

**Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення**

**КВАЛІФІКАЦІЙНА РОБОТА
ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»**

**РОЗРОБКА СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ
ВИКОРИСТОВУЮЧИ REACT, SPRING BOOT ТА MYSQL**

**DEVELOPMENT OF A DISTANCE LEARNING SYSTEM USING REACT,
SPRING BOOT AND MYSQL**

спеціальність 121 «Інженерія програмного забезпечення»
освітня програма «Інженерія програмного забезпечення»

Виконав: здобувач вищої освіти
групи ІПЗ-42
Василевський Денис Віталійович

Керівник:
Самчук Людмила Михайлівна

Кваліфікаційну роботу
допущено до захисту
«__» _____ 20__ р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна

Луцьк – 2026 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних та інформаційних технологій

Кафедра інженерії програмного забезпечення

Ступінь вищої освіти бакалавр

Галузь знань: 12 «Інформаційні технології»

Спеціальність: 121 «Інженерія програмного забезпечення»

Освітня програма: «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

«__» _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Василевському Денису Віталійовичу

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи «Розробка системи дистанційного навчання використовуючи React, Spring Boot та MySQL»

Керівник роботи: доцент Самчук Л. М.

затверджені наказом закладу вищої освіти від «31» грудня 2025 р. № 541/01-02

2. Строк подання здобувачем вищої освіти кваліфікаційної роботи бакалавра «29» травня 2026 р.

3. Вихідні дані до роботи: літературні джерела з дистанційного навчання та веброзробки; технології React, Spring Boot і MySQL; результати аналізу предметної області; вимоги до системи дистанційного навчання; програмні засоби розробки та моделювання.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): аналіз предметної області та існуючих аналогів; постановка завдання; проектування програмного забезпечення та бази даних; вибір засобів і методів розробки; реалізація системи дистанційного навчання; забезпечення захисту інформації; тестування програмного забезпечення.

5. Перелік графічного матеріалу: 13 рисунків, 8 лістингів, 6 таблиць.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	Самчук Л. М.		
Специфікація вимог до розробленої системи	Самчук Л. М.		
Розробка об'єкта проектування	Самчук Л. М.		
Нормоконтроль	Суринович О. М.		
Гарант ОП	Ліщина Н. М.		
Показник запозичень тексту	_____ %		
Академічна доброчесність	Самчук Л. М.		

7. Дата видачі завдання «3» лютого 2026 р.

№	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел по темі кваліфікаційної роботи бакалавра	до 14.02.2026 р.	
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 03.03.2026 р.	
3	Обґрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 14.03.2026 р.	
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 14.04.2026 р.	
5	Практична реалізація об'єкта проектування та розробка бази даних	до 20.04.2026 р.	
6	Тестування та налагодження об'єкта проектування	до 04.05.2026 р.	
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедрі	до 29.05.2026 р.	

Здобувач вищої освіти

_____ Денис ВАСИЛЕВСЬКИЙ

Керівник кваліфікаційної роботи

_____ Людмила САМЧУК

АНОТАЦІЯ

Василевський Д. В. Розробка системи дистанційного навчання використовуючи React, Spring Boot та MySQL. Рукопис. Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення» спеціальності «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2026.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел.

У кваліфікаційній роботі розглянуто розробку системи дистанційного навчання з використанням технологій React, Spring Boot та MySQL. Проведено аналіз предметної області та існуючих систем дистанційного навчання, визначено функціональні та нефункціональні вимоги до програмного забезпечення. У роботі спроектовано архітектуру програмної системи, побудовано UML-діаграми та розроблено структуру бази даних. Реалізовано веборієнтовану систему дистанційного навчання, яка забезпечує реєстрацію та авторизацію користувачів, керування курсами, уроками та завданнями, оцінювання результатів навчання, а також підтримку ролей студента, викладача та адміністратора. Для реалізації клієнтської частини використано React, серверної частини – Spring Boot, а для зберігання даних – MySQL. У системі реалізовано механізми захисту інформації, зокрема рольове розмежування доступу та хешування паролів. Проведено тестування програмного забезпечення, яке підтвердило коректність його функціонування та відповідність поставленим вимогам.

Ключові слова: система дистанційного навчання, вебзастосунок, React, Spring Boot, MySQL, REST API, база даних, дистанційна освіта, інформаційна система.

ABSTRACT

Vasylevskyi D. V. Development of a Distance Learning System Using React, Spring Boot and MySql. Manuscript. Qualification work for bachelor's degree in Software Engineering, specialty "Software Engineering". Lutsk National Technical University. Lutsk, 2026.

The bachelor's qualification thesis consists of an introduction, three chapters, conclusions, and a list of references.

The qualification thesis is devoted to the development of a distance learning system using React, Spring Boot, and MySQL technologies. The subject area and existing distance learning systems were analyzed, and the functional and non-functional requirements for the software were identified. The architecture of the software system was designed, UML diagrams were developed, and the database structure was created. A web-based distance learning system was implemented, providing user registration and authentication, management of courses, lessons, and assignments, assessment of learning outcomes, and support for student, teacher, and administrator roles. React was used for the implementation of the client-side application, Spring Boot for the server-side application, and MySQL for data storage. The system includes information security mechanisms, including role-based access control and password hashing. Software testing was conducted, confirming the correct operation of the system and its compliance with the specified requirements.

Keywords: distance learning system, web application, React, Spring Boot, MySQL, REST API, database, distance education, information system.

ЗМІСТ

ВСТУП.....	7
РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ РОЗРОБКИ ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПОСТАНОВКА ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ	9
1.1 Аналіз сучасного стану проблеми	9
1.2 Постановка завдання на кваліфікаційну роботу бакалавра	12
РОЗДІЛ 2 СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ...	13
2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проєктування програмного забезпечення	13
2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання ...	18
РОЗДІЛ 3 РОЗРОБКА СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ	24
3.1 Практична реалізація об'єкта проєктування.....	24
3.2 Розробка бази даних	32
3.3 Захист інформаційно-комп'ютерної системи	36
3.4 Тестування та налагодження інформаційно-комп'ютерної системи	40
ВИСНОВКИ	43
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	45

ВСТУП

Актуальність теми. У сучасних умовах стрімкого розвитку інформаційних технологій та цифрової трансформації освіти особливого значення набуває використання систем дистанційного навчання. Такі системи забезпечують доступ до освітніх ресурсів незалежно від місця перебування користувачів, підтримують взаємодію між студентами та викладачами, а також дозволяють автоматизувати значну частину навчального процесу.

Поширення дистанційної та змішаної форм навчання сприяло зростанню потреби у програмних засобах, які забезпечують організацію навчального процесу в електронному середовищі. Сучасні заклади освіти потребують платформ, що дозволяють створювати навчальні курси, розміщувати матеріали, формувати завдання, контролювати успішність здобувачів освіти та забезпечувати зручну комунікацію між учасниками освітнього процесу [1].

На сьогодні існує значна кількість систем дистанційного навчання, зокрема Moodle, Google Classroom, Canvas та інші. Незважаючи на широкий функціонал, багато з них мають певні недоліки: складність налаштування, надлишковість функцій для окремих категорій користувачів, обмежені можливості адаптації під конкретні потреби навчального закладу або організації. Крім того, актуальним залишається питання створення спеціалізованих систем, орієнтованих на конкретні вимоги користувачів та сучасні вебтехнології.

Важливим аспектом функціонування систем дистанційного навчання є забезпечення розмежування доступу між різними категоріями користувачів. Студенти, викладачі та адміністратори повинні мати доступ лише до тих функцій і даних, які необхідні для виконання їхніх обов'язків. Це потребує реалізації надійних механізмів автентифікації, авторизації та захисту даних [2].

Застосування сучасних технологій веброзробки, зокрема React для реалізації клієнтської частини, Spring Boot для створення серверної логіки та MySQL для зберігання даних, дозволяє створити ефективну, масштабовану та зручну у використанні систему дистанційного навчання.

Отже, тема кваліфікаційної роботи, присвячена розробці системи дистанційного навчання з використанням React, Spring Boot та MySQL, є актуальною, оскільки спрямована на вирішення практичних завдань автоматизації освітнього процесу та підвищення ефективності організації навчання в електронному середовищі.

Метою кваліфікаційної роботи є розробка системи дистанційного навчання з використанням технологій React, Spring Boot та MySQL.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- проаналізувати сучасні підходи до організації дистанційного навчання та існуючі програмні рішення;
- визначити функціональні та нефункціональні вимоги до програмної системи;
- обґрунтувати вибір технологій та інструментальних засобів розробки;
- спроектувати архітектуру системи та структуру бази даних;
- реалізувати веборієнтовану систему дистанційного навчання;
- забезпечити механізми автентифікації, авторизації та захисту даних;
- провести тестування та налагодження програмного забезпечення.

Об'єктом дослідження є процес організації дистанційного навчання в електронному освітньому середовищі.

Предметом дослідження є методи та засоби розробки веборієнтованих систем дистанційного навчання із використанням технологій React, Spring Boot та MySQL.

Практичне значення роботи полягає у створенні програмного продукту, який може бути використаний закладами освіти та організаціями для організації дистанційного навчання, управління навчальними курсами, контролю успішності студентів та автоматизації взаємодії між учасниками освітнього процесу.

РОЗДІЛ 1

АНАЛІЗ МЕТОДІВ РОЗРОБКИ ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПОСТАНОВКА ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

У сучасних умовах цифровізації освіти та широкого впровадження інформаційно-комунікаційних технологій дистанційне навчання стало невід'ємною складовою освітнього процесу. Розвиток мережових технологій, поширення мобільних пристроїв та необхідність забезпечення безперервності навчання сприяли активному впровадженню систем дистанційного навчання (Learning Management System, LMS) у закладах освіти різних рівнів.

Системи дистанційного навчання забезпечують організацію взаємодії між студентами, викладачами та адміністраторами, дозволяють розміщувати навчальні матеріали, створювати завдання, здійснювати контроль знань та аналізувати результати навчання. Використання таких систем сприяє підвищенню доступності освіти, автоматизації навчальних процесів та покращенню якості освітніх послуг.

На сьогодні існує значна кількість програмних платформ для організації дистанційного навчання. Найбільш поширеними серед них є Moodle, Google Classroom та Canvas LMS.

Moodle є однією з найпопулярніших систем дистанційного навчання з відкритим вихідним кодом. Платформа надає широкі можливості для створення електронних курсів, організації тестування, оцінювання студентів та управління навчальними матеріалами. Перевагами Moodle є гнучкість налаштування, розвинена система розширень та велика спільнота користувачів. Водночас система характеризується досить складною структурою адміністрування та потребує додаткових ресурсів для підтримки та налаштування [3].

