

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Веб-технології»)

на тему:

**«Вебплатформа для вивчення іноземних мов з
інтеграцією Телеграм-бота»**

Виконав студент групи ВТ-22-1
ОЛІЙНИК Дмитрій Сергійович

Керівник роботи:
ЛОКТИКОВА Тамара Миколаївна

Рецензент:
ЛОБАНЧИКОВА Надія Миколаївна

Житомир – 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ "ЖИТОМИРСЬКА ПОЛІТЕХНІКА"
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»
Зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«28» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ОЛІЙНИК Дмитрій Сергійович**

Керівник роботи: **ЛОКТИКОВА Тамара Миколаївна**

Тема роботи: **«Вебплатформа для вивчення іноземних мов з інтеграцією Телеграм-бота»**,

затверджена Наказом закладу вищої освіти від «27» лютого 2026р., №92/с

Вихідні дані для роботи: **об'єктом дослідження є процес управління он-лайн навчанням та вивченням іноземних мов у межах спеціалізованої платформи, а предметом дослідження є технології, методи та інструменти, що застосовуються для розробки клієнт-серверних застосунків, а також підходи до створення навчального контенту та адміністративних панелей.**

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Локтікова Т.М	02.03.2026	02.03.2026
2	Локтікова Т.М	02.03.2026	02.03.2026
3	Локтікова Т.М	02.03.2026	02.03.2026

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з вебплатформи та пояснювальної записки. Пояснювальна записка до кваліфікаційної роботи містить текстову частину, викладену на 83 сторінках друкованого тексту, містить 28 ілюстрацій, 31 таблицю. Список використаних джерел містить 30 найменувань і займає 2 сторінки. Також у роботі є 14 сторінок додатків. Загальний обсяг роботи складає 97 сторінок.

Мета роботи – розробка вебплатформи для навчального процесу вивчення іноземних мов, що забезпечує проходження навчальних курсів, виконання вправ, відстеження прогресу, публікацію постів про створення нових курсів, а також адміністрування системи через відповідні інструменти керування.

В роботі використано методи веброзробки, проєктування баз даних, побудови REST API, автентифікації користувачів із використанням JWT-токенів, а також розробки інтерфейсу для управління даними. Отримані результати включають спроектовану архітектуру вебплатформи, реалізовану реляційну базу даних для зберігання інформації про користувачів, курси, уроки, пости, підписки та платежі, а також реалізовані серверні модулі для управління навчальним процесом, гейміфікацією та користувачами.

Рекомендовано впровадження розробленої вебплатформи для автоматизації процесу вивчення іноземних мов, підвищення мотивації учнів за рахунок елементів гейміфікації, а також спрощення адміністрування платформи та контролю за її роботою.

КЛЮЧОВІ СЛОВА: ВЕБПЛАТФОРМА, НАВЧАННЯ, ІНОЗЕМНІ МОВИ, ТЕЛЕГРАМ-БОТ, КУРСИ, FASTAPI, REACT, PYTHON, REST API.

					ІПЗ.КР.Б – 121 – 26 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Вебплатформа для вивчення іноземних мов з інтеграцією Телеграм-бота	Літ.	Арк.	Аркуші
Розроб.		Олійник Д.С.						
Перевір.		Локтікова Т.М.					3	97
Керівник		Локтікова Т.М.				Житомирська політехніка, група ВТ-22-1		
Н. контр		Тичина О.П.						
Зав.каф.		Вакалюк Т.А.						

ABSTRACT

The bachelor's qualification thesis consists of a web platform and an explanatory note. The explanatory note to the qualification thesis contains a textual part presented on 83 pages of printed text, 28 illustrations, and 31 tables. The list of references contains 30 items and occupies 2 pages. Also, there are 14 pages of appendices in the thesis. The total volume of the thesis is 97 pages.

The aim of the thesis is to develop a web platform for the educational process of learning foreign languages, which enables taking educational courses, completing exercises, tracking progress, publishing posts about the creation of new courses, and administering the system through appropriate management tools.

The thesis utilizes methods of web development, database design, REST API building, user authentication using JWT tokens, as well as interface development for data management. The obtained results include the designed architecture of the web platform, an implemented relational database for storing information about users, courses, lessons, posts, subscriptions, and payments, as well as implemented server modules for managing the educational process, gamification, and users.

The implementation of the developed web platform is recommended to automate the process of learning foreign languages, increase students' motivation through gamification elements, and simplify the platform's administration and control over its operation.

KEYWORDS: WEB PLATFORM, LEARNING, FOREIGN LANGUAGES, TELEGRAM BOT, COURSES, FASTAPI, REACT, PYTHON, RxEEST API.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	6
ВСТУП.....	7
РОЗДІЛ 1 АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ВЕБПЛАТФОРМИ.....	9
1.1 Постановка задачі.....	9
1.2 Аналіз аналогів програмного продукту.....	12
1.3 Вибір архітектури вебплатформи.....	18
1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення.....	21
Висновки до розділу 1	24
РОЗДІЛ 2 ПРОЄКТУВАННЯ ВЕБПЛАТФОРМИ.....	25
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи	25
2.2 Розробка бази даних	36
2.3 Проєктування та реалізація алгоритмів роботи системи.....	50
2.4 Реалізація функціоналу вебплатформи	54
Висновки до розділу 2	57
РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБПЛАТФОРМОЮ	59
3.1 Порядок встановлення та налаштування системи.....	59
3.2 Структура інтерфейсу та порядок взаємодії із системою.....	62
3.3 Тестування роботи програмного продукту	74
Висновки до розділу 3	77
ВИСНОВКИ.....	78
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	80
ДОДАТКИ.....	83

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних.

СУБД – система управління базами даних.

API – програмний інтерфейс застосунку (Application Programming Interface).

REST API – архітектурний стиль взаємодії клієнта та сервера через HTTP (Representational State Transfer Application Programming Interface).

JWT – токен доступу для автентифікації користувачів (JSON Web Token).

UX – досвід користувача (User Experience).

UI – інтерфейс користувача (User Interface).

HTTP – протокол передачі гіпертексту (HyperText Transfer Protocol).

HTTPS – захищений протокол HTTP із використанням шифрування.

ORM – технологія відображення об'єктів у базу даних (Object-Relational Mapping).

JSON – текстовий формат обміну даними (JavaScript Object Notation).

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Актуальність теми зумовлена стрімким зростанням популярності он-лайн навчання та високим попитом на сучасні освітні вебпродукти. Люди прагнуть вивчати іноземні мови в зручному для себе темпі, використовуючи інтерактивні інструменти. Традиційні підходи поступово поступаються місцем цифровим платформам та мобільним застосункам, які дозволяють не лише отримувати теоретичні знання, а й одразу застосовувати їх на практиці. Відповідно, існує гостра необхідність у проектуванні надійних інформаційних систем, здатних автоматизувати процеси навчання, перевірку завдань, а також контроль за успішністю користувачів.

Ключовим фактором успіху таких продуктів є застосування передових веб-технологій. Вони дають змогу розробляти високопродуктивні та безпечні рішення, що базуються на клієнт-серверній архітектурі. Крім того, важливим трендом є інтеграція додаткових механізмів залучення аудиторії — зокрема, гейміфікації для стимулювання інтересу учнів та зручної системи публікації новин і оновлень.

Метою випускної кваліфікаційної роботи є створення вебплатформи, орієнтованої на управління процесом вивчення іноземних мов. Цей продукт має надавати інструменти для формування навчальних курсів, виконання інтерактивних завдань, моніторингу особистого прогресу та успіхів інших учнів, а також містити зручний адміністративний інтерфейс для управління всім контентом платформи.

Для досягнення поставленої мети необхідно вирішити такі **завдання**:

- 1) дослідити предметну область та виконати огляд існуючих аналогів на ринку он-лайн освіти;
- 2) розробити загальну архітектуру програмного продукту та спроектувати реляційну базу даних;
- 3) створити серверну частину (бекенд), що базується на REST API, для забезпечення обміну даними;

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

4) реалізувати безпечний механізм авторизації та ідентифікації користувачів;

5) створити програмні модулі для роботи з курсами, уроками, навчальними завданнями та новинами;

6) інтегрувати підсистему гейміфікації (бали, рівні, сертифікати) та функціонал обробки користувацьких підписок;

7) спроектувати та розробити клієнтську частину (фронтенд) для комфортної взаємодії з платформою;

8) провести тестування створеного програмного забезпечення.

Об'єктом дослідження є процес управління он-лайн навчанням та вивченням іноземних мов у межах спеціалізованої платформи.

Предметом дослідження є технології, методи та інструменти, що застосовуються для розробки клієнт-серверних застосунків, а також підходи до створення навчального контенту та адміністративних панелей.

Практичне значення отриманих результатів полягає у тому, що створена вебплатформа є повністю функціональним і готовим до розгортання рішенням для організації дистанційного вивчення іноземних мов. Наявність захищеної адміністративної панелі дозволяє уповноваженим адміністраторам (або власникам платформи) централізовано керувати всім навчальним контентом без необхідності втручання в програмний код. Крім того, інтеграція платформи у формат Telegram Mini App суттєво підвищує доступність сервісу для мобільних користувачів, а реалізовані алгоритми гейміфікації сприяють підвищенню рівня залученості та мотивації учнів до безперервного навчання.

За темою випускної кваліфікаційної роботи бакалавра були опубліковані тези: Олійник Д.С., Локтікова Т.М. Вебплатформа для вивчення іноземних мов з підтримкою Телеграм-бота: тези Всеукраїнської науково-практичної on-line конференції здобувачів вищої освіти і молодих учених, присвяченої Дню науки, м. Житомир, 11-15 травня 2026 р. Житомир: «Житомирська політехніка», 2026. С. 86-87 [1].

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

РОЗДІЛ 1 АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ВЕБПЛАТФОРМИ

1.1 Постановка задачі

Головне завдання даної кваліфікаційної роботи полягає у проектуванні та створенні комплексної вебплатформи, призначеної для організації процесу вивчення іноземних мов. Ця система має гарантувати безпечну та комфортну взаємодію між учнями та адміністраторами контенту, а також демонструвати високу стабільність роботи як клієнтської, так і серверної частин. Важливою умовою є розробка інтуїтивно зрозумілого інтерфейсу, який дозволить користувачам легко орієнтуватися в навчальних матеріалах без додаткових інструкцій.

Процес розробки спирається на передові методи побудови вебзастосунків, зокрема використання компонентного підходу на клієнтській частині та асинхронної обробки запитів на серверній, що дає можливість отримати гнучкий до навантажень та надійний програмний продукт. Усі кроки побудови системи виконуються за чітким алгоритмом: від створення макетів користувацького інтерфейсу та проектування нормалізованої бази даних до розробки REST API, інтеграції зовнішніх хмарних сервісів і фінального налаштування серверного середовища.

На першому етапі передбачено реалізацію клієнтської частини (фронтенду). Вона охоплює розробку головної сторінки з каталогом доступних мовних курсів, екранів реєстрації та входу, сторінок з детальним описом навчальних програм, а також спеціального розділу для адміністраторів, де відбувається управління контентом. Крім того, впроваджується інструментарій для пошуку та фільтрації курсів, завдяки якому учні зможуть швидко підбирати потрібні матеріали за рівнем складності або категорією. Додатково реалізовано блок новин, де відображаються інформаційні пости про появу нових уроків.

Окрему увагу приділено створенню інтерактивного середовища для проходження уроків, де користувачі виконують тестові завдання та отримують

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

миттєвий зворотний зв'язок. Також розроблено особистий кабінет студента для візуалізації прогресу, зароблених балів досвіду (XP), здобутих сертифікатів та керування преміум-підпискою. Усі інтерфейси є повністю адаптивними, що гарантує їх коректну роботу як у веббраузерах, так і у форматі Telegram Mini App.

Важливою інноваційною складовою є інтеграція клієнтської частини з месенджером у форматі Telegram Mini App (Web App). Це дозволяє користувачам запускати повноцінний додаток прямо всередині Telegram, забезпечуючи максимально швидкий та зручний доступ до навчання без необхідності відкривати браузер чи встановлювати сторонні програми на телефон.

Серверна архітектура (бекенд) складається з кількох ключових програмних модулів, які в сукупності забезпечують безперебійне функціонування платформи. До них належать: модуль управління користувачами, модуль інтеграції з Telegram, модуль навчальних курсів та уроків, модуль відстеження прогресу (виконання вправ), модуль гейміфікації, модуль обробки підписок (платежів) та модуль публікацій (постів). Кожен із цих компонентів відповідає за свою специфічну бізнес-логіку та обмінюється даними з іншими через API.

Модуль управління користувачами бере на себе функції реєстрації, автентифікації та редагування особистих профілів. Він також відповідає за розподіл прав доступу між звичайними учнями та адміністраторами системи.

Модуль відстеження прогресу відповідає за збереження результатів виконання практичних вправ та проходження уроків. Завдяки йому учні можуть бачити свою статистику та аналізувати власні досягнення.

Модуль гейміфікації керує нарахуванням балів досвіду (XP), формуванням безперервних днів навчання та видачею сертифікатів, що суттєво підвищує мотивацію користувачів повертатися до навчання.

Модуль обробки підписок відповідає за фінансовий аспект роботи платформи. Він гарантує безпечну взаємодію з платіжними шлюзами для оформлення платних рівнів доступу (преміум-підписок) та збереження історії транзакцій.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

Модуль публікацій дозволяє адміністраторам створювати інформаційні пости, щоб оперативно повідомляти аудиторію про оновлення платформи або релізи нових мовних курсів. Зі свого боку, студенти мають змогу активно взаємодіяти з цим контентом: залишати коментарі, ставити вподобайки та обговорювати навчальний процес.

Важливим кроком після написання основних модулів є налаштування безперебійного зв'язку між сервером та інтерфейсом користувача. Для цього застосовується архітектурний стиль REST API [2], який стандартизує обробку мережових запитів та передачу інформації. Використання REST-архітектури робить систему гнучкою, полегшує її подальше обслуговування та залишає простір для інтеграції зі сторонніми сервісам.

Завершальною стадією розробки є комплексне тестування створеного продукту. Воно охоплює перевірку працездатності кожного окремого модуля, аналіз безпеки взаємодії між клієнтом і сервером, а також оцінку зручності (UX) дизайну як у звичайному браузері, так і в середовищі Telegram. Особлива увага під час тестування приділяється перевірці коректності нарахування балів досвіду (XP), валідації механізму генерування сертифікатів та безперебійній роботі модуля імітації преміум-підписок. Такий підхід дозволяє ідентифікувати та виправити потенційні дефекти до офіційного запуску платформи.

Отже, результатом виконання роботи стає повноцінна освітня вебплатформа, яка надає користувачам ефективні інструменти для вивчення іноземних мов (включно з інтеграцією Telegram Mini App), а адміністраторам — надійний засіб для зручного управління навчальним контентом через автоматизовану панель. Розроблена система вирізняється сучасним кросплатформним інтерфейсом, відмовостійкою серверною частиною та високим потенціалом для подальшого масштабування. Завдяки гнучкій модульній архітектурі на базі FastAPI та React, система легко піддається впровадженню нових функцій, зокрема інтеграції реальних платіжних шлюзів, що робить її повністю готовою до практичного та комерційного використання.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

1.2 Аналіз аналогів програмного продукту

Розробка вебплатформи вивчення іноземних мов вимагає попереднього глибокого аналізу вже існуючих на ринку рішень. Сьогодні сфера освітніх онлайн-технологій (EdTech) переживає стрімке зростання, що сприяє появі безлічі платформ для проходження мовних курсів, тренування граматики та розширення словникового запасу.

Попри величезну кількість готових освітніх продуктів, більшість із них мають суттєві недоліки. Одні системи пропонують занадто несерйозний підхід, інші, навпаки, перевантажені академічним функціоналом або обмежують базові можливості жорсткими платними підписками. Також на ринку вкрай рідко зустрічається зручна інтеграція з популярними месенджерами. Тому доцільно розглянути найвідоміші аналоги, щоб виявити їхні сильні та слабкі сторони.

