але потребує додаткових налаштувань. Vitest є сучасним рішенням, однак тісно прив'язаний до Vite-екосистеми.

Jest обрано як основний інструмент тестування завдяки стабільності, широкій підтримці, простій інтеграції з TypeScript та можливості запуску тестів без складної конфігурації. У проєкті тестування виконується безпосередньо у VS Code, що спрощує процес розробки та відлагодження.

У таблиці 2.16 наведено порівняння розглянутих технологій за ключовими критеріями: продуктивність, складність впровадження, масштабованість, екосистема та відповідність вимогам LMS-системи.

Таблиця 2.16 – Порівняльний аналіз технологій

Категорія	**Розглянуті технології**	**Основні переваги**	**Основні недоліки**	**Обране рішення**
Frontend	Next.js, CRA, Vue, Angular	SSR, продуктивність, SEO	складність Angular	**Next.js**
Backend	Next.js API, Express, NestJS	простота інтеграції	обмежена гнучкість	**Next.js API Routes**
Database	SQLite, PostgreSQL, MySQL, MongoDB	простота, ACID	масштабування	**SQLite**
UI	shadcn/ui, MUI, Ant, Chakra	легкість, контроль	менше готових компонентів	**shadcn/ui**
Math Rendering	KaTeX, MathJax, MathLive	швидкість	обмежений функціонал	**KaTeX**
Testing	Jest, Mocha, Vitest	стабільність	не realtime	**Jest**

У результаті аналізу було обрано стек технологій, який забезпечує баланс між продуктивністю, простотою розробки та можливістю масштабування. Використані рішення дозволяють швидко реалізувати MVP-систему, зберігаючи можливість подальшого розширення функціоналу без суттєвих архітектурних змін.

3 РОЗРОБКА ТА ТЕСТУВАННЯ LMS

3.1 Реалізація основного функціоналу (курси, завдання, тести, індивідуальні траєкторії)

Реалізація основного функціоналу LMS-системи базується на модульному підході, де кожен функціональний блок (автентифікація, курси, навчальні матеріали, тести та прогрес навчання) реалізований як окремий логічно ізольований модуль. Це дозволяє забезпечити масштабованість системи, спрощує підтримку коду та дає можливість подальшого розширення функціоналу без суттєвих змін архітектури.

Система побудована на fullstack-підході з використанням Next.js, що дозволяє об’єднати клієнтську та серверну логіку в межах одного застосунку. Основна бізнес-логіка реалізована через API Routes, а взаємодія з користувачем здійснюється через React-компоненти з керуванням станом через Context API.

Модуль автентифікації є одним із ключових компонентів системи, оскільки забезпечує безпечний доступ до персоналізованих даних користувачів, включаючи курси, результати тестів та прогрес навчання. Реалізація базується на використанні JWT (JSON Web Token) у поєднанні з HTTP-only cookies, що дозволяє мінімізувати ризики XSS-атак та забезпечити безпечне зберігання токенів на стороні клієнта.

Процес автентифікації складається з кількох етапів:

Користувач надсилає облікові дані (email і пароль) через форму входу.

Сервер перевіряє коректність даних та виконує порівняння пароля з хешем у базі даних.

У разі успішної перевірки генерується JWT-токен, який містить ідентифікатор користувача та його роль.

Токен зберігається у HTTP-only cookie, що унеможливлює доступ до нього через JavaScript.

При кожному запиті до захищених API виконується перевірка токена та авторизація користувача.

Архітектура модуля автентифікації включає три основні рівні:

Клієнтський рівень – форма входу та React Context для зберігання стану користувача.

Серверний рівень – API endpoints для логіну, реєстрації та перевірки сесії.

Рівень безпеки – bcrypt для хешування паролів та JWT для створення токенів доступу.

Схема взаємодії показує повний цикл: від введення даних користувачем до перевірки токена сервером і повернення авторизованої сесії клієнту.

На серверній стороні використовується алгоритм bcrypt із 12 раундами сольового хешування, що забезпечує високий рівень захисту паролів навіть у разі компрометації бази даних. JWT токени генеруються з терміном дії 7 днів, що є компромісом між безпекою та зручністю користування системою (додаток А.1).

Додатково реалізовано HTTP-only cookies з параметрами:

- httpOnly – захист від доступу через JavaScript;
- secure – використання лише HTTPS у production;
- sameSite=Lax – захист від CSRF-атак;
- maxAge – автоматичне завершення сесії.

На клієнтському рівні реалізовано централізоване управління станом автентифікації через AuthProvider. Це дозволяє зберігати інформацію про користувача глобально та забезпечує доступ до неї в будь-якому компоненті системи.

При ініціалізації застосунку виконується перевірка активної сесії через запит до API /api/auth/me. Це дозволяє автоматично відновлювати стан користувача після оновлення сторінки (додаток А.2).

Модуль управління курсами є основою навчальної системи, оскільки саме він відповідає за створення, зберігання та організацію навчального контенту. Він реалізує повний CRUD-функціонал (створення, читання, оновлення та видалення курсів), а також підтримує механізми фільтрації, пошуку та аналітики.

Архітектура побудована за принципом ієрархії навчального контенту:

- курс є верхньорівневою сутністю;
- курс складається з модулів;
- модуль містить уроки та тести;
- уроки містять навчальний контент (у тому числі LaTeX);
- тести складаються з питань і варіантів відповідей.

Така структура дозволяє логічно організувати навчальний процес і забезпечує гнучкість у створенні навчальних програм різної складності.

Модуль курсів забезпечує:

- створення нових курсів викладачами;
- редагування та оновлення навчального контенту;
- публікацію та архівацію курсів;
- запис студентів на курси;
- перегляд статистики (кількість модулів, уроків, студентів);
- пошук та фільтрацію курсів за параметрами.

Особливу увагу приділено оптимізації запитів: при отриманні списку курсів додатково формується розширена інформація (кількість уроків, викладач, кількість студентів), що зменшує кількість окремих звернень до бази даних.

Серверна частина реалізована через REST API endpoints, які забезпечують взаємодію між клієнтом і базою даних (додаток А.4).

Основні операції включають:

- отримання списку курсів з фільтрацією;
- створення нового курсу з перевіркою прав доступу;
- отримання детальної інформації про курс;
- управління записами студентів.

Доступ до операцій створення та редагування курсів обмежений ролями викладача та адміністратора, що реалізовано через перевірку JWT-токена та ролі користувача.

Для зберігання даних використовується реляційна структура SQLite. При створенні курсу автоматично генерується унікальний ідентифікатор, а також

зберігаються метадані, такі як дата створення, рівень складності та статус (draft, published, archived).

Зв’язки між таблицями реалізують наступну логіку:

- один викладач може створювати багато курсів;
- курс містить багато модулів;
- кожен модуль містить уроки та тести;
- студенти можуть записуватися на багато курсів.

Система підтримує базові елементи персоналізації навчання через відстеження прогресу користувача. На основі таблиць enrollments та user_progress формується індивідуальна навчальна траєкторія, яка включає:

- прогрес проходження курсу у відсотках;
- відмітку про завершені уроки;
- результати тестів;
- історію навчальної активності.

Ці дані в подальшому можуть використовуватися для побудови адаптивних рекомендацій та персоналізованих навчальних шляхів.

Модуль уроків є одним із ключових компонентів системи, оскільки саме через нього відбувається передача навчального контенту студентам. Його особливістю є підтримка математичних формул у форматі LaTeX, що дозволяє коректно відображати складні математичні вирази, необхідні для навчального процесу (табл. 3.1).

Таблиця 3.1 – Приклади LaTeX синтаксису в системі

Тип формули	**LaTeX код**	**Результат**
Дріб	$\frac{a}{b}$	Дріб a/b
Степінь	$x^{2} + y^{2} = z^{2}$	Теорема Піфагора
Корінь	$\sqrt{x^2 + y^2}$	Квадратний корінь
Інтеграл	$$\int_{0}^{\infty} e^{-x^2} dx$$	Визначений інтеграл
Матриця	$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$	Матриця 2×2
Сума	$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$	Формула суми
Границя	$$\lim_{x \to \infty} \frac{1}{x} = 0$$	Границя функції

Для реалізації рендерингу використовується бібліотека KaTeX, яка забезпечує швидке та легке відображення формул як у блочному, так і в інлайн форматі. Важливою перевагою KaTeX є висока продуктивність, що дозволяє використовувати його без значного навантаження на клієнтську частину системи.

У системі реалізовано компонент LaTeXRenderer, який відповідає за обробку текстового контенту уроку та автоматичне перетворення LaTeX-виразів у HTML-розмітку (додаток А.5). Контент аналізується за допомогою регулярних виразів, після чого формули у форматі $$...$$ (блочні) та $...$ (інлайн) замінюються на відрендерені елементи KaTeX.

Окремо реалізовано компонент MathFormula, який використовується для рендерингу одиничних формул. Він застосовується у випадках, коли необхідно відобразити лише один математичний вираз без текстового контексту, наприклад у завданнях або поясненнях.

Тобто модуль уроків забезпечує гнучке відображення навчального матеріалу, що поєднує текстовий та математичний контент, що є критично важливим для дисциплін з математичною складовою.

У системі підтримується широкий набір математичних конструкцій LaTeX, що дозволяє викладачам створювати повноцінний навчальний матеріал без обмежень у записі формул.

Дріб ($\frac{a}{b}$) використовується для відображення раціональних виразів.

Степеневі вирази ($x^{2} + y^{2} = z^{2}$) дозволяють записувати алгебраїчні рівняння та геометричні залежності.

Корені ($\sqrt{x^2 + y^2}$) застосовуються для відображення ірраціональних виразів.

Інтеграли ($$\int_{0}^{\infty} e^{-x^2} dx$$) використовуються в математичному аналізі.

Матриці ($$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$) дозволяють працювати з лінійною алгеброю.

Суми ($$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$) використовуються для вираження числових рядів.

Границі ($$\lim_{x \to \infty} \frac{1}{x} = 0$$) застосовуються для аналізу функцій.

Це забезпечує повноцінну математичну гнучкість системи та робить її придатною для використання у вищій освіті та технічних дисциплінах.

Модуль тестування призначений для перевірки рівня засвоєння навчального матеріалу студентами (додаток А.6). Він підтримує два основні типи запитань: з однією правильною відповіддю (single choice) та з кількома правильними відповідями (multiple choice), що дозволяє формувати як прості, так і складні тестові завдання.

Процес проходження тесту реалізований як послідовність станів, що відображає повний життєвий цикл тестування – від початку до отримання результатів. Користувач переходить між станами: початковий стан, проходження тесту, перевірка відповідей та перегляд результатів. Окремо передбачено автоматичне завершення тесту у випадку закінчення часу.

Логіка станів дозволяє чітко контролювати поведінку користувача та виключає можливість некоректного проходження тестування, наприклад повторного надсилання відповідей або пропуску обов’язкових кроків.

Реалізація серверної частини тестування забезпечує повну перевірку відповідей користувача та автоматичне обчислення результатів (додаток А.7). Після отримання відповідей система виконує такі основні кроки:

- перевіряє автентифікацію користувача;
- завантажує дані тесту та відповідні питання;
- порівнює надані відповіді з правильними варіантами;
- розраховує кількість балів та відсоток виконання;
- формує детальний результат проходження тесту;
- зберігає спробу проходження в базі даних.

Особливістю реалізації є те, що перевірка відповідей виконується на сервері, що унеможливлює підміну результатів на стороні клієнта. Для кожного

питання зберігається детальна інформація про правильність відповіді, отримані бали та пояснення, що дозволяє використовувати систему не лише для оцінювання, але й для навчання.

Після завершення тесту система автоматично формує підсумкову статистику, яка включає загальну кількість питань, кількість правильних відповідей, набрані бали, максимальний бал та відсоток виконання. Також визначається факт проходження тесту залежно від прохідного балу, встановленого для конкретного тесту.

Система відстеження прогресу забезпечує автоматичний моніторинг навчальної активності користувача в межах курсу. Вона враховує завершення уроків, результати тестів та загальну активність студента.

Кожен урок після завершення позначається як виконаний і записується в таблицю прогресу користувача. На основі цих даних система автоматично розраховує відсоток проходження курсу як відношення кількості завершених уроків до загальної кількості уроків у курсі.

Окремо реалізовано механізм оновлення запису про зарахування на курс, де зберігається загальний прогрес користувача. При досягненні 100% система автоматично фіксує дату завершення курсу, що дозволяє формувати історію навчання.

Отже, створено ключові функціональні модулі системи, що забезпечують роботу з навчальним контентом, тестуванням користувачів та відстеженням їхнього навчального прогресу. Реалізовані механізми дозволяють організувати повний цикл навчання – від ознайомлення з матеріалами до перевірки знань і фіксації результатів. Особливу увагу приділено підтримці математичних виразів у форматі LaTeX, що розширює можливості подання складного навчального матеріалу. Також впроваджено систему автоматичного обчислення результатів тестування та прогресу навчання, що забезпечує більш структурований та персоналізований освітній процес.

3.2 Розробка користувацького інтерфейсу для учнів і викладачів

Розробка користувацького інтерфейсу системи спрямована на забезпечення зручної взаємодії між учнями та викладачами, а також на підтримку навчального процесу в інтуїтивно зрозумілому та візуально узгодженому середовищі. Інтерфейс побудовано за принципами сучасного UX/UI-дизайну з акцентом на простоту навігації, адаптивність та доступність.

Основою інтерфейсу є дизайн-система, реалізована з використанням бібліотек shadcn/ui та Tailwind CSS v4. Такий підхід дозволяє забезпечити єдині правила оформлення компонентів, повторне використання елементів інтерфейсу та підтримку масштабованості системи.

Кольорова схема інтерфейсу (табл. 3.2) визначає візуальну ієрархію та сприяє кращому сприйняттю інформації. Вона підтримує як світлу, так і темну тему, що підвищує зручність користування в різних умовах освітлення.

Таблиця 3.2 – Кольорова схема інтерфейсу

Призначення	CSS змінна	Світла тема	Темна тема
Фон	--background	#ffffff	#0a0a0a
Передній план	--foreground	#0a0a0a	#fafafa
Основний колір	--primary	#1d4ed8	#3b82f6
Акцент	--accent	#f1f5f9	#262626
Успіх	--success	#22c55e	#4ade80
Помилка	--destructive	#ef4444	#f87171
Попередження	--warning	#f59e0b	#fbbf24

Конфігурація глобальних стилів забезпечує єдину типографіку, кольорову модель та базові анімації інтерфейсу (додаток Б.1). Окрему увагу приділено плавним переходам та мікроанімаціям, які покращують користувацький досвід та роблять взаємодію з системою більш динамічною.