Google Classroom є хмарною платформою для організації освітнього процесу, інтегрованою з іншими сервісами Google. Система дозволяє швидко створювати курси, видавати завдання, збирати роботи студентів та здійснювати

комунікацію між учасниками навчального процесу. Основними перевагами є простота використання та інтеграція з екосистемою Google. Недоліками можна вважати обмежені можливості налаштування функціоналу та залежність від сторонніх сервісів [4].

Canvas LMS є сучасною платформою дистанційного навчання, яка широко використовується у закладах вищої освіти. Система має зручний інтерфейс користувача, підтримує сучасні стандарти електронного навчання та надає широкі можливості для інтеграції з іншими сервісами. Разом з тим використання окремих функціональних можливостей платформи може потребувати комерційної ліцензії [5].

Для кращого розуміння особливостей існуючих рішень доцільно провести їх порівняльний аналіз (табл. 1.1).

Таблиця 1.1 – Порівняння існуючих систем дистанційного навчання

Характеристика	Moodle	Google Classroom	Canvas LMS	Розроблювана система
Управління курсами	Так	Так	Так	Так
Створення уроків	Так	Частково	Так	Так
Завдання різних типів	Так	Обмежено	Так	Так
Контроль успішності	Так	Так	Так	Так
Рольова модель доступу	Так	Так	Так	Так
Відкриті та закриті курси	Так	Ні	Так	Так
Простота розгортання	Середня	Висока	Середня	Висока
Можливість адаптації під потреби користувача	Висока	Низька	Середня	Висока
Власна база даних	Так	Ні	Так	Так
Відкритий вихідний код	Так	Ні	Частково	Так

Проведений аналіз показує, що існуючі LMS-платформи забезпечують широкий спектр функціональних можливостей для організації дистанційного навчання. Проте більшість із них або мають надлишковий функціонал, який ускладнює використання системи, або характеризуються обмеженими можливостями адаптації під конкретні потреби користувачів.

Крім того, важливим аспектом є забезпечення чіткого розмежування доступу між різними категоріями користувачів. У системах дистанційного навчання студенти, викладачі та адміністратори виконують різні функції, тому програмне забезпечення повинно підтримувати ефективну рольову модель доступу та забезпечувати захист персональних даних.

Аналіз предметної області показав доцільність створення власної системи дистанційного навчання, яка поєднуватиме зручний інтерфейс користувача, сучасну вебархітектуру, підтримку різних ролей користувачів, механізми управління курсами, уроками та завданнями, а також можливість подальшого розширення функціоналу.

Розроблювана система дистанційного навчання реалізується як вебзастосунок із використанням технологій React, Spring Boot та MySQL. Вона передбачає підтримку трьох основних ролей користувачів: студента, викладача та адміністратора. Функціональні можливості системи включають створення та керування курсами, публікацію навчальних матеріалів, формування завдань, подання відповідей студентами, оцінювання результатів навчання та адміністрування користувачів.

Отже, проведений аналіз сучасного стану проблеми підтверджує актуальність розробки системи дистанційного навчання та свідчить про доцільність створення власного програмного рішення, яке забезпечить ефективну організацію навчального процесу в електронному середовищі.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Метою кваліфікаційної роботи є розробка системи дистанційного навчання з використанням технологій React, Spring Boot та MySQL, яка забезпечуватиме ефективну організацію освітнього процесу, управління навчальними курсами, взаємодію між студентами та викладачами, а також контроль результатів навчання.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- здійснити аналіз сучасного стану проблеми організації дистанційного навчання та дослідити існуючі системи керування навчанням;
- визначити функціональні та нефункціональні вимоги до розроблюваної системи дистанційного навчання;
- обґрунтувати вибір архітектурних рішень, технологій та інструментальних засобів розробки програмного забезпечення;
- спроектувати структуру системи, визначити взаємодію її компонентів та розробити модель даних;
- розробити базу даних для зберігання інформації про користувачів, курси, уроки, завдання, результати навчання та зарахування студентів на курси;
- реалізувати клієнтську частину системи засобами React та серверну частину із використанням Spring Boot;
- забезпечити підтримку різних ролей користувачів (студент, викладач, адміністратор) та реалізувати механізми авторизації й розмежування доступу;
- реалізувати функціонал створення курсів, уроків, завдань, подання відповідей студентами та оцінювання результатів навчання;
- забезпечити захист даних користувачів та безпечне зберігання облікових записів;
- провести тестування та налагодження програмного забезпечення з метою перевірки його працездатності та відповідності поставленим вимогам.

РОЗДІЛ 2

СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проєктування програмного забезпечення

Процес розробки системи дистанційного навчання передбачає детальний аналіз предметної області, визначення вимог до програмного забезпечення та проєктування архітектури майбутньої системи. На цьому етапі формуються основні функціональні можливості програмного продукту, визначаються категорії користувачів, встановлюються вимоги до безпеки та продуктивності, а також здійснюється моделювання структури програмного забезпечення.

Система дистанційного навчання призначена для автоматизації освітнього процесу та забезпечення взаємодії між усіма його учасниками в електронному середовищі. Використання таких систем дозволяє організовувати навчання незалежно від місця перебування користувачів, забезпечувати доступ до навчальних матеріалів, здійснювати контроль виконання завдань та аналізувати результати навчальної діяльності студентів. Враховуючи сучасні тенденції розвитку цифрової освіти, особливої актуальності набуває створення програмних рішень, які поєднують зручність використання, функціональність та можливість подальшого розширення.

Для реалізації поставленого завдання доцільно використовувати ітераційну модель життєвого циклу програмного забезпечення. Такий підхід дозволяє поетапно реалізовувати функціональні можливості системи, виконувати тестування окремих компонентів та поступово вдосконалювати програмний продукт відповідно до вимог користувачів. Використання ітераційної моделі також спрощує внесення змін до системи під час розробки та забезпечує більш гнучкий процес створення програмного забезпечення.

У результаті аналізу предметної області було встановлено, що система повинна забезпечувати роботу трьох основних категорій користувачів:

студентів, викладачів та адміністраторів. Кожна з цих категорій виконує власний набір функцій та має різні права доступу до інформаційних ресурсів системи.

Студент є безпосереднім учасником освітнього процесу та використовує систему для перегляду доступних курсів, ознайомлення з навчальними матеріалами, виконання завдань і контролю власних результатів навчання. Після авторизації студент отримує доступ до особистого кабінету, де може переглядати перелік курсів, на які він зарахований, відстежувати прогрес навчання, переглядати оцінки та коментарі викладачів. Викладач використовує систему для створення та адміністрування навчальних курсів. До його функцій належить підготовка навчальних матеріалів, створення уроків, формування завдань різних типів, зарахування студентів на курси, перевірка виконаних робіт та оцінювання результатів навчання. Викладач також має можливість контролювати активність студентів та відстежувати їх успішність у межах окремих курсів. Адміністратор відповідає за керування користувачами системи та підтримку її працездатності. Він здійснює створення нових облікових записів, зміну ролей користувачів, контроль структури курсів та адміністрування навчального середовища. Наявність окремої адміністративної ролі дозволяє централізовано керувати системою та забезпечувати коректне функціонування всіх її компонентів.

На основі проведеного аналізу були сформовані функціональні вимоги до розроблюваного програмного забезпечення. Система повинна забезпечувати можливість реєстрації та авторизації користувачів, підтримувати розмежування прав доступу відповідно до ролей, надавати інструменти для створення та редагування навчальних курсів, дозволяти розміщення уроків і навчальних матеріалів, підтримувати створення завдань різних типів та забезпечувати можливість подання відповідей студентами. Крім того, програмне забезпечення повинно реалізовувати механізми оцінювання результатів навчання, відображення прогресу студентів та адміністрування користувачів і навчальних курсів.

Окрім функціональних можливостей, система повинна відповідати ряду нефункціональних вимог. Насамперед програмне забезпечення має

забезпечувати зручний та зрозумілий інтерфейс користувача, який не потребуватиме тривалого навчання для роботи із системою. Важливими характеристиками також є швидкодія, надійність, безпечність зберігання даних та можливість одночасної роботи декількох користувачів. Архітектура системи повинна передбачати можливість подальшого масштабування та розширення функціональності без необхідності суттєвої зміни вже реалізованих компонентів. Для візуалізації взаємодії користувачів із системою використовується UML-діаграма варіантів використання [6]. Основні функції користувачів системи дистанційного навчання та їх взаємодію із програмним забезпеченням представлено на діаграмі варіантів використання, наведеній на рисунку 2.1.



Рисунок 2.1 – Діаграма варіантів використання

Під час проєктування системи також було визначено основні сутності предметної області та взаємозв'язки між ними. До таких сутностей належать користувачі, навчальні курси, уроки, зарахування студентів на курси, завдання та відповіді на них. Кожна із зазначених сутностей відіграє важливу роль у функціонуванні системи та забезпечує реалізацію відповідних бізнес-процесів навчального середовища.

Структуру основних сутностей системи дистанційного навчання та зв'язки між ними наведено на UML-діаграмі класів, представлений на рисунку 2.2. Діаграма дозволяє відобразити логічну структуру системи та визначити взаємозв'язки між окремими об'єктами предметної області.