Одним із найпопулярніших світових сервісів є Duolingo [3], який зробив революцію у вивченні мов завдяки впровадженню потужної системи гейміфікації. На головній сторінці застосунку користувача зустрічає так зване "дерево навичок" (рис. 1.1), яке візуалізує траєкторію навчання. Кожен модуль розбитий на серію коротких інтерактивних уроків.

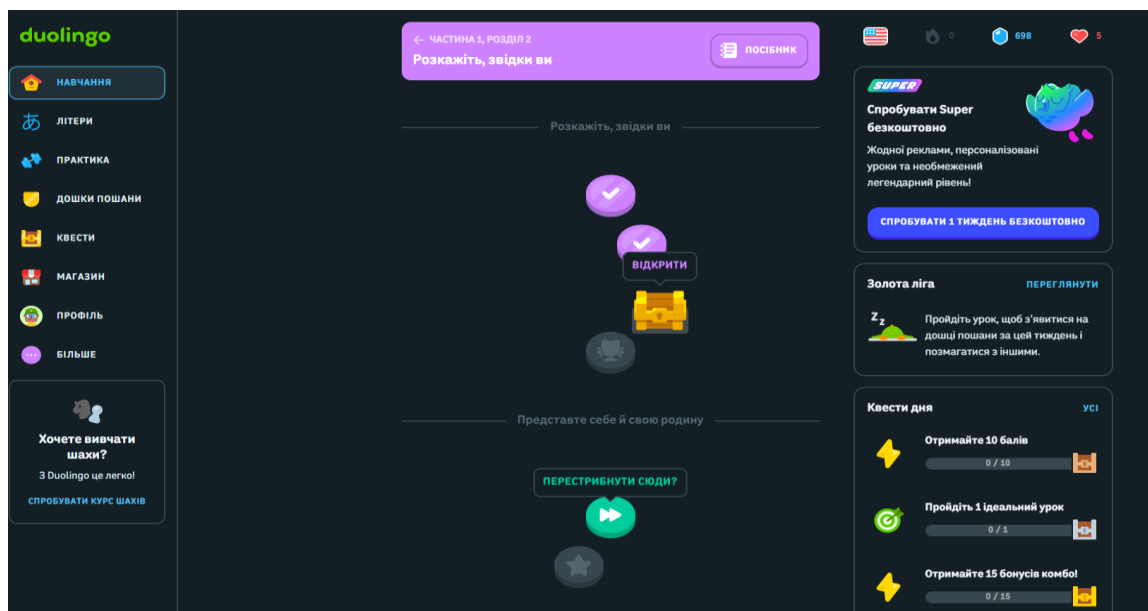


Рис. 1.1. Головна сторінка екрану навчання у Duolingo

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

Головною особливістю платформи є система утримання уваги (рис. 1.2). Користувачі заробляють бали досвіду (XP), змагаються у лігах та підтримують "стріки" (дні безперервного навчання). Такий підхід робить навчання схожим на гру.

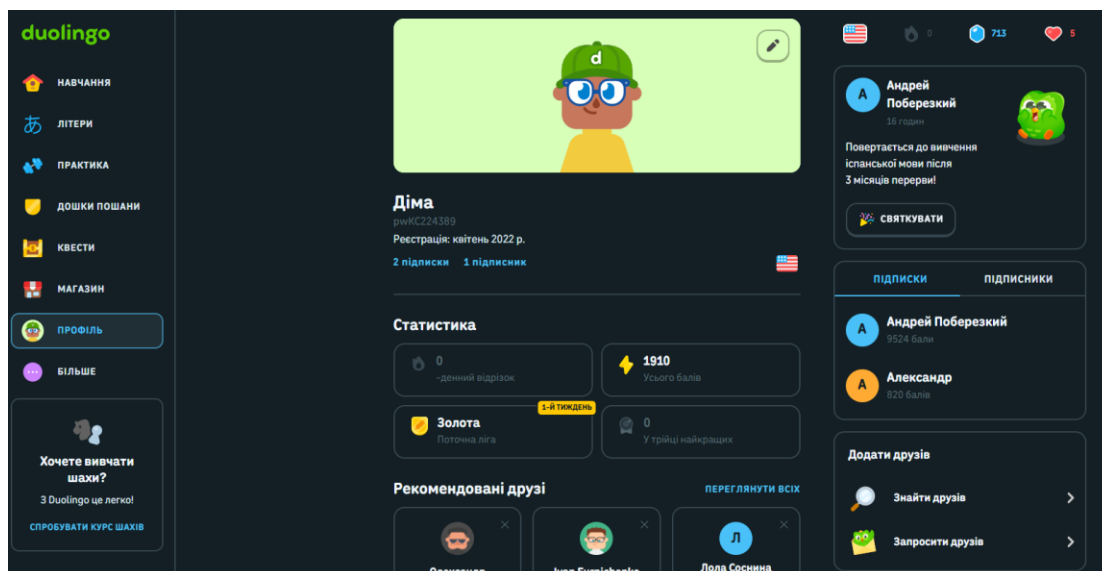


Рис. 1.2. Інтерфейс гейміфікації та рейтингів (ліг)

Проте, незважаючи на успішну ігрову механіку, сервісу часто бракує можливості публікації інформаційних постів або новин від розробників, а взаємодія з адміністраторами зведена до нуля.

Другим потужним аналогом у сфері електронного навчання є платформа Busuu [4], яка пропонує більш структурований підхід до вивчення іноземних мов, максимально наближений до класичних академічних стандартів. На відміну від Duolingo, Busuu робить акцент на повноцінні уроки, що чітко розподілені за темами та загальноєвропейськими компетенціями володіння іноземною мовою, а також приділяє значну увагу детальним поясненням граматики.

На рис. 1.3 представлено сторінку з каталогом доступних курсів англійської мови. Як видно із зображення, інтерфейс платформи дозволяє студентам гнучко обирати не лише загальні мовні програми, а й вузькоспеціалізовані напрямки залежно від їхніх персональних цілей: наприклад, англійську для подорожей, ділового спілкування, академічних потреб або підготовки до міжнародних іспитів. Важливою перевагою системи є те, що після успішного проходження всіх модулів обраного курсу та складання підсумкових

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

тестів, користувач отримує офіційний цифровий сертифікат, що підтверджує його здобутий мовний рівень.

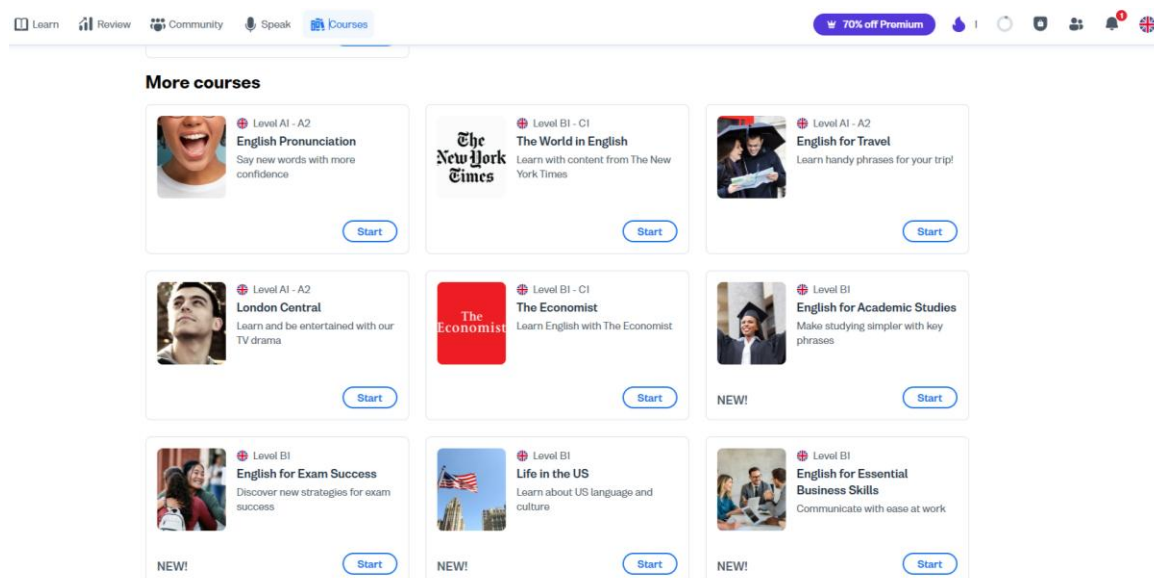


Рис. 1.3. Сторінка курсів Англійської в Busuu

Водночас платформа монетизується через систему щомісячних або річних підписок. У системі реалізовано жорстке розмежування між базовим та преміум-контентом. На рис. 1.4 представлено модуль оформлення платної підписки Busuu. Користувачі безкоштовної версії мають доступ лише до обмеженого набору карток і базових завдань. Натомість поглиблене вивчення граматики, отримання сертифікатів, завантаження уроків для офлайн-режиму та перевірка завдань іншими користувачами є платними преміум-функціями.

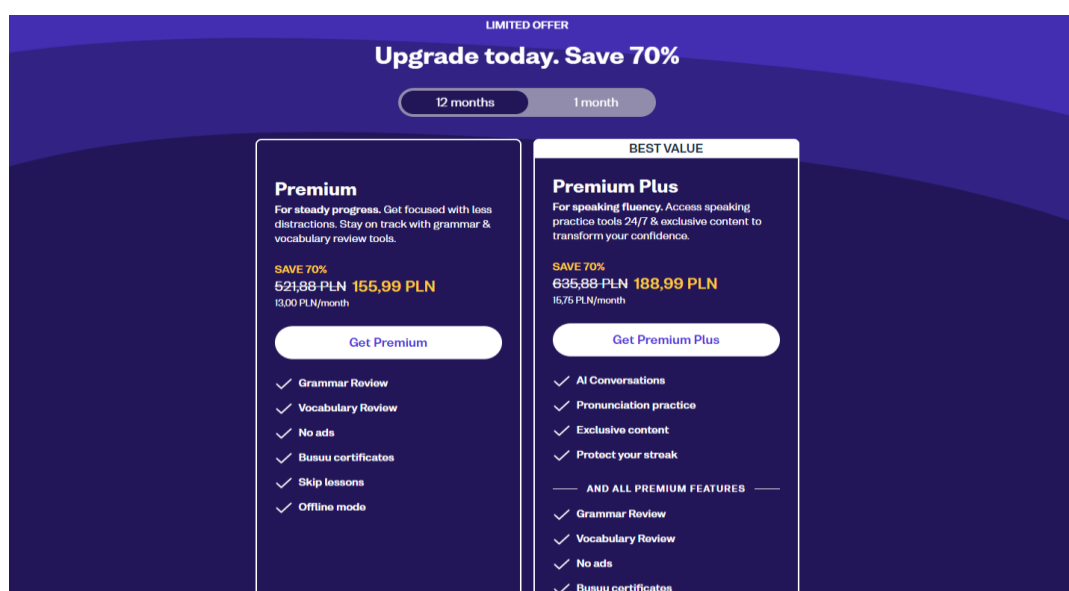


Рис. 1.4. Модуль оформлення платної підписки Busuu

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

Наступним проаналізованим сервісом є Memrise [5] — міжнародна платформа, яка спеціалізується на ефективному розширенні словникового запасу та швидкому сприйнятті мови на слух. Окрім цього, даний сервіс активно використовує короткі відеоролики з реальними носіями мови для покращення вимови.

Однією з ключових особливостей сучасного Memrise є наявність спеціалізованого модуля для підготовки до міжнародних іспитів (рис. 1.5). У розділі «Exam Prep» користувачі можуть знайти цілеспрямовані матеріали для тренування перед складанням сертифікаційних тестів, таких як IELTS, TOEFL тощо. Це дозволяє сфокусувати навчання на конкретних академічних цілях.

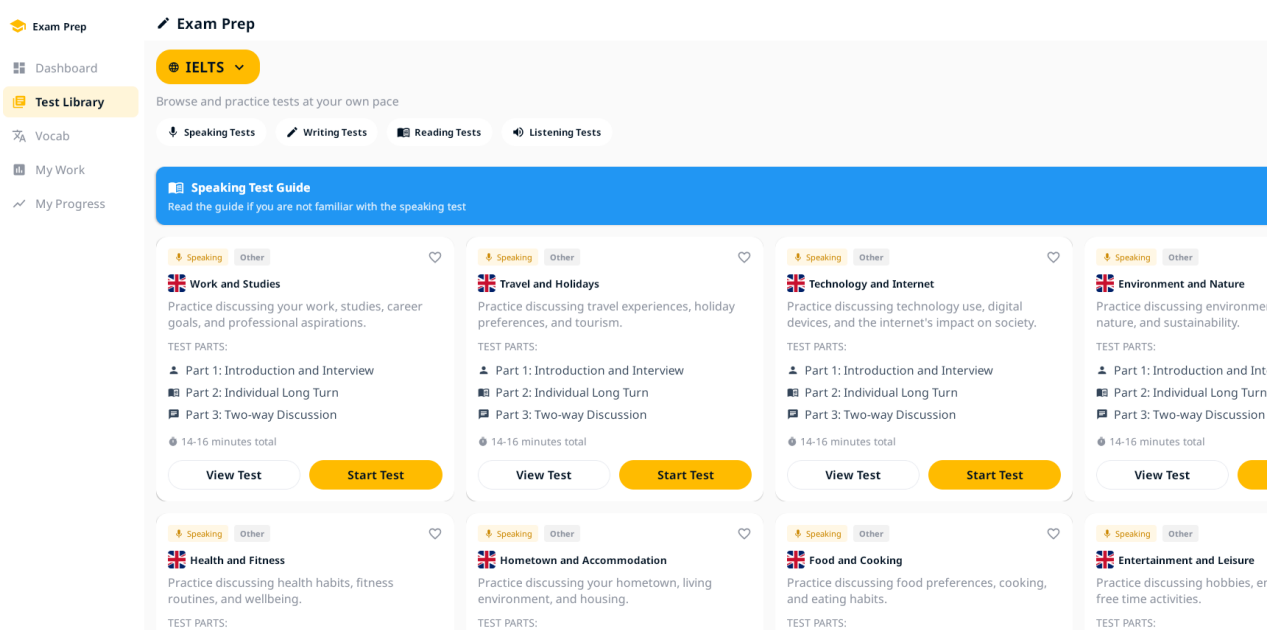


Рис. 1.5. Інтерфейс модуля підготовки до іспитів у Memrise

Система пропонує короткі відеоролики від справжніх носіїв мови та різноманітні інтерактивні тести (рис. 1.6). Проте інтерфейс адміністрування власного контенту часто виявляється занадто складним та дуже перевантаженим для новачків. Через велику кількість візуальних елементів та складну навігаційну структуру, користувачам потрібно витратити значно більше часу на створення нових навчальних модулів, що є суттєвим недоліком для сучасних освітніх платформ.

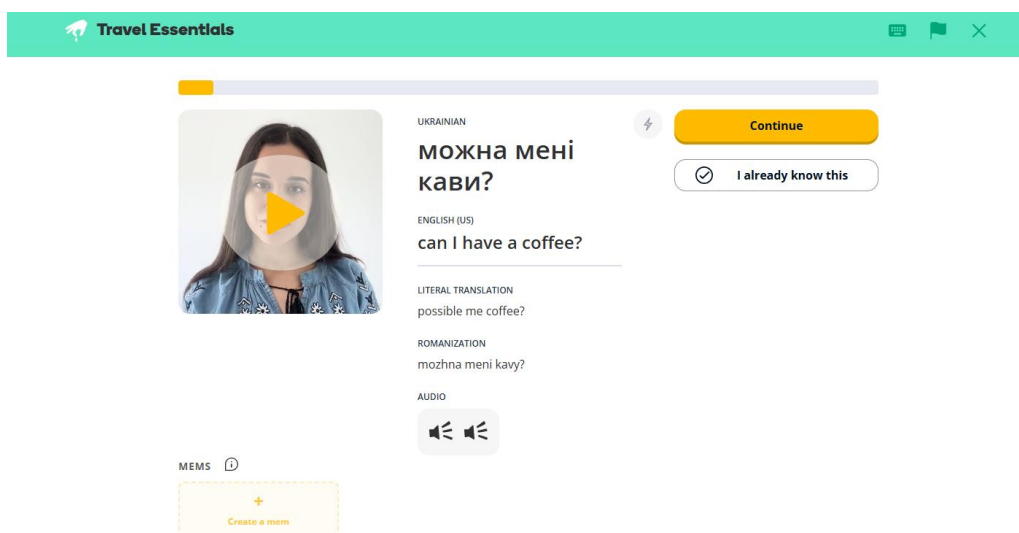


Рис. 1.6. Інтерфейс проходження практичної вправи в Memrise

Після детального розгляду функціональних можливостей досліджуваних платформ доцільно узагальнити результати у вигляді табл. 1.1.