Головна сторінка виконує не лише навігаційну, але й презентаційну функцію, створюючи візуально привабливе перше враження про систему. Для цього реалізовано анімовані геометричні елементи, які формують сучасний математично-орієнтований візуальний стиль платформи.

Компонент плаваючих фігур генерує набір випадкових геометричних об’єктів, що рухаються з різною швидкістю та затримкою (додаток Б.2). Це створює ефект динамічного фону без перевантаження інтерфейсу.

Такий підхід дозволяє підкреслити освітню спрямованість платформи, особливо в контексті вивчення математики, де геометричні форми та візуалізація відіграють важливу роль у сприйнятті матеріалу.

Панель учня реалізує централізовану систему доступу до навчальних ресурсів, включаючи курси, уроки, тести та відстеження прогресу (додаток Б.3). Інтерфейс побудовано за принципом модульності, де кожен функціональний блок відображає окремий аспект навчальної діяльності.

Особливу увагу приділено відображенню статистики навчання. Учень може швидко переглядати кількість активних курсів, завершених уроків, результати тестів та середній бал. Це сприяє формуванню усвідомленого контролю власного навчального процесу.

Навігація організована через бічне меню, що забезпечує швидкий доступ до основних розділів системи. Така структура є стандартною для освітніх платформ і знижує когнітивне навантаження на користувача.

Інтерфейс викладача орієнтований на створення та управління навчальним контентом. Основним елементом є форма створення та редагування курсів (додаток Б.4), яка дозволяє задавати ключові параметри навчального матеріалу: назву, опис, рівень складності та статус публікації.

Застосування компонентів форми з єдиної дизайн-системи забезпечує узгодженість інтерфейсу та зменшує кількість помилок під час введення даних. Використання випадаючих списків для вибору параметрів спрощує процес налаштування курсу та підвищує швидкість роботи викладача.

Таким чином, панель викладача забезпечує ефективне управління навчальним контентом і підтримує повний життєвий цикл створення освітніх матеріалів.

Модуль тестування (додаток Б.5) реалізує повноцінний сценарій проходження оцінювання з підтримкою різних типів питань, таймера та

автоматичної перевірки результатів. Інтерфейс адаптується до стану тесту: початковий екран, режим проходження та екран результатів.

Під час проходження тесту користувач взаємодіє з питаннями по черзі, використовуючи радіокнопки або чекбокси залежно від типу завдання. Реалізовано підтримку математичних формул через LaTeX, що дозволяє коректно відображати складні вирази.

Окремо реалізовано систему таймера, яка автоматично завершує тест після закінчення часу. Після завершення система надає детальний розбір відповідей із поясненнями, що має важливе дидактичне значення та сприяє кращому засвоєнню матеріалу.

Інтерфейс системи повністю адаптивний і коректно відображається на пристроях різних типів – від мобільних телефонів до широкоформатних моніторів. Використання breakpoint-системи дозволяє динамічно змінювати структуру сторінок залежно від розміру екрана.

На мобільних пристроях основна увага приділяється вертикальній навігації та спрощеній структурі контенту, тоді як на десктопах доступна розширена багатоколонкова компоновка з додатковими панелями інформації.

Застосування адаптивного дизайну забезпечує однаково комфортний користувацький досвід незалежно від пристрою, що є критично важливим для сучасних вебосвітніх платформ (додаток Б.7).

Розроблений користувацький інтерфейс наведено в додатку В.

Отже, користувацький інтерфейс системи було розроблено з урахуванням потреб двох основних груп користувачів – учнів і викладачів. Запропонована дизайн-система забезпечує єдність візуального стилю, зручність навігації та адаптивність на різних пристроях. Реалізовані компоненти дозволяють ефективно виконувати ключові сценарії навчального процесу: перегляд і проходження курсів, виконання тестів, створення та редагування навчального контенту. Завдяки цьому інтерфейс сприяє інтуїтивній взаємодії з системою та підвищує загальну ефективність використання платформи.

3.3 Тестування системи та оцінка ефективності системи

Тестування розробленої LMS було спрямоване на перевірку коректності роботи основних функціональних модулів, стабільності системи та відповідності реалізованої логіки вимогам проєкту. Основний акцент зроблено на модульному тестуванні (Unit Testing), що дозволяє ізольовано перевіряти окремі функції та компоненти системи, мінімізуючи ризик появи прихованих помилок при подальшому розвитку програмного продукту. Такий підхід є доцільним для навчальних платформ, оскільки дозволяє швидко виявляти помилки в бізнес-логіці ще на ранніх етапах розробки.

Для реалізації тестування було використано фреймворк Jest у середовищі Node.js із застосуванням TypeScript, що забезпечило типізацію тестових сценаріїв та підвищило їх надійність. Тестування охоплювало критичні частини системи, зокрема модулі валідації даних користувачів, обчислення прогресу навчання, систему тестування знань, механізми авторизації та розмежування доступу, а також функціонал фільтрації курсів. Кожен із цих модулів перевірявся окремо, що дозволило забезпечити повну ізоляцію тестових сценаріїв і точність результатів.

Особлива увага приділялася перевірці коректності вхідних даних. У межах тестування валідації було перевірено обробку електронних адрес та паролів різної довжини і формату. Результати показали, що система коректно відхиляє невалідні дані та приймає лише ті значення, що відповідають встановленим правилам безпеки. Це є важливим аспектом, оскільки забезпечує захист системи від некоректного введення та потенційних помилок користувачів.

Окремо було протестовано алгоритм обчислення прогресу навчання, який визначає відсоток виконання курсу на основі завершених і загальної кількості елементів. Тестування підтвердило коректність розрахунків у різних сценаріях, включаючи повне завершення курсу, частковий прогрес та відсутність виконаних завдань. Також було враховано граничні випадки, що дозволило

уникнути можливих помилок ділення на нуль або некоректного відображення результатів.

Система тестування знань була перевірена на правильність оцінювання відповідей користувачів. Зокрема, тестувалася логіка перевірки як одиночного, так і множинного вибору відповідей, а також алгоритм визначення результату проходження тесту залежно від прохідного балу. Результати показали, що система правильно визначає успішність проходження тестів і коректно обробляє як повністю правильні, так і частково правильні відповіді.

Важливим етапом стало тестування системи авторизації та розмежування прав доступу. Перевірено коректність роботи рольової моделі, що включає користувачів з ролями учня, викладача та адміністратора. Система успішно забезпечує обмеження доступу до функцій відповідно до ролі користувача, що гарантує безпечне використання платформи та виключає можливість виконання несанкціонованих дій.

Також було протестовано функціонал фільтрації та пошуку курсів, який дозволяє користувачам швидко знаходити необхідні навчальні матеріали за назвою, рівнем складності або статусом курсу. Результати тестування підтвердили коректну роботу механізмів фільтрації та точність отриманих результатів.

Загалом виконане тестування підтвердило працездатність усіх основних модулів системи та відсутність критичних помилок у реалізованій логіці. Отримані результати свідчать про те, що LMS стабільно функціонує, коректно обробляє користувацькі запити та забезпечує надійну роботу основних освітніх процесів.

Результати тестування наведені на рис. 3.1.

Оцінка ефективності системи показала, що розроблена платформа забезпечує зручну організацію навчального процесу, підтримує індивідуальний підхід до навчання та дозволяє підвищити якість засвоєння матеріалу завдяки структурованим курсам, тестуванню та відстеженню прогресу. Таким чином,

LMS може бути ефективно використана як інструмент підтримки індивідуального навчання математики в освітньому середовищі.

```
 PASS  __tests__/utils.test.ts
  Валідація email
    √ повертає true для коректного email (58 ms)
    √ повертає false для некоректного email (16 ms)
  Валідація пароля
    √ повертає true для пароля >= 6 символів (13 ms)
    √ повертає false для короткого пароля (17 ms)
  Обчислення прогресу курсу
    √ обчислює правильний відсоток (24 ms)
    √ повертає 0 при відсутності уроків (15 ms)
  Обчислення результатів тесту
    √ обчислює результат тесту коректно (16 ms)
    √ визначає непройдений тест (10 ms)
    √ обробляє граничний випадок прохідного балу (10 ms)
  Форматування часу
    √ форматує хвилини (17 ms)
    √ форматує години (8 ms)
    √ форматує години з хвилинами (11 ms)
  Форматування дати
    √ форматує дату в українському форматі (525 ms)
  Перевірка ролей користувачів
    √ вчитель може керувати курсами (8 ms)
    √ адмін може керувати курсами (6 ms)
    √ учень не може керувати курсами (11 ms)
    √ тільки адмін може керувати користувачами (11 ms)
  Перевірка відповідей тесту
    √ правильна одинична відповідь (10 ms)
    √ неправильна одинична відповідь (8 ms)
    √ правильна множинна відповідь (8 ms)
    √ неправильна множинна відповідь - зайвий варіант (7 ms)
    √ неправильна множинна відповідь - не всі обрані (7 ms)
  Фільтрація курсів
    √ фільтрує за складністю (16 ms)
    √ фільтрує за статусом (14 ms)
    √ фільтрує за пошуком (12 ms)
    √ комбінує фільтри (18 ms)

Test Suites: 1 passed, 1 total
Tests:       26 passed, 26 total
Snapshots:   0 total
Time:        6.162 s
Ran all test suites.
```

Рисунок 3.1 – Результати тестування

3.4 Інструкція користувача

Інструкція користувача розробленої системи призначена для забезпечення цілісного розуміння принципів роботи платформи всіма категоріями користувачів та охоплює повний цикл взаємодії з навчальним середовищем – від першого входу до виконання навчальних завдань, створення контенту та адміністрування системи. Основна ідея побудови інструкції полягає в тому, щоб зробити процес використання системи максимально інтуїтивним, послідовним і доступним незалежно від рівня цифрової підготовки користувача, а також забезпечити швидке освоєння основного функціоналу без необхідності додаткового навчання.

Робота з системою починається з процесу реєстрації, який є однаковим для всіх типів користувачів. Для створення облікового запису необхідно перейти на головну сторінку платформи та заповнити реєстраційну форму, в якій зазначаються базові персональні дані, електронна адреса та пароль. Важливим елементом є вибір ролі користувача, оскільки саме вона визначає подальший функціонал системи. Після підтвердження реєстрації створюється обліковий запис, і користувач автоматично перенаправляється до відповідної панелі керування. Аналогічно, процес входу в систему передбачає введення email та пароля, після чого система виконує авторизацію та надає доступ до персоналізованого інтерфейсу, адаптованого під конкретну роль.

Після входу в систему починається безпосередня взаємодія з навчальним середовищем, яка для учня зосереджена на проходженні курсів та виконанні навчальних завдань. Учень отримує доступ до каталогу курсів, де представлені всі доступні навчальні програми, структуровані за рівнями складності та тематикою. Для полегшення пошуку передбачено механізми фільтрації та сортування, що дозволяє швидко знаходити необхідний курс. Після вибору курсу користувач може ознайомитися з його описом, структурою та змістом, після чого здійснюється запис на курс. Ця дія є ключовою, оскільки після запису курс

автоматично додається до персонального списку «Мої курси», де відбувається подальше навчання.

Навчальний процес у межах курсу організований у вигляді послідовних уроків, кожен з яких містить структурований теоретичний матеріал, приклади та, за необхідності, математичні формули, що відображаються у форматі LaTeX. Це дозволяє забезпечити коректне подання складних математичних виразів і підвищує наочність навчального матеріалу. Після опрацювання уроку учень має можливість позначити його як завершений, що автоматично впливає на оновлення прогресу в курсі. Таким чином формується прозора система відстеження навчальних досягнень, яка дозволяє користувачу контролювати власний темп навчання.

Наступним важливим етапом є проходження тестів, які призначені для перевірки засвоєних знань. Тестові завдання можуть містити питання різного типу, зокрема з однією або кількома правильними відповідями. Під час проходження тесту користувач має можливість послідовно переходити між питаннями, змінювати відповіді та контролювати свій вибір до моменту остаточного завершення тесту. У випадку наявності обмеження часу система автоматично відслідковує залишок часу, що додає елементу самодисципліни та моделює реальні умови перевірки знань. Після завершення тесту користувач отримує результат із детальним аналізом відповідей, включаючи правильні варіанти та пояснення, що сприяє кращому засвоєнню матеріалу.

Окрім безпосереднього навчального процесу, система надає учню інструменти для аналізу власного прогресу. У відповідному розділі відображається статистика навчальної діяльності, зокрема кількість завершених курсів, середній результат тестів, рівень активності та загальна динаміка навчання. Це дозволяє користувачу не лише фіксувати свій поточний стан, але й планувати подальші етапи навчання, визначаючи слабкі місця та напрями для покращення.

Функціонал викладача є більш розширеним і орієнтований на створення, редагування та управління навчальним контентом. Після входу викладач

отримує доступ до інструментів створення курсів, де він може визначати основні параметри навчальної програми, такі як назва, опис, рівень складності та статус публікації. Після створення курсу викладач формує його структуру, додаючи модулі, які поділяють матеріал на логічні частини, а також уроки, що містять детальний навчальний контент.

Особливістю системи є підтримка математичного контенту, що дозволяє викладачам використовувати LaTeX-синтаксис для відображення формул без втрати точності та читабельності. Це є особливо важливим для технічних та математичних дисциплін, де коректне подання формул безпосередньо впливає на якість навчання. Крім того, викладач може створювати тести, формуючи питання, варіанти відповідей, визначаючи правильні варіанти та додаючи пояснення, які відображаються студенту після проходження тесту.

Додатково викладач має доступ до аналітичних інструментів, які дозволяють оцінювати ефективність навчального процесу. На основі зібраної статистики можна аналізувати рівень засвоєння матеріалу студентами, складність окремих питань, а також загальну успішність групи. Це створює умови для постійного вдосконалення навчального контенту та адаптації його до потреб студентів.

Окрему роль у системі виконує адміністратор, який відповідає за загальне управління платформою. Його функціонал включає контроль користувачів, зміну ролей, активацію або деактивацію облікових записів, а також модерацію курсів. Адміністратор може перевіряти навчальні матеріали, змінювати статус курсів, а за необхідності видаляти некоректний або застарілий контент, забезпечуючи стабільність і якість навчальної платформи.

У процесі використання системи можуть виникати типові проблеми, більшість з яких пов’язані з помилками введення даних або технічними аспектами відображення контенту. Найпоширенішими є неправильний пароль при вході, некоректний LaTeX-синтаксис у формулах, а також ситуації, коли курс або тест не відображається через відповідний статус або невиконані умови

доступу. У більшості випадків ці проблеми вирішуються перевіркою введених даних, оновленням сторінки або зверненням до викладача чи адміністратора.

