МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ВЕБ ДОДАТКУ ДЛЯ ОНЛАЙН-ШКОЛИ»**

Виконав: студент 4 курсу, групи 6.1221-з
спеціальності 122 комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми комп'ютерні науки
(назва освітньої програми)
В.В. Виставкін
(ініціали та прізвище)
Керівник старший викладач кафедри комп'ютерних наук, Лебедєва-Дичко А.С.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент завідувач кафедри програмної інженерії , доцент, к.ф.-м.н. Лісняк А. О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя 2025

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
Завідувач кафедри комп'ютерних наук,
д.т.н., професор

_____ Шило Г.М..
(підпис)

" ____ " _____ 2024 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Виставкіну Володимиру Володимировичу
(прізвище, ім'я та по-батькові)

1. Тема роботи Розробка веб додатку для онлайн-школи

керівник роботи Лебедєва-Дичко А.С., старший викладач кафедри комп'ютерних наук
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 20 » грудня 2024 року № 2002с

2. Строк подання студентом роботи 15.05.2025

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

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.

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
	завідувач кафедри , д.т.н., професор Шило Г.М.		

7. Дата видачі завдання 01.01.2025

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.	10.01.2025	
2.	Збір вихідних даних.	31.01.2025	
3.	Обробка методичних та теоретичних	28.02.2025	
	джерел.		
4.	Розробка першого та другого розділу.	10.04.2025	
5.	Розробка третього розділу.	10.05.2025	
6.	Оформлення та нормоконтроль	20.05.2025	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	29.05.2025	

Студент _____ В.В. Виставкін
(підпис) (ініціали та прізвище)

Керівник роботи _____ А.С Лебедєва-Дичко
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

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

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка веб-додатку для онлайн-школи»: 37 с., 11 рис., 3 табл., 4 джерела.

БАЗА ДАНИХ, ІНТЕРФЕЙС КОРИСТУВАЧА, НАВЧАННЯ, ОНЛАЙН-ШКОЛА, ПРОГРАМУВАННЯ, СИСТЕМА УПРАВЛІННЯ НАВЧАННЯМ, ТЕСТУВАННЯ, ФРЕЙМВОРК, ФУНКЦІОНАЛЬНІСТЬ, ХОСТИНГ, ШАБЛОН, ШКОЛА DJANGO, PYTHON.

Об’єкт дослідження – веб-додатки для організації освітнього процесу.

Мета роботи: створення веб-додатку для онлайн-школи, який забезпечує ефективну взаємодію між студентами, викладачами та адміністрацією закладу освіти.

Метод дослідження – аналіз предметної області, об’єктно-орієнтоване програмування, моделювання структури баз даних, застосування фреймворку Django, тестування, документування результатів.

У роботі проаналізовано потреби онлайн-освіти, обґрунтовано доцільність використання обраних технологій, побудовано архітектуру веб-додатку, реалізовано ключові функціональні модулі для користувачів з різними ролями. Проведено тестування функціональності, здійснено розгортання проєкту на сервері та виконано аналіз результатів. Результати можуть бути використані як основа для створення власної навчальної платформи або впровадження в діяльність малих освітніх закладів. Робота має практичне значення й демонструє ефективне поєднання технологій Python, Django та MySQL у сфері освітнього програмного забезпечення.

SUMMARY

Master's qualifying paper «Development of the Web Application for an Online School»: 37 pages, 11 figures, 3 tables, 4 references.

DATABASE, USER INTERFACE , LEARNING, ONLINE SCHOOL, PROGRAMMING, LEARNING MANAGEMENT SYSTEM, , TESTING, FRAMEWORK, FUNCTIONALITY, HOSTING, TEMPLATE, SCHOOL, DJANGO, PYTHON.

Object of the study – web applications for organizing the educational process.

Aim of the study: to create a web application for an online school that ensures effective interaction between students, teachers, and administration.

Method of research – subject area analysis, object-oriented programming, database modeling, use of the Django framework, testing, documentation of results.

The paper analyzes the needs of online education, substantiates the choice of technologies used, develops the architecture of the web application, and implements key functional modules for users with different roles. Functional testing was performed, the project was deployed on a server, and the results were analyzed. The outcomes can serve as a basis for launching a personal educational platform or integrating it into small educational institutions. The project is of practical value and demonstrates the effective integration of Python, Django, and MySQL technologies in the development of educational software.

ЗМІСТ

Завдання на кваліфікаційну роботу……..2

Реферат 4

Summary 5

Зміст 6

Вступ 7

1 Аналіз предметної області та постановка задачі 9

1.1 Огляд сучасного стану систем онлайн-навчання 9

1.2 Постановка задачі 12

1.3 Архітектура веб-додатку 14

1.4 Вибір технологій 16

2 Проектування та реалізація веб-додатку 19

2.1 Модульна структура системи 20

2.2 Проєктування структури бази даних 21

2.3 Реалізація функціональних модулів 25

2.4 Безпека, розмежування доступу та валідація 26

3 Тестування, розгортання та аналіз результатів 28

3.1 Тестування функціональності 28

3.2 Розгортання веб-додатку 31

3.3 Аналіз результатів впровадження 33

Висновки 35

Перелік посилань 37

ВСТУП

У сучасному світі, що стрімко змінюється під впливом інформаційних технологій, освіта також зазнає значної трансформації. Онлайн-навчання, яке ще кілька десятиліть тому здавалося футуристичною концепцією, сьогодні стало повсякденним явищем. Воно забезпечує доступ до знань незалежно від місця перебування, дає змогу формувати індивідуальні освітні траєкторії та поєднувати навчання з іншими сферами життя. Пандемія COVID-19 лише прискорила цей процес, висвітливши необхідність швидкої адаптації навчальних закладів до дистанційного формату.

Наукові дослідження, такі як [1], підкреслюють, що ключовими перевагами онлайн-освіти є гнучкість, індивідуалізація навчання та доступ до контенту світового рівня незалежно від географічного розташування. Також зростає значення мобільного навчання, що відображено у [2], де наголошено на ролі пристроїв та адаптивних технологій у розширенні освітнього простору.

Розвиток інформаційних систем у сфері освіти дозволяє автоматизувати процеси адміністрування, взаємодії між студентами й викладачами, а також зберігати, обробляти та аналізувати дані про успішність та активність користувачів. Особливої актуальності набувають веб-додатки як ефективні інструменти створення повноцінного онлайн-освітнього середовища. За даними публікацій в [3] та [4], застосування сучасних інформаційних технологій суттєво підвищує ефективність освітнього процесу, зокрема в частині адаптації контенту, інтерактивності та аналітики результатів.

Дана кваліфікаційна робота присвячена розробці веб-додатку для онлайн-школи з використанням сучасного стеку технологій: мови програмування Python, фреймворку Django та системи управління базами даних MySQL. Основною метою є створення зручного, безпечного та функціонального рішення для керування навчальним процесом.

Об’єктом дослідження виступають веб-додатки, орієнтовані на підтримку освітніх процесів у дистанційній формі.

Предметом дослідження є процес проєктування, реалізації та розгортання інформаційної системи онлайн-школи на базі Django.

У роботі використано методи аналізу програмних рішень, об’єктно-орієнтованого програмування, моделювання структури бази даних, а також тестування реалізованих функціональних компонентів системи.

Результатом дослідження є повнофункціональний веб-додаток, вихідний код якого розміщено у відкритому репозиторії.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд сучасного стану систем онлайн-навчання

Стрімкий розвиток освітніх технологій у поєднанні з пандемією COVID-19 суттєво підвищили попит на онлайн-освіту в усьому світі. Особливо це стосується шкільної освіти, де виникла гостра потреба у веб-платформах, що мають забезпечувати повноцінне функціонування навчального процесу в дистанційному форматі. Серед основних функціональних вимог до таких систем можна виокремити: інтерактивність, гнучке управління користувачами (рольовий доступ), підтримку занять у режимі реального часу, автоматизовану перевірку завдань, збереження статистики успішності та відвідуваності. Окрім цього, важливими є: інтеграція з сервісами відеозв’язку, підтримка мобільних пристроїв, захищене середовище для обміну матеріалами, система сповіщень та персоналізовані освітні траєкторії.