Таблиця 1.1

Порівняльний аналіз освітніх платформ

Критерій	Duolingo	Busuu	Memrise
Структуровані навчальні курси	✓	✓ (розширений)	✓
Пошук та фільтрація курсів	відсутній	✓	✓
Ролі (користувач / адмін)	мінімальні	частково	✓
Нарахування балів (XP) та стріки	✓ (розвинена)	відсутня	✓ базова
Видача сертифікатів	відсутня	✓	✓
Система платних підписок	✓	✓	✓
Блок інформаційних постів/новин	відсутній	відсутній	відсутній

Продовження табл. 1.1

Точка входу через Telegram Mini App	відсутній	відсутній	відсутній
API для інтеграції	закрите	закрите	обмежене
Зручність інтерфейсу	дуже проста	середня	перевантажена
Можливість стежити за іншими учнями	✓ (друзі, ліги)	✓ (спільнота)	мінімальна
Модуль підготовки до іспитів	окремий сервіс	✓	обмежене
Адаптивність інтерфейсу (Web / Mobile)	✓	✓	✓

Порівняння розглянутих рішень показує, що кожна платформа має унікальну спеціалізацію. Одні сервіси роблять ставку на гейміфікацію (Duolingo), інші — на вивчення та сертифікацію (Busuu), а треті — на створення контенту спільнотою (Memrise). Водночас жодне рішення не поєднує ідеальний баланс зручного адміністрування, гейміфікації, системи інформування та швидкого доступу через месенджери.

Таким чином, аналоги не повністю задовольняють потреби учнів та викладачів. Це створює передумови для розробки нової веб-платформи, яка об'єднує функції аналогів та доповнює їх розширеними можливостями. Зокрема, у системі передбачено управління курсами, систему гейміфікації (бали, рівні), обробку підписок, створення новинних постів та інтеграцію Telegram Mini App.

Розроблювана вебплатформа у даному контексті розглядається як більш цілісне та сучасне рішення, спрямоване на усунення обмежень існуючих платформ та забезпечення максимально комфортного процесу вивчення мов.

1.3 Вибір архітектури вебплатформи

Визначення архітектурних рішень є фундаментальним кроком під час створення будь-якого програмного забезпечення, адже від цього залежить відмовостійкість, потенціал для розширення та зручність підтримки коду. Оскільки розроблювана освітня платформа орієнтована на одночасну роботу багатьох учнів та адміністраторів, критично важливо було спроектувати надійну ієрархію компонентів та забезпечити швидкісний обмін інформацією між ними.

В основу розробки покладено клієнт-серверну модель взаємодії з використанням REST-API архітектури. Цей принцип передбачає чітке розмежування застосунку на дві ізольовані складові: фронтенд, який відповідає за графічне відображення та взаємодію з користувачем, і бекенд, що бере на себе виконання бізнес-логіки, розрахунки та комунікацію з базою даних. Подібний розподіл обов'язків робить продукт надзвичайно гнучким, дозволяє легко масштабувати ресурси та паралельно вдосконалювати різні частини системи.

Серверна складова виступає центральним вузлом обробки даних, координуючи функціонування всіх підсистем. Вона керує процесами авторизації, управління навчальними курсами, фіксацією прогресу учнів, нарахуванням ігрових балів (гейміфікацією) та обробкою підписок. Саме завдяки такій централізації вдається гарантувати цілісність інформації та суворий контроль доступу до приватних ресурсів.

Питання кібербезпеки та захисту даних є одним із пріоритетних напрямків. У системі впроваджено надійний механізм ідентифікації користувачів і суворе розмежування повноважень між звичайними студентами та адміністраторами. Це унеможливорює несанкціоноване втручання в навчальний контент або службові налаштування платформи.

Клієнтська частина реалізована у вигляді сучасного односторінкового застосунку (SPA) та додатково інтегрована як Telegram Mini App. Через ці інтерфейси учень може переглядати доступні мовні курси, фільтрувати їх за рівнем складності, проходити уроки, виконувати вправи та відслідковувати свою

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		18

статистику. Уся актуальна інформація підтягується із сервера в режимі реального часу через програмний інтерфейс (API).

Комунікація між фронтом та бекендом відбувається через захищені HTTP-запити. Для стандартизації обміну даними обрано формат JSON, який є універсальним для більшості сучасних технологій і чудово підходить для інтеграцій. Для спрощення архітектури та забезпечення високої швидкості роботи всі мультимедійні матеріали (зображення курсів чи аватари) інтегруються шляхом збереження прямих URL-посилань у базі даних без фізичного завантаження файлів на сервер. Такий чіткий протокол взаємодії гарантує незалежність модулів та полегшує майбутню модернізацію.

Для спрощення процесу розробки та подальшого технічного супроводу архітектура розбита на функціональні блоки (модулі). Кожен окремий блок відповідає за свій ізольований напрямок: користувачі, курси, тести, гейміфікація, платежі та новини. Такий принцип слабкої зв'язності дозволяє оновлювати або змінювати одну частину системи, не порушуючи роботу інших.

У бекенді активно застосовується патерн сервісного шару (Service Layer). Його суть полягає у фізичному відокремленні складної бізнес-логіки від контролерів, які просто приймають HTTP-запити. Наприклад, алгоритми нарахування досвіду (XP) за пройдени уроки, перевірка правильності відповідей або формування статистики винесені в окремі сервісні класи. Контролери (роутери) лише приймають запит від користувача і передають його в сервіс. Це робить програмний код набагато чистішим, зручнішим для тестування і придатним для повторного використання.

Основним інструментом для побудови серверної частини обрано сучасний асинхронний фреймворк FastAPI. На відміну від класичних рішень, він забезпечує максимальну швидкість обробки запитів завдяки підтримці асинхронності. Робота з базою даних здійснюється за допомогою бібліотеки SQLAlchemy, яка виконує роль ORM-системи, а валідація даних та їх серіалізація відбувається через Pydantic. У загальному вигляді архітектура поєднує React-фронтенд, REST API на FastAPI, сервісний шар та реляційну базу даних.

		Олійник Д.С.			ПІЗ.КР.Б–121–26–ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		19

З метою забезпечення високої продуктивності при великих навантаженнях було застосовано ряд оптимізацій. Зокрема, для зменшення кількості звернень до бази даних впроваджено систему кешування на базі Redis, що дозволяє миттєво віддавати часто запитувану інформацію. Також під час виведення великих масивів інформації використовується пагінація, що зменшує навантаження на мережу.

Велику увагу приділено безпеці транзакцій та сесій. Для підтримки авторизованого стану застосовуються безпечні JWT-токени. Система автоматично валідує всі вхідні дані від клієнта та має механізми захисту від спаму (Rate Limiting). Підтвердження оформлення підписок і доступ до преміум-контенту реалізовано через захищені внутрішні перевірки бази даних, що виключає можливість несанкціонованого доступу.

Архітектура також тісно взаємодіє зі сторонніми сервісами. Оскільки весь медіаконтент підвантажується через зовнішні URL-посилання, це значно розвантажує основний сервер та не вимагає підтримки власного файлового сховища. Відправка системних сповіщень реалізована через інтеграцію з поштовими сервісами за протоколом SMTP. Крім того, ключовою інтеграцією є використання платформи Telegram для роботи бота та запуску Mini App.

З архітектурної точки зору розроблений продукт є модульним монолітом. Це означає, що всі підсистеми розгорнуті в рамках одного серверного додатка і працюють з єдиною базою даних, але на рівні програмного коду вони жорстко ізольовані одна від одної. Таке рішення є оптимальною серединою: воно дозволяє уникнути надмірної складності мікросервісних архітектур, але при цьому гарантує простоту розгортання, підтримки та масштабування.

Отже, спроектована архітектура вебплатформи створює надійний фундамент для ефективного управління навчальним процесом. Обрані підходи і технології забезпечують високу швидкість обробки даних, безпеку особистої інформації та залишають широкі можливості для додавання нового функціоналу в майбутньому.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		20

1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Для успішної реалізації освітньої вебплатформи було сформовано сучасний стек технологій, покликаний гарантувати максимальну швидкодію, безпеку даних та легкість подальшого масштабування. Процес вибору інструментарію базувався на архітектурних вимогах: необхідності створення клієнт-серверного вебзастосунку з використанням REST API, здатного обробляти велику кількість паралельних запитів від учнів, керувати мультимедійним контентом та підтримувати обробку підписок [6].

Серверна частина (бекенд) написана мовою програмування Python [7] із застосуванням інноваційного асинхронного вебфреймворку FastAPI. Вибір саме цього інструменту, а не класичних рішень на кшталт Django, пояснюється його надзвичайною продуктивністю (яка наближається до NodeJS та Go), нативною підтримкою асинхронного програмування та автоматичною генерацією документації (Swagger). Використання FastAPI у поєднанні з бібліотекою Pydantic забезпечує строгу типізацію та автоматичну валідацію вхідних даних, що зводить до мінімуму кількість помилок на етапі обробки запитів.

Для захисту персональних даних та контролю сесій інтегровано механізм JSON Web Token (JWT) [9]. Цей стандарт дозволяє реалізувати автентифікацію без збереження стану (stateless), тобто серверу не потрібно тримати дані про сесії у пам'яті. Використання короткострокових access-токенів та довгострокових refresh-токенів створює ідеальний баланс між високим рівнем безпеки та комфортом користувача. Додатковий рівень захисту забезпечується інструментами обмеження частоти запитів (Rate Limiting) на базі системи Redis.

Клієнтський інтерфейс розроблено на основі екосистеми React. Ця бібліотека є індустріальним стандартом для створення інтерактивних односторінкових додатків (SPA). Вибір React обумовлений необхідністю миттєво реагувати на дії учня під час виконання мовних вправ, перегляду курсів чи оновлення ігрових балів (XP) без перезавантаження сторінки. Завдяки

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

концепції віртуального DOM, React забезпечує плавну роботу навіть складних анімацій та дашбордів.

Для пришвидшення процесу збірки фронтенду використовується сучасний бандлер Vite, а для стилізації інтерфейсу — фреймворк Tailwind CSS, який дозволяє швидко створювати адаптивний дизайн без написання об'ємних CSS-файлів. Окремою перевагою обраного фронтенд-стеку є його здатність безшовно інтегруватися в екосистему Telegram у форматі Mini App, що робить платформу доступною безпосередньо з месенджера.

Особливої уваги заслуговує архітектура взаємодії з платформою Telegram. Завдяки використанню Telegram Web Apps, автентифікація користувачів відбувається автоматично та цілком безпечно на основі криптографічного підпису, який сервер перевіряє за допомогою секретного токена бота. Це позбавляє учнів необхідності вигадувати та запам'ятовувати складні паролі, а також суттєво знижує ризик несанкціонованого доступу до облікових записів. Крім того, такий підхід гарантує безперервний зв'язок між клієнтом та сервером, забезпечуючи миттєву доставку сповіщень про нові уроки або досягнення. Для забезпечення найвищого рівня продуктивності всі мережеві запити між месенджером та бекендом додатково шифруються за протоколом HTTPS, що відповідає сучасним стандартам захисту конфіденційної інформації у складних сучасних освітніх вебзастосунках.

Основним сховищем даних обрано реляційну систему керування базами даних PostgreSQL. Оскільки освітня платформа оперує великою кількістю взаємопов'язаних таблиць (користувачі, курси, уроки, прогрес, стріки, підписки), саме реляційна модель гарантує абсолютну цілісність та консистентність інформації. PostgreSQL відрізняється надійністю при виконанні складних транзакцій та агрегацій даних. Для взаємодії з базою даних на рівні коду використовується ORM-система SQLAlchemy у зв'язці з асинхронним драйвером asynpcrg, а для контролю версій бази даних (міграцій) — інструмент Alembic.

Зберігання статичного контенту (зображення для курсів, аватари користувачів, аудіоматеріали) винесено на хмарне об'єктне сховище Cloudflare

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

R2 [10]. Воно працює за сумісним із S3 протоколом, але пропонує швидку глобальну CDN-мережу доставки. Такий підхід повністю знімає навантаження з основного сервера та пришвидшує завантаження медіафайлів для кінцевих користувачів.

Для зручного управління платформою було інтегровано адміністративну панель на базі бібліотеки FastCRUD. Це рішення автоматично генерує інтерфейси для керування базою даних (створення нових уроків, модерація користувачів, додавання постів) без необхідності писати багато шаблонного коду, що значно спрощує роботу адміністратора.

У ході написання коду застосовувалися сучасні інтегровані середовища розробки (IDE), такі як Visual Studio Code та PyCharm. Для командної роботи, контролю версій та збереження історії змін використовувалася система Git. Для автоматизації процесу розгортання та гарантування ізоляції середовищ використовується технологія контейнеризації Docker. Вона дозволяє надійно запакувати серверний додаток, систему управління базами даних та всі необхідні залежності у спеціальний образ, який миттєво запускається на сервері без жодного ризику виникнення системних конфліктів. Для запуску асинхронного серверного додатка використовується сучасний високопродуктивний ASGI-сервер Uvicorn. Він безперебійно обробляє велику кількість одночасних з'єднань, що є критично важливим для забезпечення швидкого відгуку платформи під час масового напливу користувачів та гарантує стабільну роботу платформи в умовах постійного навантаження.

Щоб гарантувати стабільну роботу платформи, було визначено базові вимоги до апаратної частини. Для локального розгортання та розробки (середовище розробника) потрібен комп'ютер із мінімум двоядерним процесором та 8 ГБ оперативної пам'яті. Для розгортання серверної частини (продакшену) цілком достатньо хмарного сервера з 2-4 ГБ оперативної пам'яті. З боку клієнта жодних особливих апаратних вимог немає — учням достатньо мати сучасний веббраузер (Google Chrome, Safari тощо) на комп'ютері або встановлений застосунок Telegram на смартфоні.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

Висновки до розділу 1

У першому розділі кваліфікаційної роботи було здійснено детальний аналіз предметної області та доведено актуальність створення нової вебплатформи для вивчення іноземних мов. У межах розділу було чітко сформульовано мету та головні завдання проєкту, які зводяться до розробки надійного вебзастосунку на базі REST API архітектури з безпечною системою автентифікації та гейміфікації.

Під час виконання роботи було виконано огляд найпопулярніших на ринку аналогів, серед яких Duolingo, Busuu та Memrise. Аналіз виявив недоліки існуючих систем: від несерйозного підходу та обмежень до перевантаженого інтерфейсу і відсутності зручної точки входу. Це стало вагомим аргументом для створення власного продукту, який гармонійно поєднує уроки, мотивацію і доступність.

На базі проведеного аналізу було сформовано перелік ключових вимог до розроблюваної платформи. Серед них: реалізація системи управління мовними курсами, інтеграція підписок, публікація інформаційних постів та забезпечення зручного пошуку матеріалів. Особливий акцент було зроблено на проектуванні оптимізованої серверної архітектури, яка здатна обробляти запити, керувати даними та безперебійно взаємодіяти з клієнтською частиною (в тому числі у форматі Telegram Mini App).