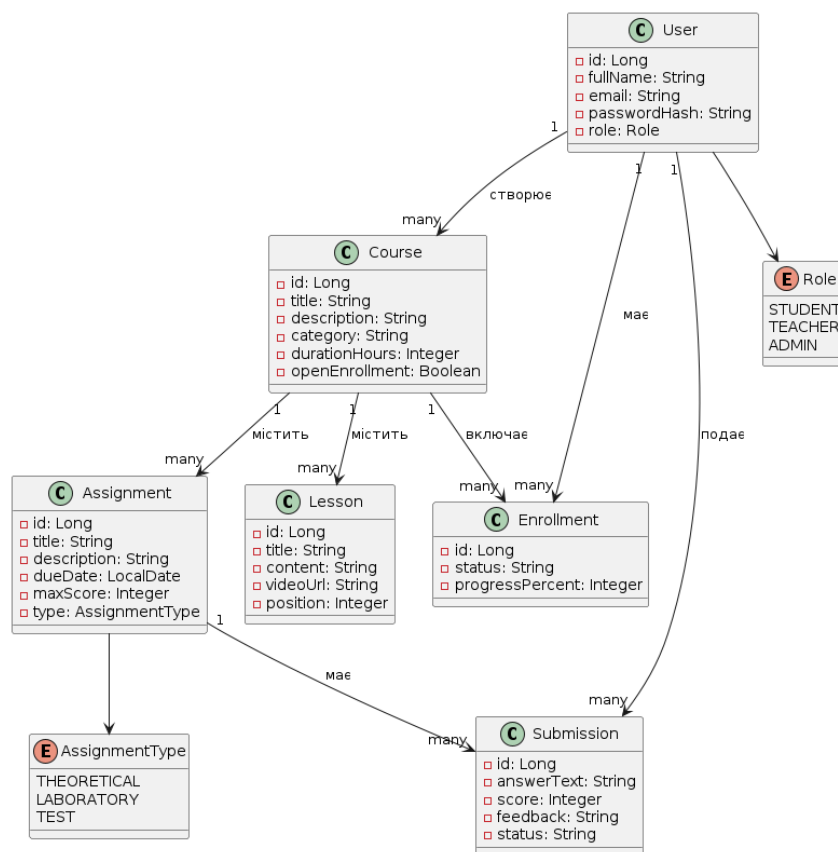


Рисунок 2.2 – Діаграма класів

У системі передбачено, що один викладач може створювати декілька навчальних курсів, кожен курс може містити множину уроків та завдань, а студенти можуть бути зараховані одночасно на декілька курсів. Для збереження

інформації про результати виконання завдань використовується сутність відповіді, яка містить дані про подану роботу, отриману оцінку та коментар викладача.

Для аналізу логіки функціонування системи доцільно використовувати UML-діаграми діяльності та послідовності. Послідовність дій під час виконання навчального завдання представлена на діаграмі діяльності, наведеній на рисунку 2.3. На діаграмі відображено процес авторизації студента, вибору курсу, відкриття завдання, введення відповіді, її надсилання до системи, перевірки викладачем та отримання результату оцінювання.



Рисунок 2.3 – Діаграма діяльності

Процес взаємодії компонентів системи під час подання студентом відповіді на завдання представлено на діаграмі послідовності, наведений на рисунку 2.4. Дана діаграма відображає обмін повідомленнями між клієнтською частиною, сервером застосунку та базою даних, що дозволяє детально проаналізувати роботу програмного забезпечення під час виконання основних операцій.

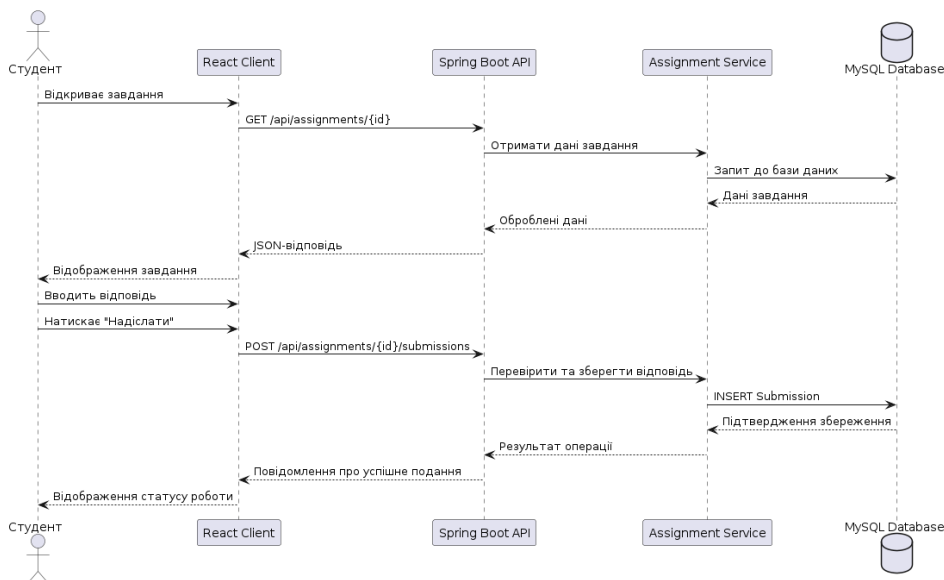


Рисунок 2.4 – Діаграма послідовності

Таким чином, у результаті проведеного аналізу предметної області було сформовано вимоги до системи дистанційного навчання, визначено основні ролі користувачів та виконано початкове проєктування програмного забезпечення. Отримані результати є основою для подальшого вибору технологій розробки та практичної реалізації програмного продукту.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Для реалізації системи дистанційного навчання важливим етапом є обґрунтування вибору технологій, інструментальних засобів, архітектурного підходу та алгоритмів, які забезпечують коректну роботу програмного продукту.

Оскільки розроблювана система є веборієнтованою та передбачає одночасну взаємодію різних категорій користувачів, доцільним є використання клієнт-серверної архітектури. Такий підхід дозволяє розділити логіку відображення інтерфейсу користувача, бізнес-логіку та рівень збереження даних.

Клієнтська частина системи реалізується з використанням бібліотеки React. React є сучасним засобом розробки користувацьких інтерфейсів, який дозволяє створювати динамічні вебсторінки та ефективно оновлювати їхній вміст без повного перезавантаження сторінки. Використання компонентного підходу дає змогу поділити інтерфейс системи на окремі логічні частини, зокрема сторінку входу, кабінет студента, кабінет викладача, адміністративну панель, сторінки курсів, уроків, завдань та оцінок. Це спрощує супровід програмного коду, підвищує його повторне використання та полегшує подальше розширення функціональності системи [7].

Для збирання та запуску клієнтської частини використовується Vite. Цей інструмент забезпечує швидке розгортання середовища розробки, оперативне оновлення сторінки під час внесення змін до коду та оптимізоване складання готового застосунку. У межах розроблюваної системи Vite використовується разом із React для створення сучасного, швидкого та зручного вебінтерфейсу [8].

Серверна частина системи реалізується з використанням Spring Boot. Даний фреймворк є одним із поширених засобів створення вебзастосунків мовою Java та дозволяє швидко розробляти REST API, реалізовувати бізнес-логіку, організовувати взаємодію з базою даних і налаштовувати конфігурацію застосунку. Використання Spring Boot є доцільним для цієї роботи, оскільки система дистанційного навчання потребує надійної серверної частини, яка відповідатиме за обробку запитів користувачів, перевірку прав доступу, керування курсами, уроками, завданнями, зарахуваннями та оцінками [9-10].

Для організації взаємодії між клієнтською і серверною частинами використовується REST API. Такий підхід передбачає передавання даних між React-застосунком і Spring Boot-сервером за допомогою HTTP-запитів. Клієнтська частина надсилає запити на отримання, створення, редагування або

видалення даних, а серверна частина обробляє ці запити та повертає відповідь у структурованому форматі. REST API дозволяє чітко розмежувати відповідальність між frontend і backend частинами системи та забезпечує можливість подальшого розвитку програмного продукту.

Для роботи з базою даних у серверній частині використовується Spring Data JPA та Hibernate. Spring Data JPA спрощує реалізацію доступу до даних за рахунок використання репозиторіїв, а Hibernate забезпечує об'єктно-реляційне відображення між класами Java та таблицями бази даних. Такий підхід дозволяє працювати з даними на рівні об'єктів, що робить програмний код більш зрозумілим та структурованим. У системі дистанційного навчання за допомогою JPA-сутностей описуються користувачі, курси, уроки, зарахування студентів, завдання та відповіді на завдання [11].

Для збереження даних обрано систему керування базами даних MySQL. Вона забезпечує надійне зберігання структурованої інформації, підтримує реляційну модель даних та є придатною для реалізації інформаційних систем із великою кількістю взаємопов'язаних сутностей. У розроблюваній системі MySQL використовується для збереження облікових записів користувачів, навчальних курсів, уроків, завдань, зарахувань студентів, відповідей на завдання, оцінок і коментарів викладачів [12]. Характеристику основних технологій, використаних під час розробки системи, наведено в таблиці 2.1.

Таблиця 2.1 – Технології, використані для розробки системи

Компонент	Технологія	Призначення
Frontend	React, Vite	Інтерфейс користувача, кабінети ролей, робота з REST API
UI icons	lucide-react	Іконки для навігації, дій і статусів
Backend	Spring Boot 2.7	REST API, бізнес-логіка, робота з базою даних
Persistence	Spring Data JPA, Hibernate	ORM-мапінг сутностей LMS у таблиці MySQL
Database	MySQL	Зберігання користувачів, курсів, уроків, завдань і оцінок
Password hashing	BCrypt	Збереження паролів у вигляді хешів
Build tools	Maven, npm	Збирання backend і frontend частин

Важливим аспектом розробки системи дистанційного навчання є забезпечення безпеки облікових записів користувачів. Для цього у системі використовується хешування паролів за допомогою BCrypt. Застосування такого підходу дозволяє не зберігати паролі користувачів у відкритому вигляді. Під час реєстрації або створення нового користувача пароль перетворюється на хешоване значення, яке зберігається в базі даних. Під час авторизації введений користувачем пароль порівнюється з хешем, що дозволяє перевірити правильність облікових даних без розкриття самого пароля.

Архітектура серверної частини системи побудована з урахуванням розподілу відповідальності між окремими логічними рівнями. Рівень контролерів відповідає за приймання HTTP-запитів від клієнтської частини та повернення відповідей. Сервісний рівень реалізує бізнес-логіку системи, зокрема перевірку прав доступу, обробку курсів, створення завдань, зарахування студентів і виставлення оцінок. Рівень репозиторіїв забезпечує доступ до бази даних, а рівень моделей описує основні сутності предметної області. Така структура підвищує зрозумілість програмного коду та спрощує його подальшу підтримку.

Під час розробки системи було використано алгоритм авторизації користувача. Його суть полягає в тому, що користувач вводить електронну пошту та пароль у формі входу, після чого клієнтська частина надсилає ці дані на сервер. Сервер здійснює пошук користувача за електронною поштою, перевіряє відповідність введеного пароля збереженому хешу та, у разі успішної перевірки, повертає інформацію про користувача і його роль. На основі отриманої ролі система відкриває відповідний інтерфейс: кабінет студента, викладача або адміністратора. Якщо облікові дані введено неправильно, користувач отримує повідомлення про помилку авторизації.

Окрему роль у функціонуванні системи відіграє алгоритм зарахування студента на курс. Якщо курс є відкритим, студент може самостійно записатися на нього через каталог курсів. У цьому випадку система перевіряє, чи не був студент уже зарахований на відповідний курс, після чого створює запис про

зарахування в базі даних. Якщо курс є закритим, зарахування здійснює викладач. Для цього викладач обирає потрібний курс і студента, після чого система створює відповідний запис у таблиці зарахувань. Такий підхід дозволяє підтримувати як відкриту, так і контрольовану модель доступу до навчальних курсів.