Наукове підґрунтя розвитку онлайн-навчання описане у фундаментальній праці [1], де акцентується на важливості створення інтерактивного навчального середовища, орієнтованого на студента. Автор виділяє моделі взаємодії в онлайн-курсі: студент–контент, студент–викладач і студент–студент, які повинні бути належно підтримані платформами.

Публікації в журналах [3] та [4] також наголошують, що ефективні платформи онлайн-навчання повинні забезпечувати не лише передачу знань, а й аналітику участі, адаптацію контенту до стилів навчання користувача та можливість зворотного зв'язку.

У [2] особлива увага приділяється мобільності, як ключовому чиннику у майбутньому онлайн-навчання. Зокрема, платформи мають підтримувати адаптивні інтерфейси та персоналізований контент.

Платформа Coursera співпрацює з провідними університетами світу та надає доступ до академічних курсів з можливістю сертифікації. Її головна

перевага – висока якість контенту, однак вартість курсів часто є високою, а можливості адаптації обмежені орієнтована на академічне навчання, має велику базу якісного контенту від провідних університетів, проте є закритою та менш гнучкою для викладачів і може не відповідати чинному законодавству об персональних даних учнів.

Комерційна платформа Udemy має великий каталог курсів, створених користувачами. Перевагами є широкий вибір тем і гнучка модель монетизації, проте якість матеріалів не завжди відповідає академічним стандартам відкрита для авторів курсів, має широкий вибір, але якість контенту різниться через відсутність академічної перевірки та менш гнучкою для викладачів і може не відповідати чинному законодавству об персональних даних учнів.

Заснована MIT та Гарвардом платформа edX надає доступ до академічних курсів, підтримує інтерактивні інструменти, сертифікацію та навіть повноцінні магістерські програми. Водночас її використання вимагає значних ресурсів і технічної підтримки що вимагай підвищення витрати ресурсів які можуть бути обмеженні у навчального закладу.

Платформа Khan Academy – некомерційна з безкоштовними матеріалами для шкільної освіти. Вона забезпечує інтерактивність і простоту інтерфейсу, однак не охоплює спеціалізовані дисципліни вищої освіти.

На сьогодні для онлайн освіти використовується активно Google Classroom. Це зручна безкоштовна платформа для організації навчального процесу з інтеграцією до G Suite, але з обмеженою аналітикою та кастомізацією також неінтуїтивним графічним інтерфейсом що потребує запровадження тренінгів та навчальних матеріалів для вчителів.

Також для онлайн освіти використовуються Zoom, Microsoft Teams та BigBlueButton – системи для проведення занять у реальному часі з підтримкою відеозв’язку, обміну екранами та запису сесій, які часто використовуються як додаткові компоненти до навчальних платформ. Ці платформи більше застосовуються як інструментом відео та аудіо зв’язку і можуть слугувати гарним доповнення для освітнього процесу але не бути повноціною освітньою

платформою. У таблиці 1.1 наведена характеристика популярних освітніх платформ.

Таблиця 1.1 – Порівняльний аналіз освітніх платформ

Платформа	Вартість	Відкритість	Технічна складність	Якість контенту	Гнучкість
Coursera	Висока	Закрита	Низька	Висока	Обмежена
Udemy	Середня	Закрита	Низька	Змінна	Висока
edX	Висока	Часткова	Висока	Висока	Середня
Khan Academy	Безкоштовна	Закрита	Низька	Висока	Середня
Zoom/MS Teams	Умовно безкоштовна	Закрита	Середня	Середня	Висока
BigBlueButton	Умовно безкоштовна	Закрита	Середня	Середня	Висока
Google Classroom	Безкоштовна	Закрита	Низька	Середня	Середня

Порівняльний аналіз освітніх платформ показав, що кожна з них має свої переваги та обмеження залежно від мети використання. Такі платформи, як Coursera та edX, забезпечують високу якість контенту, проте мають високу вартість і нижчу гнучкість. Udemy відзначається високою гнучкістю, але якість матеріалів може бути нерівномірною. Безкоштовні рішення на кшталт Khan Academy та Google Classroom є доступними, але поступаються за технічними можливостями. Zoom, MS Teams та BigBlueButton більше підходять для синхронної взаємодії, пропонуючи середню якість та високу гнучкість. Вибір платформи має базуватись на вимогах конкретного навчального процесу.

Зокрема, слід порівняти доступні платформи за їх технологічними характеристиками: яку мову та фреймворк використовують для фронтенду та

бекенду, які мають переваги та обмеження щодо продуктивності, безпеки, масштабованості та підтримки мобільних пристроїв.

Серед ключових викликів для освітніх веб-систем варто виокремити:

- безпека даних, зокрема персональних даних неповнолітніх користувачів, що регламентується міжнародними стандартами (наприклад, GDPR);
- масштабованість, що передбачає стабільну роботу системи при збільшенні навантаження в години пікової активності або зростанні кількості користувачів;
- UX/UI для різних вікових груп – від дітей молодшого шкільного віку до дорослих слухачів курсів;
- адаптація під мобільні пристрої – включає адаптивний дизайн, підтримку жестів та ефективне завантаження контенту в умовах нестабільного інтернет-з’єднання.

Аналіз технологічних аспектів освітніх платформ свідчить про необхідність врахування не лише функціональності, а й технічних рішень у розробці систем. Ключовими викликами залишаються забезпечення безпеки персональних даних, масштабованість при зростанні аудиторії, адаптивність під мобільні пристрої та зручність інтерфейсу для різних вікових груп. Вибір технологій для фронтенду та бекенду має базуватись на вимогах до продуктивності, безпеки й підтримки сучасних стандартів UX/UI.

1.2 Постановка задачі

Попри наявність великої кількості готових платформ Google Classroom, Zoom тощо, більшість з них не повною мірою задовольняють вимоги приватних або спеціалізованих онлайн-шкіл, зокрема в аспектах кастомізації інтерфейсу, розширення функціональності, інтеграції з внутрішніми сервісами, аналітики успішності та захисту персональних даних. Проблема полягає в необхідності

створення гнучкого, масштабованого веб-додатку, який би дозволяв: управляти користувачами, проводити онлайн-заняття в реальному часі, перевіряти завдання та тести, зберігати історію навчання, забезпечувати багаторівневу автентифікацію, захист облікових записів, вести аналітику активності й адаптуватися під мобільні пристрої різного типу. У контексті цих вимог і виявлених обмежень існуючих рішень основною метою даної роботи є розробка веб-додатку для онлайн-школи, що відповідає зазначеним критеріям і враховує сучасні виклики освітньої сфери.

Функціональні вимоги до веб-додатку:

- адаптивний інтерфейс для десктопних і мобільних пристроїв;
- масштабованість та гнучка структура для подальшого розширення функціоналу;
- надання зворотного зв’язку, комунікації в рамках курсів;
- підтримка багаторівневої системи користувачів (адміністратор, викладач, учень);
- проведення онлайн-занять у реальному часі (інтеграція з відеозв’язком);
- реєстрація та автентифікація з урахуванням безпеки даних;
- створення, редагування та управління курсами, модулями, завданнями;
- збереження статистики навчання, аналітика активності.
- Відповідно, завданням даної кваліфікаційної роботи є розробка веб-додатку для онлайн-школи, який реалізує:
- вибір та обґрунтування архітектури програмного забезпечення;
- інтеграцію з сервісами відеоконференцій (наприклад, Jitsi або Zoom API);
- підготовку технічної документаці;
- проєктування інтерфейсу користувача з урахуванням адаптивності;
- реалізацію базового функціоналу (реєстрація, створення курсів, розклад, завдання, оцінювання);
- розробку системи авторизації та ролей користувачів;

– тестування функціоналу та оцінювання його ефективності;
– формалізацію функціональних вимог до системи.

Типові CMS, такі як WordPress із навчальними плагінами, мають певний рівень функціональності, однак не дозволяють повністю реалізувати всі вимоги без значного втручання в код або використання сторонніх сервісів. Саме тому було прийнято рішення про створення індивідуального рішення на базі фреймворку Django, що дозволяє реалізувати комплексну систему з високим рівнем безпеки, структурованості та масштабованості.