Також у розділі обґрунтовано вибір технологій. Бекенд базується на Python та асинхронному фреймворку FastAPI, що забезпечує гнучкість і масштабованість системи. Безпека доступу реалізована через JWT-токени. Дані зберігаються у СУБД PostgreSQL, а медіафайли — у хмарному сховищі Cloudflare R2. Для керування контентом обрано адмін-панель FastCRUD, а кешування через Redis значно пришвидшує роботу платформи.

Отже, у першому розділі було закладено теоретичні основи для створення освітньої вебплатформи, визначено вимоги до її модулів та підібрано оптимальні інструменти. Зібрана інформація та рішення дозволяють перейти до проектування бази даних та практичної реалізації системи.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

РОЗДІЛ 2 ПРОЄКТУВАННЯ ВЕБПЛАТФОРМИ

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

Вебплатформа для вивчення іноземних мов надає зручний інтерфейс, за допомогою якого учні можуть переглядати каталог доступних освітніх матеріалів, проходити навчальні курси, виконувати практичні вправи, оформлювати підписки та відстежувати власний прогрес через систему гейміфікації. Для адміністраторів платформи передбачено окрему панель керування, яка дозволяє створювати та редагувати курси, структурувати уроки, публікувати інформаційні пости, а також керувати обліковими записами користувачів. Адміністративний модуль містить інструменти для управління контентом та перегляду системної статистики, що гарантує стабільний розвиток та модерацію платформи. Для забезпечення безпеки та цілісності даних усі користувачі чітко розподілені за рівнем доступу, що обмежує їхні можливості відповідно до призначеної ролі.

Короткий опис акторів (ролей) системи представлено в табл. 2.1.

Таблиця 2.1

Визначення акторів системи

Актор	Короткий опис
Студент (Учень)	Людина, яка реєструється або входить у систему (зокрема через Telegram) з метою вивчення мов. Після авторизації студент може редагувати свій профіль, переглядати список доступних курсів, проходити уроки, заробляти бали досвіду (ХР), підтримувати серію безперервного навчання (стріки), оформлювати преміум-підписку та слідкувати за прогресом інших учасників.
Адміністратор	Користувач із розширеними (максимальними) правами доступу. Має доступ до захищеної адміністративної панелі, через яку здійснює створення навчального контенту, публікує новинні пости, відслідковує активність користувачів та модерує систему загалом.

Визначивши основних акторів, можна побудувати діаграму варіантів використання (Use Case Diagram), яка наочно ілюструє їхню взаємодію із вебзастосунком. На рис. 2.1 показано загальну діаграму прецедентів системи.

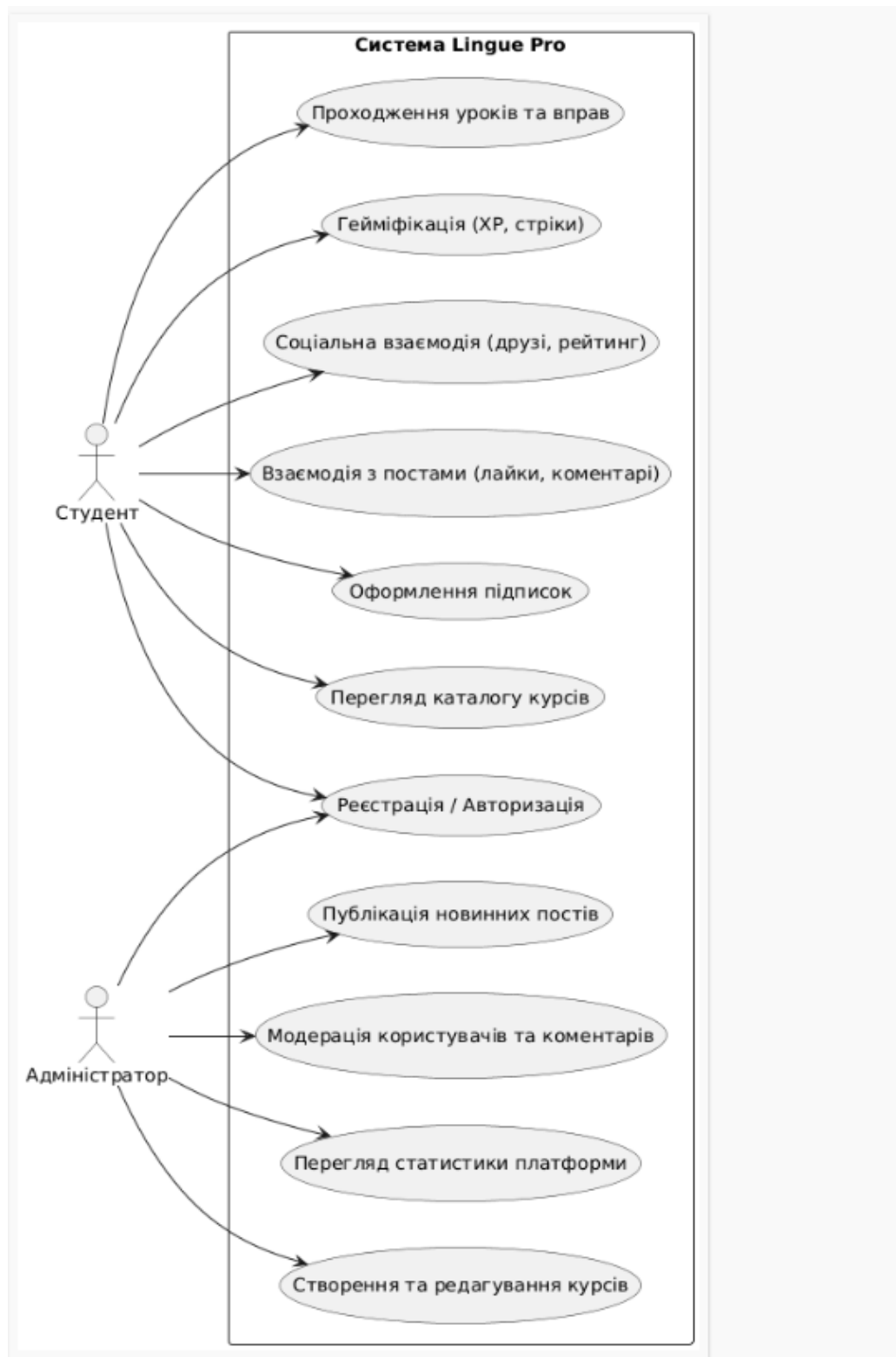


Рис. 2.1. Діаграма варіантів використання системи

Опис діаграми варіантів використання наведено в табл. 2.2 – 2.14.

У табл. 2.2 подано опис прецеденту авторизації та керування профілем, що є першим і необхідним кроком для доступу до функціоналу платформи.

Таблиця 2.2

Опис прецеденту «Авторизація користувача та профіль»

Назва варіанту використання	Реєстрація / Авторизація
Основні актори	Студент, Адміністратор
Короткий опис	Користувач створює обліковий запис або входить у систему за допомогою класичного методу (email/пароль) чи через інтеграцію з Telegram. Після входу отримує доступ до особистого профілю, статистики навчання та можливості редагувати свої дані.

Після успішної авторизації студент отримує доступ до пошуку навчальних матеріалів. У табл. 2.3 представлено опис прецеденту перегляду каталогу курсів, що включає інструменти фільтрації та вибору необхідної програми.

Таблиця 2.3

Опис прецеденту «Перегляд каталогу курсів»

Назва варіанту використання	Перегляд каталогу курсів
Основні актори	Студент
Короткий опис	Користувач переглядає сторінку з переліком доступних курсів, має можливість використовувати інструменти пошуку та фільтрації для вибору потрібного матеріалу відповідно до своїх інтересів.

Безпосередня взаємодія з навчальним контентом є основним етапом роботи студента в системі. У табл. 2.4 детально описано прецедент проходження

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		27

інтерактивних уроків та виконання практичних завдань із автоматичною перевіркою.

Таблиця 2.4

Опис прецеденту «Проходження уроків та виконання вправ»

Назва варіанту використання	Проходження уроків та виконання вправ
Основні актори	Студент
Короткий опис	Користувач розпочинає обраний курс, проходить теоретичні блоки (модулі/уроки) та виконує практичні завдання. Система автоматично перевіряє правильність відповідей і зберігає результати проходження в базу даних.

Для підвищення рівня залученості та підтримки мотивації учнів на платформі реалізовано систему заохочень. У табл. 2.5 подано опис прецеденту гейміфікації, що охоплює нарахування балів досвіду та відстеження безперервного прогресу.

Таблиця 2.5

Опис прецеденту «Гейміфікація та відстеження прогресу»

Назва варіанту використання	Гейміфікація та відстеження прогресу
Основні актори	Студент
Короткий опис	За кожен успішно виконану вправу або урок система автоматично нараховує учню бали досвіду (XP). Також фіксується регулярність відвідувань для формування "стріків" (днів безперервного навчання).

Монетизація платформи забезпечується через гнучку систему рівнів доступу. У табл. 2.6 представлено прецедент управління підписками, за допомогою якого студент може отримати розширений преміум-функціонал.

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

Таблиця 2.6

Опис прецеденту «Управління підписками (Tiers)»

Назва варіанту використання	Оформлення рівнів доступу (підписки)
Основні актори	Студент
Короткий опис	Користувач переглядає доступні рівні (Tiers) на платформі та може здійснити оформлення розширеного доступу до ексклюзивних курсів або закритого функціоналу платформи.

Окрім безпосереднього навчання, платформа пропонує можливості для комунікації студентів. У табл. 2.7 описано прецедент взаємодії з інформаційними постами, що дозволяє користувачам залишати лайки та коментарі.

Таблиця 2.7

Опис прецеденту «Взаємодія з інформаційними постами»

Назва варіанту використання	Взаємодія з постами (лайки, коментарі)
Основні актори	Студент
Короткий опис	Користувач переглядає новинні пости, опубліковані адміністратором. Він може ставити лайки, залишати власні коментарі під публікаціями та вести обговорення з іншими учасниками платформи.

Для створення активного студентського ком'юніті реалізовано механізми здорової конкуренції. У табл. 2.8 подано прецедент соціальної взаємодії, який охоплює додавання друзів та участь у загальних рейтингах.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

Таблиця 2.8

Опис прецеденту «Соціальна взаємодія та рейтинги»

Назва варіанту використання	Соціальна взаємодія (друзі, рейтинг)
Основні актори	Студент
Короткий опис	Користувач може додавати інших студентів у друзі, переглядати їхній прогрес та змагатися за місця у загальному рейтингу лідерів, що значно підвищує рівень залученості та мотивації.

Особистий кабінет дозволяє кожному учню самостійно управляти своїми даними. У табл. 2.9 подано опис прецеденту керування профілем, що включає редагування конфіденційної інформації та перегляд сертифікатів.

Таблиця 2.9

Опис прецеденту «Керування профілем»

Назва варіанту використання	Керування профілем
Основні актори	Студент
Короткий опис	У профілі користувач редагує власні дані, змінює пароль, переглядає отримані сертифікати, відстежує активність або видаляє акаунт.

Окрім функціоналу для студентів, система містить розширені можливості для управління навчальним контентом. У табл. 2.10 описано прецедент створення та редагування курсів безпосередньо адміністратором платформи.

Таблиця 2.10

Опис прецеденту «Створення та редагування курсів»

Назва варіанту використання	Створення та редагування курсів
-----------------------------	---------------------------------

Основні актори	Адміністратор
Короткий опис	Адміністратор створює новий навчальний курс або редагує існуючий: додає модулі, уроки, текстові та медіаматеріали (зображення, аудіо), а також формує тестові вправи та практичні завдання для перевірки знань учнів.

Для своєчасного інформування спільноти про оновлення передбачено спеціальний новинний блок. У табл. 2.11 представлено опис прецеденту публікації інформаційних дописів.

Таблиця 2.11

Опис прецеденту «Публікація новинних постів»

Назва варіанту використання	Публікація новинних постів
Основні актори	Адміністратор
Короткий опис	Адміністратор створює інформаційні дописи для сповіщення студентів про вихід нових курсів, оновлення функціоналу платформи або інші важливі події, додаючи текстовий опис та обкладинку.

Забезпечення безпечного та комфортного середовища на платформі вимагає надійних інструментів контролю. У табл. 2.12 подано опис прецеденту модерації користувачів та управління коментарями.

Таблиця 2.12

Опис прецеденту «Модерація користувачів та коментарів»

Назва варіанту використання	Модерація користувачів та коментарів
-----------------------------	--------------------------------------

Основні актори	Адміністратор
Короткий опис	Адміністратор забезпечує контроль користувачів: він має змогу блокувати порушників та видаляти спам-коментарі під своїми публікаціями.

Нарешті, для ефективного розвитку проекту необхідно постійно відстежувати його ключові показники. У табл. 2.13 описано прецедент перегляду загальної статистики платформи адміністратором.

Таблиця 2.13

Опис прецеденту «Перегляд статистики платформи»

Назва варіанту використання	Перегляд статистики платформи
Основні актори	Адміністратор
Короткий опис	Адміністратор за допомогою панелі керування переглядає загальну аналітику: кількість активних учнів, популярність конкретних курсів, динаміку оформлення підписок та інші ключові метрики для оцінки розвитку платформи.

Комплексний аналіз предметної області, огляд існуючих рішень на ринку та чітка постановка задачі із виявленням акторів дозволили сформулювати функціональні та нефункціональні вимоги, які необхідно врахувати під час реалізації вебплатформи. Функціональні вимоги визначають взаємодію користувачів із системою, тоді як нефункціональні охоплюють технічні та якісні характеристики платформи.

Зведені вимоги до системи подано у табл. 2.14.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

Таблиця 2.14

Вимоги до системи

Категорія	Підкатегорія	Вимога
Функціональні вимоги	Реєстрація та профіль	Вхід через email чи Telegram, налаштування профілю, перегляд сертифікатів і статистики.
	Каталог курсів	Забезпечення перегляду списку доступних мовних курсів із можливістю пошуку та фільтрації за складністю чи мовою.
	Навчальний процес	Можливість проходити модулі, уроки та виконувати практичні вправи з автоматичною перевіркою правильності відповідей.
	Гейміфікація	Автоматичне нарахування балів досвіду (XP) за виконані завдання та відстеження безперервних днів навчання (стріків).
	Оформлення підписок	Можливість оформлення користувачем преміум-підписок (Tiers) для отримання доступу до закритого контенту.
	Взаємодія з контентом	Можливість ставити лайки та залишати коментарі під інформаційними публікаціями.
	Адмініструва ння	Керування користувачами, створення курсів, публікація новин та статистика в адмін-панелі.
	Зручність використанн я	Інтуїтивно зрозумілий інтерфейс, адаптований для браузера та Telegram Mini App.

Нефункціональні вимоги	Продуктивність	Підтримка швидкої відповіді сервера при одночасній роботі 100-1000 учнів завдяки асинхронній обробці запитів.
	Надійність	Стабільна робота бекенду з мінімальними затримками завдяки використанню системи кешування Redis.
	Безпека	Захищений доступ через механізм JWT-токенів, надійне хешування паролів та захист від спам-запитів (Rate Limiting).

Аналіз функціональних вимог допоміг сформувати об'єктно-орієнтовану структуру (архітектуру роутерів та сервісів) системи для вивчення іноземних мов. Нижче наведено діаграму класів системи, яка демонструє логічні зв'язки між основними програмними компонентами платформи (рис. 2.2).