Для реалізації навчального процесу важливим є алгоритм створення завдань. Викладач у межах конкретного курсу заповнює назву завдання, опис, тип, максимальний бал і кінцевий термін виконання. Після надсилання форми дані передаються на сервер, де перевіряються на коректність і зберігаються в базі даних. Система підтримує різні типи завдань, зокрема теоретичні, лабораторні та тестові, що дозволяє адаптувати її до різних форматів навчальної діяльності.

Алгоритм подання відповіді студентом передбачає вибір завдання, введення текстової відповіді та надсилання її до системи. Після цього сервер перевіряє, чи має студент доступ до відповідного курсу та завдання, а також чи не була відповідь уже подана раніше. У разі успішної перевірки відповідь зберігається в базі даних зі статусом очікування перевірки. Надалі викладач може переглянути подану роботу, виставити оцінку та додати коментар.

Алгоритм оцінювання робіт забезпечує взаємодію між викладачем і студентом у межах навчального процесу. Викладач відкриває список поданих відповідей, переглядає зміст роботи, вказує кількість балів і додає текстовий відгук. Після збереження результатів система змінює статус роботи та робить оцінку доступною для студента. Студент у своєму кабінеті може переглянути отриманий бал, коментар викладача та загальний стан виконання завдань.

Для реалізації адміністративних функцій використовується алгоритм керування користувачами. Адміністратор має можливість створювати нові облікові записи, редагувати контактні дані, змінювати ролі користувачів і видаляти неактуальні записи. Завдяки цьому забезпечується централізоване адміністрування системи та контроль за складом її користувачів.

Обрані технології та алгоритми відповідають поставленим завданням кваліфікаційної роботи, оскільки дозволяють реалізувати повноцінну систему

дистанційного навчання з підтримкою різних ролей користувачів, управлінням курсами, уроками, завданнями та оцінками. Використання React забезпечує зручний та динамічний інтерфейс користувача, Spring Boot відповідає за надійну серверну логіку, MySQL забезпечує структуроване зберігання даних, а REST API організовує ефективну взаємодію між клієнтською та серверною частинами системи.

Таким чином, вибір засобів, методів і алгоритмів розробки є обґрунтованим та відповідає особливостям предметної області. Запропонований технологічний стек дозволяє створити масштабовану, зручну та функціональну систему дистанційного навчання, яка може використовуватися для організації освітнього процесу в електронному середовищі.

РОЗДІЛ 3

РОЗРОБКА СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ

3.1 Практична реалізація об'єкта проєктування

Практична реалізація системи дистанційного навчання виконана із використанням сучасних вебтехнологій та клієнт-серверної архітектури. Основною метою реалізації було створення програмного продукту, який забезпечує ефективну взаємодію між студентами, викладачами та адміністраторами, а також підтримує всі основні процеси організації дистанційного навчання. Під час розробки системи для реалізації клієнтської частини використано бібліотеку React та інструмент Vite, які забезпечують швидке створення сучасних користувацьких інтерфейсів і високу продуктивність вебзастосунку. Серверну частину реалізовано на базі фреймворку Spring Boot, який відповідає за обробку запитів користувачів, виконання бізнес-логіки та взаємодію з базою даних. Для зберігання інформації використано систему керування базами даних MySQL.

Загальна архітектура системи побудована за принципом розподілу функціональності між окремими компонентами. Клієнтська частина працює у веббраузері користувача та забезпечує відображення інтерфейсу, введення даних і взаємодію з сервером. Серверна частина реалізує REST API, здійснює обробку запитів, перевірку прав доступу та збереження даних у базі даних (табл. 3.1).

Таблиця 3.1 – Backend структура

Пакет	Опис
model	JPA-сутності: User, Course, Lesson, Enrollment, Assignment, Submission
repository	Spring Data JPA репозиторії для доступу до таблиць
service	Бізнес-логіка: курси, зарахування, завдання, оцінювання
controller	REST API контролери для frontend
dto	Об'єкти запитів для створення і редагування даних
config	CORS-конфігурація і початкові демо-дані

У процесі реалізації програмного забезпечення було створено окремі модулі для роботи з користувачами, курсами, уроками, завданнями,

зарахуваннями та результатами навчання. Такий підхід дозволив забезпечити логічне розділення функціональності системи та спростити подальший супровід програмного коду.

Після запуску програмного забезпечення користувачеві відображається сторінка авторизації, яка забезпечує вхід до системи відповідно до ролі користувача. Для входу необхідно вказати електронну пошту та пароль, після чого система перевіряє введені дані та надає доступ до відповідного функціоналу.

Для реалізації механізму авторизації та реєстрації користувачів у системі було створено окремий REST-контролер. Контролер забезпечує обробку запитів від клієнтської частини та передає їх до сервісного рівня для подальшої обробки (ліст. 3.1).

Лістинг 3.1 – Контролер авторизації користувачів

```
@RestController
@RequestMapping("/api/auth")
public class AuthController {
    private final AuthService authService;
    public AuthController(AuthService authService) {
        this.authService = authService;
    }
    @PostMapping("/register")
    public User register(@RequestBody RegisterRequest request) {
        return authService.register(request);
    }
    @PostMapping("/login")
    public User login(@RequestBody LoginRequest request) {
        return authService.login(request);
    }
}
```

кінець лістингу 3.1

Для зберігання інформації про користувачів використовується окрема сутність User, яка відповідає таблиці користувачів у базі даних та містить основні атрибути облікового запису (ліст. 3.2).

Лістинг 3.2 – Сутність користувача системи

```
@Entity
@Table(name = "users")
public class User {
    @Id
```

```

@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
@Column(nullable = false)
private String fullName;
@Column(nullable = false, unique = true)
private String email;
@JsonIgnore
@Column(nullable = false)
private String passwordHash;
@Enumerated(EnumType.STRING)
private Role role;
}

```

кінець лістингу 3.2

Після успішної авторизації студент потрапляє до власного кабінету, де відображається інформація про доступні курси, кількість активних навчальних дисциплін, результати навчання та швидкий доступ до основних функцій системи (рис. 3.1).

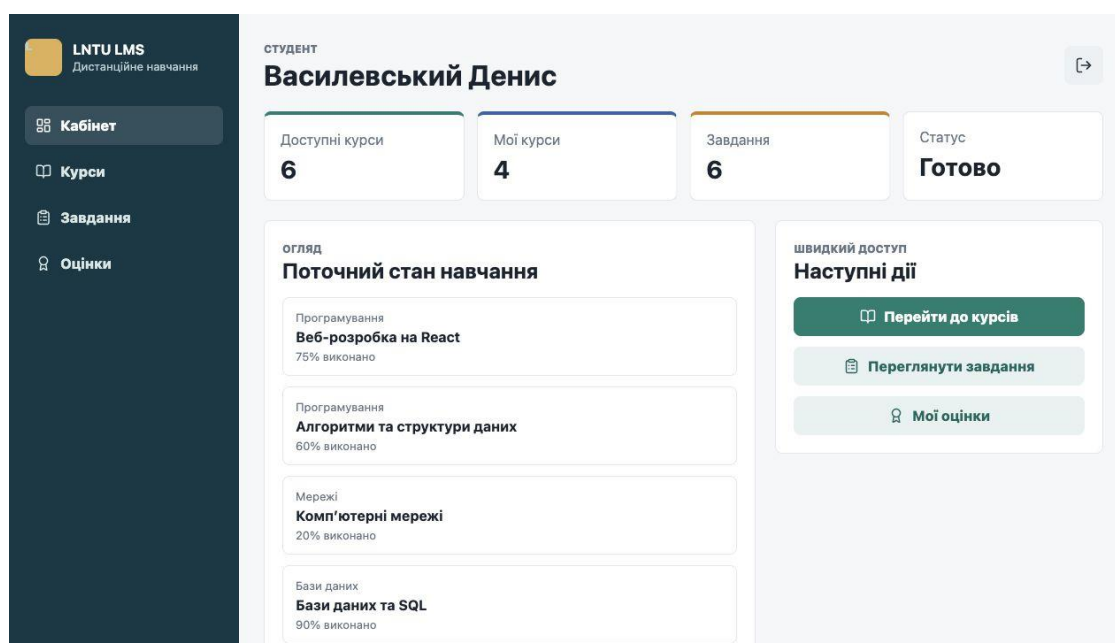


Рисунок 3.1 – Кабінет студента

Для перегляду навчальних курсів реалізовано окрему сторінку каталогу курсів. На цій сторінці студент може ознайомитися з описом курсів, переглянути інформацію про викладача та виконати запис на відкритий курс. Сторінку перегляду курсів наведено на рисунку 3.2.

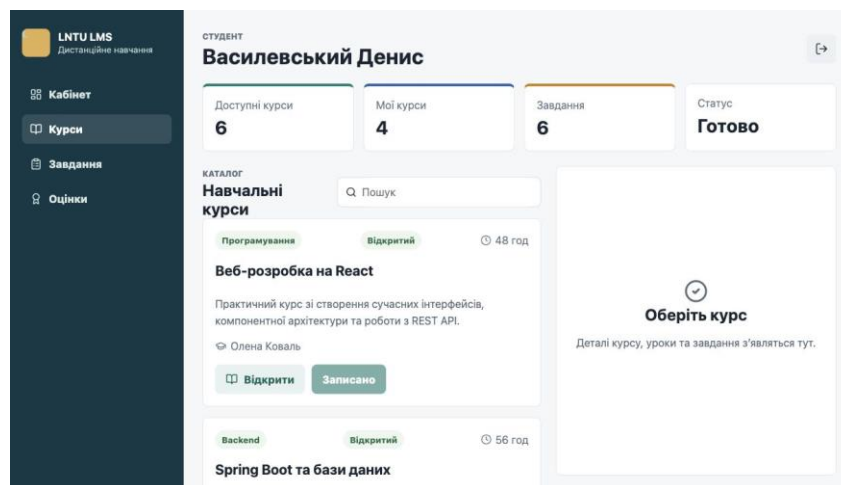


Рисунок 3.2 – Перегляд навчальних курсів

Для відображення списку курсів клієнтська частина взаємодіє із сервером через REST API. Після відкриття сторінки формується HTTP-запит до серверної частини, яка повертає перелік доступних курсів у форматі JSON. Отримані дані використовуються для формування інтерфейсу користувача. Фрагмент програмного коду наведено у лістингу 3.3.