Отже, інструкція користувача охоплює всі ключові сценарії взаємодії з системою та забезпечує цілісне розуміння її функціонування. Вона сприяє ефективному використанню платформи, мінімізує кількість помилок і забезпечує комфортну взаємодію між усіма категоріями користувачів у межах єдиного навчального середовища.

ВИСНОВКИ

У процесі виконання роботи було досягнуто поставленої мети та послідовно реалізовано всі визначені завдання, що стосуються розробки системи індивідуального навчання математики на основі LMS-підходу. Загалом проведене дослідження та практична реалізація підтвердили актуальність створення спеціалізованих освітніх платформ, орієнтованих на персоналізацію навчального процесу, інтерактивність подачі матеріалу та автоматизацію контролю знань.

На першому етапі було здійснено аналіз предметної області індивідуального навчання математики та сучасних підходів до використання LMS. Встановлено, що цифровізація освіти активно змінює підходи до організації навчального процесу, зокрема переходом від уніфікованих програм до адаптивних моделей навчання. Особливо важливим є те, що математика як дисципліна потребує не лише теоретичного подання матеріалу, але й візуалізації, інтерактивного опрацювання формул та постійного зворотного зв’язку. Дослідження показало, що ефективне індивідуальне навчання можливе лише за умови поєднання структурованого контенту, автоматизованого оцінювання та можливості відстеження індивідуального прогресу здобувача освіти.

Другим етапом було дослідження існуючих систем управління навчанням. У результаті аналізу популярних LMS було виявлено, що такі системи, як правило, забезпечують базовий функціонал організації курсів, завантаження матеріалів, проведення тестування та комунікації між учасниками освітнього процесу. Водночас встановлено низку суттєвих обмежень: недостатній рівень персоналізації навчання, слабка підтримка складних математичних виразів у зручному форматі, обмежені можливості адаптації контенту до індивідуального рівня студента, а також недостатньо розвинені механізми аналітики навчального прогресу. Саме ці недоліки стали підставою для розробки власного рішення, орієнтованого на більш гнучку та адаптивну модель навчання.

На етапі проєктування було розроблено архітектуру LMS-системи та структуру бази даних. Архітектура побудована за модульним принципом, що дозволяє розділити систему на логічні компоненти: керування користувачами, управління курсами, модуль навчального контенту, система тестування та аналітичний модуль. Такий підхід забезпечує масштабованість системи, спрощує її супровід та дозволяє в майбутньому розширювати функціональність без суттєвого перероблення базової структури. Окрему увагу було приділено проєктуванню бази даних, яка включає взаємопов'язані сутності для збереження інформації про користувачів, курси, модулі, уроки, завдання, тести та результати їх виконання. Це дозволяє забезпечити цілісність даних та ефективну організацію навчального процесу.

Практична реалізація системи включала розробку основних функціональних модулів, які забезпечують повний цикл навчальної взаємодії. Було реалізовано функціонал створення та управління курсами, додавання навчальних матеріалів, формування тестових завдань, а також механізм проходження тестів і отримання результатів. Особливу увагу приділено підтримці індивідуальних навчальних траєкторій, що дозволяє адаптувати процес навчання відповідно до результатів користувача, його темпу навчання та рівня засвоєння матеріалу. Додатково реалізовано механізми відображення математичних формул у форматі LaTeX, що забезпечує коректне та наочне подання складних математичних виразів, що є критично важливим для даної предметної області.

Окремим важливим етапом стало тестування системи, яке дозволило перевірити коректність роботи всіх основних модулів. У процесі тестування було підтверджено стабільність функціонування платформи, правильність обробки навчальних даних, коректність роботи тестових завдань та системи підрахунку результатів. Також було оцінено зручність користувацького інтерфейсу, який забезпечує інтуїтивну навігацію для учнів, викладачів та адміністраторів. Отримані результати свідчать про те, що система є працездатною та може ефективно використовуватися в навчальному процесі.

Отже, виконана робота підтвердила доцільність розробки спеціалізованої LMS для індивідуального навчання математики. Запропоноване рішення дозволяє поєднати структуроване подання навчального матеріалу, інтерактивне тестування та можливість адаптації навчального процесу під індивідуальні потреби здобувачів освіти.

Реалізована система може слугувати основою для подальшого розвитку адаптивних освітніх платформ, а також бути розширена за рахунок впровадження елементів штучного інтелекту для більш точного персоналізованого навчання та прогнозування навчальних результатів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Глобін О. І. та ін. Компетентнісно орієнтована методика навчання математики в основній школі : метод. посіб. Київ : Педагогічна думка, 2015. 245 с.

2. Онопрієнко О. В. Дидактико-методичні засади контролю й оцінювання навчальних досягнень молодших школярів : монографія. Київ : Педагогічна думка, 2020. 400 с.

3. Спірін О. М. Системи управління навчальним контентом. *Енциклопедія освіти.* Київ : *Юрінком Інтер*, 2021. С. 919.

4. Стрілець С. І., Запорожченко Т. П. Методика навчання освітньої галузі «Математика» : навч.-метод. посіб. Чернігів : Чернігівський національний педагогічний університет імені Т. Г. Шевченка, 2014. 188 с.

5. Щербина Ю. М. Сучасні системи дистанційного навчання (LMS). URL : https://repo.snau.edu.ua/bitstream/123456789/9052/1/%D0%A9%D0%B5%D1%80%D0%B1%D0%B8%D0%BD%D0%B0%20%D0%AE.%20%D0%9C.%20%D0%A1%D1%83%D1%87%D0%B0%D1%81%D0%BD%D1%96%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8.pdf (дата звернення : 20.05.2026).

6. Andersen G. 10 Essential Features for the Future of Learning Management Systems – Unlocking Tomorrow's Educational Potential. *Custom software development company*. URL : https://moldstud.com/articles/p-10-essential-features-for-the-future-of-learning-management-systems-unlocking-tomorrows-educational-potential (date of access : 12.05.2026).

7. Angular : Overview. URL : https://angular.dev/overview (date of access : 12.05.2026).

8. Ant Design – The world’s second most popular React UI framework. URL : https://ant.design/ (date of access : 12.05.2026).

9. Chakra UI. URL : https://chakra-ui.com/ (дата звернення : 13.05.2026).

10. Connor C. M., Mazzocco M. M. M., Kurz-Thompson T. Using assessment to individualize early mathematics instruction. *Journal of School Psychology.* 2018. Vol. 66. P. 97–113. DOI : https://doi.org/10.1016/j.jsp.2017.04.005.

11. Dardak V., Shevchuk L. Personalized adaptive learning in higher education: A scoping review of key characteristics and impact on academic performance and engagement. *PMC*. 2024. URL : https://pmc.ncbi.nlm.nih.gov /articles/PMC11544060/ (date of access : 12.05.2026).

12. DreamBox Learning. Online math & reading programs for students. URL : https://www.dreambox.com/ (date of access : 12.05.2026).

13. Gerpe M. R. Beyond the traditional LMS : The modern learning platform in 2026. *Docebo*. 2026. URL : https://www.docebo.com/learning-network/blog/modern-lms-vs-traditional-lms/ (date of access : 12.05.2026).

14. Gupta N. Nest.js vs. Express.js – Which Should Aspiring Developers Choose? 2024. URL : https://www.linkedin.com/pulse/nestjs-vs-expressjs-which-should-aspiring-developers-nikhil-gupta-repne/ (date of access : 12.05.2026).

15. Hodge S. Adapted Physical Activity for Students with Special Needs. In : Elsevier eBooks. 2010. P. 518–529. DOI : https://doi.org/10.1016/b978-0-08-044894-7.01124-6.

16. Internet Archive. ERIC ED646968 : Revolutionizing Math Education : The Power of Personalized Learning. 2024. URL : https://archive.org/details/ ERIC_ED646968/mode/1up?ui=embed&wrapper=false (date of access : 12.05.2026).

17. IXL Learning. IXL | Math, Language Arts, Science, Social Studies, and Spanish. URL : https://www.ixl.com/ (date of access : 12.05.2026).

18. Jest : Getting Started. URL : https://jestjs.io/docs/getting-started (date of access : 12.05.2026).

19. KaTeX – The fastest math typesetting library for the web. URL : https://katex.org/ (date of access : 12.05.2026).

20. Khan Academy. Efficacy Results, November 2024. URL : https://blog.khanacademy.org/khan-academy-efficacy-results-november-2024/ (date of access : 12.05.2026).

21. Material UI : React components that implement Material Design. URL : https://mui.com/material-ui/ (date of access : 12.05.2026).

22. MathJax. URL : https://www.mathjax.org/ (date of access : 12.05.2026).

23. MathLive. Scientific computing for everyone. URL : https://mathlive.io/ (date of access : 12.05.2026).

24. Mocha. URL : https://mochajs.org/ (date of access : 12.05.2026).

25. Model-View-Presenter : шаблон архітектури мобільних додатків – Brander. *IT-компанія з розробки цифрових продуктів в Україні – Brander*. URL : https://brander.ua/technologies/mvp (дата звернення : 12.05.2026).

26. MongoDB. The world's leading modern data platform. URL : https://www.mongodb.com/ (date of access : 12.05.2026).

27. Moodle Plugins directory : Adaptive Quiz (CAT implementation for Moodle). URL : https://moodle.org/plugins/mod_adaptivequiz/versions (date of access : 12.05.2026).

28. Moodle в Україні: що таке Moodle. URL : https://moodle.org/mod/page/view.php?id=8174 (date of access : 12.05.2026).

29. MySQL Documentation. URL : https://dev.mysql.com/doc/ (дата звернення : 13.05.2026).

30. National Council of Teachers of Mathematics. Principles to actions : Ensuring mathematical success for all. Reston, VA : Author, 2014.

31. Next.js by Vercel – the React framework. URL : https://nextjs.org/ (date of access : 12.05.2026).

32. Ouassam E., Hmina N., Bouikhalene B. Adaptive approach for skills assessment on Moodle: use of big data and artificial intelligence. *In: Proceedings of ICACTCE* 2024. Springer, 2024. URL : https://link.springer.com/chapter/10.1007/978-3-031-94623-3_16 (date of access : 12.05.2026).

33. PostgreSQL Documentation. The PostgreSQL Global Development Group. URL : https://www.postgresql.org/docs/ (date of access : 12.05.2026).

34. Pritula Academy. Система управління навчанням (LMS) : ключові переваги та особливості. URL : https://ukr.pritula.academy/tpost/ksuxc8cpa1 (дата звернення : 20.05.2026).

35. React. URL : https://react.dev/ (date of access : 12.05.2026).

36. Room to Discover. The Top 4 Math Sites for Adaptive Learning. URL : https://roomtodiscover.com/adaptive-learning/ (date of access : 12.05.2026).

37. Santos M. et al. Adaptive learning the use of the Khan platform Academy in mathematics teaching. 2021. URL : https://ui.adsabs.harvard.edu/abs/2021FMA.....3...46S/abstract (date of access : 12.05.2026).

38. shadcn/ui : The foundation for your design system. URL : https://ui.shadcn.com/ (date of access : 12.05.2026).

39. Shkoukani M. Explore the Major Characteristics of Learning Management Systems and their Impact on e-Learning Success. *International Journal of Advanced Computer Science and Applications.* 2019. Vol. 10, no. 1. DOI : https://doi.org/10.14569/ijacsa.2019.0100139.

40. SQLite Documentation. URL : https://sqlite.org/docs.html (date of access : 12.05.2026).

41. Think Academy US. Educational Technology in K–12 : Interactive and Adaptive Learning Tools. 2025. URL : https://www.thethinkacademy.com/blog/edubriefs-educational-technology-in-k-12-interactive-and-adaptive-learning-tools/ (date of access : 12.05.2026).

42. Vercel. Routing : API routes. Next.js documentation. URL : https://nextjs.org/docs/pages/building-your-application/routing/api-routes (date of access : 12.05.2026).

43. Vitest : Next Generation Testing Framework. URL : https://vitest.dev/ (date of access : 12.05.2026).

44. Vue.js : The Progressive JavaScript Framework. URL : https://vuejs.org/ (date of access : 12.05.2026).

ДОДАТОК А

Лістинг реалізації основного функціоналу

Лістинг А.1 Реалізація хешування паролів та генерації токенів (lib/auth.ts)

```
import { compare, hash } from 'bcryptjs'
import jwt from 'jsonwebtoken'
import { cookies } from 'next/headers'

const JWT_SECRET = process.env.JWT_SECRET || 'your-secret-key'
const COOKIE_NAME = 'auth_token'
const SALT_ROUNDS = 12

// Хешування пароля з використанням bcrypt
export async function hashPassword(password: string): Promise<string> {
  return hash(password, SALT_ROUNDS)
}

// Порівняння пароля з хешем
export async function verifyPassword(
  password: string,
  hashedPassword: string
): Promise<boolean> {
  return compare(password, hashedPassword)
}

// Генерація JWT токена
export function createToken(payload: JWTPayload): string {
  return jwt.sign(payload, JWT_SECRET, { expiresIn: '7d' })
}

// Верифікація токена
export function verifyToken(token: string): JWTPayload | null {
  try {
    return jwt.verify(token, JWT_SECRET) as JWTPayload
```

```
  } catch {
    return null
  }
}

// Встановлення HTTP-only cookie
export async function setAuthCookie(token: string): Promise<void> {
  const cookieStore = await cookies()
  cookieStore.set(COOKIE_NAME, token, {
    httpOnly: true,      // Захист від XSS
    secure: process.env.NODE_ENV === 'production',
    sameSite: 'lax',     // Захист від CSRF
    maxAge: 60 * 60 * 24 * 7, // 7 днів
    path: '/',
  })
}
```

Лістинг А.2 API endpoint для входу в систему (app/api/auth/login/route.ts)

```
import { NextRequest, NextResponse } from 'next/server'
import { login } from '@/lib/auth'

export async function POST(request: NextRequest) {
  try {
    const { email, password } = await request.json()

    // Валідація вхідних даних
    if (!email || !password) {
      return NextResponse.json(
        { error: 'Email та пароль обов\'язкові' },
        { status: 400 }
      )
    }

    // Спроба автентифікації
    const result = await login(email, password)

    if (!result) {
```

```
      return NextResponse.json(
        { error: 'Невірний email або пароль' },
        { status: 401 }
      )
    }

    return NextResponse.json({
      user: result.user,
      message: 'Успішний вхід'
    })
  } catch (error) {
    return NextResponse.json(
      { error: 'Помилка сервера' },
      { status: 500 }
    )
  }
}
```