1.3 Архітектура веб-додатку

Архітектура веб-додатку Moodle2.0 побудована за моделлю клієнт-сервер. Це підхід, при якому клієнт наприклад, веб-браузер користувача надсилає запити до сервера, що обробляє ці запити, звертається до бази даних або зовнішніх сервісів і повертає результат клієнту. Така модель дозволяє чітко розмежувати відповідальність між компонентами системи, забезпечити масштабованість, централізовану обробку даних і спрощене технічне обслуговування. Вона також допомагає реалізувати продемонстровану модель взаємодії додатку з користувачем, зображену на рисунку 1.1.

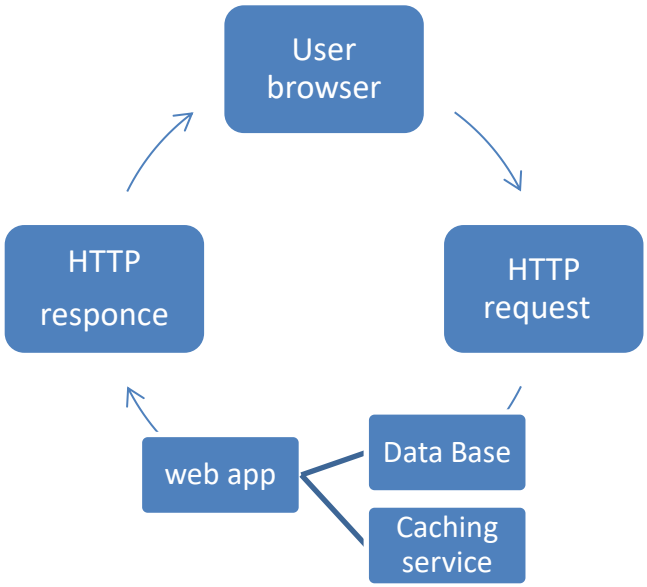


Рисунок 1.1 – Модель взаємодії користувача з веб додатком

Структурно система поділяється на такі основні рівні:

- клієнтська частина (Front-end) – відповідає за відображення інформації користувачу, взаємодію з інтерфейсом і передавання запитів до серверної частини;
- серверна частина (Back-end) –обробляє запити, керує логікою роботи додатку, забезпечує доступ до бази даних, контроль прав доступу, авторизацію та автентифікацію;
- база даних – містить структуровані дані щодо курсів, користувачів, розкладів, оцінювання тощо;
- адміністративний інтерфейс – дозволяє адміністраторам системи керувати всіма об’єктами: створювати нові курси, реєструвати користувачів, переглядати статистику тощо;
- середовище розгортання – система контейнеризована за допомогою Docker, що забезпечує незалежність від конфігурацій ОС. Для розгортання використовуються Nginx як проксі-сервер.
- Додаткові компоненти включають такі елементи:
- модулі безпеки охоплюють CSRF-захист, шифрування паролів та розмежування прав доступу.
- поштова система реалізована з підтримкою протоколу SMTP, що дозволяє надсилати сповіщення та виконувати процедуру відновлення паролів.
- інтеграція з сервісами відеоконференцій, зокрема Jitsi або Zoom, передбачена на етапі реалізації проєкту, однак не була включена до прототипу.

Така архітектура дозволяє масштабувати систему у випадку зростання кількості користувачів, легко адаптувати інтерфейс під різні пристрої та швидко впроваджувати нові функції без впливу на наявний функціонал.

1.4 Вибір технологій

Під час розробки веб-додатку важливим етапом є обґрунтований вибір технологій, які забезпечують стабільну роботу, легкість у супроводі, безпеку та можливість подальшого масштабування. Вибір здійснювався з урахуванням сучасних тенденцій у веб-розробці, сумісності компонентів між собою та специфіки проєкту.

У процесі аналізу було розглянуто низку популярних технологічних стеків. ASP.NET Core є потужним фреймворком із глибокою інтеграцією в екосистему Microsoft. Проте його складність у розгортанні, меншою придатністю для швидкого створення MVP та високими вимогами до кваліфікації розробників і ресурсів обумовлює обмеження для застосування в академічних та малобюджетних проєктах.

Laravel (PHP) відзначається зручним синтаксисом, однак вимагає реалізації багатьох функцій, які у Django вже є стандартними (ORM, авторизація, маршрутизація тощо). Крім того, Laravel залежить від великої кількості сторонніх бібліотек, а масштабування проектів на його основі може бути менш гнучким.

Node.js з Express є гнучким рішенням, але потребує значних зусиль на налаштування інфраструктури: реалізацію ORM, валідації, системи безпеки тощо, що ускладнює швидку розробку MVP.

Ruby on Rails забезпечує високу швидкість розробки, однак поступається за популярністю і підтримкою в українських реаліях, що створює труднощі з наймом фахівців та технічним супроводом.

З урахуванням зазначених факторів, для реалізації серверної частини обрано Python — популярну мову з читабельним синтаксисом і широкою екосистемою. Вона підтримується великою спільнотою та активно використовується у сфері освіти, що робить її ідеальною для даного проєкту.

Фреймворком для серверної логіки обрано Django, який базується на архітектурній моделі MVT (Model-View-Template). Django включає в себе багато

вбудованих рішень — ORM, систему аутентифікації, маршрутизацію, панель адміністратора, що значно скорочує час розробки.

Для фронтенду застосовується Bootstrap, що дозволяє швидко створювати адаптивні та сучасні інтерфейси. У поєднанні з HTML, CSS та JavaScript це дає змогу забезпечити зрозумілий, зручний інтерфейс, що адаптується до різних типів пристроїв.

У якості системи керування базами даних використано MySQL — реляційну СУБД, яка забезпечує стабільну роботу, підтримку транзакцій, високу продуктивність та масштабованість. Її ефективна інтеграція з Django ORM дозволяє гнучко працювати з даними.

Для спрощення процесу розгортання застосовано Docker, який дозволяє створити єдине середовище для розробки та продакшну. Як зворотний проксі використано Nginx, а для запуску Django-сервера – Gunicorn, що забезпечує надійність і продуктивність.

Серед додаткових функцій реалізовано підтримку SMTP для надсилання сповіщень. Також передбачена можливість інтеграції з відеосервісами Jitsi або Zoom, що дозволяє проводити онлайн-заняття безпосередньо через систему.

Для наочності наведемо таблицю порівняння популярних технологічних стеків для веб-додатків освітнього призначення у таблиці 1.2.

Таблиця 1.2 – Порівняльний аналіз веб-технологій

Технології	Переваги	Недоліки
Python + Django	Висока безпека, вбудована ORM, готова адмін-панель, активна спільнота	Менша продуктивність у порівнянні з Node.js у високонавантажених системах
PHP + Laravel	Гнучкість, проста структура, багата екосистема	Відсутність вбудованої адмін-панелі, потреба в додаткових пакетах

Продовження таблиці 1.2

Технології	Переваги	Недоліки
Node.js + Express	Висока продуктивність, асинхронна обробка	Потребує ручного налаштування багатьох базових компонентів
ASP.NET Core	Потужна підтримка, висока продуктивність, інтеграція з MS Azure	Складне розгортання, висока вартість інфраструктури
Ruby on Rails	Швидкість розробки, принцип "конвенція понад конфігурацію"	Менш популярний, обмежена підтримка в Україні

Порівняння розглянутих технологій за ключовими критеріями – функціональністю, зручністю розгортання, безпекою, підтримкою мобільних пристроїв та масштабованістю – дозволяє зробити обґрунтований висновок.

Обраний стек – Python + Django + MySQL + Docker – надає збалансоване поєднання простоти реалізації, потужних можливостей для розробника, високого рівня захисту даних та швидкого розгортання в різних середовищах.

Завдяки Django забезпечується структурованість коду, наявність готових модулів ORM, авторизація, адміністрування, що знижує поріг входу для команди. MySQL як СУБД дозволяє ефективно працювати зі структурованими даними, а Docker гарантує узгодженість середовища між розробкою, тестуванням та продуктивом. Отже, саме ця комбінація технологій найкраще відповідає вимогам поставленої задачі.