Лістинг 3.3 – Отримання списку курсів у React

```
useEffect(() => {
  fetch("http://localhost:8080/api/courses")
    .then(response => response.json())
    .then(data => setCourses(data))
    .catch(error => console.error(error));
}, []);
```

кінець лістингу 3.3

Для роботи з навчальними курсами реалізовано окремий контролер, який забезпечує отримання інформації про курси та виконання основних операцій керування ними (ліст. 3.4).

Лістинг 3.4 – Контролер роботи з курсами

```
@RestController
@RequestMapping("/api/courses")
public class CourseController {
  private final CourseService courseService;
  public CourseController(CourseService courseService) {
    this.courseService = courseService;
  }
  @GetMapping
  public List<Course> getCourses() {
```

```

        return courseService.getCourses();
    }
    @PostMapping
    public Course createCourse(@RequestBody CourseRequest request) {
        return courseService.createCourse(request);
    }
}

```

кінець лістингу 3.4

Основна бізнес-логіка роботи з курсами реалізована на сервісному рівні. Сервіс відповідає за обробку даних та взаємодію між контролерами і базою даних (ліст. 3.5).

Лістинг 3.5 – Сервіс роботи з курсами

```

@Service
public class CourseService {
    private final CourseRepository courseRepository;
    private final UserRepository userRepository;
    public CourseService(
        CourseRepository courseRepository,
        UserRepository userRepository) {
        this.courseRepository = courseRepository;
        this.userRepository = userRepository;
    }
}

```

кінець лістингу 3.5

Викладач після авторизації отримує доступ до власного робочого кабінету. У цьому розділі реалізовано функції створення нових курсів, редагування навчальних матеріалів, формування завдань та перевірки робіт студентів. Інтерфейс кабінету викладача представлено на рисунку 3.3.

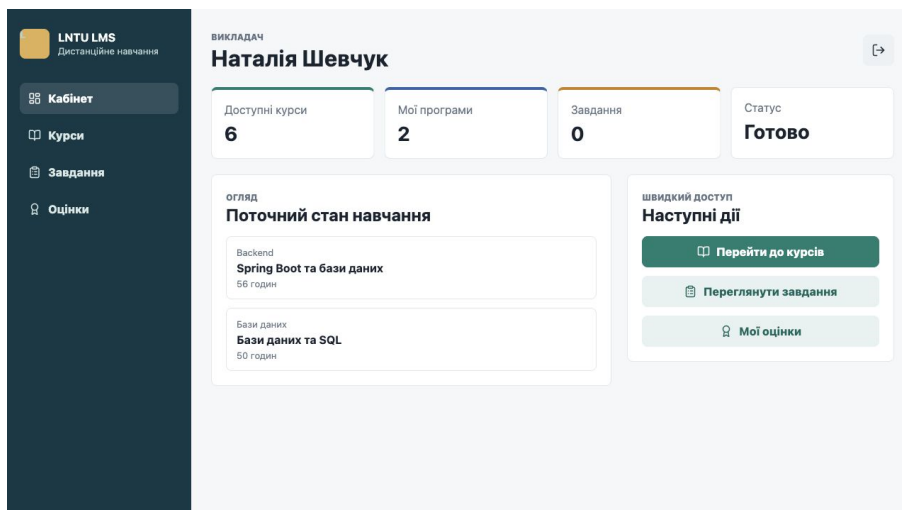


Рисунок 3.3 – Кабінет викладача

Для керування окремим курсом реалізовано сторінку редагування курсу. Викладач має можливість додавати уроки, створювати завдання, переглядати список зарахованих студентів та контролювати процес навчання. Інтерфейс керування курсом наведено на рисунку 3.4.

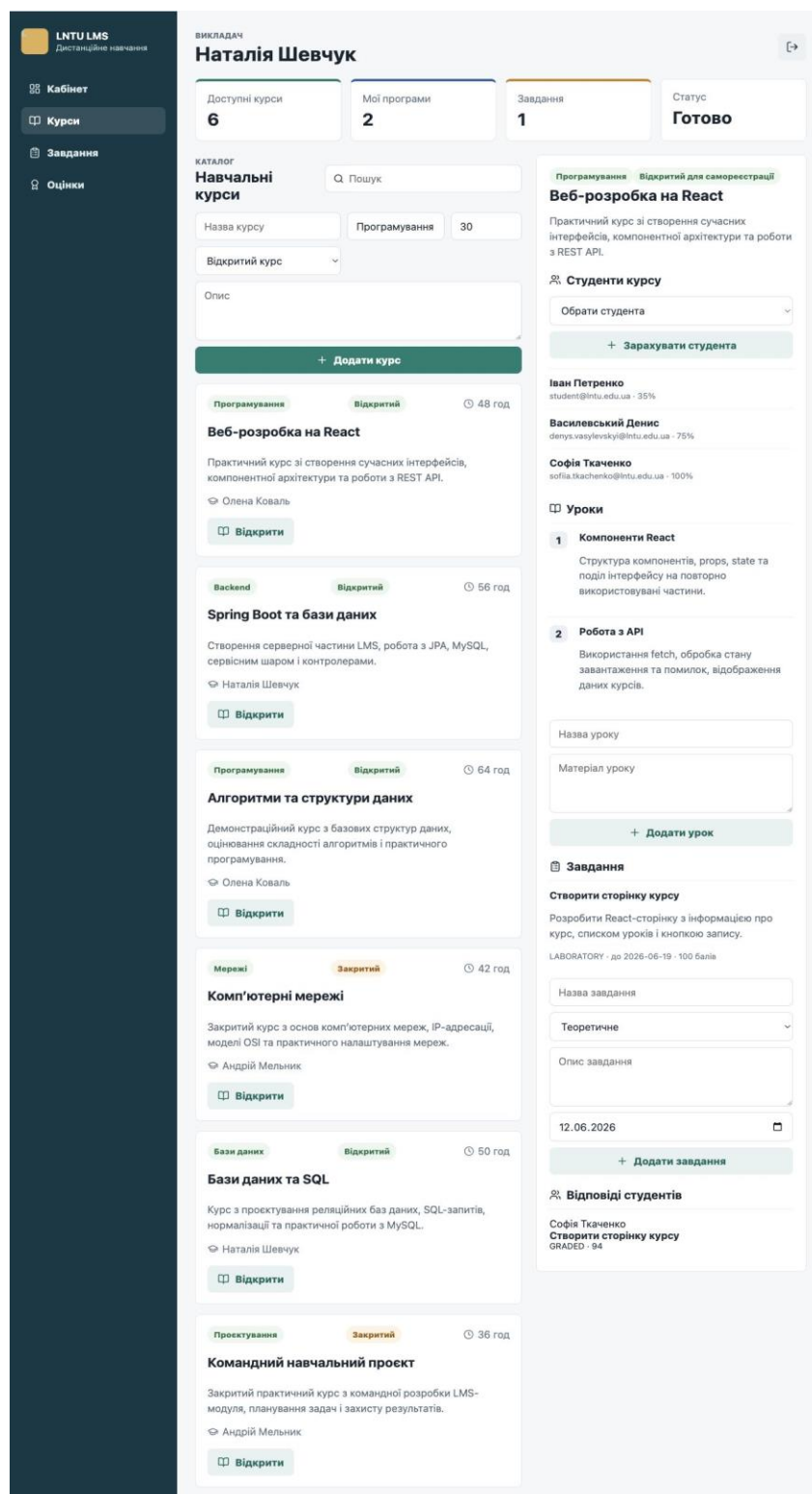


Рисунок 3.4 – Керування навчальним курсом

Однією з основних функцій системи є створення та оцінювання завдань. Під час створення завдання викладач вказує його назву, опис, тип завдання, максимальний бал та кінцевий термін виконання. Після надсилання форми інформація зберігається у базі даних і стає доступною для студентів відповідного курсу.

Процес оцінювання робіт реалізований через окремий інтерфейс, який дозволяє переглядати відповіді студентів, виставляти оцінки та додавати текстові коментарі. Інтерфейс оцінювання робіт представлено на рисунку 3.5.

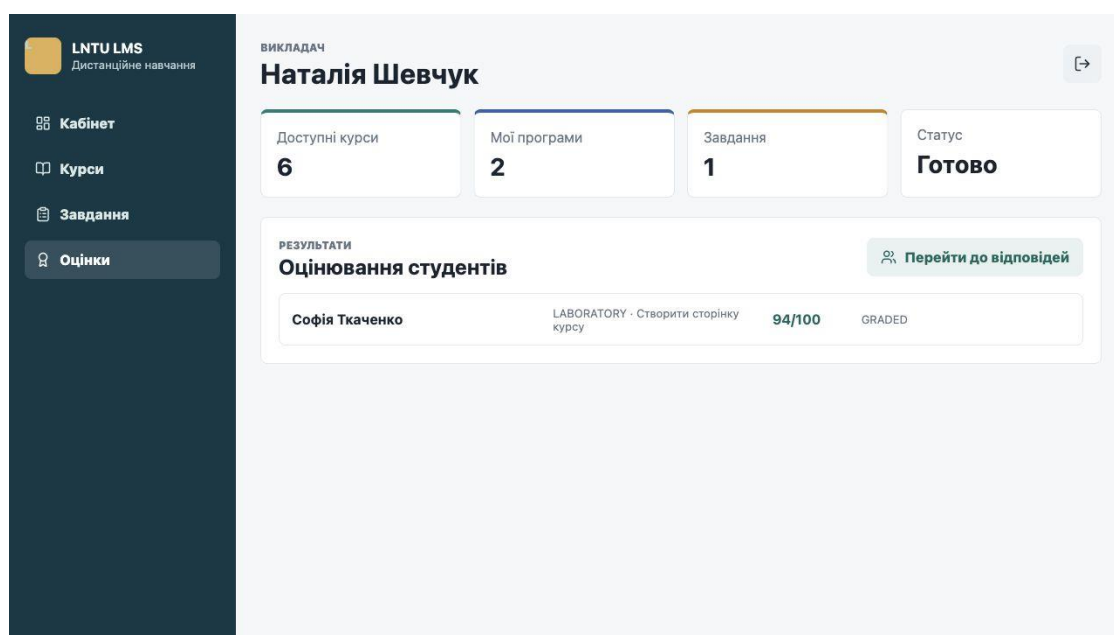


Рисунок 3.5 – Оцінювання відповідей студентів

Для адміністрування системи реалізовано окрему панель керування користувачами. Адміністратор має можливість створювати нові облікові записи, редагувати інформацію про користувачів, змінювати ролі та видаляти неактуальні записи. Інтерфейс адміністрування користувачів наведено на рисунку 3.6.