Лістинг А.3 React Context для управління станом автентифікації (components/auth/auth-provider.tsx)

```
'use client'

import { createContext, useContext, useState, useEffect, ReactNode } from 'react'
import { useRouter } from 'next/navigation'
import type { UserPublic } from '@/lib/types'

interface AuthContextType {
  user: UserPublic | null
  isLoading: boolean
  login: (email: string, password: string) => Promise<boolean>
  logout: () => Promise<void>
  register: (data: RegisterData) => Promise<boolean>
}

const AuthContext = createContext<AuthContextType | undefined>(undefined)

export function AuthProvider({ children }: { children: ReactNode }) {
  const [user, setUser] = useState<UserPublic | null>(null)
```

```
const [isLoading, setIsLoading] = useState(true)
const router = useRouter()

// Перевірка сесії при завантаженні
useEffect(() => {
  checkAuth()
}, [])

const checkAuth = async () => {
  try {
    const response = await fetch('/api/auth/me')
    if (response.ok) {
      const data = await response.json()
      setUser(data.user)
    }
  } finally {
    setIsLoading(false)
  }
}

const login = async (email: string, password: string): Promise<boolean> => {
  const response = await fetch('/api/auth/login', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ email, password }),
  })

  if (response.ok) {
    const data = await response.json()
    setUser(data.user)
    router.push('/dashboard')
    return true
  }
  return false
}

return (
  <AuthContext.Provider value={{ user, isLoading, login, logout, register }}>
    {children}
  </AuthContext.Provider>
```

```
  )
}

export const useAuth = () => {
  const context = useContext(AuthContext)
  if (!context) throw new Error('useAuth must be used within AuthProvider')
  return context
}
```

Лістинг А.4 API для отримання списку курсів (app/api/courses/route.ts)

```
import { NextRequest, NextResponse } from 'next/server'
import { getDB, persist } from '@/lib/db'
import { verifyAuth } from '@/lib/auth'

export async function GET(request: NextRequest) {
  try {
    const data = getDB()
    const { searchParams } = new URL(request.url)

    const search = searchParams.get('search')?.toLowerCase()
    const difficulty = searchParams.get('difficulty')
    const status = searchParams.get('status') || 'published'

    // Фільтрація курсів
    let courses = data.courses.filter(c => c.status === status)

    if (search) {
      courses = courses.filter(c =>
        c.title.toLowerCase().includes(search) ||
        c.description?.toLowerCase().includes(search)
      )
    }

    if (difficulty) {
      courses = courses.filter(c => c.difficulty === difficulty)
    }

    // Додавання інформації про викладача та статистики
```

```
    const coursesWithDetails = courses.map(course => {
      const teacher = data.users.find(u => u.id === course.teacher_id)
      const modules = data.modules.filter(m => m.course_id === course.id)
      const enrollments = data.enrollments.filter(e => e.course_id === course.id)

      const lessonCount = modules.reduce((acc, m) => {
        return acc + data.lessons.filter(l => l.module_id === m.id).length
      }, 0)

      return {
        …course,
        teacher_name: teacher?.name || 'Невідомий',
        module_count: modules.length,
        lesson_count: lessonCount,
        enrollment_count: enrollments.length,
      }
    })

    return NextResponse.json(coursesWithDetails)
  } catch (error) {
    return NextResponse.json({ error: 'Помилка сервера' }, { status: 500 })
  }
}

export async function POST(request: NextRequest) {
  try {
    // Перевірка авторизації
    const auth = await verifyAuth(request)
    if (!auth || !['teacher', 'admin'].includes(auth.role)) {
      return NextResponse.json({ error: 'Доступ заборонено' }, { status: 403 })
    }

    const body = await request.json()
    const { title, description, difficulty } = body

    if (!title) {
      return NextResponse.json(
        { error: 'Назва курсу обов\'язкова' },
        { status: 400 }
      )
```

```
    }

    const data = getDB()
    const newId = (data._counters.courses || 0) + 1
    data._counters.courses = newId

    const newCourse = {
     id: newId,
     title,
     description: description || ‘‘,
     teacher_id: auth.userId,
     status: ‘draft’,
     difficulty: difficulty || ‘beginner’,
     image_url: null,
     created_at: new Date().toISOString(),
     updated_at: new Date().toISOString(),
    }

    data.courses.push(newCourse)
    persist()

    return NextResponse.json(newCourse, { status: 201 })
   } catch (error) {
    return NextResponse.json({ error: ‘Помилка сервера’ }, { status: 500 })
   }
}
```

Лістинг А.5 Компонент рендерингу LaTeX (components/math/latex-renderer.tsx)

```
‘use client’

import { useEffect, useRef } from ‘react’
import katex from ‘katex’
import ‘katex/dist/katex.min.css’

interface LaTeXRendererProps {
 content: string
 className?: string
```

```
}

export function LaTeXRenderer({ content, className = '' }: LaTeXRendererProps) {
  const containerRef = useRef<HTMLDivElement>(null)

  useEffect(() => {
    if (!containerRef.current || !content) return

    // Регулярний вираз для пошуку LaTeX формул
    // Підтримка: $$...$$ (блочні) та $...$ (інлайн)
    const processedContent = content
      // Спочатку обробляємо блочні формули
      .replace(/\$\$([\s\S]*?)\$\$/g, (_, formula) => {
        try {
          return katex.renderToString(formula.trim(), {
            displayMode: true,
            throwOnError: false,
            trust: true,
          })
        } catch {
          return `<span class="text-red-500">[Помилка формули]</span>`
        }
      })
      // Потім інлайн формули
      .replace(/\$([^$\n]+?)\$/g, (_, formula) => {
        try {
          return katex.renderToString(formula.trim(), {
            displayMode: false,
            throwOnError: false,
            trust: true,
          })
        } catch {
          return `<span class="text-red-500">[Помилка]</span>`
        }
      })

    containerRef.current.innerHTML = processedContent
  }, [content])

  return (
```

```
    <div
      ref={containerRef}
      className={`latex-content prose prose-slate max-w-none ${className}`}
    />
  )
}

// Компонент для відображення однієї формули
export function MathFormula({
  formula,
  display = false
}: {
  formula: string
  display?: boolean
}) {
  const ref = useRef<HTMLSpanElement>(null)

  useEffect(() => {
    if (ref.current) {
      katex.render(formula, ref.current, {
        displayMode: display,
        throwOnError: false,
      })
    }
  }, [formula, display])

  return <span ref={ref} />
}
```

Лістинг А.6 API для проходження тесту (app/api/quizzes/[id]/attempt/route.ts)

```
import { NextRequest, NextResponse } from 'next/server'
import { getDB, persist } from '@/lib/db'
import { verifyAuth } from '@/lib/auth'

interface AnswerSubmission {
  question_id: number
  selected_options: number[]
```

```
}

export async function POST(
  request: NextRequest,
  { params }: { params: Promise<{ id: string }> }
) {
  try {
    const auth = await verifyAuth(request)
    if (!auth) {
      return NextResponse.json({ error: 'Не авторизовано' }, { status: 401 })
    }

    const { id } = await params
    const quizId = parseInt(id)
    const { answers } = await request.json() as { answers: AnswerSubmission[] }

    const data = getDB()
    const quiz = data.quizzes.find(q => q.id === quizId)

    if (!quiz) {
      return NextResponse.json({ error: 'Тест не знайдено' }, { status: 404 })
    }

    // Отримуємо питання тесту
    const questions = data.quiz_questions.filter(q => q.quiz_id === quizId)

    // Створюємо спробу
    const attemptId = (data._counters.quiz_attempts || 0) + 1
    data._counters.quiz_attempts = attemptId

    let totalScore = 0
    let maxScore = 0
    const resultsDetails: any[] = []

    // Перевіряємо кожну відповідь
    for (const question of questions) {
      maxScore += question.points

      const answer = answers.find(a => a.question_id === question.id)
      const options = data.quiz_options.filter(o => o.question_id === question.id)
```

```
const correctOptions = options.filter(o => o.is_correct).map(o => o.id)

let isCorrect = false
let pointsEarned = 0

if (answer) {
  const selected = answer.selected_options.sort()
  const correct = correctOptions.sort()

  // Перевірка правильності
  isCorrect = selected.length === correct.length &&
    selected.every((val, idx) => val === correct[idx])

  if (isCorrect) {
    pointsEarned = question.points
    totalScore += pointsEarned
  }
}

// Зберігаємо відповідь
const answerId = (data._counters.quiz_answers || 0) + 1
data._counters.quiz_answers = answerId

data.quiz_answers.push({
  id: answerId,
  attempt_id: attemptId,
  question_id: question.id,
  selected_options: JSON.stringify(answer?.selected_options || []),
  is_correct: isCorrect ? 1 : 0,
  points_earned: pointsEarned,
})

resultsDetails.push({
  question_id: question.id,
  question_text: question.question_text,
  is_correct: isCorrect,
  points_earned: pointsEarned,
  max_points: question.points,
  explanation: question.explanation,
  correct_options: correctOptions,
```

```
    selected_options: answer?.selected_options || [],
  })
 }

 // Обчислюємо результат
 const percentage = maxScore > 0 ? (totalScore / maxScore) * 100 : 0
 const passed = percentage >= quiz.passing_score

 // Зберігаємо спробу
 const attempt = {
  id: attemptId,
  quiz_id: quizId,
  user_id: auth.userId,
  score: totalScore,
  max_score: maxScore,
  percentage: Math.round(percentage * 100) / 100,
  passed: passed ? 1 : 0,
  started_at: new Date().toISOString(),
  completed_at: new Date().toISOString(),
 }

 data.quiz_attempts.push(attempt)
 persist()

 return NextResponse.json({
  attempt,
  results: resultsDetails,
  summary: {
   total_questions: questions.length,
   correct_answers: resultsDetails.filter(r => r.is_correct).length,
   score: totalScore,
   max_score: maxScore,
   percentage: attempt.percentage,
   passed,
   passing_score: quiz.passing_score,
  }
 })
} catch (error) {
 console.error('Quiz attempt error:', error)
 return NextResponse.json({ error: 'Помилка сервера' }, { status: 500 })
```

```
  }
}
```

Лістинг А.7 API для позначення уроку завершеним (app/api/lessons/[id]/complete/route.ts)

```
import { NextRequest, NextResponse } from 'next/server'
import { getDB, persist } from '@/lib/db'
import { verifyAuth } from '@/lib/auth'

export async function POST(
  request: NextRequest,
  { params }: { params: Promise<{ id: string }> }
) {
  try {
    const auth = await verifyAuth(request)
    if (!auth) {
      return NextResponse.json({ error: 'Не авторизовано' }, { status: 401 })
    }

    const { id } = await params
    const lessonId = parseInt(id)
    const data = getDB()

    // Перевіряємо чи урок існує
    const lesson = data.lessons.find(l => l.id === lessonId)
    if (!lesson) {
      return NextResponse.json({ error: 'Урок не знайдено' }, { status: 404 })
    }

    // Перевіряємо чи учень записаний на курс
    const module = data.modules.find(m => m.id === lesson.module_id)
    const enrollment = data.enrollments.find(
      e => e.course_id === module?.course_id && e.user_id === auth.userId
    )

    if (!enrollment) {
      return NextResponse.json(
        { error: 'Ви не записані на цей курс' },
```

```
    { status: 403 }
  )
}

// Перевіряємо чи вже завершено
const existingProgress = data.user_progress.find(
  p => p.user_id === auth.userId && p.lesson_id === lessonId
)

if (existingProgress) {
  if (existingProgress.completed) {
    return NextResponse.json({
      message: 'Урок вже завершено',
      progress: existingProgress
    })
  }
  // Оновлюємо існуючий запис
  existingProgress.completed = 1
  existingProgress.completed_at = new Date().toISOString()
} else {
  // Створюємо новий запис
  const progressId = (data._counters.user_progress || 0) + 1
  data._counters.user_progress = progressId

  data.user_progress.push({
    id: progressId,
    user_id: auth.userId,
    lesson_id: lessonId,
    completed: 1,
    completed_at: new Date().toISOString(),
  })
}

// Оновлюємо прогрес курсу
const courseModules = data.modules.filter(m => m.course_id === module?.course_id)
const courseLessons = courseModules.flatMap(m =>
  data.lessons.filter(l => l.module_id === m.id)
)
const completedLessons = data.user_progress.filter(
  p => p.user_id === auth.userId &&
```

```
        p.completed &&
        courseLessons.some(l => l.id === p.lesson_id)
    )

    const progress = courseLessons.length > 0
      ? (completedLessons.length / courseLessons.length) * 100
      : 0

    enrollment.progress = Math.round(progress * 100) / 100

    if (progress >= 100 && !enrollment.completed_at) {
      enrollment.completed_at = new Date().toISOString()
    }
    persist()
    return NextResponse.json({
      message: 'Урок завершено',
      course_progress: enrollment.progress,
    })
  } catch (error) {
    return NextResponse.json({ error: 'Помилка сервера' }, { status: 500 })
  }
}
```

ДОДАТОК Б

Лістинг реалізації користувацького інтерфейсу

Лістинг Б.1 Конфігурація теми (app/globals.css)

```
@import ‘tailwindcss’;

@theme inline {
  --font-sans: ‘Inter’, ‘Inter Fallback’, system-ui, sans-serif;
  --font-mono: ‘JetBrains Mono’, ‘JetBrains Mono Fallback’, monospace;

  --color-background: oklch(100% 0 0);
  --color-foreground: oklch(14.08% 0 0);
  --color-primary: oklch(54.6% 0.215 262.87);
  --color-primary-foreground: oklch(100% 0 0);
  --color-secondary: oklch(96.08% 0.013 247.84);
  --color-accent: oklch(96.08% 0.013 247.84);
  --color-muted: oklch(96.08% 0.013 247.84);
  --color-border: oklch(91.37% 0.013 247.84);

  --radius: 0.625rem;
}

/* Анімації для математичних елементів */
@keyframes float {
  0%, 100% { transform: translateY(0) rotate(0deg); }
  50% { transform: translateY(-20px) rotate(5deg); }
}

@keyframes pulse-glow {
  0%, 100% { opacity: 0.4; }
  50% { opacity: 0.8; }
}

.animate-float {
```

```
  animation: float 6s ease-in-out infinite;
}

.animate-pulse-glow {
  animation: pulse-glow 3s ease-in-out infinite;
}
```