2 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ

Розділ присвячений процесу практичної реалізації веб-додатку Moodle2.0, з урахуванням попередньо визначених архітектурних рішень, обраних технологій та функціональних вимог.

Розробка системи реалізована у вигляді модульної архітектури, де кожен модуль виконує окрему функцію. Основні модулі наведено в таблиці 1.2.

Таблиця 2.1 – Опис функціональності модулів

Назва модуля	Опис функціональності
Student	Зберігання інформації про студентів, оцінки, відвідуваність, нотатки, батьків
Schedule	Формування розкладу занять, призначення викладачів до предметів і класів
Configuration	Керування довідниками: предмети, класи, приміщення, навчальні набори
User	Реєстрація, авторизація користувачів, призначення ролей, пов’язання з курсами
System	Адміністрування, логування подій, система прав доступу

Такий підхід дозволяє масштабувати систему, легко оновлювати окремі модулі та ефективно організувати структуру коду.

На основі поставленої задачі та аналізу предметної області було визначено наступні вимоги до системи:

Функціональні вимоги:

- генерація статистичних звітів та аналітики;
- завантаження, перегляд і перевірка навчальних завдань;
- можливість створення, редагування та публікації курсів;

- оцінювання студентів та ведення статистики успішності;
- підтримка інтерактивних модулів, коментарів, повідомлень;
- реєстрація та автентифікація користувачів (адміністратор, викладач, студент);
- формування розкладу занять та розподіл викладачів за класами;
- збереження історії навчальної діяльності.
- Нефункціональні вимоги:
- адаптивність інтерфейсу для мобільних і десктопних пристроїв;
- захист від CSRF та SQL-ін’єкцій;
- легкість розгортання в продакшн-середовищі (Docker);
- масштабованість додатку при збільшенні кількості користувачів;
- надійна система безпеки з шифруванням облікових даних;
- підтримка інтеграції з зовнішніми сервісами (Zoom, SMTP) ;
- швидкість завантаження сторінок і ефективність взаємодії.

У цьому розділі представлено практичну реалізацію системи Moodle2.0, побудовану за принципами модульної архітектури. Такий підхід дозволяє забезпечити гнучкість, масштабованість і зручність супроводу. Визначені функціональні та нефункціональні вимоги охоплюють ключові аспекти сучасної освітньої платформи — від створення курсів і керування навчальним процесом до інтеграції з зовнішніми сервісами, безпеки та адаптивності інтерфейсу.

2.1 Модульна структура системи

Розробка веб-додатку здійснювалася з використанням модульного підходу. Система поділена на функціонально ізольовані частини, що відповідають за окремі аспекти освітнього процесу:

- configuration module – довідникові дані: список класів, предметів, приміщень, навчальних груп;

– schedule module – формування навчального розкладу, прив’язка вчителів до предметів і секцій;
– student module – зберігання інформації про студентів, їхню відвідуваність, оцінки, дисциплінарні випадки, батьків;
– system module – адміністрування, логування дій, права доступу;
– user module – управління користувачами, ролями, правами та зв’язком з курсами й класами.

Кожен модуль реалізовано у вигляді окремого Django-додатку app, що спрощує підтримку, масштабування та повторне використання.

2.2 Проєктування структури бази даних

Структура бази даних реалізована на основі реляційної моделі з урахуванням вимог до гнучкості, нормалізації й підтримки зв’язків між сутностями. Схеми бази даних відображають такі логічні модулі:

Модуль Student включає таблиці student_student, student_attendances, student_parents, student_disciplines, student_notes та інші, які відображають повну інформацію про учнів, їхню присутність, оцінки, дисциплінарні випадки, контакти батьків та пов’язані документи. Даний модуль формує основу навчального процесу дивись рисунок 2.1.

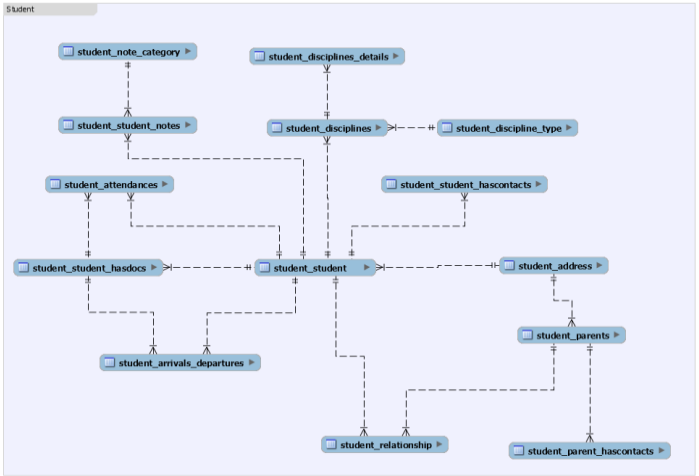


Рисунок 2.1 – Діаграма модулю Student

Модуль Schedule охоплює таблиці schedule_schedule, schedule_schedule_teacher, schedule_sections, які забезпечують побудову розкладу занять, прив’язку викладачів до секцій та керування навчальним графіком дивись рисунок 2.2.

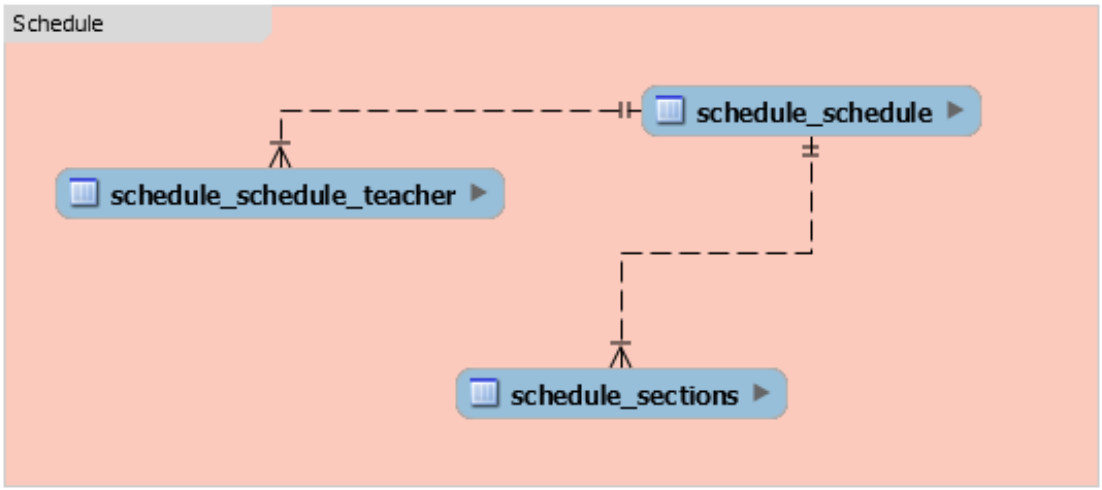


Рисунок 1.2 – Діаграма модулю Schedule

Модуль Configuration включає довідкові таблиці configuration_subjects, configuration_classes, configuration_location, які забезпечують налаштування загальноосвітнього середовища: список предметів, навчальних класів, приміщень і навчальних наборів дивись рисунок 2.3.

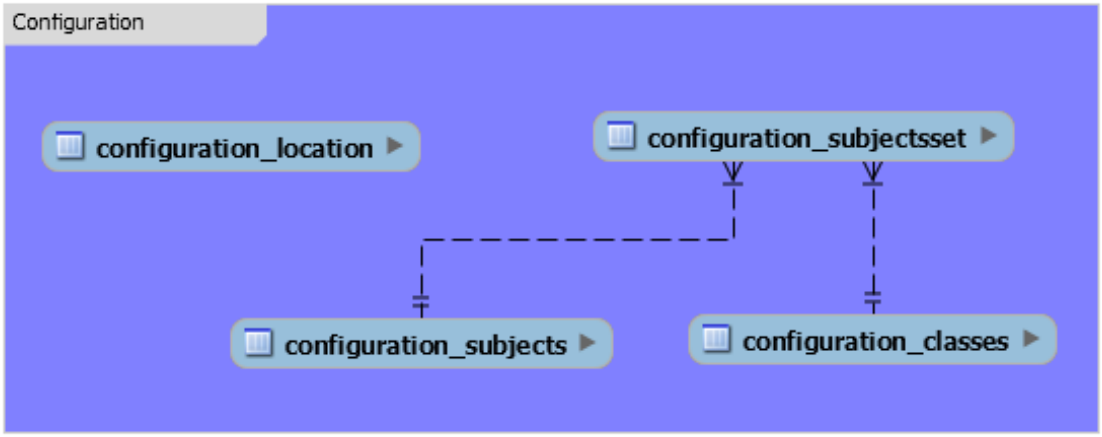


Рисунок 2.3 – Діаграма модулю Configuration

Модуль System реалізує базову систему безпеки, зокрема через таблиці auth_permission, auth_group, auth_group_permissions, django_admin_log – це стандартна частина Django для адміністрування, реєстрації подій та керування правами доступу дивись рисунок 2.4.