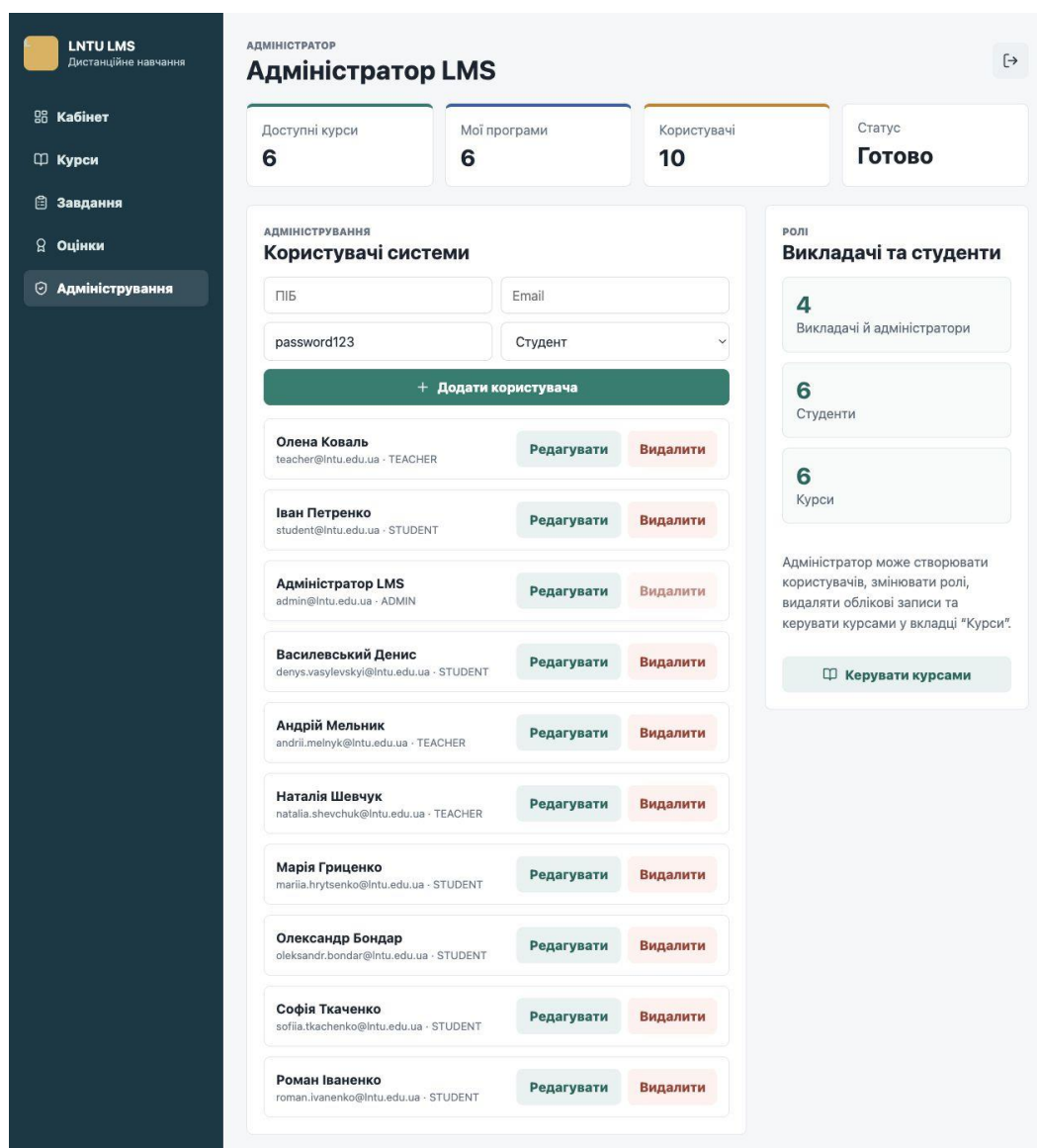


Рисунок 3.6 – Панель адміністрування користувачів

У процесі реалізації програмного забезпечення також було розроблено серверну частину системи, яка містить контролери, сервіси, репозиторії та моделі даних. Контролери відповідають за приймання HTTP-запитів, сервіси реалізують бізнес-логіку системи, репозиторії забезпечують доступ до бази даних, а моделі відображають основні сутності предметної області.

Для організації взаємодії із базою даних використовується механізм репозиторіїв Spring Data JPA. Репозиторії забезпечують виконання операцій пошуку, створення, редагування та видалення даних без необхідності написання великої кількості SQL-запитів. Приклад реалізації репозиторію користувачів наведено у лістингу 3.6.

Лістинг 3.6 – Репозиторій користувачів

```
@Repository
public interface UserRepository
    extends JpaRepository<User, Long> {

    Optional<User> findByEmail(String email);

    boolean existsByEmail(String email);
}
```

кінець лістингу 3.6

3.2 Розробка бази даних

Одним із найважливіших етапів створення системи дистанційного навчання є проєктування та реалізація бази даних. Від правильності побудови структури бази даних залежить ефективність зберігання інформації, швидкість доступу до даних, цілісність інформаційних ресурсів та можливість подальшого розширення функціональності програмного забезпечення.

Для зберігання даних у розроблюваній системі використовується система керування базами даних MySQL. Дане рішення забезпечує підтримку реляційної моделі даних, дозволяє ефективно працювати з великими обсягами інформації та підтримує механізми забезпечення цілісності даних за допомогою первинних і зовнішніх ключів.

На основі проведеного аналізу предметної області було визначено основні сутності, необхідні для функціонування системи дистанційного навчання. До них належать користувачі, навчальні курси, уроки, зарахування студентів, завдання та відповіді на завдання. Кожна із зазначених сутностей відповідає окремій таблиці бази даних і забезпечує збереження необхідної інформації для реалізації навчального процесу.

Логічну структуру бази даних системи дистанційного навчання представлено на ER-діаграмі, наведеній на рисунку 3.7.

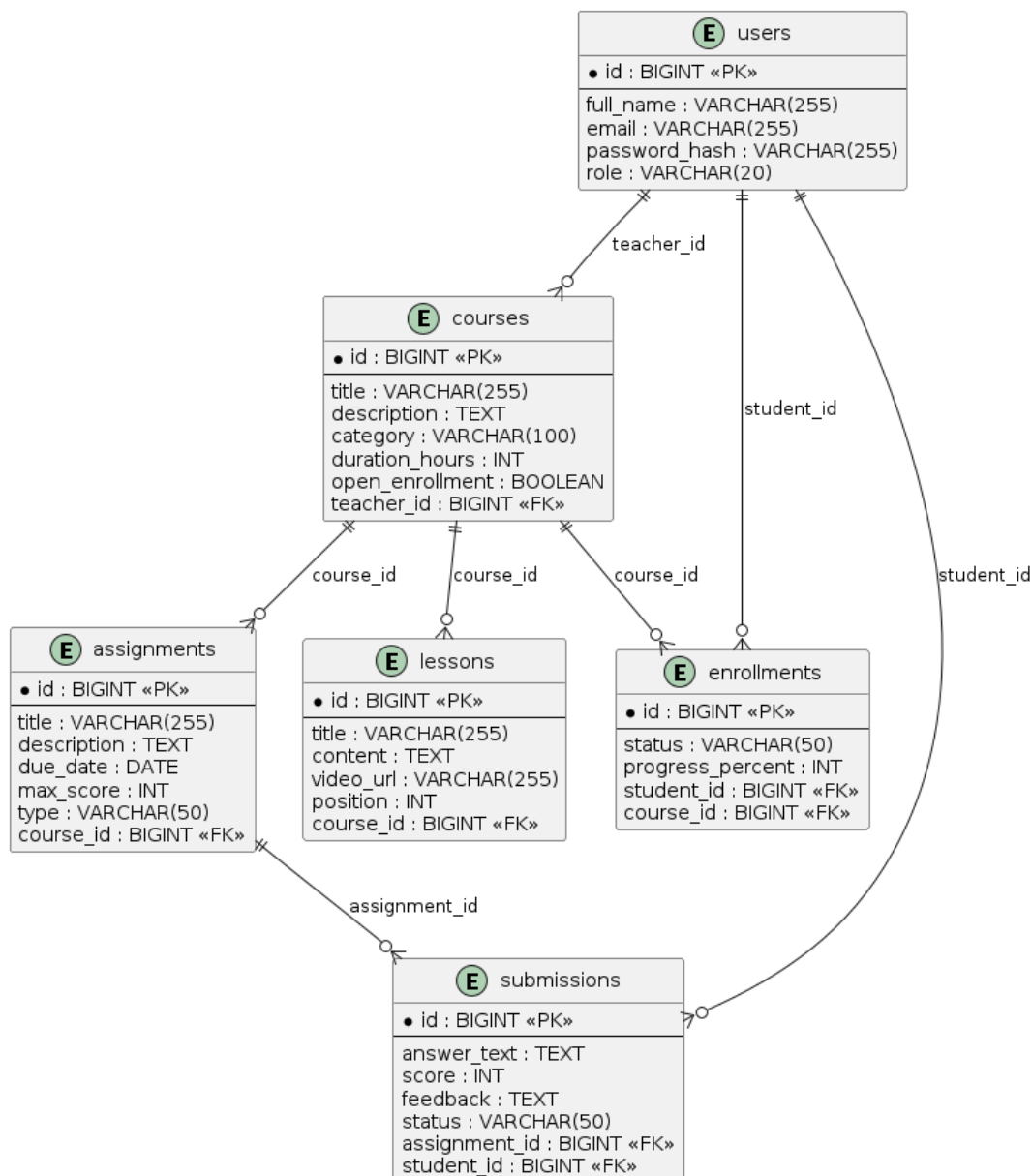


Рисунок 3.7 – ER-діаграма бази даних системи дистанційного навчання

Центральною таблицею бази даних є таблиця користувачів Users. Вона містить інформацію про всіх зареєстрованих користувачів системи незалежно від їхньої ролі. Для кожного користувача зберігаються унікальний ідентифікатор, прізвище та ім'я, адреса електронної пошти, пароль у хешованому вигляді та роль користувача. Структуру таблиці користувачів наведено в таблиці 3.2.