Лістинг Б.2 Компонент плаваючих геометричних фігур (components/landing/floating-shapes.tsx)

```
'use client'

import { useEffect, useState } from 'react'

interface Shape {
  id: number
  type: 'circle' | 'triangle' | 'square' | 'hexagon' | 'plus'
  size: number
  x: number
  y: number
  duration: number
  delay: number
  color: string
}

const COLORS = [
  'rgba(59, 130, 246, 0.15)',   // blue
  'rgba(147, 51, 234, 0.12)',   // purple
  'rgba(236, 72, 153, 0.12)',   // pink
  'rgba(20, 184, 166, 0.15)',   // teal
  'rgba(249, 115, 22, 0.12)',   // orange
]

export function FloatingShapes() {
  const [shapes, setShapes] = useState<Shape[]>([])

  useEffect(() => {
    // Генеруємо випадкові фігури
    const generated: Shape[] = Array.from({ length: 15 }, (_, i) => ({
```

```
    id: i,
    type: [‘circle’, ‘triangle’, ‘square’, ‘hexagon’, ‘plus’][
      Math.floor(Math.random() * 5)
    ] as Shape[‘type’],
    size: Math.random() * 60 + 30,
    x: Math.random() * 100,
    y: Math.random() * 100,
    duration: Math.random() * 10 + 15,
    delay: Math.random() * 5,
    color: COLORS[Math.floor(Math.random() * COLORS.length)],
  }))
  setShapes(generated)
}, [])

const renderShape = (shape: Shape) => {
  const baseStyle = {
    width: shape.size,
    height: shape.size,
    left: `${shape.x}%`,
    top: `${shape.y}%`,
    animationDuration: `${shape.duration}s`,
    animationDelay: `${shape.delay}s`,
  }

  switch (shape.type) {
    case ‘circle’:
      return (
        <div
          key={shape.id}
          className="absolute rounded-full animate-float"
          style={{ ...baseStyle, backgroundColor: shape.color }}
        />
      )
    case ‘triangle’:
      return (
        <div
          key={shape.id}
          className="absolute animate-float"
          style={{
            ...baseStyle,
```

```
            width: 0,
            height: 0,
            borderLeft: `${shape.size / 2}px solid transparent`,
            borderRight: `${shape.size / 2}px solid transparent`,
            borderBottom: `${shape.size}px solid ${shape.color}`,
          }}
        />
      )
    case 'square':
      return (
        <div
          key={shape.id}
          className="absolute rounded-lg animate-float"
          style={{
            ...baseStyle,
            backgroundColor: shape.color,
            transform: 'rotate(45deg)',
          }}
        />
      )
    // ... інші фігури
  }
}

return (
  <div className="fixed inset-0 overflow-hidden pointer-events-none z-0">
    {shapes.map(renderShape)}
  </div>
)
}
```

Лістинг Б.3 Компонент Dashboard учня (components/dashboard/student-dashboard.tsx)

```
'use client'

import { useEffect, useState } from 'react'
import Link from 'next/link'
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card'
```

```
import { Button } from '@/components/ui/button'
import { Progress } from '@/components/ui/progress'
import { BookOpen, Clock, CheckCircle, TrendingUp } from 'lucide-react'

interface StudentStats {
  enrolled_courses: number
  completed_courses: number
  completed_lessons: number
  quizzes_passed: number
  average_score: number
}

interface EnrolledCourse {
  id: number
  title: string
  progress: number
  last_accessed: string
}

export function StudentDashboard() {
  const [stats, setStats] = useState<StudentStats | null>(null)
  const [courses, setCourses] = useState<EnrolledCourse[]>([])
  const [loading, setLoading] = useState(true)

  useEffect(() => {
    Promise.all([
      fetch('/api/student/stats').then(r => r.json()),
      fetch('/api/student/my-courses').then(r => r.json()),
    ]).then(([statsData, coursesData]) => {
      setStats(statsData)
      setCourses(coursesData)
      setLoading(false)
    })
  }, [])

  if (loading) {
    return <DashboardSkeleton />
  }

  return (
```

```
<div className="space-y-6">
  <div>
    <h1 className="text-3xl font-bold">Панель учня</h1>
    <p className="text-muted-foreground">
      Ваш прогрес та активні курси
    </p>
  </div>

  {/* Статистика */}
  <div className="grid gap-4 md:grid-cols-2 lg:grid-cols-4">
    <StatCard
      title="Активні курси"
      value={stats?.enrolled_courses || 0}
      icon={<BookOpen className="h-4 w-4" />}
      description="записаних курсів"
    />
    <StatCard
      title="Завершено курсів"
      value={stats?.completed_courses || 0}
      icon={<CheckCircle className="h-4 w-4" />}
      description="пройдено повністю"
    />
    <StatCard
      title="Пройдено уроків"
      value={stats?.completed_lessons || 0}
      icon={<Clock className="h-4 w-4" />}
      description="завершених уроків"
    />
    <StatCard
      title="Середній бал"
      value={`${stats?.average_score || 0}%`}
      icon={<TrendingUp className="h-4 w-4" />}
      description="за всі тести"
    />
  </div>

  {/* Активні курси */}
  <Card>
    <CardHeader>
      <CardTitle>Мої курси</CardTitle>
```

```
      </CardHeader>
      <CardContent>
       {courses.length === 0 ? (
        <EmptyCoursesState />
       ) : (
        <div className="space-y-4">
         {courses.map(course => (
          <CourseProgressCard key={course.id} course={course} />
         ))}
        </div>
       )}
      </CardContent>
     </Card>
    </div>
  )
}

function CourseProgressCard({ course }: { course: EnrolledCourse }) {
  return (
    <div className="flex items-center gap-4 p-4 border rounded-lg">
     <div className="flex-1">
      <h3 className="font-medium">{course.title}</h3>
      <div className="flex items-center gap-2 mt-2">
       <Progress value={course.progress} className="flex-1" />
       <span className="text-sm text-muted-foreground">
        {Math.round(course.progress)}%
       </span>
      </div>
     </div>
     <Button asChild>
      <Link href={`/dashboard/student/courses/${course.id}`}>
       Продовжити
      </Link>
     </Button>
    </div>
  )
}
```

Лістинг Б.4 Форма створення/редагування курсу

```
‘use client’

import { useState } from ‘react’
import { useRouter } from ‘next/navigation’
import { Button } from ‘@/components/ui/button’
import { Input } from ‘@/components/ui/input’
import { Textarea } from ‘@/components/ui/textarea’
import { Label } from ‘@/components/ui/label’
import {
  Select,
  SelectContent,
  SelectItem,
  SelectTrigger,
  SelectValue,
} from ‘@/components/ui/select’

interface CourseFormProps {
  initialData?: {
    id?: number
    title: string
    description: string
    difficulty: string
    status: string
  }
}

export function CourseForm({ initialData }: CourseFormProps) {
  const router = useRouter()
  const [loading, setLoading] = useState(false)
  const [formData, setFormData] = useState({
    title: initialData?.title || ‘‘,
    description: initialData?.description || ‘‘,
    difficulty: initialData?.difficulty || ‘beginner’,
    status: initialData?.status || ‘draft’,
  })

  const isEditing = !!initialData?.id
```

```
const handleSubmit = async (e: React.FormEvent) => {
  e.preventDefault()
  setLoading(true)

  try {
    const url = isEditing
      ? `/api/courses/${initialData.id}`
      : '/api/courses'

    const response = await fetch(url, {
      method: isEditing ? 'PUT' : 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(formData),
    })

    if (response.ok) {
      const course = await response.json()
      router.push(`/dashboard/teacher/courses/${course.id}/edit`)
    }
  } finally {
    setLoading(false)
  }
}

return (
  <form onSubmit={handleSubmit} className="space-y-6">
    <div className="space-y-2">
      <Label htmlFor="title">Назва курсу *</Label>
      <Input
        id="title"
        value={formData.title}
        onChange={(e) => setFormData(prev => ({
          ...prev,
          title: e.target.value
        }))}
        placeholder="Введіть назву курсу"
        required
      />
    </div>
```

```
<div className="space-y-2">
  <Label htmlFor="description">Опис курсу</Label>
  <Textarea
    id="description"
    value={formData.description}
    onChange={(e) => setFormData(prev => ({
      ...prev,
      description: e.target.value
    }))}
    placeholder="Опишіть зміст та цілі курсу"
    rows={4}
  />
</div>

<div className="grid gap-4 md:grid-cols-2">
  <div className="space-y-2">
    <Label>Рівень складності</Label>
    <Select
      value={formData.difficulty}
      onValueChange={(value) => setFormData(prev => ({
        ...prev,
        difficulty: value
      }))}
    >
      <SelectTrigger>
        <SelectValue />
      </SelectTrigger>
      <SelectContent>
        <SelectItem value="beginner">Початковий</SelectItem>
        <SelectItem value="intermediate">Середній</SelectItem>
        <SelectItem value="advanced">Просунутий</SelectItem>
      </SelectContent>
    </Select>
  </div>

  <div className="space-y-2">
    <Label>Статус</Label>
    <Select
      value={formData.status}
      onValueChange={(value) => setFormData(prev => ({
```

```
            ...prev,
            status: value
          }))}
        >
          <SelectTrigger>
            <SelectValue />
          </SelectTrigger>
          <SelectContent>
            <SelectItem value="draft">Чернетка</SelectItem>
            <SelectItem value="published">Опубліковано</SelectItem>
            <SelectItem value="archived">В архіві</SelectItem>
          </SelectContent>
        </Select>
      </div>
    </div>

    <div className="flex gap-4">
      <Button type="submit" disabled={loading}>
        {loading ? 'Збереження...' : isEditing ? 'Зберегти' : 'Створити'}
      </Button>
      <Button
        type="button"
        variant="outline"
        onClick={() => router.back()}
      >
        Скасувати
      </Button>
    </div>
  </form>
 )
}
```

Лістинг Б.5 Компонент проходження тесту (app/(dashboard)/dashboard/student/quizzes/[id]/page.tsx)

```
'use client'

import { useState, useEffect } from 'react'
import { useParams, useRouter } from 'next/navigation'
```

```
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card'
import { Button } from '@/components/ui/button'
import { RadioGroup, RadioGroupItem } from '@/components/ui/radio-group'
import { Checkbox } from '@/components/ui/checkbox'
import { Label } from '@/components/ui/label'
import { Progress } from '@/components/ui/progress'
import { LaTeXRenderer } from '@/components/math/latex-renderer'
import { CheckCircle, XCircle, Clock } from 'lucide-react'

type QuizState = 'loading' | 'ready' | 'in-progress' | 'submitted' | 'results'

export default function QuizPage() {
  const params = useParams()
  const router = useRouter()
  const quizId = params.id as string

  const [state, setState] = useState<QuizState>('loading')
  const [quiz, setQuiz] = useState<Quiz | null>(null)
  const [currentQuestion, setCurrentQuestion] = useState(0)
  const [answers, setAnswers] = useState<Record<number, number[]>>({})
  const [results, setResults] = useState<QuizResults | null>(null)
  const [timeLeft, setTimeLeft] = useState<number | null>(null)

  // Завантаження тесту
  useEffect(() => {
    fetch(`/api/quizzes/${quizId}`)
      .then(r => r.json())
      .then(data => {
        setQuiz(data)
        setState('ready')
        if (data.time_limit) {
          setTimeLeft(data.time_limit * 60)
        }
      })
  }, [quizId])

  // Таймер
  useEffect(() => {
    if (state !== 'in-progress' || timeLeft === null) return
```

```
  const timer = setInterval(() => {
   setTimeLeft(prev => {
    if (prev === null || prev <= 1) {
     handleSubmit()
     return 0
    }
    return prev – 1
   })
  }, 1000)

  return () => clearInterval(timer)
 }, [state, timeLeft])

 const handleStartQuiz = () => {
  setState('in-progress')
 }

 const handleAnswerChange = (questionId: number, optionId: number, isMultiple: boolean)
=> {
  setAnswers(prev => {
   if (isMultiple) {
    const current = prev[questionId] || []
    const updated = current.includes(optionId)
     ? current.filter(id => id !== optionId)
     : [...current, optionId]
    return { ...prev, [questionId]: updated }
   }
   return { ...prev, [questionId]: [optionId] }
  })
 }

 const handleSubmit = async () => {
  setState('submitted')

  const formattedAnswers = Object.entries(answers).map(([qId, opts]) => ({
   question_id: parseInt(qId),
   selected_options: opts,
  }))

  const response = await fetch(`/api/quizzes/${quizId}/attempt`, {
```

```
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({ answers: formattedAnswers }),
   })

   const data = await response.json()
   setResults(data)
   setState('results')
  }

  // Екран результатів
  if (state === 'results' && results) {
   return (
    <div className="max-w-3xl mx-auto space-y-6">
     <Card>
      <CardHeader>
       <CardTitle className="flex items-center gap-2">
        {results.summary.passed ? (
         <CheckCircle className="h-6 w-6 text-green-500" />
        ) : (
         <XCircle className="h-6 w-6 text-red-500" />
        )}
        {results.summary.passed ? 'Тест пройдено!' : 'Тест не пройдено'}
       </CardTitle>
      </CardHeader>
      <CardContent className="space-y-4">
       <div className="grid grid-cols-2 gap-4">
        <div className="text-center p-4 bg-muted rounded-lg">
         <div className="text-3xl font-bold">
          {results.summary.percentage}%
         </div>
         <div className="text-sm text-muted-foreground">
          Ваш результат
         </div>
        </div>
        <div className="text-center p-4 bg-muted rounded-lg">
         <div className="text-3xl font-bold">
          {results.summary.correct_answers}/{results.summary.total_questions}
         </div>
         <div className="text-sm text-muted-foreground">
```

```
          Правильних відповідей
        </div>
      </div>
    </div>

    <Progress
      value={results.summary.percentage}
      className="h-3"
    />
  </CardContent>
</Card>

{/* Детальний розбір відповідей */}
<div className="space-y-4">
  <h3 className="text-lg font-semibold">Розбір відповідей</h3>
  {results.results.map((result, index) => (
    <Card
      key={result.question_id}
      className={result.is_correct ? 'border-green-200' : 'border-red-200'}
    >
      <CardContent className="pt-4">
        <div className="flex items-start gap-2">
          {result.is_correct ? (
            <CheckCircle className="h-5 w-5 text-green-500 mt-1" />
          ) : (
            <XCircle className="h-5 w-5 text-red-500 mt-1" />
          )}
          <div className="flex-1">
            <div className="font-medium">
              Питання {index + 1}
            </div>
            <LaTeXRenderer content={result.question_text} />

            {result.explanation && (
              <div className="mt-2 p-3 bg-blue-50 rounded-lg text-sm">
                <strong>Пояснення:</strong>
                <LaTeXRenderer content={result.explanation} />
              </div>
            )}
          </div>
```

```
            </div>
          </CardContent>
        </Card>
      ))}
    </div>
  </div>
 )
}

// Екран питань
if (state === 'in-progress' && quiz) {
 const question = quiz.questions[currentQuestion]
 const isMultiple = question.question_type === 'multiple'
 const selectedAnswers = answers[question.id] || []

 return (
  <div className="max-w-3xl mx-auto space-y-6">
   {/* Прогрес та таймер */}
   <div className="flex items-center justify-between">
    <div className="text-sm text-muted-foreground">
     Питання {currentQuestion + 1} з {quiz.questions.length}
    </div>
    {timeLeft !== null && (
     <div className="flex items-center gap-2 text-sm">
      <Clock className="h-4 w-4" />
      {Math.floor(timeLeft / 60)}:{(timeLeft % 60).toString().padStart(2, '0')}
     </div>
    )}
   </div>

   <Progress
    value={(currentQuestion / quiz.questions.length) * 100}
   />

   {/* Питання */}
   <Card>
    <CardHeader>
     <CardTitle className="text-lg">
      <LaTeXRenderer content={question.question_text} />
     </CardTitle>
```

```
            <p className="text-sm text-muted-foreground">
              {isMultiple
                ? 'Оберіть всі правильні відповіді'
                : 'Оберіть одну відповідь'}
            </p>
          </CardHeader>
          <CardContent>
            {isMultiple ? (
              <div className="space-y-3">
                {question.options.map(option => (
                  <div
                    key={option.id}
                    className="flex items-center space-x-3 p-3 border rounded-lg hover:bg-muted/50"
                  >
                    <Checkbox
                      id={`option-${option.id}`}
                      checked={selectedAnswers.includes(option.id)}
                      onCheckedChange={() =>
                        handleAnswerChange(question.id, option.id, true)
                      }
                    />
                    <Label
                      htmlFor={`option-${option.id}`}
                      className="flex-1 cursor-pointer"
                    >
                      <LaTeXRenderer content={option.option_text} />
                    </Label>
                  </div>
                ))}
              </div>
            ) : (
              <RadioGroup
                value={selectedAnswers[0]?.toString()}
                onValueChange={(value) =>
                  handleAnswerChange(question.id, parseInt(value), false)
                }
              >
                {question.options.map(option => (
                  <div
```

```
                key={option.id}
                className="flex items-center space-x-3 p-3 border rounded-lg hover:bg-
muted/50"
              >
                <RadioGroupItem
                  value={option.id.toString()}
                  id={`option-${option.id}`}
                />
                <Label
                  htmlFor={`option-${option.id}`}
                  className="flex-1 cursor-pointer"
                >
                  <LaTeXRenderer content={option.option_text} />
                </Label>
              </div>
            ))}
          </RadioGroup>
        )}
      </CardContent>
    </Card>

    {/* Навігація */}
    <div className="flex justify-between">
      <Button
        variant="outline"
        onClick={() => setCurrentQuestion(prev => prev – 1)}
        disabled={currentQuestion === 0}
      >
        Назад
      </Button>

      {currentQuestion < quiz.questions.length – 1 ? (
        <Button
          onClick={() => setCurrentQuestion(prev => prev + 1)}
        >
          Далі
        </Button>
      ) : (
        <Button onClick={handleSubmit}>
          Завершити тест
```

```
          </Button>
        )}
      </div>
    </div>
  )
 }

 // Початковий екран
 return (
  <Card className="max-w-2xl mx-auto">
   <CardHeader>
    <CardTitle>{quiz?.title}</CardTitle>
   </CardHeader>
   <CardContent className="space-y-4">
    <p>{quiz?.description}</p>
    <div className="grid grid-cols-2 gap-4 text-sm">
     <div>Кількість питань: {quiz?.questions.length}</div>
     <div>Прохідний бал: {quiz?.passing_score}%</div>
     {quiz?.time_limit && (
      <div>Обмеження часу: {quiz.time_limit} хв</div>
     )}
    </div>
    <Button onClick={handleStartQuiz} className="w-full">
     Почати тест
    </Button>
   </CardContent>
  </Card>
 )
}
```