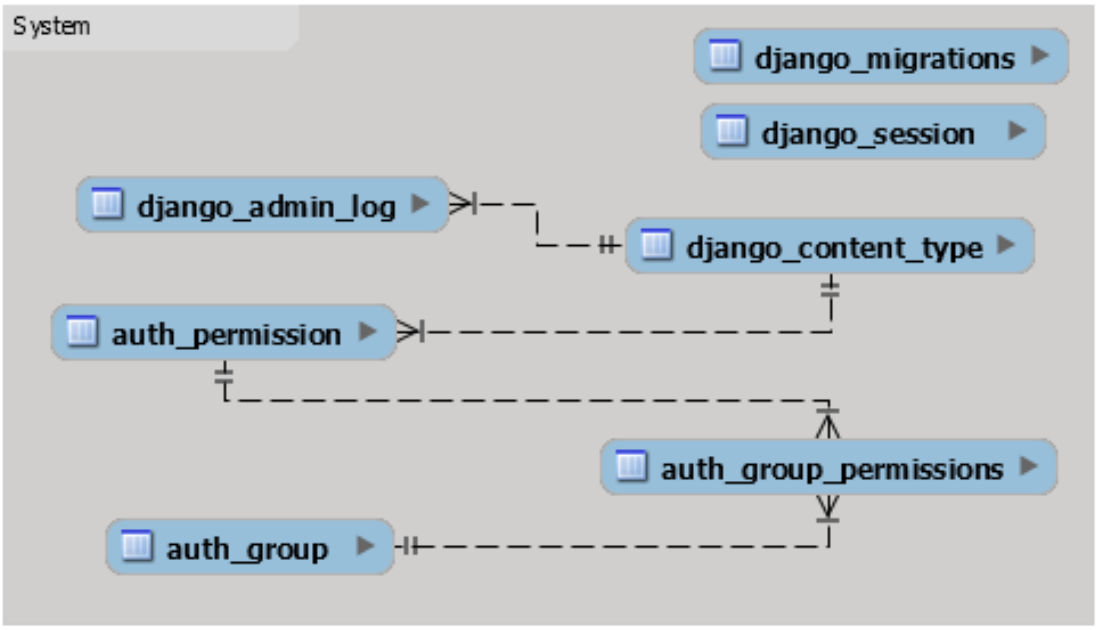


Рисунок 2.4 – Діаграма модулю System

Модуль User реалізує керування користувачами, їхніми правами, ролями та зв’язками з класами й предметами (user_user, user_user_groups, user_user_user_permissions, user_user_subject, user_classeownership).

На основі цієї структури реалізовано повну інтеграцію даних та забезпечено збереження логічних зв’язків між ключовими об’єктами системи: студентами, викладачами, батьками, курсами, розкладом дивись рисунок 2.5.

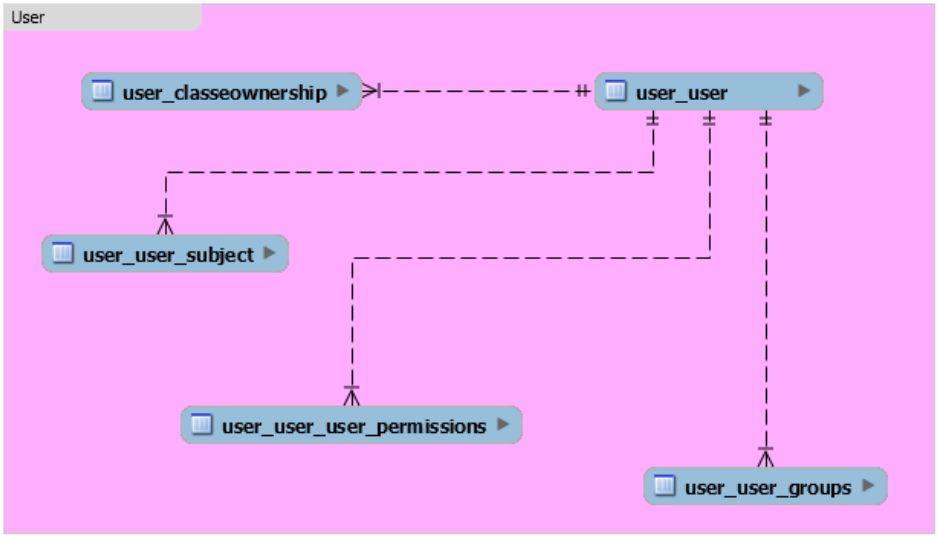


Рисунок 2.5 – Діаграма модулю User

Отже, структура бази даних побудована на основі реляційної моделі з урахуванням принципів нормалізації та логічної модульності. Такий підхід дозволив забезпечити цілісність даних, гнучкість при розширенні функціоналу та ефективне управління зв’язками між ключовими сутностями системи. Модульний поділ на Student, Schedule, Configuration, System і User гарантує зрозумілу структуру, зручність супроводу та масштабованість рішення в умовах реального освітнього середовища.

2.3 Реалізація функціональних модулів

Кожен із модулів веб-додатку реалізує окремий функціонал та обробляє відповідну бізнес-логіку. Дані, зібрані через інтерфейс, обробляються сервером і зберігаються у відповідних таблицях бази даних. Основні модулі реалізовано наступним чином:

- configuration module: доступний лише адміністраторам, містить таблиці subjects, rooms, groups, academic_years. Забезпечує налаштування параметрів системи;
- schedule module: адміністратори або викладачі формують розклад, який зберігається у таблицях schedule, teacher_subject, room. Розклад автоматично відображається в особистих кабінетах учнів і вчителів;
- student module: користувачі з роллю "учень" мають доступ до свого профілю, розкладу, оцінок, завдань. Дані зберігаються в таблицях students, grades, attendance;
- system module: логування подій logs, управління правами доступу до функціоналу middleware, журналювання змін у системі. Забезпечує прозорість дій і безпеку;
- user module: реалізовано через вбудовану систему Django auth.User із розширенням для ролей адміністратор, учень, викладач через таблиці profiles, user_roles. Забезпечує аутентифікацію, авторизацію та управління користувачами.

Модулі взаємодіють між собою за допомогою внутрішніх запитів до бази даних і передачі даних між шаблонами та представленнями (views). Запити оптимізовано через Django ORM. Усі ключові операції супроводжуються перевіркою прав доступу.

Реалізація забезпечує надійне функціонування кожного компоненту системи, даючи змогу користувачам ефективно взаємодіяти із платформою згідно з призначеною роллю.

2.4 Безпека, розмежування доступу та валідація

У процесі реалізації функціональних модулів важливо враховувати потенційні загрози, з якими може стикатися веб-додаток у сфері онлайн-освіти. Основні ризики включають:

- атаки типу CSRF (Cross-Site Request Forgery) та SQL-ін’єкції;
- відсутність шифрування конфіденційної інформації (наприклад, паролів, контактних даних);
- витік персональних даних, зокрема учнів і викладачів, які зберігаються у базі даних;
- несанкціонований доступ до облікових записів, особливо через слабкі паролі або повторне використання паролів;
- недостатній контроль ролей користувачів, що може призвести до ескалації прав доступу;
- недостатня адаптація до мобільних пристроїв, що може створити вразливості через нестандартні інтерфейси або слабкий UX;
- перевантаження серверу через велику кількість одночасних запитів;
- Щоб мінімізувати ці загрози, у веб-додатку реалізовано такі захисні механізми:
- аутентифікація з використанням хешування паролів алгоритм PBKDF2;
- docker-контейнери, які ізолюють середовище виконання і полегшують контроль над оновленнями й безпекою системи;
- захист від CSRF вбудованими механізмами Django, з використанням токенів у формах;
- обмеження доступу до адмін-панелі лише для авторизованих користувачів з відповідною роллю;
- ORM Django, яка унеможливлює ручне введення SQL-запитів і тим самим запобігає SQL-ін’єкціям;
- регулярне логування подій в системі безпеки та адміністрування дій користувачів audit log;

- система ролей і прав доступу, яка суворо обмежує доступ до функціоналу відповідно до ролі користувача;
- системи сповіщень, що повідомляють про підозрілі активності або помилки в системі.