Таблиця 3.2 – Структура таблиці Users

Поле	Тип даних	Призначення
id	BIGINT	Унікальний ідентифікатор
full_name	VARCHAR(255)	ПІБ користувача
email	VARCHAR(255)	Електронна пошта
password_hash	VARCHAR(255)	Хеш пароля
role	VARCHAR(20)	Роль користувача

Для зберігання інформації про навчальні курси використовується таблиця Courses. У ній містяться назва курсу, опис, категорія, тривалість навчання, інформація про відкритість курсу та викладача, який його створив. Структуру таблиці курсів наведено в таблиці 3.3.

Таблиця 3.3 – Структура таблиці Courses

Поле	Тип даних	Призначення
id	BIGINT	Ідентифікатор курсу
title	VARCHAR(255)	Назва курсу
description	TEXT	Опис курсу
category	VARCHAR(100)	Категорія
duration_hours	INTEGER	Тривалість
open_enrollment	BOOLEAN	Ознака відкритого курсу
teacher_id	BIGINT	Ідентифікатор викладача

Кожен навчальний курс може містити декілька уроків. Для їх зберігання використовується таблиця Lessons. Вона містить назву уроку, текст навчального матеріалу, посилання на відеоматеріали та порядковий номер уроку в межах курсу. Зв'язок між курсами та уроками реалізовано за принципом «один до багатьох», тобто одному курсу може відповідати багато уроків, тоді як кожен урок належить лише одному курсу. Для обліку студентів, які проходять навчання на курсах, використовується таблиця Enrollments. У цій таблиці зберігається інформація про зарахування студента на курс, його поточний статус та відсоток виконання програми навчання. Реалізація таблиці Enrollments дозволяє організувати зв'язок типу «багато до багатьох» між користувачами та курсами. Один студент може навчатися на декількох курсах одночасно, а кожен курс може містити довільну кількість студентів. Для реалізації навчальних завдань використовується таблиця Assignments. У ній зберігаються назва завдання, його опис, тип завдання, максимальний бал та кінцевий термін виконання.

Кожне завдання належить певному курсу та може бути виконане багатьма студентами. Завдяки цьому забезпечується можливість організації теоретичних, лабораторних і тестових завдань у межах навчального процесу.

Для збереження результатів виконання завдань студентами використовується таблиця Submissions. Вона містить текст відповіді студента, отриманий бал, коментар викладача та поточний статус перевірки роботи.

Зв'язок між таблицями Assignments та Submissions реалізований за принципом «один до багатьох», оскільки одному завданню можуть відповідати відповіді багатьох студентів. Крім того, таблиця Submissions містить зовнішній ключ на таблицю Users, що дозволяє визначити автора кожної роботи.

Під час реалізації бази даних особливу увагу було приділено забезпеченню цілісності інформації. Для цього використовуються первинні ключі, які забезпечують унікальність записів, та зовнішні ключі, що реалізують взаємозв'язки між таблицями. Завдяки цьому виключається можливість появи некоректних записів або втрати зв'язків між даними.

Для автоматизації роботи з базою даних у серверній частині застосунку використовуються технології Spring Data JPA та Hibernate. Кожна таблиця бази даних відповідає окремій Java-сутності, що дозволяє працювати з даними на рівні об'єктів без необхідності написання великої кількості SQL-запитів. Приклад реалізації сутності навчального курсу наведено у лістингу 3.7.

Лістинг 3.7 – Сутність Course

```
@Entity
@Table(name = "courses")
public class Course {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String title;
    private String description;
    private String category;
    private Integer durationHours;
    private Boolean openEnrollment;
}
```

кінець лістингу 3.7

Для реалізації зв'язків між таблицями використовуються відповідні JPA-анотації. Це дозволяє автоматично формувати структуру бази даних та підтримувати цілісність зв'язків між сутностями.

Приклад опису зв'язку між курсом та уроками наведено у лістингу 3.8.

Лістинг 3.8 – Зв'язок між Course та Lesson

```
@OneToMany(mappedBy = "course")
private List<Lesson> lessons;
```

кінець лістингу 3.8

Таким чином, у результаті проєктування та реалізації бази даних було створено структуру, яка повністю забезпечує потреби системи дистанційного навчання. Розроблена база даних підтримує зберігання інформації про користувачів, курси, уроки, завдання, відповіді студентів та результати навчання, а також забезпечує цілісність і надійність збереження даних.

3.3 Захист інформаційно-комп'ютерної системи

Одним із важливих аспектів розробки системи дистанційного навчання є забезпечення належного рівня захисту інформації. Оскільки система працює з персональними даними користувачів, результатами навчання, навчальними матеріалами та адміністративною інформацією, необхідно реалізувати комплекс заходів, спрямованих на запобігання несанкціонованому доступу, втраті або модифікації даних [13].

Під час проєктування системи було враховано основні принципи інформаційної безпеки, серед яких конфіденційність, цілісність та доступність інформації. Конфіденційність забезпечує доступ до даних лише уповноваженим користувачам, цілісність гарантує захист інформації від несанкціонованих змін, а доступність забезпечує можливість використання системи відповідно до прав користувачів.

Одним із головних механізмів захисту системи є автентифікація користувачів. Перед отриманням доступу до функціональних можливостей

програмного забезпечення користувач повинен пройти процедуру авторизації шляхом введення адреси електронної пошти та пароля. Серверна частина системи виконує перевірку введених облікових даних і лише після успішної ідентифікації надає доступ до відповідних ресурсів. Процес авторизації користувача в системі представлено на рисунку 3.8.

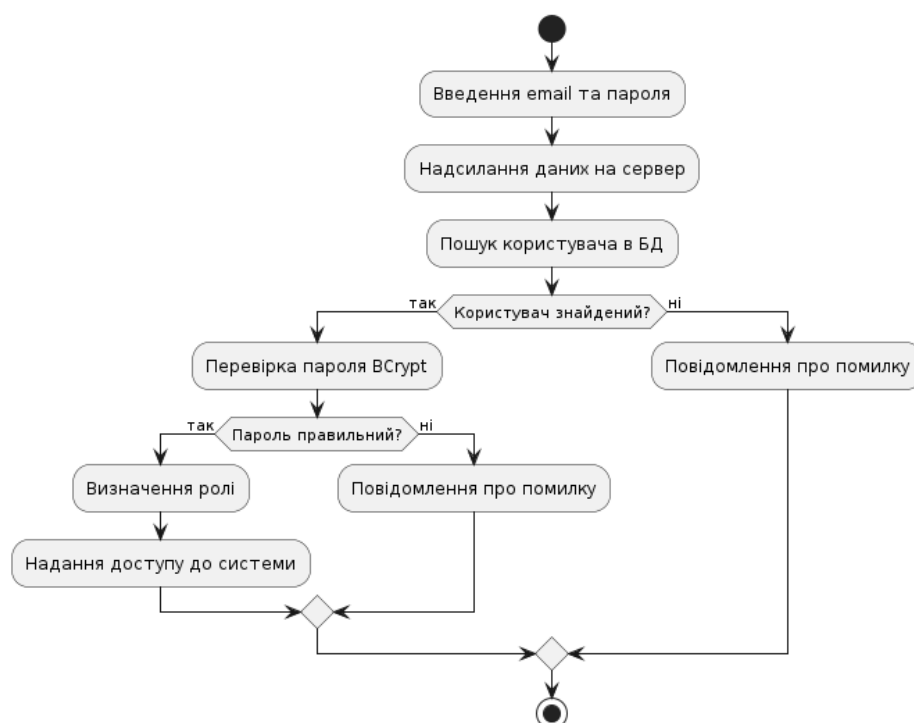


Рисунок 3.8 – Схема авторизації користувача

Для підвищення рівня безпеки паролі користувачів не зберігаються у відкритому вигляді. Під час створення нового облікового запису або зміни пароля використовується алгоритм хешування BCrypt. Даний алгоритм дозволяє перетворити пароль у спеціальний хешований рядок, який зберігається у базі даних. Під час входу до системи виконується порівняння введеного пароля із збереженим хешем без необхідності розкриття самого пароля.

Застосування BCrypt значно підвищує рівень захисту облікових записів користувачів навіть у випадку отримання сторонніми особами доступу до бази даних. Завдяки використанню сольових значень та багаторазового шифрування алгоритм забезпечує стійкість до атак перебором і словникових атак.

Важливим елементом захисту інформаційної системи є реалізація рольового контролю доступу. У межах розробленої системи використовується три основні ролі користувачів: студент, викладач та адміністратор. Кожна роль має власний набір дозволених функцій та обмежень.

Студент має доступ лише до власних курсів, навчальних матеріалів, завдань та результатів навчання. Він не може змінювати навчальні матеріали, створювати курси або переглядати дані інших користувачів.

Викладач має право створювати та редагувати навчальні курси, додавати уроки, формувати завдання, перевіряти відповіді студентів і виставляти оцінки. При цьому викладач працює лише з тими курсами, які належать йому або були призначені для адміністрування.

Адміністратор має найвищий рівень доступу до системи. Він здійснює керування користувачами, змінює ролі, контролює структуру навчальних курсів та забезпечує працездатність програмного забезпечення. Розподіл прав доступу між категоріями користувачів наведено на рисунку 3.9.

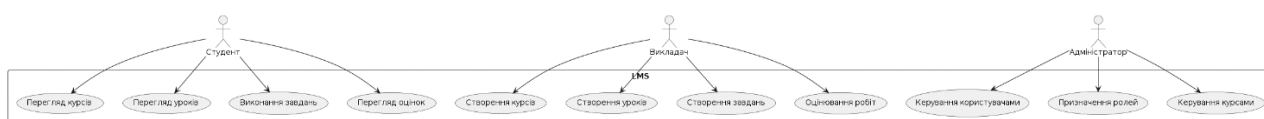


Рисунок 3.9 – Рольова модель доступу системи

Для захисту серверної частини програмного забезпечення реалізовано перевірку прав доступу під час виконання кожного запиту. Перед виконанням операції система перевіряє роль користувача та визначає, чи має він право виконувати відповідну дію. У разі порушення політики доступу сервер повертає повідомлення про відмову в доступі.

Особливу увагу приділено захисту бази даних від помилкового або несанкціонованого втручання. Для забезпечення цілісності даних використовуються первинні та зовнішні ключі, які підтримують коректність взаємозв'язків між таблицями. Додатково реалізовано перевірку коректності введених даних перед їх збереженням у базі даних.

У системі застосовується механізм валідації даних як на стороні клієнта, так і на стороні сервера. На клієнтському рівні перевіряється правильність заповнення форм, формат електронної пошти та обов'язковість введення необхідних полів. Серверна частина виконує додаткову перевірку отриманих даних, що дозволяє уникнути помилок та підвищує стійкість системи до спроб передачі некоректної інформації.