Лістинг Б.6 Приклад адаптивної сітки

```
// Адаптивна сітка для карток курсів
<div className="grid gap-4 sm:grid-cols-2 lg:grid-cols-3 xl:grid-cols-4">
 {courses.map(course => (
  <CourseCard key={course.id} course={course} />
 ))}
</div>
```

```
// Адаптивний sidebar
<div className="flex min-h-screen">
  {/* Sidebar – прихований на мобільних */}
  <aside className="hidden md:flex md:w-64 lg:w-72 flex-col border-r">
    <Sidebar />
  </aside>

  {/* Mobile navigation */}
  <div className="md:hidden">
    <MobileNav />
  </div>

  {/* Main content */}
  <main className="flex-1 p-4 md:p-6 lg:p-8">
    {children}
  </main>
</div>
```

ДОДАТОК В

Інтерфейс розробленої LMS

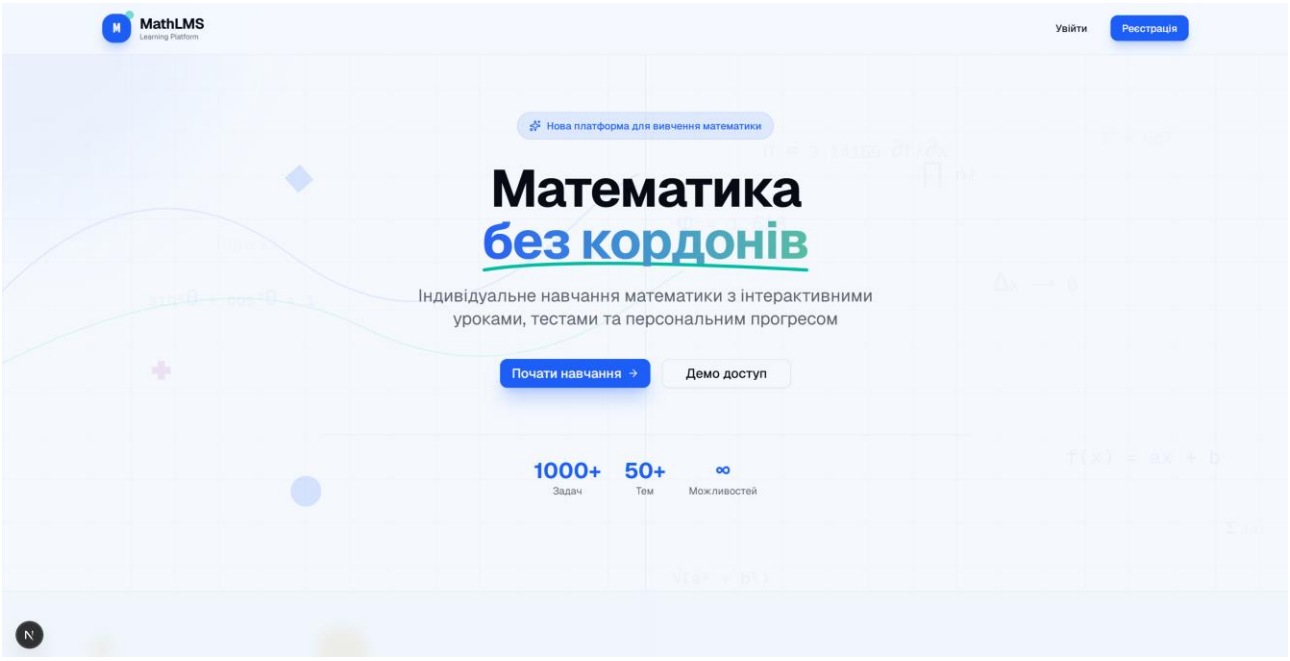


Рисунок В.1 – Головна сторінка

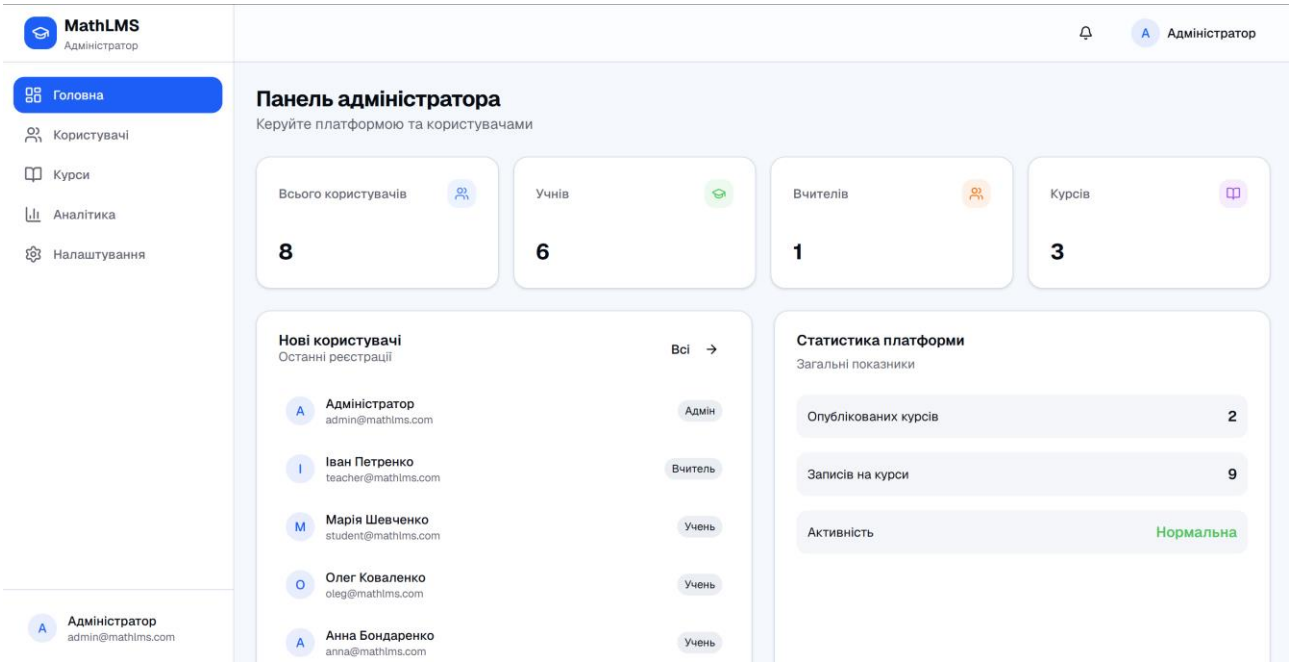


Рисунок В.2 – Дашборд в адміністратора

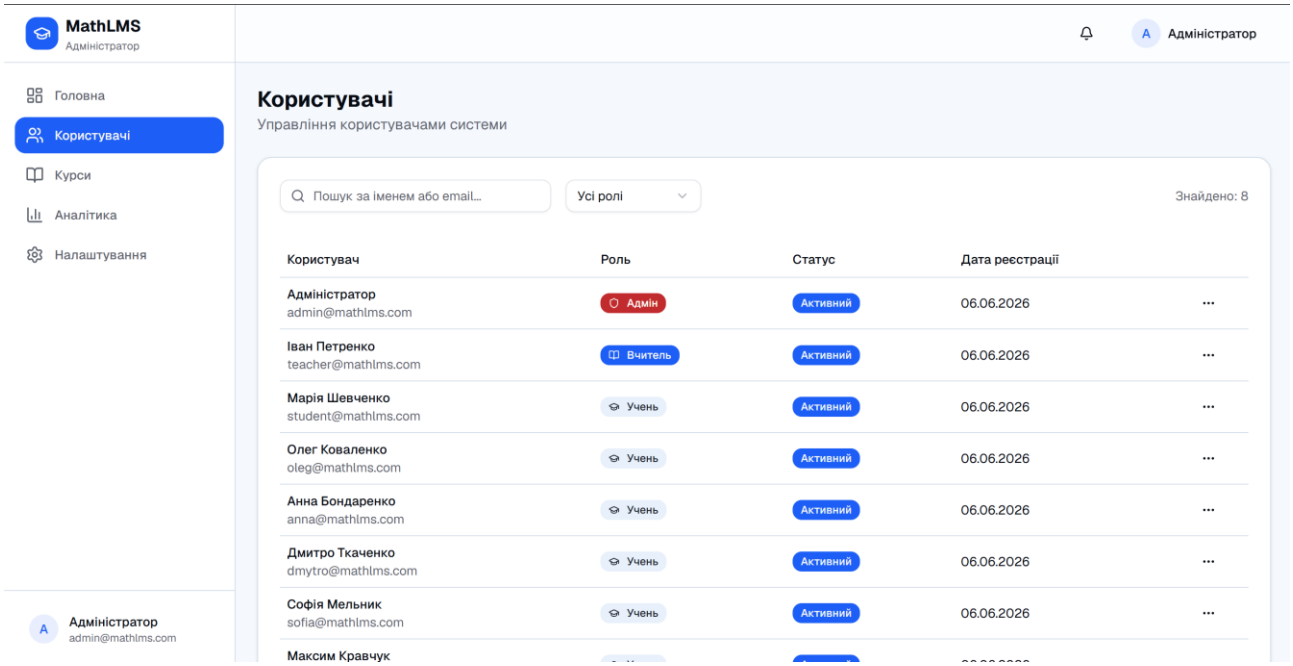


Рисунок В.3 – Сторінка користувачів у адміністратора

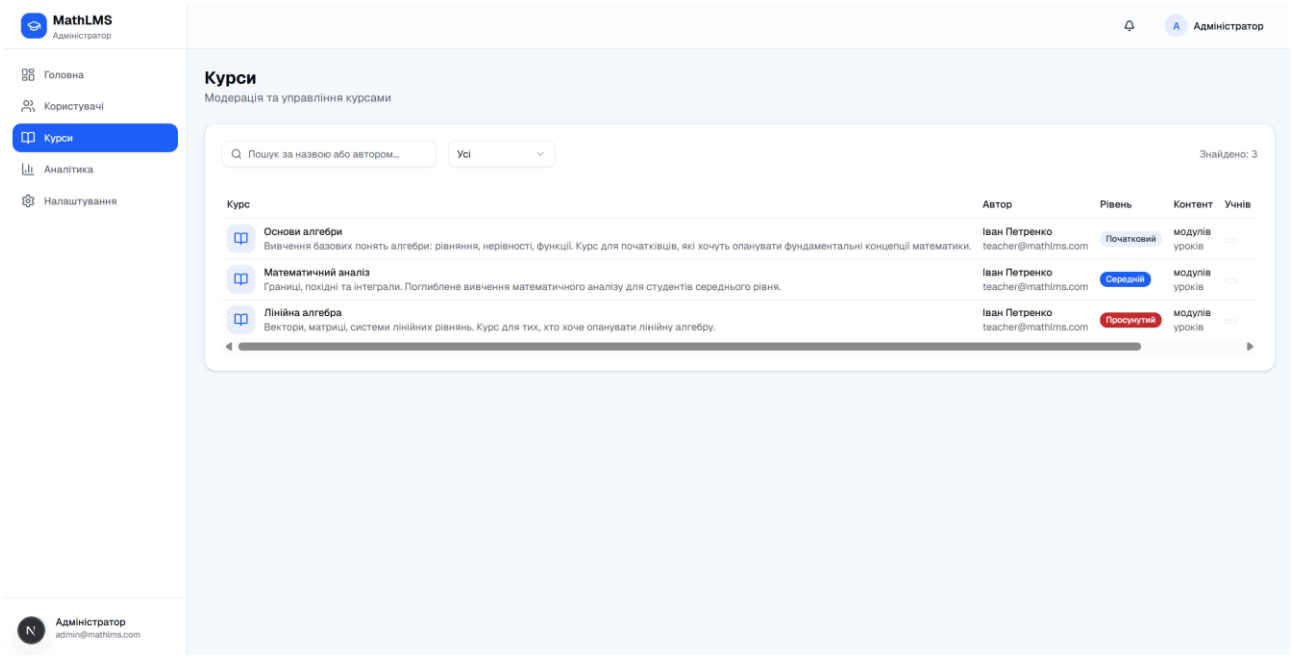


Рисунок В.4 – Сторінка курсів в адміністратора

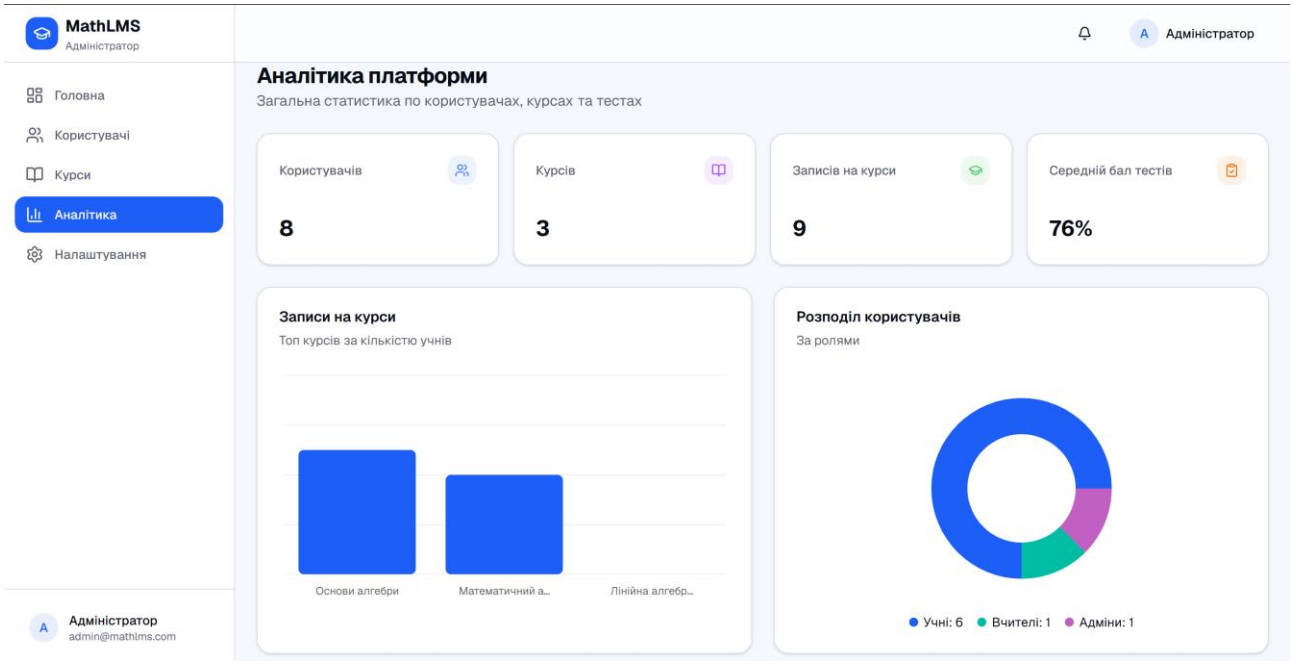


Рисунок В.5 – Сторінка аналітики в адміністатора

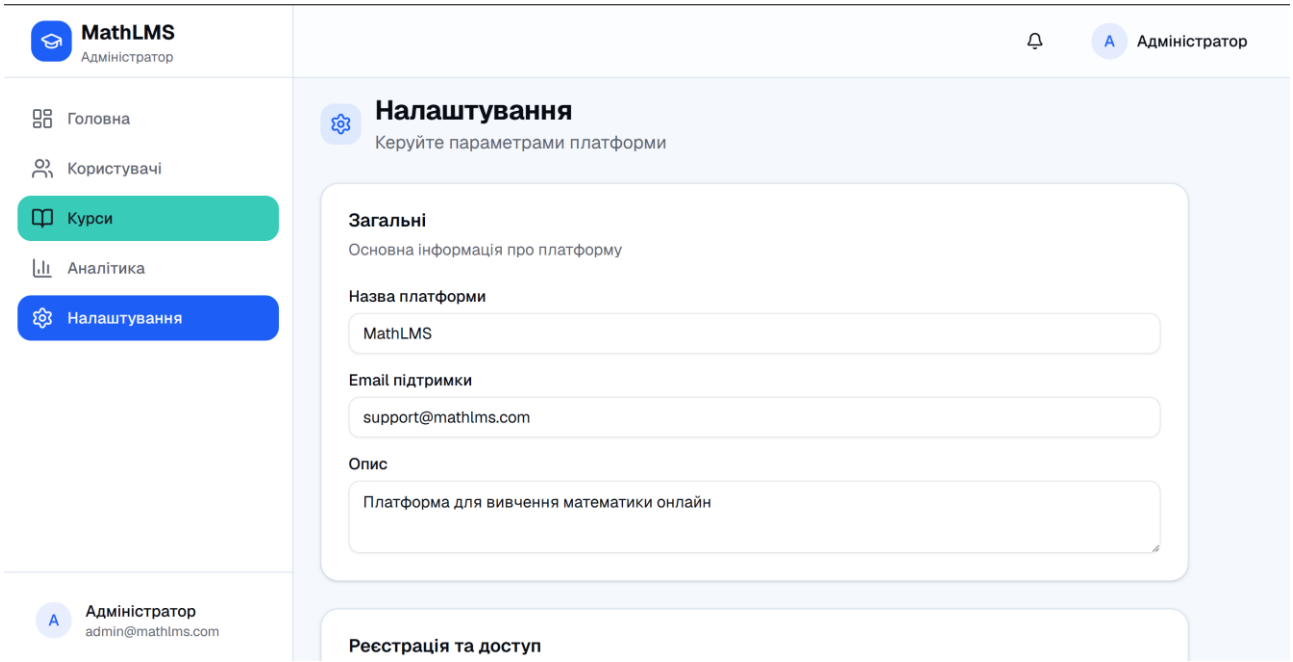


Рисунок В.6 – Сторінка налаштування в адміністратора

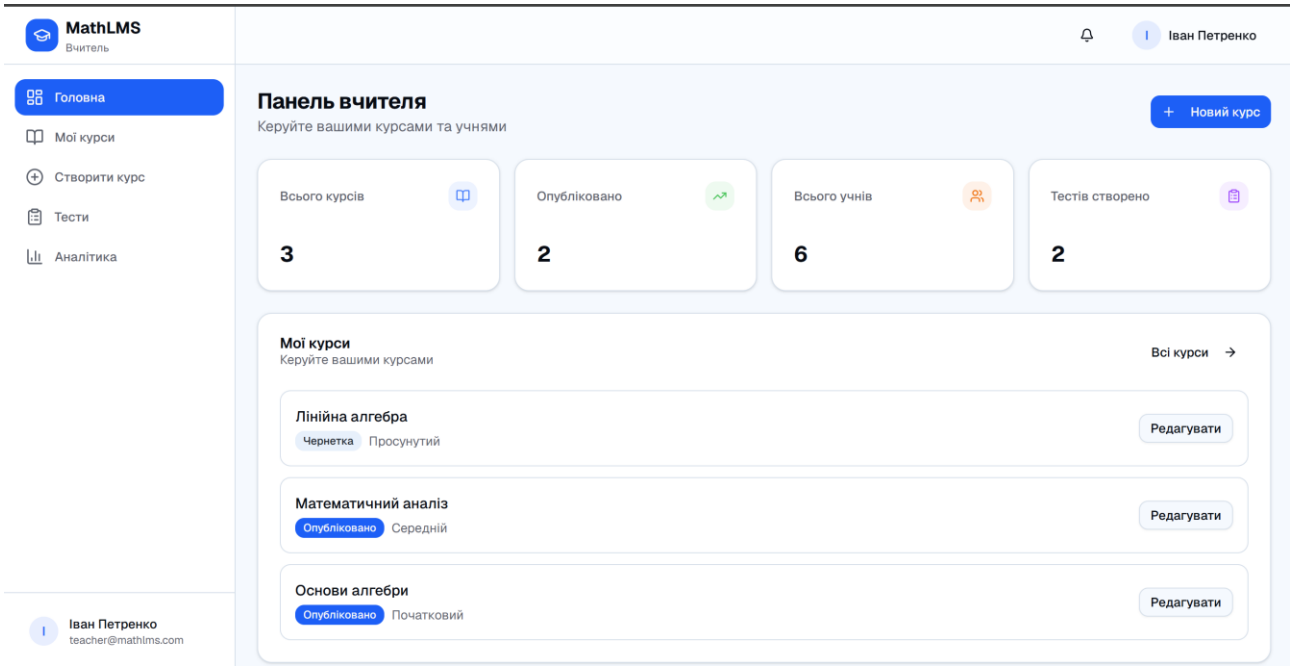