Завдяки цим заходам веб-додаток Moodle2.0 забезпечує високий рівень захисту персональних даних, стабільність у роботі та відповідність сучасним вимогам до кібербезпеки освітніх платформ.

2 ТЕСТУВАННЯ, РОЗГОРТАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

3.1 Тестування функціональності

Модульне тестування – це процес перевірки роботи окремих логічних одиниць програмного забезпечення, таких як функції, методи, моделі чи форми. У середовищі Django це зазвичай означає створення окремих тестів для кожної моделі, форми чи представлення (view), які виконуються незалежно від усієї системи. Основною метою є виявлення помилок у логіці та валідації ще на ранньому етапі розробки.

Після завершення розробки основних функціональних модулів Moodle2.0 було проведено повноцінне модульне та інтеграційне тестування з метою перевірки коректності реалізації та взаємодії компонентів системи. Для модульного тестування використовувались інструменти стандартної бібліотеки unittest, а також pytest, що дало змогу реалізувати більш складні сценарії перевірки.

На рівні модульного тестування особливу увагу приділено перевірці створення, редагування та збереження об'єктів моделі student_student, а також валідації унікальності email при реєстрації користувача. Була протестована автентифікація та доступ користувачів до курсів відповідно до їх ролей (рис. 3.1).

Перевірка охопила також коректну роботу форм, зокрема форм реєстрації та створення розкладу, а також логіку редагування даних у конфігураційних модулях.

Інтеграційне тестування здійснювалося з метою перевірки взаємодії між модулями системи, що особливо важливо в умовах модульної архітектури.

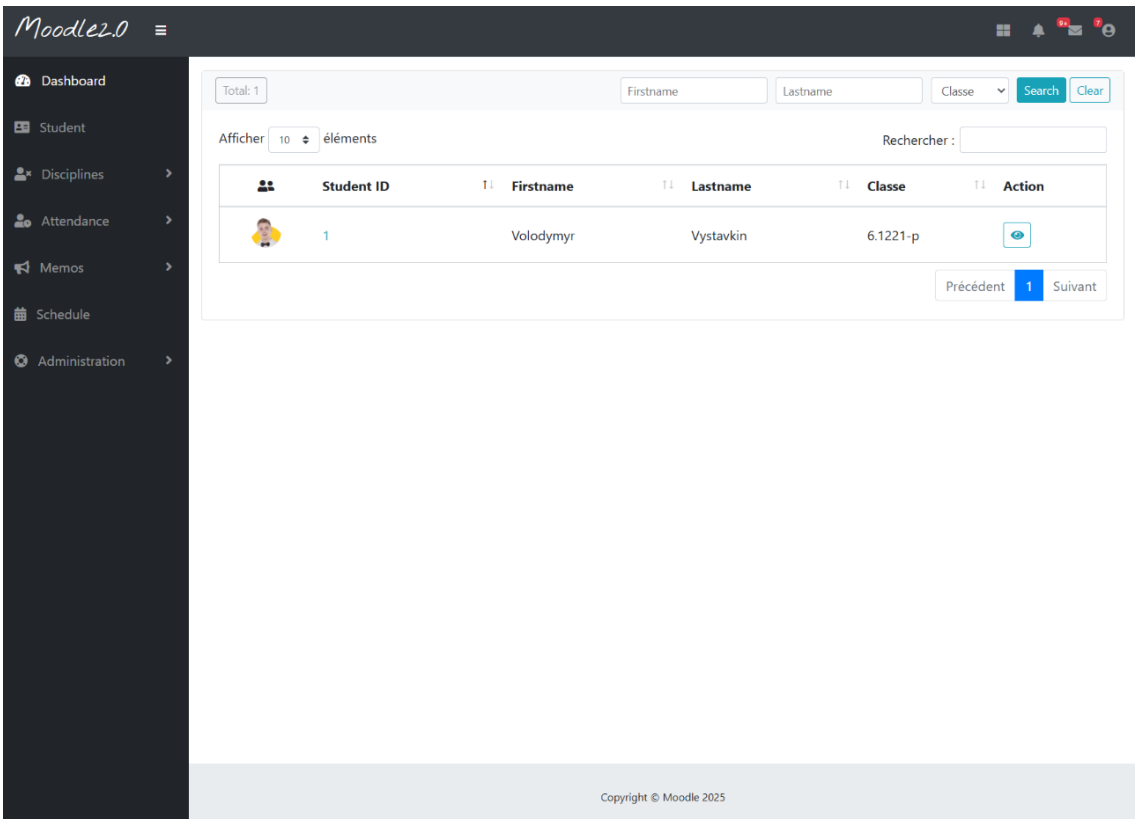


Рисунок 3.1 – Перевірка доступу до курсів залежно від ролі користувача

У межах інтеграційного тестування було перевірено створення розкладу з використанням наявних викладачів і предметів (рис.3.2).

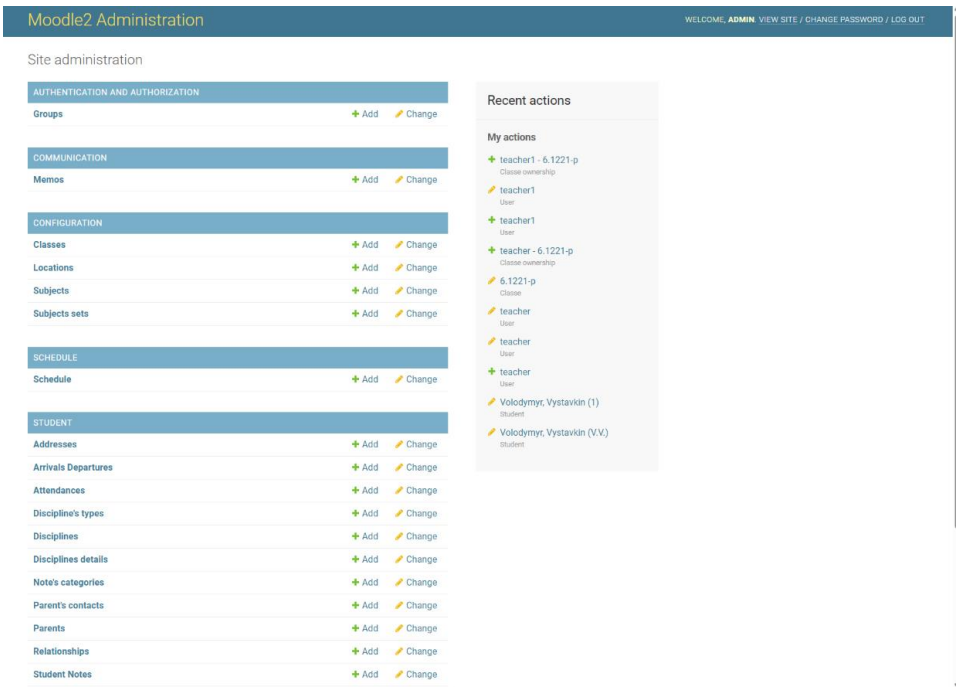


Рисунок 3.2 – Інтерфейс адміністративної панелі для створення розкладу

Було протестовано доступ викладачів до закріплених за ними курсів (рис.3.3).