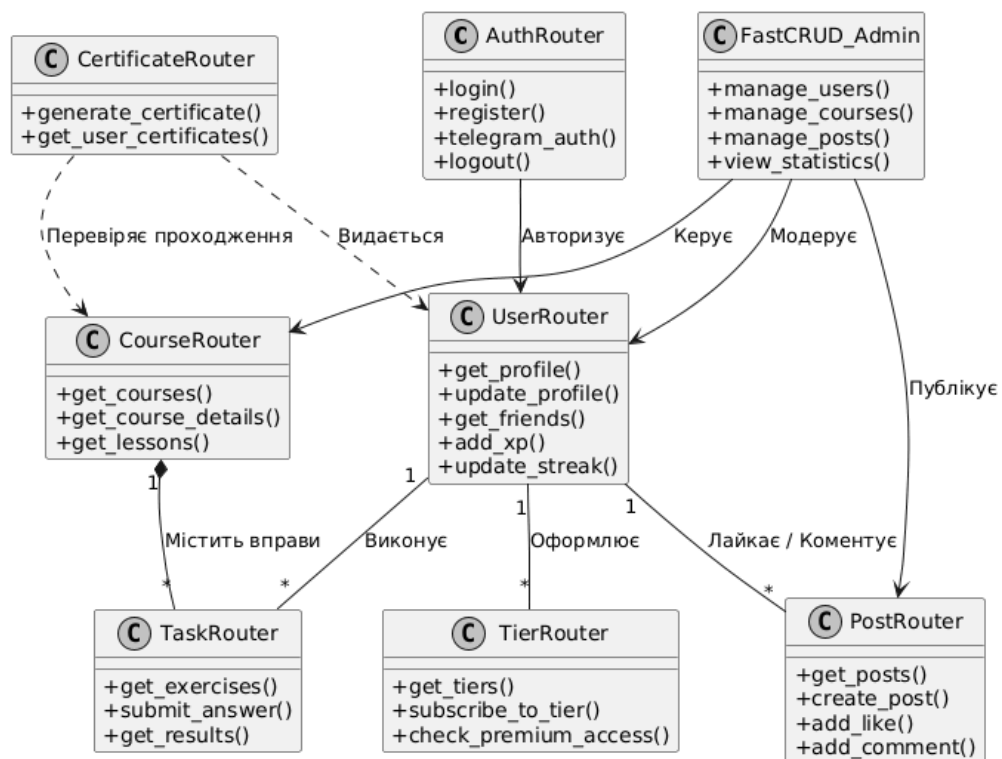


Рис. 2.2. Діаграма класів системи

Основним компонентом для роботи з користувачами є AuthRouter та UserRouter, які забезпечують створення облікових записів, вхід у систему (зокрема через інтеграцію з Telegram), управління особистим профілем, а також відстеження соціальних зв'язків (списку друзів) користувача.

Бізнес-логіка навчального процесу зосереджена у CourseRouter та супутніх сервісах. Цей компонент відповідає за перегляд каталогу курсів, отримання структурованих модулів і уроків. Для взаємодії з практичними вправами, перевірки правильності відповідей та фіксації результатів використовується TaskRouter.

Нарахування балів досвіду (XP) та контроль за регулярністю відвідувань реалізовано через модуль гейміфікації (на рівні UserService). Він оновлює статистику учня після кожного успішно пройденого уроку, забезпечуючи високий рівень мотивації.

Для управління правами доступу до закритого контенту застосовується TierRouter (система рівнів підписок). Цей компонент координує процеси переходу користувача на преміум-рівні (Tiers) і відкриває відповідні привілеї безпосередньо в базі даних.

Інформування студентів про оновлення та вихід нових матеріалів реалізовано через PostRouter. Він забезпечує створення новинних публікацій адміністраторами, а також обробляє взаємодію учнів із цим контентом, зокрема збереження лайків та коментарів.

Окремою важливою ланкою є адміністративна панель, реалізована за допомогою розширення FastCRUD. Вона виступає універсальним інструментом для адміністраторів, що інкапсулює базові операції створення, читання, оновлення та видалення для всіх сутностей бази даних, дозволяючи швидко модерувати платформу без додаткового написання коду.

Такий розподіл відповідальності між компонентами відповідає принципам чистої архітектури, забезпечує високу підтримуваність програмного забезпечення та полегшує його масштабування.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

2.2 Розробка бази даних

При створенні програмного продукту розробка бази даних є одним із ключових етапів, оскільки саме вона визначає структуру зберігання, обробки та взаємозв'язків інформації в системі. Для вебплатформи вивчення іноземних мов було спроектовано оптимізовану реляційну базу даних, яка забезпечує надійне зберігання інформації про користувачів, навчальні курси, прогрес виконання завдань (XP та стріки), підписки (Tiers) та новинні публікації.

У якості системи керування базами даних (СКБД) використовується PostgreSQL [12] — потужна реляційна система з відкритим вихідним кодом. Її вибір зумовлений високою продуктивністю, безпекою та підтримкою складних реляційних зв'язків, що виникають між учнями, уроками та практичними вправами. Завдяки підтримці асинхронних запитів (через драйвер `asynpg`) та строгій транзакційності, PostgreSQL ідеально підходить для сучасних високонавантажених вебзастосунків. Взаємодія з базою даних на рівні програмного коду здійснюється за допомогою технології ORM (Object-Relational Mapping) через бібліотеку SQLAlchemy, що дозволяє безпечно та ефективно працювати з таблицями як з об'єктами.

З огляду на значну кількість сутностей та складності їхніх взаємозв'язків, структуру бази даних було логічно розділено на три основні підсистеми для зручності проєктування та подальшої підтримки. Першою з них є підсистема керування користувачами, соціальною взаємодією та гейміфікацією, яка відповідає за зберігання профілів, балансу досвіду (XP), стріків та списків друзів. Другою виступає підсистема навчального контенту та оцінювання, що включає таблиці для структуризації курсів, модулів, уроків, практичних вправ і результатів їх виконання. Третьою є підсистема підписок та інформаційного контенту, яка забезпечує управління преміум-рівнями доступу (Tiers), новинними публікаціями, а також лайками та коментарями користувачів.

На рис. 2.3 представлено структуру бази даних першої підсистеми — керування користувачами, соціальною взаємодією та гейміфікацією. Вона

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

включає ключову сутність користувача (User), таблицю соціальних зв'язків для реалізації списку друзів (UserFriendship), а також таблицю для збереження накопичених ігрових балів (UserXPBalance). Дана підсистема забезпечує надійне зберігання даних про досягнення учнів та їхню взаємодію один з одним, що є основою мотиваційної складової платформи.

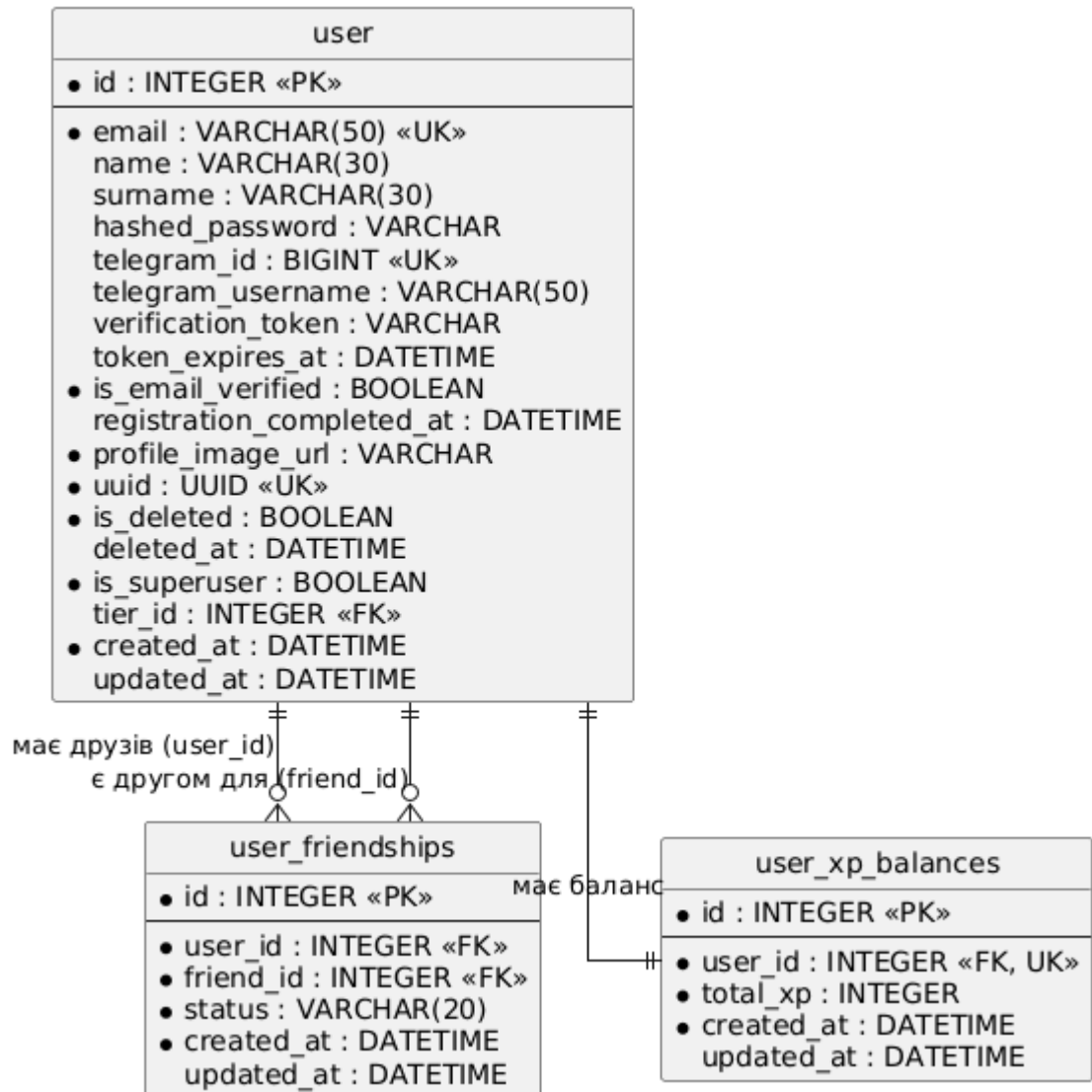


Рис. 2.3. Структура БД підсистеми керування користувачами та соціальною взаємодією

На рис. 2.4 зображено структуру бази даних підсистеми навчального контенту та оцінювання. Вона охоплює сутності мовних курсів (courses), навчальних модулів (course_units), уроків (lessons), практичних вправ (exercises), а також таблиці для збереження реєстрацій студентів на курс

(course_enrollments), прогресу проходження уроків (lesson_progress) та результатів виконання завдань (exercise_attempts). Дана підсистема забезпечує гнучке формування навчальних матеріалів, а також веде чіткий і деталізований облік успішності кожного користувача.

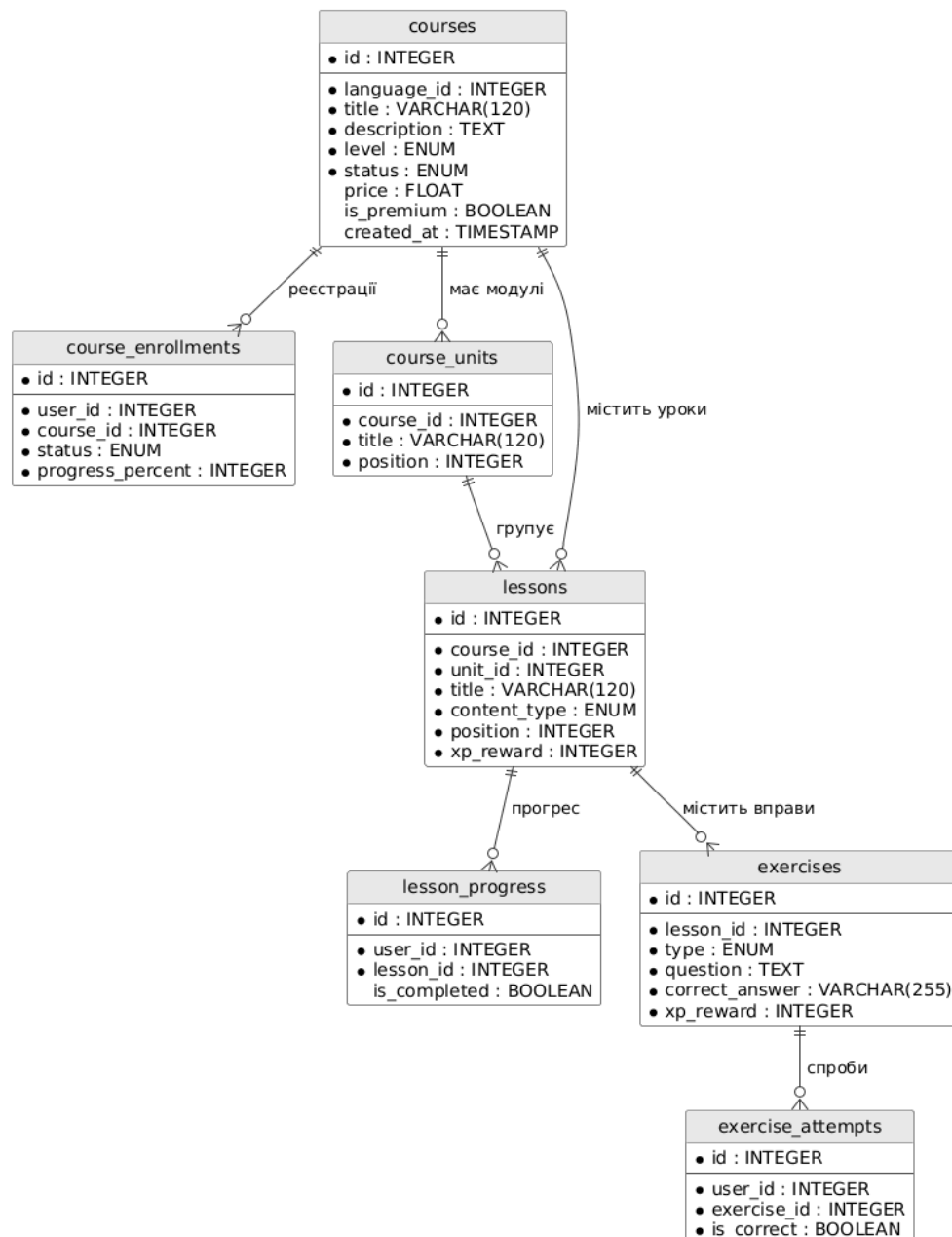


Рис. 2.4. Структура БД підсистеми навчального контенту та оцінювання

На рис. 2.5 представлено структуру бази даних підсистеми підписок та інформаційного контенту. Вона включає сутності преміум-рівнів (tier), оформлених студентами підписок із фіксацією розрахункових даних (user_tier_subscriptions), а також таблиці для новинних публікацій (post), коментарів (post_comments) та лайків (post_likes). Дана підсистема забезпечує управління доступом користувачів до закритого контенту платформи, а також надає інструменти для інформаційної взаємодії адміністрації зі студентами.

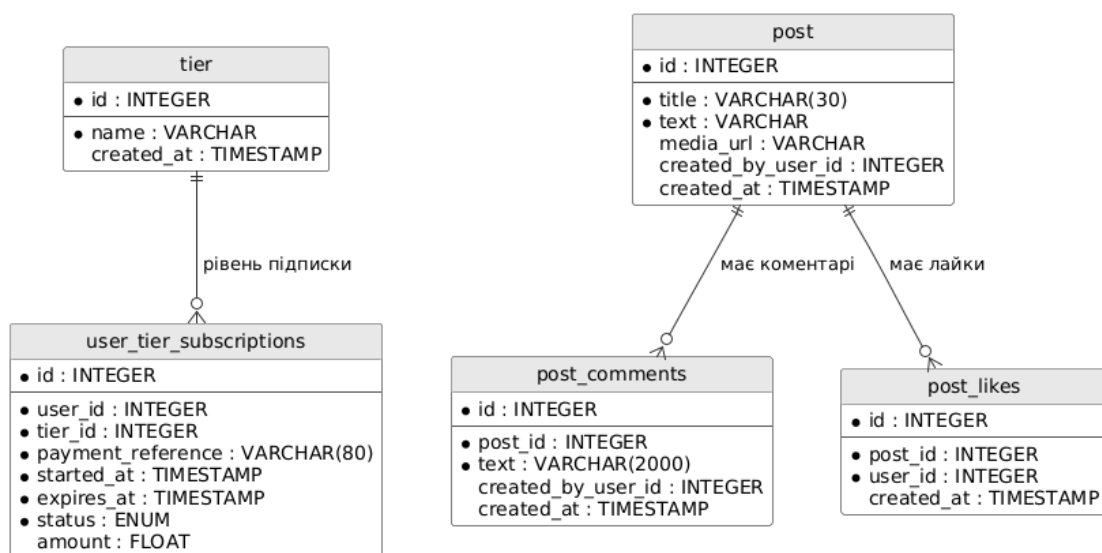


Рис. 2.5. Структура БД підсистеми підписок та інформаційного контенту

База даних вебплатформи містить 15 ключових таблиць.