Рисунок В.7 – Дашборд у вчителя

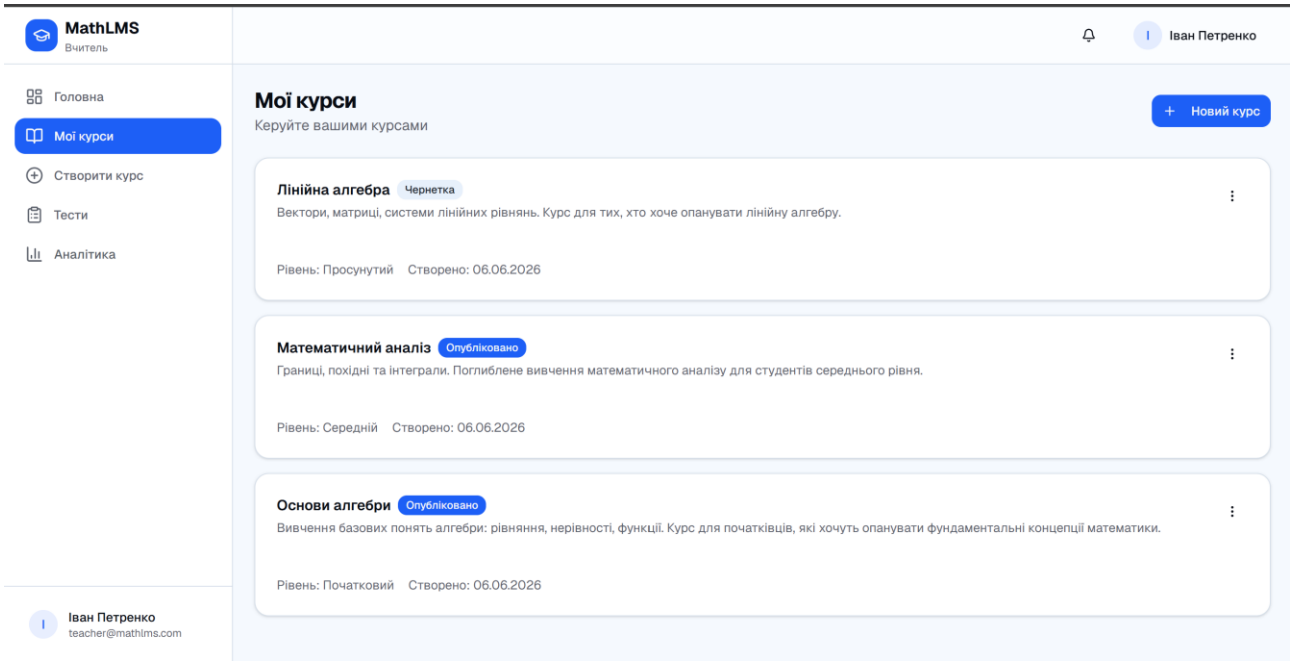


Рисунок В.8 – Сторінка курсів у вчителя

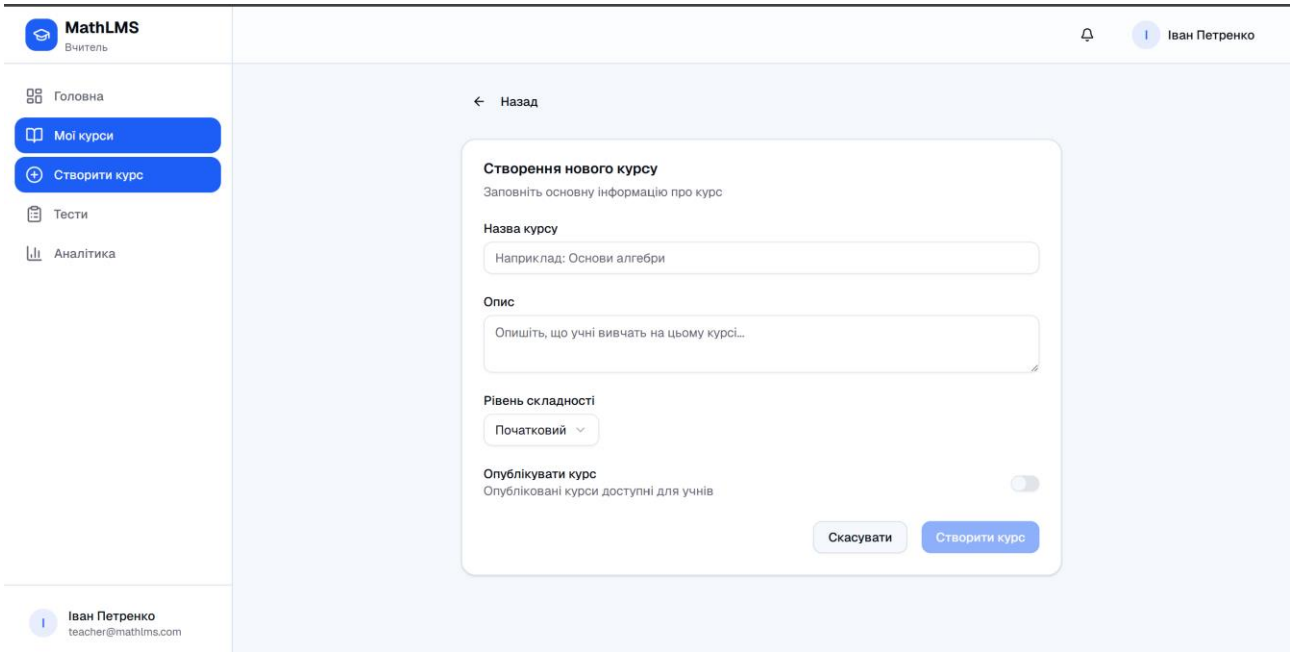


Рисунок В.9 – Сторінка створення курсу в учителя

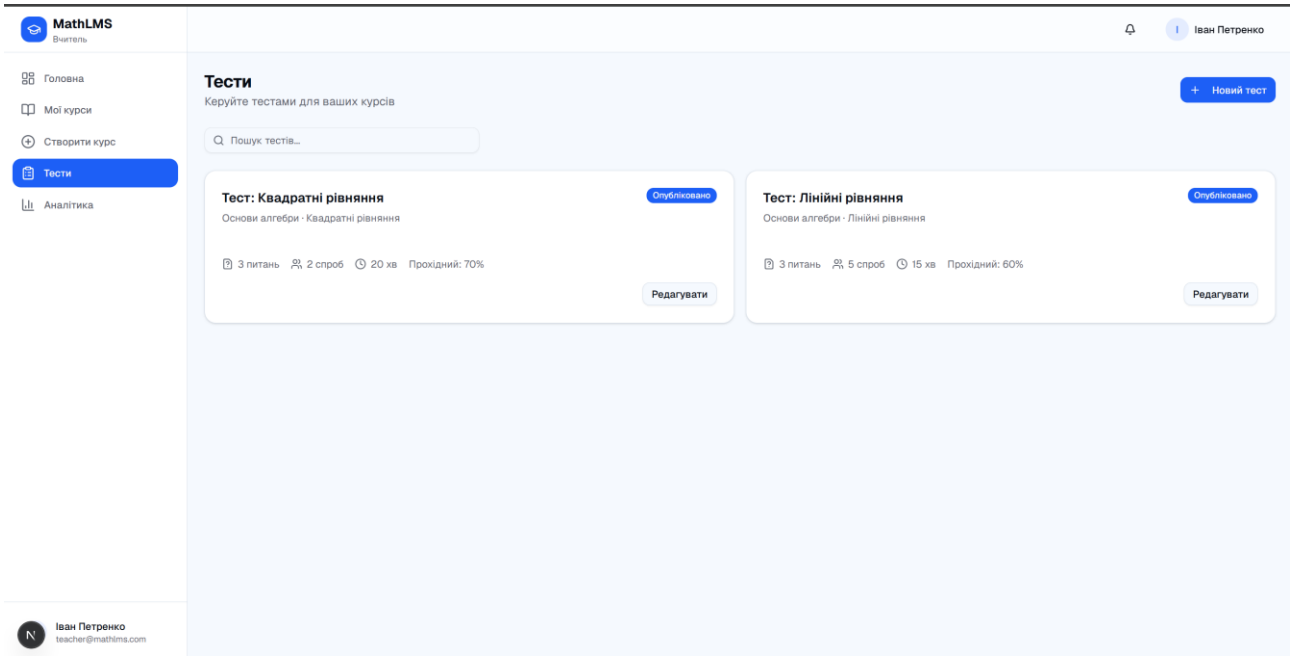


Рисунок В.10 – Сторінка тестів в учителя

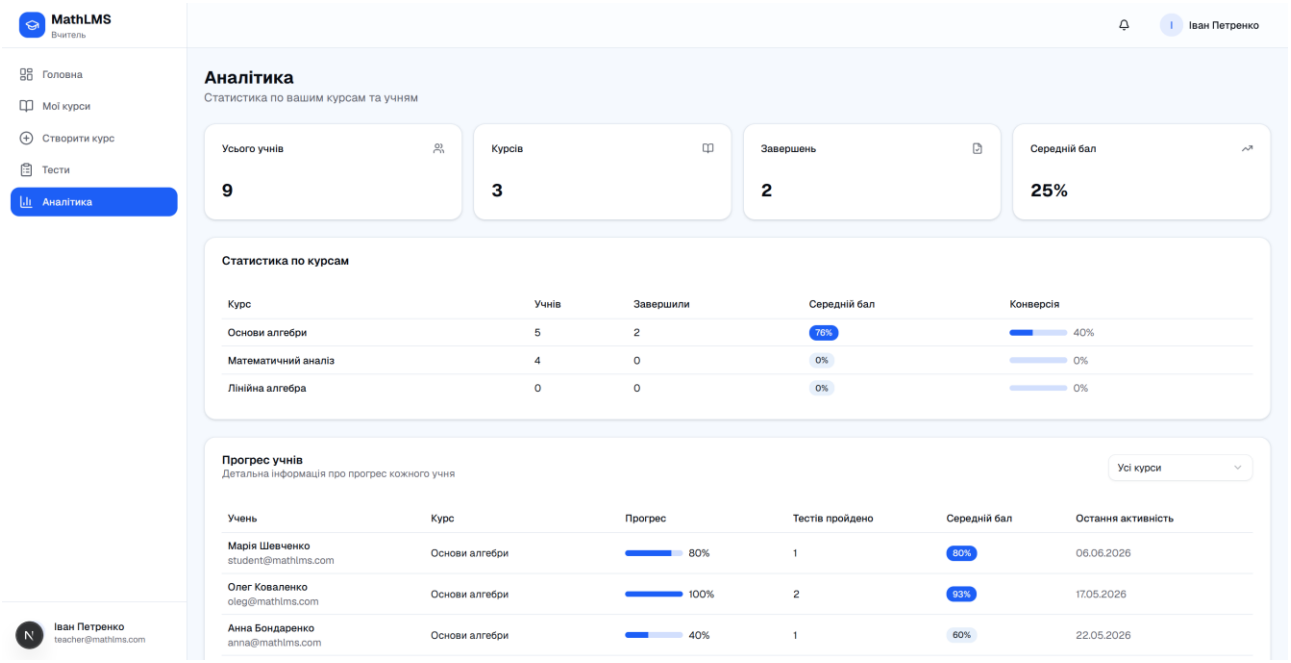


Рисунок В.11 – Сторінка аналітики у вчителя

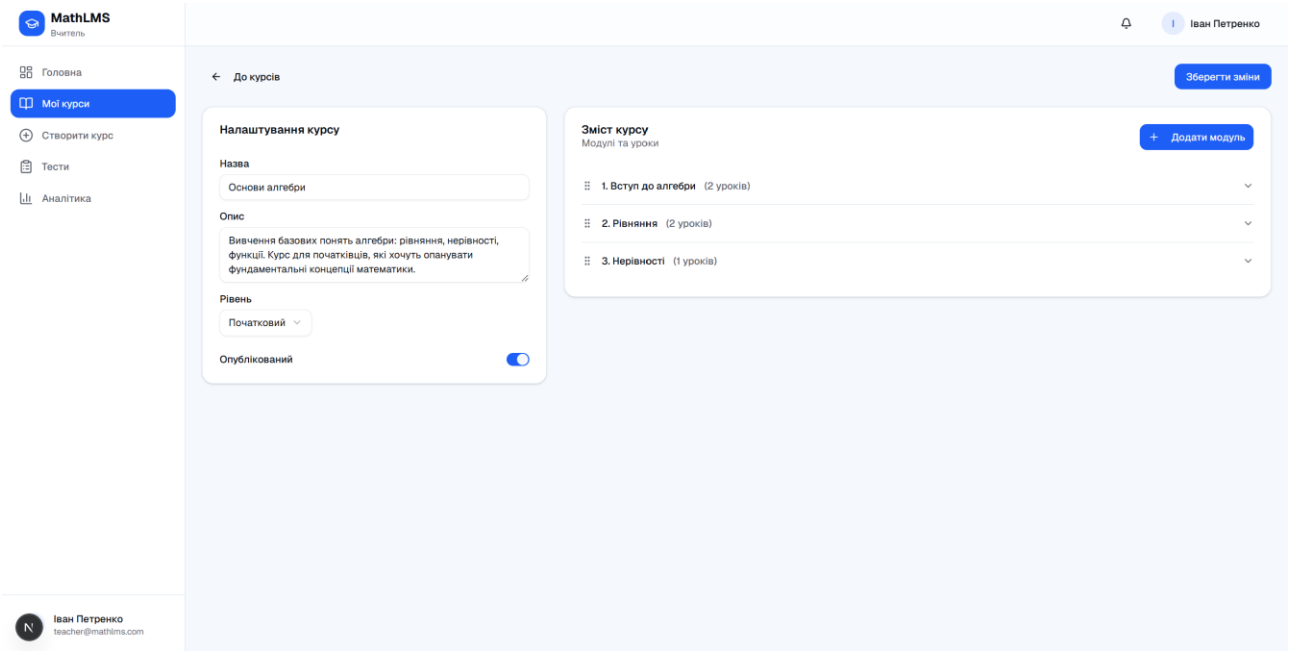


Рисунок В.12 – Сторінка редагування курсу в учителя

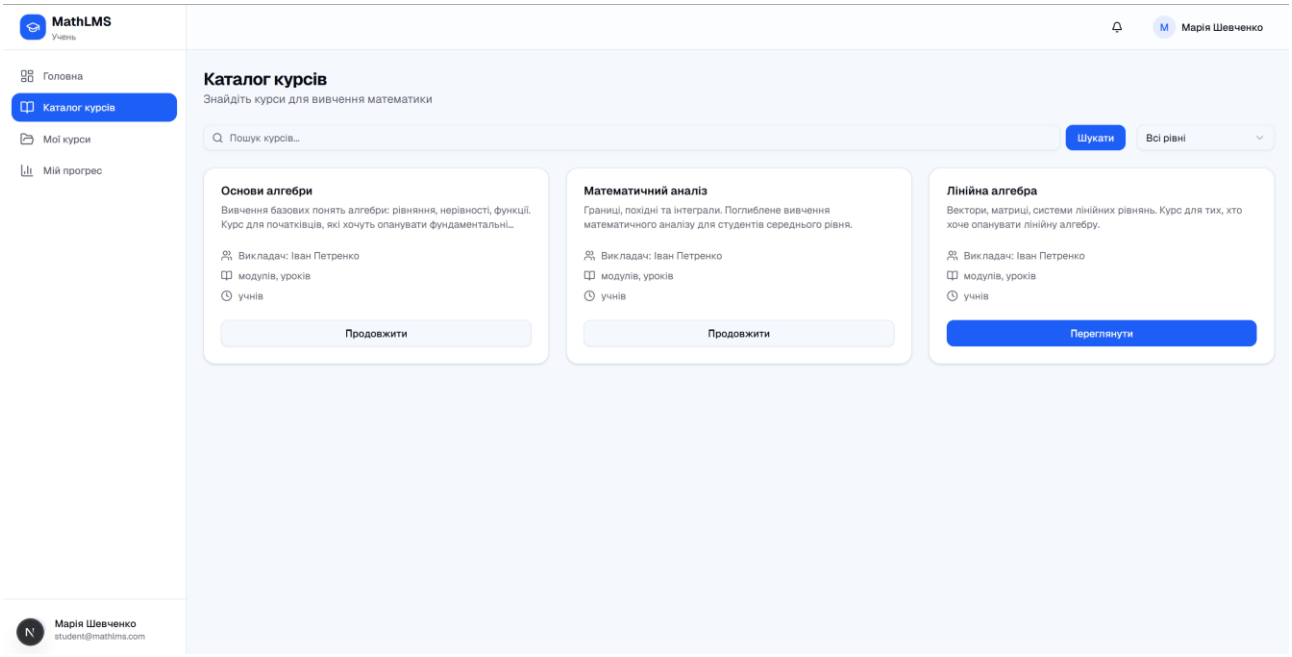


Рисунок В.13 – Сторінка каталогу курсів в учня

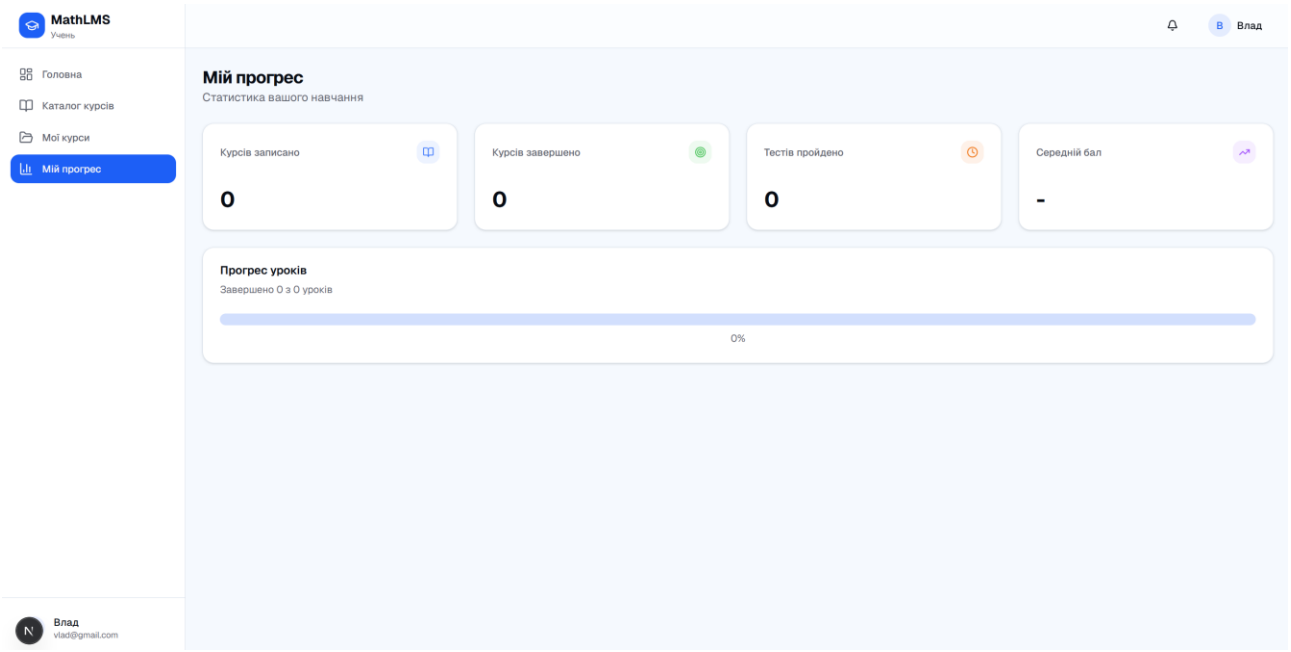


Рисунок В.14 – Сторінка прогресу в учня