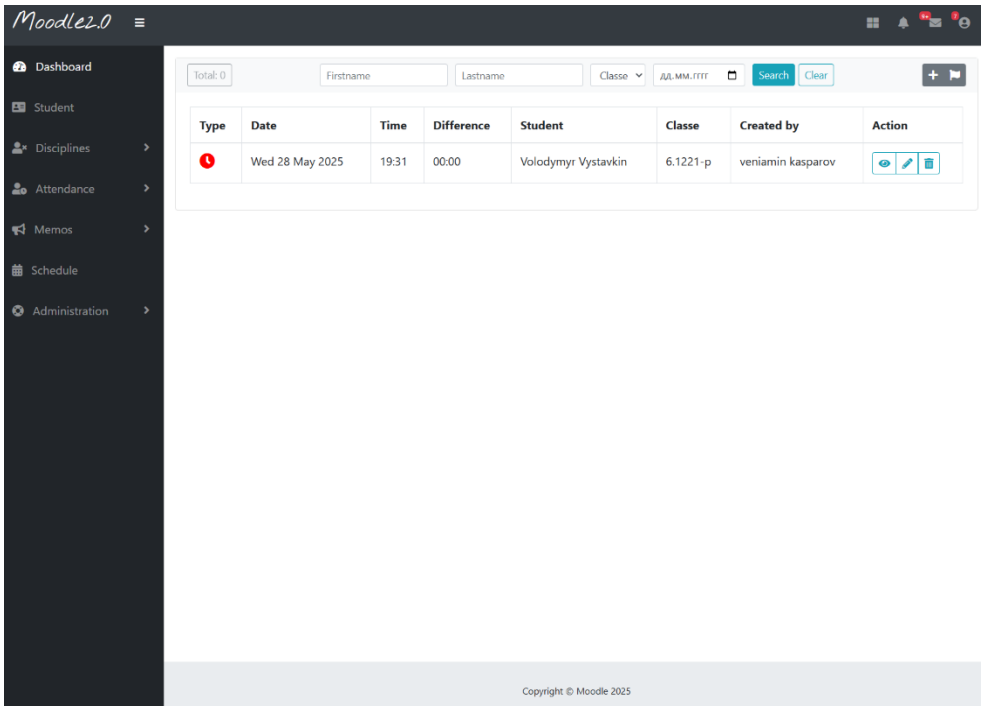


Рисунок 3.3 – Вигляд кабінету викладача з доступом до закріплених курсів

Окрему увагу приділено перевірці відображення аналітики щодо успішності студентів для адміністратора.

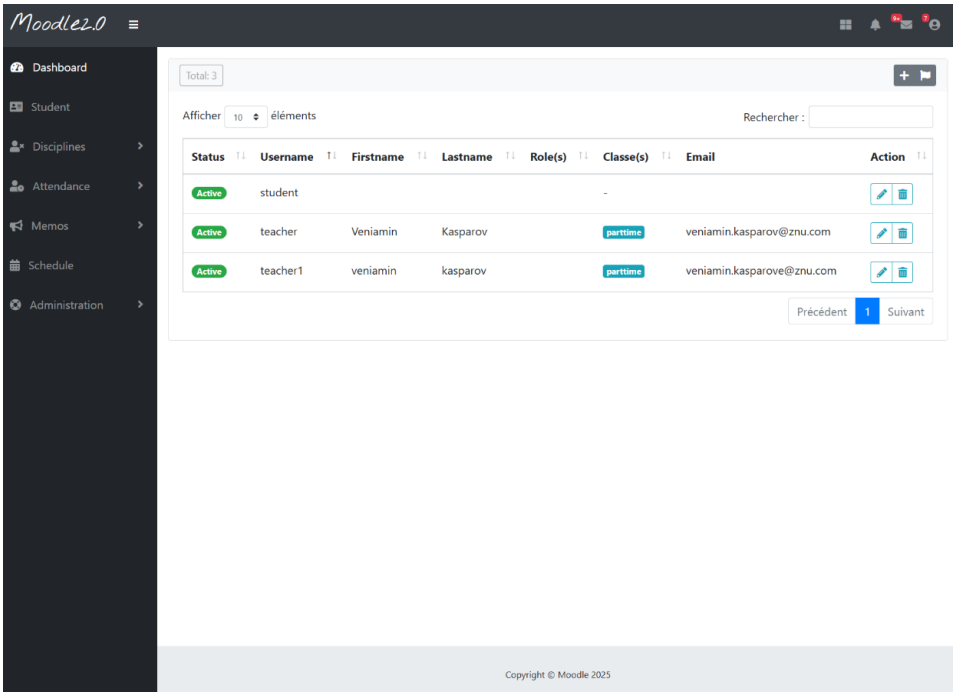


Рисунок 3.4 – Панель аналітики для адміністратора щодо успішності студентів

Також здійснено перевірку функціональності взаємодії між студентами та викладачами у межах курсу.

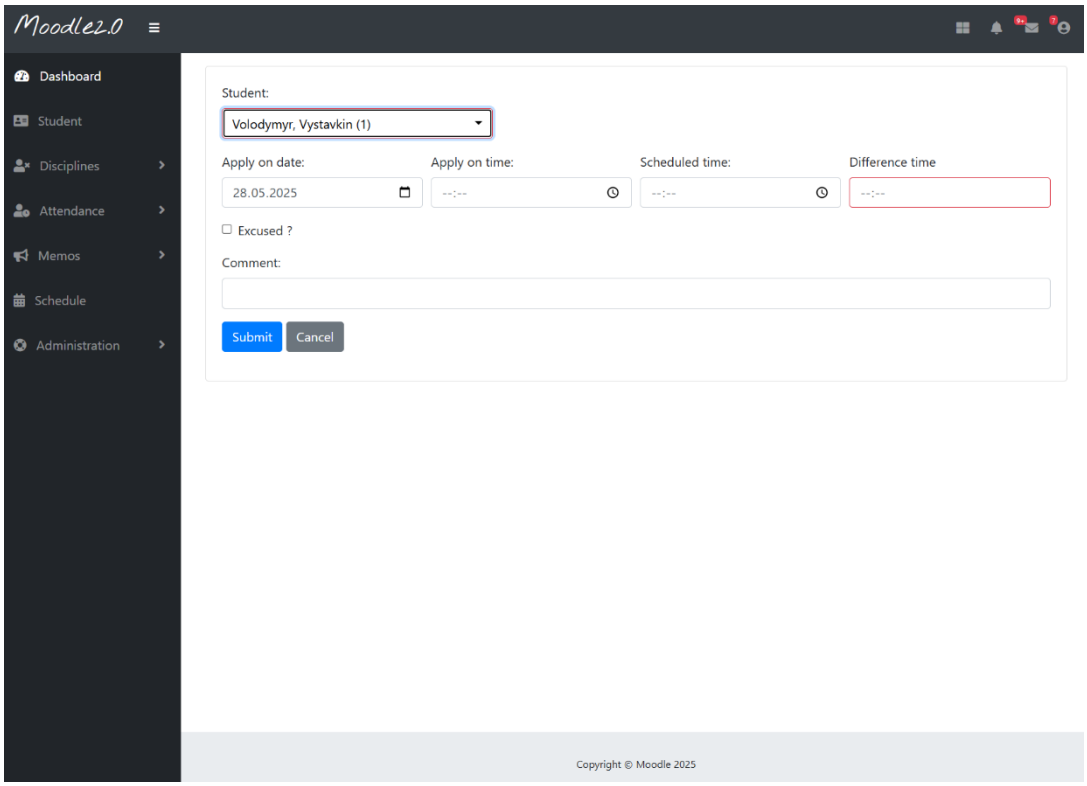


Рисунок 3.5 – Інтерфейс взаємодії студента з викладачем у межах курсу

Окрім основних перевірок, тестування дозволило виявити потенційні edge-cases, зокрема спроби доступу до системи неавторизованими користувачами або введення некоректних даних у форми. Усі критично важливі шляхи користувача було протестовано, що дало змогу підтвердити стабільність і надійність системи перед її впровадженням.

3.2 Розгортання веб-додатку

Для ізоляції всіх компонентів і забезпечення масштабованості використано Docker і Docker Compose. Кожен сервіс (база даних, бекенд, веб-сервер) функціонує в окремому контейнері. Тестування проводилося у середовищі Ubuntu 22.04 з Docker Engine.