Структура та опис полів кожної таблиці подані в табл. 2.15 – 2.29.

Зокрема, у табл. 2.15 наведено структуру головної таблиці «User», яка зберігає ключову інформацію про облікові записи студентів та адміністраторів.

Таблиця 2.15

Структура таблиці «User»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор користувача
email	VARCHAR	Електронна пошта (використовується як логін)

Продовження табл. 2.15

name	VARCHAR	Ім'я користувача
surname	VARCHAR	Прізвище користувача
hashed_password	VARCHAR	Захешований пароль для безпечного зберігання
telegram_id	BIGINT	Унікальний ID користувача в месенджері Telegram
telegram_username	VARCHAR	Нікнейм користувача з Telegram
verification_token	VARCHAR	Токен для підтвердження електронної пошти
token_expires_at	TIMESTAMP	Час закінчення дії токена верифікації
is_email_verified	BOOLEAN	Статус підтвердження електронної пошти
registration_completed_at	TIMESTAMP	Дата та час повного завершення реєстрації
profile_image_url	VARCHAR	Посилання на аватарку профілю
uuid	UUID	Унікальний ідентифікатор для зовнішніх інтеграцій (v7)
is_superuser	BOOLEAN	Права глобального адміністратора платформи
tier_id	INTEGER (FK)	Посилання на активний рівень підписки (преміум)
created_at	TIMESTAMP	Дата та час створення запису в базі

У табл. 2.16 представлено структуру таблиці «UserFriendship», яка відповідає за організацію соціальних зв'язків між користувачами. Вона містить

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

ідентифікатори користувачів, що надсилають та отримують заявки на дружбу, дозволяючи формувати список друзів у системі.

Таблиця 2.16

Структура таблиці «UserFriendship»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор запису
user_id	INTEGER (FK)	Посилання на користувача, який надіслав заявку
friend_id	INTEGER (FK)	Посилання на користувача, якому надіслано заявку
status	VARCHAR	Статус дружби (наприклад: "accepted")
created_at	TIMESTAMP	Дата та час створення соціального зв'язку
updated_at	TIMESTAMP	Дата та час останньої зміни статусу

Підсистема гейміфікації вимагає надійного збереження досягнень кожного студента для подальшого формування рейтингів. У табл. 2.17 представлено структуру таблиці «UserXpBalance», яка використовується для обліку загальної кількості накопичених балів досвіду (XP) та пов'язана безпосередньо з профілем користувача.

Таблиця 2.17

Структура таблиці «UserXpBalance»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор запису
user_id	INTEGER (FK)	Посилання на користувача, якому належить баланс
total_xp	INTEGER	Загальна кількість накопичених балів досвіду (XP)

Поле	Тип	Призначення
created_at	TIMESTAMP	Дата та час створення запису
updated_at	TIMESTAMP	Дата та час останнього нарахування балів

Центральним об'єктом навчальної підсистеми є безпосередньо курс, який містить організаційні метадані. У табл. 2.18 наведено структуру таблиці «Course», що агрегує детальну інформацію про мову вивчення, рівень складності, статус публікації та тип доступу до матеріалу (преміум або безкоштовний).

Таблиця 2.18

Структура таблиці «Course»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор курсу
language_id	INTEGER (FK)	Посилання на мову вивчення
title	VARCHAR	Назва мовного курсу
description	Text	Детальний опис курсу
level	ENUM	Рівень складності (наприклад, A1, B2)
study_goal	ENUM	Мета вивчення мови (робота, подорожі тощо)
image_url	VARCHAR	Посилання на обкладинку курсу
certificate_url	VARCHAR	Посилання на шаблон сертифіката
status	ENUM	Статус публікації (DRAFT, PUBLISHED)
estimated_hours	INTEGER	Орієнтовна тривалість курсу в годинах
difficulty_score	INTEGER	Числовий індекс складності
price	FLOAT	Вартість доступу до курсу

is_premium	BOOLEAN	Прапорець, що означає преміум-курс
------------	---------	------------------------------------

Для зручності сприйняття навчальний матеріал курсу логічно розбивається на окремі модулі або розділи. У табл. 2.19 подано структуру таблиці «CourseUnit», яка визначає ієрархію модулів та їхній порядковий номер у межах конкретної програми.

Таблиця 2.19

Структура таблиці «CourseUnit»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор модуля
course_id	INTEGER (FK)	Посилання на курс, якому належить модуль
title	VARCHAR	Назва модуля (розділу)
description	TEXT	Опис навчального модуля
position	INTEGER	Порядковий номер у курсі
created_at	TIMESTAMP	Дата створення
updated_at	TIMESTAMP	Дата оновлення

Кожен модуль поділяється на дрібніші інформаційні блоки — уроки. У табл. 2.20 представлено структуру таблиці «Lesson», яка містить вичерпний теоретичний матеріал, правила граматики, посилання на мультимедійні ресурси та розмір винагороди (XP) за успішне завершення уроку.

Таблиця 2.20

Структура таблиці «Lesson»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор уроку
course_id	INTEGER (FK)	Посилання на курс

Продовження табл. 2.20

title	VARCHAR	Назва уроку
objective	TEXT	Навчальна ціль уроку
unit_id	INTEGER (FK)	Посилання на модуль (якщо урок згруповано)
summary	VARCHAR	Короткий підсумок уроку
theory_text	TEXT	Теоретичний матеріал
rules_text	TEXT	Граматичні правила
examples_text	TEXT	Приклади вживання
content_type	ENUM	Тип контенту уроку (MIXED, VIDEO, AUDIO)
image_url	VARCHAR	Посилання на зображення
video_url	VARCHAR	Посилання на відео
audio_url	VARCHAR	Посилання на аудіоматеріал
transcript	TEXT	Текстова розшифровка аудіо/відео
materials_json	TEXT	Додаткові матеріали у форматі JSON
position	INTEGER	Порядковий номер уроку
xp_reward	INTEGER	Кількість балів XP за проходження
created_at	TIMESTAMP	Дата створення

Для перевірки знань та закріплення матеріалу до кожного уроку можуть додаватися практичні завдання. У табл. 2.21 описано структуру таблиці «Exercise», що зберігає умови завдання, можливі варіанти відповідей у форматі JSON та правильну відповідь із поясненнями.

Таблиця 2.21

Структура таблиці «Exercise»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор вправи
lesson_id	INTEGER (FK)	Посилання на урок

Продовження табл. 2.21

title	VARCHAR	Назва вправи
prompt	TEXT	Завдання для учня
exercise_type	ENUM	Тип завдання (тест, введення тексту)
content_type	ENUM	Формат завдання (TEXT, AUDIO тощо)
audio_url	VARCHAR	Аудіозапис до вправи
video_url	VARCHAR	Відеофрагмент до вправи
image_url	VARCHAR	Зображення до вправи
options_json	TEXT	Варіанти відповідей у форматі JSON
correct_answer	TEXT	Правильна відповідь
explanation	TEXT	Пояснення до правильної відповіді
time_limit_seconds	INTEGER	Обмеження часу на виконання
points	INTEGER	Бали за правильне виконання
position	INTEGER	Порядковий номер у межах уроку
created_at	TIMESTAMP	Дата створення

Щоб система могла фіксувати, до яких курсів долучився студент, використовується механізм зарахувань. У табл. 2.22 наведено структуру таблиці «CourseEnrollment», яка зберігає статус реєстрації учня на курс та динамічно оновлює відсоток його проходження.

Таблиця 2.22

Структура таблиці «CourseEnrollment»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор реєстрації
user_id	INTEGER (FK)	Посилання на студента
course_id	INTEGER (FK)	Посилання на обраний курс
status	ENUM	Статус реєстрації (ENROLLED, COMPLETED)

progress_percent	INTEGER	Відсоток проходження курсу
created_at	TIMESTAMP	Дата реєстрації на курс
updated_at	TIMESTAMP	Дата останньої зміни прогресу

Деталізація навчального процесу вимагає обліку результатів на рівні окремих занять. У табл. 2.23 подано структуру таблиці «LessonProgress», яка відстежує статус проходження конкретного уроку студентом та кількість зароблених ним балів.

Таблиця 2.23

Структура таблиці «LessonProgress»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор запису
user_id	INTEGER (FK)	Посилання на користувача
lesson_id	INTEGER (FK)	Посилання на урок
status	ENUM	Статус (NOT_STARTED, COMPLETED)
score	INTEGER	Кількість зароблених балів на уроці
started_at	TIMESTAMP	Час початку проходження уроку
completed_at	TIMESTAMP	Час успішного завершення уроку
created_at	TIMESTAMP	Дата створення запису
updated_at	TIMESTAMP	Дата оновлення прогресу

Для збору аналітики щодо успішності виконання практичних завдань фіксується кожна дія учня. У табл. 2.24 представлено структуру таблиці «ExerciseAttempt», що зберігає надані студентом відповіді, витрачений на виконання час та результати автоматичної перевірки.

Таблиця 2.24

Структура таблиці «ExerciseAttempt»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор спроби
user_id	INTEGER (FK)	Посилання на учня
exercise_id	INTEGER (FK)	Посилання на виконану вправу
answer	TEXT	Відповідь, яку надав користувач
is_correct	BOOLEAN	Результат перевірки (вірно/невірно)
score	INTEGER	Отримані бали за спробу
time_spent_seconds	INTEGER	Витрачений час у секундах
created_at	TIMESTAMP	Дата та час виконання вправи

Керування доступом до ексклюзивного контенту базується на гнучкій системі тарифних планів. У табл. 2.25 наведено структуру таблиці «Tier», яка містить базову інформацію про доступні рівні підписок (наприклад, безкоштовний або преміум-доступ).

Таблиця 2.25

Структура таблиці «Tier»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор рівня підписки
name	VARCHAR	Назва преміум-рівня (наприклад, "Free", "Pro")
created_at	TIMESTAMP	Дата створення рівня
updated_at	TIMESTAMP	Дата останнього редагування

Фіксація розширених прав доступу та облік фінансових операцій відбувається за допомогою проміжної таблиці. У табл. 2.26 подано структуру

таблиці «UserTierSubscription», яка відстежує активні підписки користувачів, терміни їхньої дії та деталі імітованих платіжних транзакцій.

Таблиця 2.26

Структура таблиці «UserTierSubscription»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор оформленої підписки
user_id	INTEGER (FK)	Посилання на користувача
tier_id	INTEGER (FK)	Посилання на обраний рівень підписки
payment_reference	VARCHAR	Унікальний ідентифікатор платіжної транзакції
started_at	TIMESTAMP	Час активації преміум-доступу
expires_at	TIMESTAMP	Час закінчення дії підписки
status	ENUM	Статус підписки (ACTIVE, EXPIRED, CANCELLED)
payment_provider	VARCHAR	Назва платіжного провайдера
amount	FLOAT	Сума сплачених коштів
currency	VARCHAR	Валюта платежу (наприклад, USD)
created_at	TIMESTAMP	Дата створення запису
updated_at	TIMESTAMP	Дата оновлення статусу

Для інформування студентів про оновлення платформи реалізовано підсистему новин. У табл. 2.27 представлено структуру таблиці «Post», яка зберігає текстовий контент публікацій, посилання на медіаматеріали та інформацію про адміністратора-автора.

Таблиця 2.27

Структура таблиці «Post»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор поста
title	VARCHAR	Заголовок публікації
text	VARCHAR	Основний текст новини або статті
created_by_user_id	INTEGER (FK)	Посилання на адміністратора, що створив пост
uuid	UUID	Унікальний публічний ідентифікатор для URL
media_url	VARCHAR	Посилання на прикріплене зображення/відео
is_deleted	BOOLEAN	Прапорець архівації поста (м'яке видалення)
deleted_at	TIMESTAMP	Дата перенесення в архів
created_at	TIMESTAMP	Дата публікації
updated_at	TIMESTAMP	Дата редагування

Зворотний зв'язок від аудиторії та комунікація між користувачами реалізується через систему обговорень під новинами. У табл. 2.28 описано структуру таблиці «PostComment», яка фіксує текстові коментарі, прив'язуючи їх до відповідної публікації та профілю автора.

Таблиця 2.28

Структура таблиці «PostComment»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор коментаря
post_id	INTEGER (FK)	Посилання на публікацію

text	VARCHAR	Текст залишеного коментаря
created_by_user_id	INTEGER (FK)	Посилання на користувача (автора коментаря)
created_at	TIMESTAMP	Дата та час створення

Окрім коментарів, студенти мають змогу швидко висловлювати свою зацікавленість новинами за допомогою вподобань. У табл. 2.29 подано структуру таблиці «PostLike», що зберігає інформацію про реакції користувачів на опубліковані пости.

Таблиця 2.29

Структура таблиці «PostLike»

Поле	Тип	Призначення
id	INTEGER (PK)	Унікальний ідентифікатор лайка
post_id	INTEGER (FK)	Посилання на публікацію
user_id	INTEGER (FK)	Посилання на користувача, який вподобав пост
created_at	TIMESTAMP	Дата та час натискання лайка

2.3 Проєктування та реалізація алгоритмів роботи системи

Створена вебплатформа має інтуїтивно зрозумілий інтерфейс і забезпечує зручну навігацію, що дозволяє учням ефективно взаємодіяти з основними функціональними модулями системи. Процес взаємодії користувача з платформою починається з головної сторінки, яка виконує роль центральної точки входу до системи та містить каталог мовних курсів, механізми пошуку та фільтрації, а також блоки з популярними напрямками навчання.

Після вибору курсу користувач переходить на сторінку його детального перегляду, де отримує повну інформацію, включаючи опис, кількість модулів, список уроків та орієнтовний час проходження. Даний рівень деталізації

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

необхідний для прийняття учнем рішення щодо початку навчання та допомагає оцінити обсяг матеріалу.

Взаємодія користувача з платформою також реалізована через соціальні механізми, зокрема додавання вподобань та коментарів під новинними публікаціями (Додаток А). Такі дії доступні лише авторизованим користувачам, що дозволяє контролювати якість контенту та зменшити кількість спам-взаємодій.

Процес авторизації передбачає класичну реєстрацію або швидкий вхід за допомогою інтеграції з Telegram. Після успішної автентифікації користувач отримує доступ до особистого профілю, де відображається його статистика навчання, баланс ігрових балів (ХР), дні безперервного навчання (стріки), списки друзів та історія виконаних вправ.

Початок навчання на обраному курсі ініціює створення запису про зарахування (CourseEnrollment) у базі даних. У випадку вибору преміум-курсу система автоматично перевіряє наявність активної підписки (Tier) у профілі користувача. Якщо підписка відсутня, система блокує доступ і пропонує перейти на преміум-рівень (Додаток Б). Після отримання доступу студент може покроково проходити уроки, виконуючи інтерактивні практичні вправи, відповіді на які миттєво перевіряються серверною частиною.

Для адміністраторів передбачено зручний функціонал (реалізований за допомогою бібліотеки FastCRUD), що дозволяє керувати контентом через автоматично згенеровану панель управління. Це дає змогу швидко створювати нові мовні курси, структурувати їх за модулями, додавати тестові завдання та модерувати користувачів (Додаток В), зберігаючи при цьому повний контроль над усім освітнім процесом платформи.

Окремо реалізовано алгоритми гейміфікації навчання. Система нараховує бали досвіду (ХР) за кожну правильно виконану вправу і відстежує щоденну активність учня. Це дозволяє утримувати високий рівень мотивації студентів і заохочувати їх до регулярного проходження уроків (Додаток Г). Завершення всіх

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		51

модулів курсу ініціює автоматичну генерацію іменного сертифіката, який офіційно підтверджує отримані знання.