Для запобігання втраті даних передбачена можливість резервного копіювання бази даних. Створення резервних копій дозволяє відновити працездатність системи у випадку апаратних збоїв, пошкодження бази даних або інших непередбачуваних ситуацій. Використання резервного копіювання є важливою складовою забезпечення безперервності роботи інформаційної системи.

Оскільки система функціонує як вебзастосунок, значну роль відіграє захист мережевої взаємодії між клієнтською та серверною частинами. Передача даних між компонентами системи здійснюється через REST API, що дозволяє централізовано контролювати доступ до ресурсів та забезпечувати перевірку всіх вхідних запитів. Завдяки цьому зменшується ризик несанкціонованого доступу до функціональних можливостей системи.

Розроблені механізми захисту забезпечують належний рівень безпеки системи дистанційного навчання та дозволяють захистити персональні дані користувачів, результати навчання та інші інформаційні ресурси від випадкового або навмисного пошкодження. Реалізація автентифікації, рольового контролю доступу, хешування паролів, перевірки даних та резервного копіювання створює комплексну систему захисту, яка відповідає вимогам сучасних веборієнтованих інформаційних систем.

Таким чином, у процесі розробки було реалізовано основні механізми захисту інформаційно-комп'ютерної системи, що забезпечують безпечне функціонування програмного продукту та надійне зберігання інформації в електронному освітньому середовищі.

3.4 Тестування та налагодження інформаційно-комп'ютерної системи

Завершальним етапом розробки системи дистанційного навчання є проведення тестування та налагодження програмного забезпечення. Основною метою тестування є перевірка працездатності всіх функціональних компонентів системи, виявлення можливих помилок та підтвердження відповідності програмного продукту поставленим вимогам.

Під час тестування особлива увага приділялася перевірці основних функцій системи, зокрема авторизації користувачів, керуванню навчальними курсами, роботі з навчальними матеріалами, створенню та виконанню завдань, оцінюванню результатів навчання, а також функціям адміністрування користувачів [14].

Для перевірки коректності роботи програмного забезпечення використовувалося функціональне тестування, яке дозволяє оцінити правильність виконання системою поставлених завдань з точки зору кінцевого користувача. Крім того, виконувалося інтеграційне тестування взаємодії між клієнтською частиною, сервером застосунку та базою даних.

Перед початком тестування було сформовано набір тестових сценаріїв, які охоплюють основні бізнес-процеси системи дистанційного навчання. Результати проведеного тестування наведено в таблиці 3.4.

Таблиця 3.4 – Результати функціонального тестування системи

№	Тестовий сценарій	Очікуваний результат	Отриманий результат
1	Авторизація користувача з коректними даними	Успішний вхід до системи	Виконано успішно
2	Авторизація користувача з некоректним паролем	Повідомлення про помилку	Виконано успішно
3	Реєстрація нового користувача	Створення нового облікового запису	Виконано успішно
4	Створення нового курсу викладачем	Курс збережено в системі	Виконано успішно
5	Запис студента на відкритий курс	Студента зараховано на курс	Виконано успішно
6	Створення нового завдання	Завдання доступне студентам	Виконано успішно
7	Подання відповіді на завдання	Робота збережена в системі	Виконано успішно

Продовження таблиці 3.4

№	Тестовий сценарій	Очікуваний результат	Отриманий результат
8	Оцінювання роботи студентів	Оцінка відображається студенту	Виконано успішно
9	Редагування користувача адміністратором	Дані користувача оновлено	Виконано успішно
10	Видалення курсу адміністратором	Курс видалено з системи	Виконано успішно

У процесі тестування було перевірено коректність роботи механізму авторизації. Користувачі різних ролей успішно проходили процедуру входу до системи та отримували доступ лише до тих функцій, які передбачені їхніми правами доступу. Також підтверджено правильність роботи механізму хешування паролів і перевірки облікових даних.

Для перевірки функціональності навчальних курсів виконувалося створення нових курсів, редагування їхніх параметрів, додавання уроків та завдань. Результати тестування показали, що всі операції виконуються коректно, а внесені зміни миттєво відображаються в інтерфейсі користувача.

Особлива увага приділялася перевірці процесу подання та оцінювання завдань. Було протестовано сценарій, у межах якого студент обирає курс, відкриває завдання, надсилає відповідь, а викладач переглядає роботу та виставляє оцінку. Після завершення перевірки оцінка та коментар успішно відображалися в особистому кабінеті студента.

Для підтвердження коректної взаємодії між компонентами системи виконувалося інтеграційне тестування. У процесі тестування було перевірено обмін даними між клієнтською частиною React, сервером Spring Boot та базою даних MySQL. Отримані результати підтвердили правильність передавання даних та стабільність роботи системи під час виконання основних операцій.

Під час налагодження програмного забезпечення було усунуто незначні помилки, пов'язані з обробкою некоректних даних користувача та відображенням окремих елементів інтерфейсу. Після внесення необхідних змін система пройшла повторне тестування, яке підтвердило її стабільну роботу.

Для оцінки працездатності системи було виконано комплексне тестування за участю користувачів різних ролей. Студенти перевіряли функції перегляду курсів, виконання завдань та перегляду оцінок. Викладачі тестували створення курсів, формування навчальних матеріалів та оцінювання робіт. Адміністратори здійснювали перевірку функцій керування користувачами та курсами. Усі тестові сценарії були виконані успішно.

Отримані результати свідчать про те, що розроблена система дистанційного навчання коректно виконує покладені на неї функції та відповідає сформованим вимогам. Проведене тестування підтвердило працездатність програмного забезпечення, правильність реалізації бізнес-логіки та стабільність взаємодії між усіма компонентами системи.

Таким чином, у результаті проведення тестування та налагодження було підтверджено готовність системи дистанційного навчання до практичного використання. Усі основні функції працюють коректно, а виявлені під час розробки недоліки були успішно усунені.

ВИСНОВКИ

У кваліфікаційній роботі було розглянуто питання розробки системи дистанційного навчання з використанням сучасних вебтехнологій React, Spring Boot та MySQL. Актуальність теми зумовлена зростанням потреби в ефективних програмних засобах організації освітнього процесу в електронному середовищі, автоматизації взаємодії між учасниками навчання та забезпеченні доступу до освітніх ресурсів незалежно від місця перебування користувачів.

У процесі виконання роботи було проведено аналіз сучасних підходів до організації дистанційного навчання та досліджено можливості існуючих систем керування навчанням. Виконаний аналіз дозволив визначити основні переваги та недоліки наявних програмних рішень і обґрунтувати доцільність створення власної системи дистанційного навчання.

У результаті дослідження було сформовано функціональні та нефункціональні вимоги до програмного забезпечення. Визначено основні ролі користувачів системи, їхні функціональні можливості та особливості взаємодії із програмним продуктом. Для моделювання структури системи та процесів її функціонування побудовано UML-діаграми варіантів використання, класів, діяльності та послідовності.

У роботі обґрунтовано вибір технологій React, Spring Boot, Spring Data JPA, Hibernate та MySQL для реалізації програмного забезпечення. Використання клієнт-серверної архітектури дозволило забезпечити розділення інтерфейсу користувача, бізнес-логіки та рівня доступу до даних, що сприяє підвищенню гнучкості та масштабованості системи.

У ході виконання роботи було спроектовано структуру програмного забезпечення та розроблено базу даних для зберігання інформації про користувачів, навчальні курси, уроки, зарахування студентів, завдання та результати навчання. Для моделювання структури даних побудовано ER-діаграму та реалізовано відповідні зв'язки між сутностями предметної області.

Практичним результатом роботи стала реалізована система дистанційного навчання, яка забезпечує реєстрацію та авторизацію користувачів, керування навчальними курсами, створення уроків і завдань, подання відповідей студентами, оцінювання результатів навчання та адміністрування користувачів. Розроблений програмний продукт підтримує роботу трьох категорій користувачів: студентів, викладачів та адміністраторів.

У системі реалізовано механізми захисту інформації, зокрема рольовий контроль доступу, перевірку прав користувачів, хешування паролів за допомогою алгоритму BCrypt та валідацію вхідних даних. Застосовані засоби захисту забезпечують безпечне функціонування системи та захист персональних даних користувачів.

Для перевірки працездатності програмного забезпечення проведено функціональне та інтеграційне тестування. Отримані результати підтвердили коректність роботи всіх основних функцій системи, стабільність взаємодії між клієнтською та серверною частинами, а також відповідність програмного продукту сформованим вимогам.

Таким чином, мету кваліфікаційної роботи досягнуто, а всі поставлені завдання виконано в повному обсязі. Розроблена система дистанційного навчання може використовуватися для організації освітнього процесу в закладах освіти та інших організаціях, що потребують ефективного управління навчальними курсами та контролю результатів навчання в електронному середовищі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. OECD. Digital Education Outlook 2023. Paris: OECD Publishing, 2023.
2. UNESCO. Global Education Monitoring Report 2024: Technology in Education. Paris : UNESCO Publishing, 2024.
3. Moodle Documentation. URL: <https://docs.moodle.org> (дата звернення: 10.03.2026).
4. Google Classroom Help Center. URL: <https://support.google.com/edu/classroom> (дата звернення: 10.04.2026).
5. Canvas LMS Guides. URL: <https://community.canvaslms.com> (дата звернення: 10.04.2026).
6. Object Management Group. Unified Modeling Language (UML) Specification Version 2.5.1. URL: <https://www.omg.org/spec/UML/2.5.1/> (дата звернення: 10.04.2026).
7. React Documentation. URL: <https://react.dev/> (дата звернення: 20.04.2026).
8. Vite Documentation. URL: <https://vitejs.dev/> (дата звернення: 20.04.2026).
9. Spring Boot Documentation. URL: <https://docs.spring.io/spring-boot/reference/> (дата звернення: 10.05.2026).
10. Spring Data JPA Documentation. URL: <https://spring.io/projects/spring-data-jpa> (дата звернення: 10.05.2026).
11. Hibernate ORM Documentation. URL: <https://hibernate.org/orm/documentation/> (дата звернення: 10.05.2026).
12. MySQL Documentation. URL: <https://dev.mysql.com/doc/> (дата звернення: 10.05.2026).
13. OWASP Foundation. OWASP Top 10: Web Application Security Risks. URL: <https://owasp.org/www-project-top-ten/> (дата звернення: 10.05.2026).
14. Postman Learning Center. API Testing Documentation. URL: <https://learning.postman.com/docs/tests-and-scripts/write-scripts/test-scripts/> (дата звернення: 10.05.2026).