Налаштування проєкту:

- використано файл .env для конфігурації змінних середовища (ключ безпеки, параметри БД, режим налагодження);
- файл docker-compose.yml містить опис сервісів: додатку (Gunicorn), бази даних (MySQL), та зворотного проксі-сервера (Nginx);
- налаштовано розділення статичних (static/) і медіа-файлів (media/) із відповідними правами доступу;
- сконфігуровано nginx.conf для обробки HTTP-запитів та перенаправлення трафіку до Gunicorn через сокет.
- Запуск сервера:
- ініціалізація бази даних через python manage.py migrate;
- створення суперкористувача: python manage.py createsuperuser;
- збір статичних файлів: python manage.py collectstatic;
- запуск контейнерів: docker-compose up --build -d.

Після запуску система доступна за локальною IP-адресою через Nginx. Доступ до панелі адміністратора обмежений за IP-адресою. Production-середовище додатково захищено базовим файрволом ufw і налаштованим обмеженням до root-доступу. на персональному комп’ютері розробника з використанням наступного технологічного стеку:

- Docker + Docker Compose – контейнеризація додатку, бази даних та серверів;
- Nginx – як зворотний проксі-сервер;
- Gunicorn – для запуску WSGI-додатку Django;
- MySQL – як окремий контейнер для зберігання даних.

Для налаштування середовища використано файл .env, у якому зберігалися змінні середовища, зокрема секретний ключ, логін і пароль до бази даних та інші параметри безпеки. Статичні файли та медіа було винесено до окремих директорій, які обробляються Nginx відповідно до правил конфігурації.

Розгортання у production-середовищі включало:

- перенесення всіх сервісів на сервер;

- налаштування обмеженого доступу до адмін-панелі за IP-адресами;
- встановлення базових правил безпеки на рівні файрволу та серверної конфігурації;
- запуск проєкту за допомогою docker-compose up у відповідному конфігураційному оточенні.

Застосування контейнеризації за допомогою Docker і Docker Compose забезпечило ізольованість сервісів, стабільність роботи системи та гнучкість при розгортанні. Завдяки використанню окремих контейнерів для бази даних, бекенду й проксі-сервера вдалося створити уніфіковане середовище як для розробки, так і для production-розгортання. Реалізація захисту на рівні доступу, конфігурації серверу та використання змінних середовища сприяє високому рівню безпеки та готовності системи до масштабування.

3.3 Аналіз результатів впровадження

Після завершення розробки та локального розгортання веб-додатку для онлайн-школи було проведено тестування функціональності, зручності інтерфейсу та відповідності поставленим вимогам. Метою аналізу є визначення ефективності реалізованої системи, її стабільності, повноти реалізації функцій, а також виявлення можливостей подальшого удосконалення.

Функціональна повнота: Усі заплановані функції, зокрема реєстрація користувачів, створення курсів, додавання завдань, формування розкладу, ролі доступу, а також аналітика успішності – були реалізовані. Жодна з базових вимог не залишилася без покриття, що підтверджується результатами модульного та інтеграційного тестування.

Зручність інтерфейсу: Було перевірено UX для трьох основних типів користувачів – адміністратора, викладача та студента. Інтерфейс є інтуїтивно зрозумілим, адаптивним до різних розмірів екранів, перевірено коректність

роботи на смартфонах, планшетах та ПК. Усі ключові дії (наприклад, створення курсу або перегляд оцінок) можна виконати в кілька кліків.

Надійність і безпека: Завдяки використанню механізмів автентифікації, авторизації, CSRF-захисту, хешування паролів та ізоляції середовищ у Docker-контейнерах, додаток виявився стійким до типових векторів атак. Логування подій і обмеження прав доступу забезпечують додатковий рівень контролю.

Масштабованість: Інфраструктура дозволяє швидко перенести систему на хмарний сервер або додати нові модулі без зміни основної архітектури. Docker-образи можуть бути розгорнуті в CI/CD-процесах у майбутньому.

Можливості для вдосконалення: Серед напрямів подальшого розвитку варто зазначити реалізацію відеозв’язку через Jitsi або Zoom API, інтеграцію з електронною поштою для повідомлень та створення API для зовнішніх клієнтів або мобільного застосунку.

У підсумку, реалізований веб-додаток Moodle2.0 повністю відповідає поставленим завданням, має гнучку архітектуру, зручний інтерфейс і високий рівень захищеності, що дозволяє використовувати його в освітніх закладах різного масштабу та профілю.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було здійснено повноцінний цикл проєктування, реалізації, тестування та впровадження інформаційної системи для онлайн-школи. Розроблений веб-додаток Moodle2.0 демонструє приклад адаптивного, масштабованого, безпечного та функціонального рішення на базі відкритих технологій: Python, Django, MySQL і Docker.

Аналіз предметної області, сучасних викликів у сфері онлайн-освіти, зокрема наслідків пандемії COVID-19, показав необхідність створення спеціалізованих рішень, що забезпечують повну підтримку навчального процесу в цифровому середовищі. Було з’ясовано, що існуючі платформи не повною мірою відповідають вимогам приватних або спеціалізованих шкіл, зокрема щодо кастомізації інтерфейсу, безпеки, аналітики та інтеграції з внутрішніми сервісами.

На основі сформульованих вимог було реалізовано наступне:

– проведено аналіз цільової аудиторії, типових сценаріїв використання, визначено набір необхідних функцій, серед яких: управління користувачами, створення курсів, завдань, розкладу, ведення аналітики успішності;

– побудована система за принципами MVT-моделі (Model–View–Template) фреймворку Django, що дозволяє забезпечити гнучку модульну структуру, зручне тестування та легке розширення функціоналу;

– розроблено UI інтерфейс з урахуванням різних типів користувачів (адміністратор, викладач, студент), протестований на різних типах пристроїв і забезпечує зручність взаємодії та мінімізацію навчального порогу;

– реалізовано функціональні модулі системи, які включають: Student, Schedule, Configuration, System, User, кожен з яких відповідає за певну частину навчального процесу та адміністрування. Це забезпечує гнучкість у керуванні інформацією;

– проєкт підтримує підключення до API сервісів відеозв'язку, таких як Zoom та Jitsi, що дозволяє проводити заняття в реальному часі без виходу за межі платформи;

– впроваджено багаторівневу систему авторизації та автентифікації, використано механізми захисту від CSRF-атак, хешування паролів, логування дій користувачів;

– проведено модульне та інтеграційне тестування основних компонентів системи. Залучено бібліотеки unittest і pytest. Виявлені та усунуті критичні помилки, забезпечено стабільність у роботі;

– проєкт контейнеризовано за допомогою Docker, що спрощує запуск та розгортання на будь-якому середовищі. Підготовлено повну технічну документацію для супроводу та подальшого розвитку проєкту.

Проєкт Moodle2.0 вже на рівні прототипу демонструє здатність замінити типові платформи для дистанційного навчання там, де потрібна більша гнучкість і контроль. Він може використовуватись у малих школах, курсах, гуртках, які прагнуть цифровізувати навчальний процес з урахуванням власної специфіки.

Отже, мета кваліфікаційної роботи виконана. Розроблено прототип сучасного програмного продукту, що не лише задовольняє поточні потреби освітніх установ, а й має значний потенціал для масштабування, інтеграції з іншими освітніми системами та подальшого вдосконалення. Виконана робота створює передумови для розробки повноцінної екосистеми електронного навчання в Україні на базі відкритих стандартів і технологій.

ПЕРЕЛІК ПОСИЛАНЬ

1. Anderson T. The Theory and Practice of Online Learning. 2nd ed. Athabasca University Press, 2008. 472 с.

2. Ally M. Mobile Learning: Transforming the Delivery of Education and Training. Athabasca University Press, 2009. 284 с.

3. The potential and challenges of Integrating Generative AI in Higher Education as Received by Teaching Staff. *A Phenomenological Study Information Systems Education Journal*. ISSN: 1545-679X. URL : https://isedj.org/ (дата звернення : 28.02.2025).

4. Unlocking Success. *Key Features of College Online Pedagogical Practices that Predict Better Performance Computers & Education*. ISSN: 0360-1315. Elsevier. URL : https://www.sciencedirect.com/journal/computers-and-education (дата звернення : 28.02.2025).