Для детального опису процесів взаємодії користувача з системою та відображення основної бізнес-логіки платформи було побудовано діаграму активності (рис. 2.6). Згідно з діаграмою, алгоритм розпочинається з відкриття користувачем головної сторінки та перегляду каталогу курсів. Після вибору дисципліни спрацьовує перший логічний блок — перевірка статусу авторизації. Якщо користувач не є авторизованим у системі, алгоритм перенаправляє його на форму автентифікації email.

У разі успішної авторизації ініціюється вибір курсу для вивчення. На цьому кроці алгоритм виконує комплексну перевірку доступу до навчального контенту. Спочатку система визначає статус курсу — чи є він безкоштовним, чи належить до категорії преміум. Якщо обраний курс є преміальним, виконується додаткова перевірка наявності активної підписки (Tier). У випадку її відсутності потік виконання переривається на обов'язковий крок оформлення преміум-підписки. Лише після цього система дозволяє перейти до наступного етапу — безпосередньої реєстрації на курс (Enrollment).

Наступний блок діаграми описує циклічний алгоритм проходження навчального матеріалу, який є ядром взаємодії клієнта та сервера. Він включає послідовне відкриття уроку та ознайомлення з теорією. Після теорії учень виконує практичні вправи, які підлягають автоматичній перевірці серверною частиною. Ця перевірка здійснюється за допомогою асинхронних запитів, що дозволяє миттєво повертати результат та підтримувати високий темп навчання без відчутних затримок.

У випадку виявлення помилок під час автоматичної перевірки відповідей, сервер миттєво формує JSON-відповідь із зазначенням некоректних полів та підказок для виправлення. Користувач залишається на етапі виконання практичних вправ доти, доки не надасть правильний варіант. Кожна спроба транзакційно фіксується у базі даних через ORM-модель для подальшого аналізу успішності студента. Такий підхід гарантує цілісність освітнього процесу,

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

унеможливлнює пропуск складних тем та надає адміністраторам вичерпну статистику щодо найважчих завдань курсу.

За умови правильних відповідей алгоритм ініціює блок гейміфікації: учневі нараховуються бали досвіду (XP) та оновлюється індикатор щоденної активності (навчальний стрік). Після цього система перевіряє умову завершення курсу: якщо пройдено не всі модулі, цикл повертається до початку, пропонуючи відкриття наступного уроку. Якщо ж алгоритм фіксує повне проходження матеріалу, цикл розривається, і система автоматично генерує електронний сертифікат. Даний документ підтверджує завершення навчання, після чого алгоритм переходить до свого фінального стану.



Рис. 2.6. Діаграма активності процесу навчання

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

2.4 Реалізація функціоналу вебплатформи

Реалізація вебплатформи побудована на клієнт–серверній архітектурі з чітким розділенням відповідальності між рівнем представлення, бізнес-логікою та рівнем доступу до даних. Система реалізована у вигляді модульного моноліту: серверна частина представлена потужним асинхронним API-застосунком, структурованим за предметними модулями (доменами), тоді як клієнтська частина є самостійним односторінковим застосунком (SPA). Такий підхід забезпечує логічну ізоляцію даних, високу швидкість обробки запитів та спрощує підтримку транзакційної цілісності бази даних.

Серверна частина реалізована з використанням мови Python та сучасного асинхронного фреймворку FastAPI [14], що дозволяє будувати масштабоване REST API. Структуру бекенду організовано у вигляді окремих маршрутизаторів (Routers), таких як auth, courses, users, posts та tiers. Доступ до даних здійснюється через технологію ORM за допомогою бібліотеки SQLAlchemy та асинхронного драйвера asyncpg. Архітектура не виділяє окремого шару репозиторіїв, оскільки SQLAlchemy повністю інкапсулює операції з даними, а бізнес-логіка зосереджена у відповідних сервісах.

Основна бізнес-логіка розбита на доменні модулі. CourseService відповідає за формування каталогу мовних курсів, фільтрацію, структурування модулів та уроків. TaskService забезпечує перевірку правильності виконання практичних вправ та нарахування балів. Модуль гейміфікації (інтегрований у UserService) веде облік досвіду (XP) та безперервних днів навчання (стріків). TierService координує управління преміум-підписками та розмежування прав доступу, а PostService забезпечує роботу з інформаційними новинами, лайками та коментарями.

Клієнтська частина реалізована як SPA на базі бібліотеки React. Взаємодія з сервером здійснюється через REST API з використанням JWT-аутентифікації. Стан авторизації глобально керується через сучасний стейт-менеджер Zustand (authStore), а доступ до захищених маршрутів реалізовано через компонент-

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

обгортку RequireAuth. Для роботи з API використовується окремий клієнтський інстанс (Axios), який уніфікує запити та автоматично додає токени доступу (Access Token). Обмін усіма даними, включаючи URL-посилання на медіафайли, виконується виключно у форматі JSON. Між клієнтом і сервером додатково налаштовано CORS-політику для безпечного міждоменового зв'язку.

На рівні бази даних розроблено детальну реляційну схему, що включає таблиці для збереження профілів (Users), навчальних матеріалів (Courses, Modules, Lessons) та результатів виконання тестів. Завдяки декларативному підходу SQLAlchemy, усі зв'язки між таблицями (відношення один-до-багатьох та багато-до-багатьох) налаштовані за допомогою зовнішніх ключів (Foreign Keys) та каскадного видалення. Це виключає появу осиротілих записів у разі видалення курсу чи профілю. Додатково, для забезпечення консистентності даних при одночасних запитах, застосовуються механізми транзакційних блокувань. Усі міграції схеми бази даних виконуються в автоматизованому режимі за допомогою утиліти Alembic, що дозволяє безпечно оновлювати структуру таблиць без втрати вже існуючої інформації.

Для забезпечення стабільної роботи серверної частини впроваджено централізовану систему обробки винятків (Exception Handling). Усі непередбачувані помилки перехоплюються на рівні глобальних middleware, після чого бекенд формує уніфіковану JSON-відповідь із відповідним HTTP-кодом статусу. Це гарантує, що клієнтська частина завжди отримує зрозуміле повідомлення про помилку, незалежно від того, чи це збій валідації Pydantic, чи відсутність запису в базі даних. Також налаштовано систему логування, яка фіксує всі критичні події, спроби несанкціонованого доступу та час виконання тривалих запитів, що є критично важливим для подальшого моніторингу.

На клієнтському рівні значна увага приділяється адаптивності та оптимізації інтерфейсу. Для стилізації використовується фреймворк Tailwind CSS, який завдяки системі утиліт дозволяє створювати складні візуальні компоненти без розростання файлів стилів. Користувацький інтерфейс автоматично підлаштовується під роздільну здатність екрана, забезпечуючи

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

однаковий рівень комфорту як на десктопних моніторах, так і на мобільних пристроях. Окремим архітектурним рішенням стала розробка ізольованих UI-компонентів (кнопки, вікна, картки), що підвищує рівень повторного використання коду.

Каталог курсів реалізований через API-ендпоінти, які підтримують фільтрацію за мовою вивчення, рівнем складності та метою. На стороні сервера виконується агрегація даних та повернення результату у форматі JSON. Після цього фронтенд динамічно відображає список доступних курсів у вигляді карток із зазначенням орієнтовного часу проходження та рівня складності.

Сторінка детального перегляду курсу надає учню повну інформацію про обрану програму, включаючи опис, список модулів, уроків та відсоток власного прогресу. Процес навчання ініціюється автоматичним створенням запису CourseEnrollment. У випадку доступу до закритого контенту система перевіряє наявність активної підписки (Tier).

Проходження уроку реалізовано як послідовність кроків: ознайомлення з теорією та виконання інтерактивних завдань (текстових, аудіо- чи відеоформатів). Відповіді користувача відправляються на бекенд, де миттєво валідуються. У разі правильної відповіді оновлюється статистика LessonProgress та нараховується нагорода (XP).

Процес управління контентом реалізовано через адміністративну панель за допомогою розширення FastCRUD. Це дозволяє адміністраторам зручно створювати курси, вставляти медіафайли (обкладинки, аудіо), які автоматично зберігаються в хмарному сховищі Cloudflare R2, та модерувати облікові записи без необхідності писати додатковий код для базових операцій.

Система авторизації базується на двох підходах: класичній реєстрації через email із видачею пари токенів (Access та Refresh) та швидкому вході через Telegram Web App. Для захисту API від зловживань, DDoS-атак та брутфорс-перебору паролів застосовано механізм Rate Limiting на базі in-memory сховища Redis. Розмежування прав доступу реалізовано за допомогою вбудованих інструментів FastAPI (Depends) та JWT-декодування.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		56

Особистий кабінет користувача об'єднує перегляд прогресу навчання, історії підписок, списку друзів (соціальний модуль) та налаштувань профілю. Додатково користувач може переглядати отримані цифрові сертифікати за успішно пройдени курси.

У результаті реалізації було створено високопродуктивну та сучасну вебплатформу, яка забезпечує користувачам можливість зручного пошуку курсів, інтерактивного проходження уроків із системою мотивації (гейміфікації), а адміністраторам надає потужні інструменти для створення навчального контенту та управління платформою. Дане технічне рішення повністю відповідає сучасним стандартам якості розробки програмного забезпечення.

Висновки до розділу 2

Під час проєктування вебплатформи для вивчення іноземних мов проведено аналіз основних сценаріїв використання системи з урахуванням ролей користувачів, зокрема студентів та адміністраторів, а також визначено ключові функціональні та нефункціональні вимоги до програмного забезпечення. На основі цього було спроектовано сучасну архітектуру платформи з чітким розділенням серверної (на базі FastAPI) та клієнтської (на базі React) частин і визначено принципи їх взаємодії через REST API.

Було розроблено логіку роботи основних бізнес-процесів системи, включаючи перегляд та фільтрацію каталогу курсів, детальний перегляд навчальних модулів і уроків, процес реєстрації на курс, оформлення преміум-підписок (Tiers), а також виконання практичних вправ із автоматичною перевіркою та нарахуванням ігрового досвіду (XP). Це дозволило формалізувати ключові етапи роботи користувача з платформою та забезпечити їх коректну реалізацію.

Спроектовано оптимізовану структуру реляційної бази даних на основі PostgreSQL, яка налічує 15 ключових таблиць та забезпечує надійне зберігання інформації про профілі користувачів, навчальний контент (курси, уроки, вправи),

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

прогрес навчання, підписки, а також соціальну взаємодію (друзі, пости, лайки та коментарі). Особливу увагу приділено зв'язкам між сутностями для коректного збереження статистики навчання кожного студента.

Була інтегрована багатофункціональна адміністративна панель на базі бібліотеки FastCRUD, що забезпечує інтуїтивно зрозумілий інтерфейс для управління базою даних без написання додаткового коду. Вона використовується адміністраторами для модерації облікових записів, створення навчальних курсів, завантаження контенту (зокрема у хмарне сховище Cloudflare R2) та контролю загального стану системи.

У процесі реалізації продемонстровано основний функціонал платформи, що охоплює безпечну реєстрацію та авторизацію користувачів (зокрема через Telegram), проходження уроків із миттєвою перевіркою результатів, а також систему гейміфікації (стріки та ХР) для підвищення залученості студентів.

Таким чином, у другому розділі спроектовано та реалізовано повноцінний програмний прототип серверної та клієнтської частин вебплатформи для вивчення іноземних мов, що забезпечує безперервний життєвий цикл навчання — від реєстрації користувача на платформі до успішного проходження курсу та генерації електронного сертифіката.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБПЛАТФОРМОЮ

3.1 Порядок встановлення та налаштування системи

Для забезпечення безперебійної роботи вебплатформи використовується сучасна хмарна клієнт–серверна архітектура, яка включає клієнтську частину, бекенд-сервер, бази даних та зовнішні сервіси. Такий підхід забезпечує централізоване зберігання даних, віддалений доступ до платформи з будь-якого пристрою та суттєво спрощує масштабування програмного продукту під час зростання навантаження. Розподіл функціональності між мікрокомпонентами підвищує загальну надійність платформи та полегшує її подальший супровід.

Клієнтська частина реалізована як односторінковий вебзастосунок (SPA) на базі бібліотеки React. Збірка оптимізованого фронтенду виконується командою `npm run build`, після чого статичні файли розміщуються на серверах провайдера. Для підключення клієнта до бекенду використовується змінна оточення `VITE_API_URL`, що визначає базову адресу API. Користувач взаємодіє із системою через класичний браузер або інтерфейс Telegram Mini App. Обмін даними здійснюється через REST API із обов'язковим використанням протоколу HTTPS.

Серверна частина реалізована на мові Python із використанням високопродуктивного асинхронного фреймворку FastAPI і розгортається на хмарному хостингу у вигляді ізольованого Docker-контейнера [22]. Контейнер збирається за допомогою інструкцій Dockerfile, де встановлюються залежності проєкту (через `poetry` або `pip`), після чого застосунок запускається через ASGI-сервер Uvicorn [23]. Автоматичне виконання міграцій бази даних здійснюється за допомогою інструменту Alembic, що гарантує актуальність структури таблиць після кожного релізу.

Бекенд-сервер забезпечує обробку HTTP-запитів, реалізацію бізнес-логіки навчання та взаємодію з базою даних. На цьому рівні реалізовано механізми JWT-автентифікації, управління навчальними курсами, перевірки практичних завдань, розрахунку ігрового досвіду (XP) та надання доступу до преміум-

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

підписок (Tiers). Між клієнтом і сервером жорстко налаштовано CORS-політику, що дозволяє крос-доменну взаємодію виключно з дозволених джерел.

Для надійного збереження основних даних використовується реляційна система PostgreSQL, підключення до якої здійснюється через асинхронний драйвер asyncpg. Крім цього, до інфраструктури підключено in-memory сховище Redis, яке відповідає за швидке кешування та захист системи від перевантажень (Rate Limiting). Усі медіафайли (аудіо, відео, обкладинки курсів) автоматично завантажуються та зберігаються у хмарному об'єктному сховищі Cloudflare R2 (S3-сумісне), що суттєво зменшує навантаження на основний сервер.

Система інтегрована з Telegram, що дозволяє користувачам проходити швидку авторизацію через Telegram-бота та користуватися платформою у форматі мобільного Mini App. Це інноваційне рішення надійно забезпечує безперешкодний доступ до навчальних матеріалів без необхідності встановлення додаткових мобільних застосунків через офіційні магазини. Завдяки використанню нативних вебтехнологій всередині месенджера, клієнтський інтерфейс миттєво адаптується під системні налаштування пристрою, а синхронізація прогресу з основним сервером відбувається у фоновому режимі, що значно підвищує рівень залученості цільової аудиторії.

Для зручного адміністрування платформи інтегровано розширення FastCRUD, яке автоматично генерує сучасний інтерфейс управління базою даних. Адміністратор отримує можливість створювати курси, модерувати користувачів та відстежувати статистику навчання без необхідності прямого доступу до SQL-консолі.

Загальну архітектуру розгортання вебплатформи наведено на рис. 3.1. Діаграма відображає основні вузли системи: клієнтські пристрої, CDN для фронтенду, Docker-контейнер із FastAPI-сервером, бази даних PostgreSQL та Redis, а також зовнішнє сховище Cloudflare R2 та API Telegram.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		60

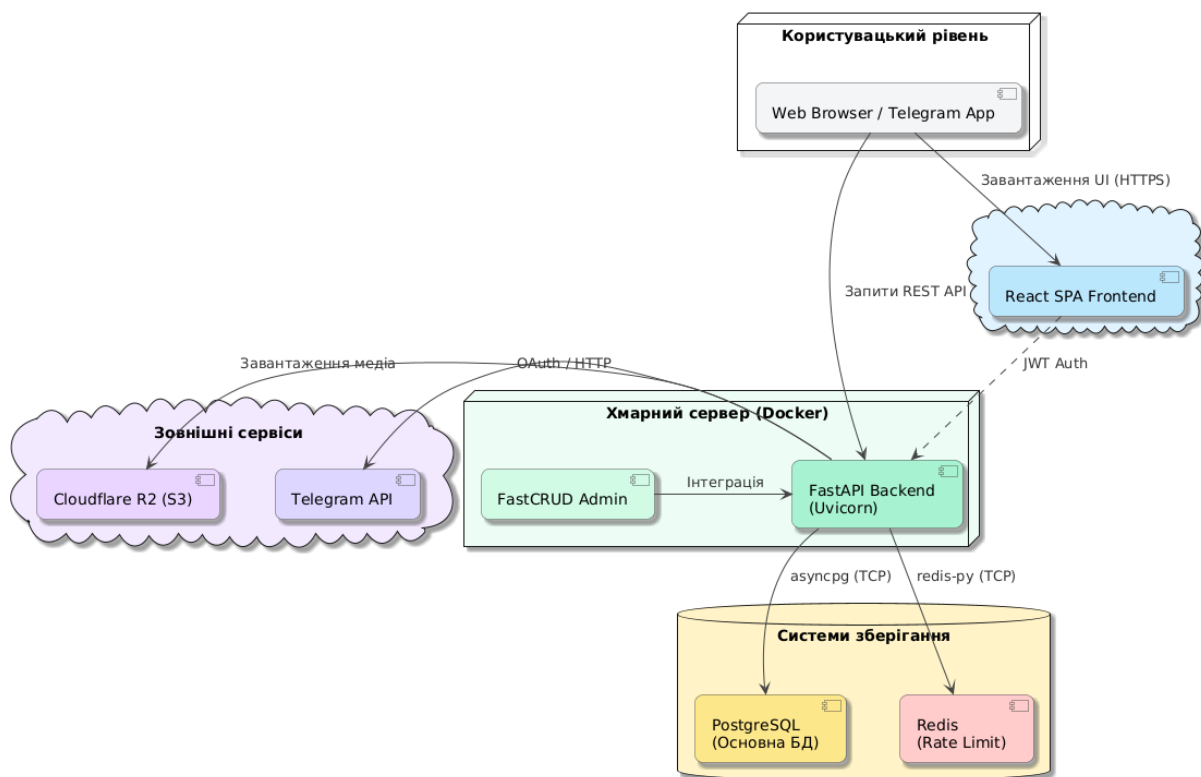


Рис. 3.1. Діаграма розгортання

Представлена архітектура забезпечує стабільне функціонування системи, захищену передачу даних між усіма компонентами, високу відмовостійкість, а також можливість незалежного масштабування окремих частин вебплатформи відповідно до навантаження. Додатково така структура сприяє спрощенню супроводу системи, дозволяє оновлювати окремі мікрокомпоненти без впливу на інші частини платформи та забезпечує гнучкість при подальшому розширенні функціоналу.

Це особливо важливо для сучасних освітніх платформ із мінливим навантаженням, де кількість одночасних навчальних сесій учнів та обсяг оброблюваної статистики прогресу можуть динамічно зростати. Тому використання хмарних технологій та надійних баз даних є критично необхідним кроком для успішної комерційної реалізації та довгострокового розвитку проєкту. Впроваджені архітектурні рішення мінімізують ризики простою, гарантують безпеку конфіденційних даних та створюють міцний фундамент для інтеграції нових інтерактивних навчальних модулів.

		Олійник Д.С.		
		Локтікова Т.М.		
Змн.	Арк.	№ докум.	Підпис	Дата

3.2 Структура інтерфейсу та порядок взаємодії із системою

Головна сторінка є точкою входу в систему. У верхній частині розміщено логотип платформи та елементи навігації для авторизації або швидкого переходу до особистого кабінету. Центральна частина сторінки містить короткий опис можливостей сервісу та блок із рекомендованими мовними програмами. Основний функціональний блок представлений каталогом курсів, який включає вкладки для перегляду всіх доступних програм, панель фільтрації (за мовою вивчення, рівнем складності та метою) та список карток курсів. Дані підвантажуються динамічно (з використанням пагінації), що забезпечує швидку роботу інтерфейсу (рис. 3.2).

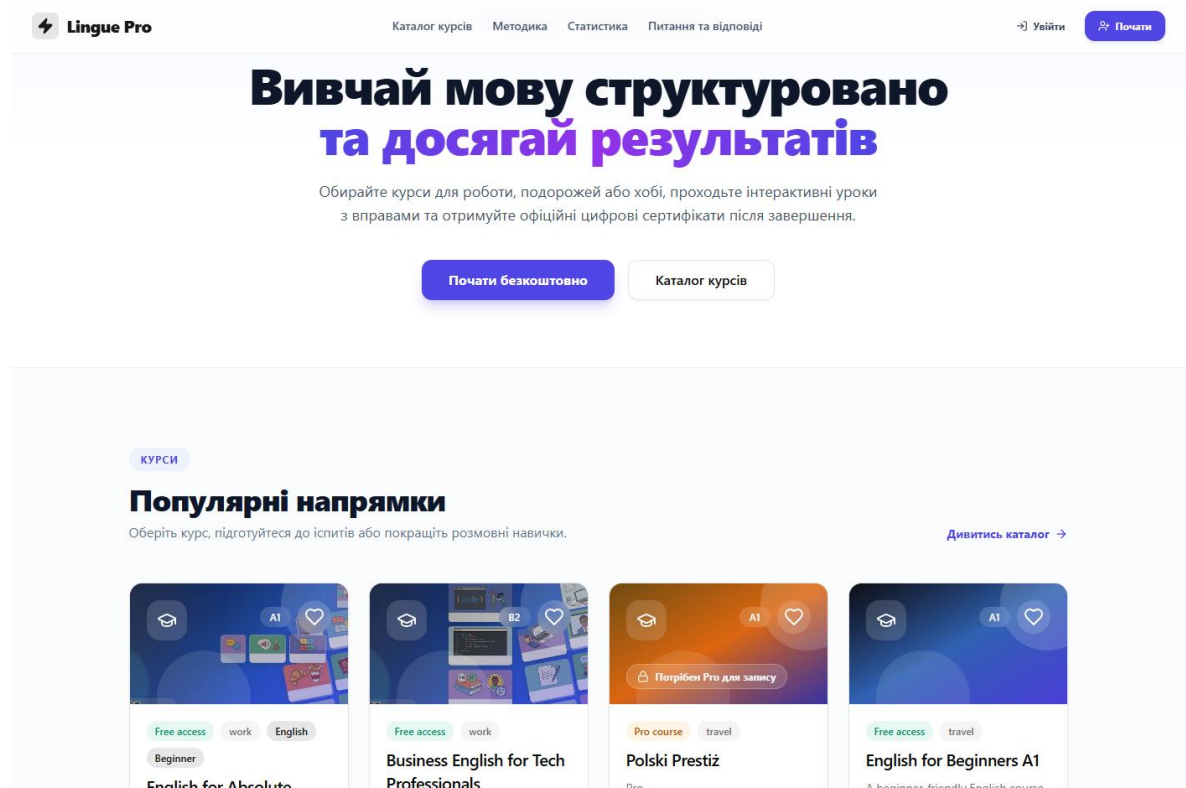


Рис. 3.2. Головна сторінка з каталогом курсів

Сторінка обраного курсу відображає повну інформацію про програму: обкладинку, назву, рівень складності (наприклад, A1-B2), орієнтовний час проходження та детальний опис. Для зручності студентів у нижній частині сторінки відображається детальна структура навчального плану — список модулів та розбивка по уроках. Це дозволяє учню оцінити обсяг матеріалу перед початком навчання. Для преміум-курсів додатково відображається інформація

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

про вартість доступу та необхідність оформлення підписки (Tier). Якщо курс є у вільному доступі або користувач вже має активну підписку, він може розпочати навчання безпосередньо зі сторінки курсу без необхідності переходу до інших розділів системи. Такий підхід забезпечує зручний процес взаємодії з платформою. Сторінка детального перегляду курсу представлена на рис. 3.3.

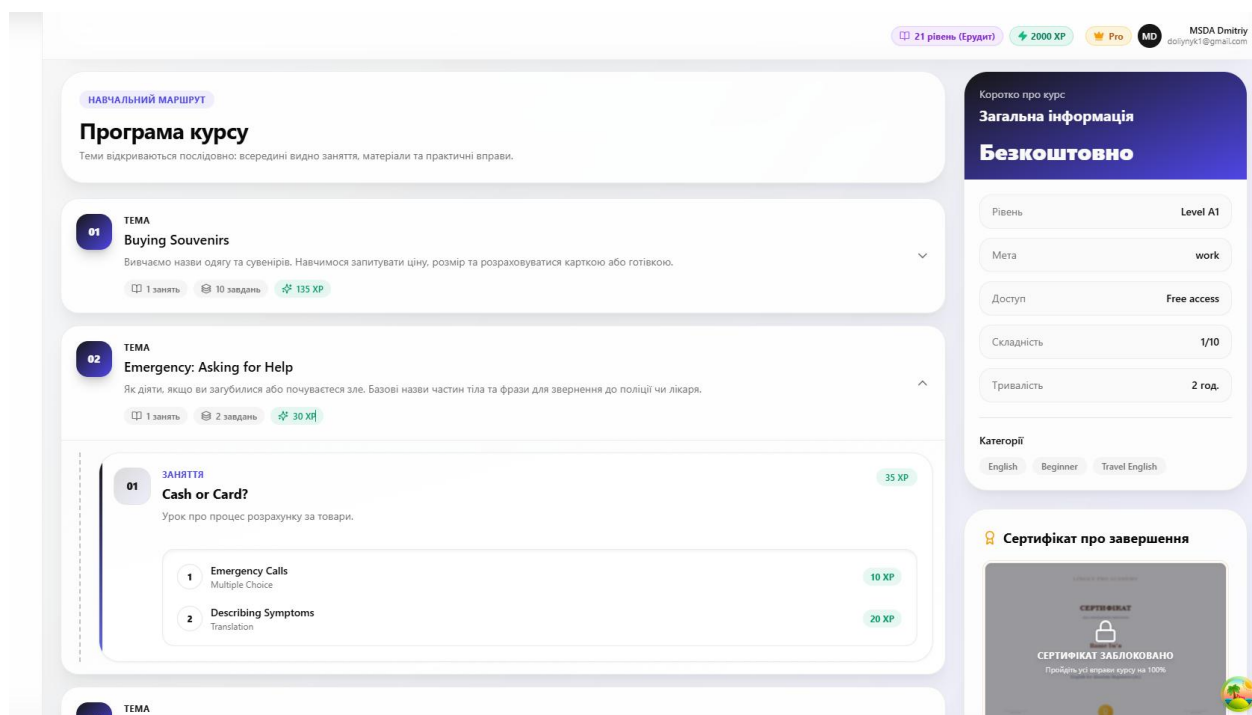


Рис. 3.3. Сторінка детального перегляду курсу

Сам процес навчання відбувається в інтерактивному режимі. Авторизований користувач знайомиться з теоретичним матеріалом (граматика, правила, приклади), після чого переходить до виконання практичних вправ (тести, введення тексту, аудіо-завдання). Після успішного виконання вправ система автоматично перевіряє відповіді та нараховує ігрові бали (XP). Крім того, на платформі реалізовано стрічку новин, де користувачі можуть залишати вподобання та коментарі під публікаціями від адміністрації, формуючи навколо платформи активну спільноту (рис. 3.4).

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

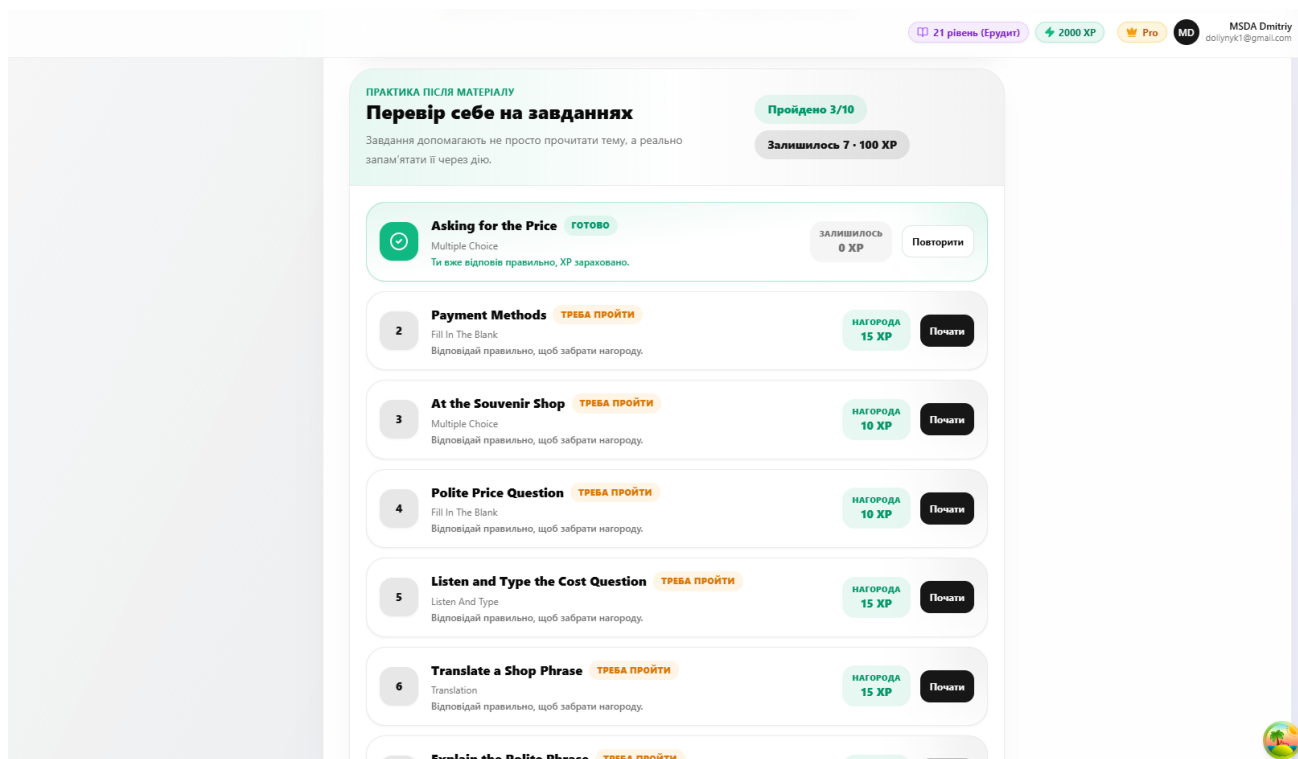


Рис. 3.4. Блок інтерактивних вправ

Окремою функціональною особливістю платформи є блок виконання завдань із таймером. Для певних типів вправ (наприклад, тестових запитань на швидкість реакції або завдань з аудіювання) адміністратор системи має можливість встановлювати жорсткий ліміт часу на надання відповіді (`time_limit_seconds`).

У такому випадку в інтерфейсі практичної вправи відображається динамічний таймер зворотного відліку (у вигляді прогрес-бару або цифрового годинника). Це створює ефект занурення, імітує умови реального мовного іспиту та стимулює студента приймати рішення швидше, не використовуючи сторонні словники. Якщо користувач не встигає дати відповідь до завершення часу, система автоматично зараховує спробу як невірну (рис. 3.5).

ВИКОНАННЯ ЗАВДАННЯ

Закрити

Translate a Shop Phrase

Завдання 6 з 10 · Translation · 15 XP

КРОК ПРАКТИКИ

Час: 1:24

Подумай і вибери одну відповідь

6/10 · Відповідь обрано

Проходь завдання по черзі. Після підтвердження відповіді відкриється перехід до наступного завдання.

ПЕРЕКЛАД ТА ПРАКТИКА

Переклади англійською: Я хотів би купити це.

Введи відповідь тут...

Підтвердити відповідь

← Попереднє заняття

Курс пройдено 🏆

Рис. 3.5. Блок виконання завдань із таймером зворотного відліку

Після надання відповіді або вичерпання ліміту часу клієнтський застосунок фіксує фактично витрачений користувачем час і відправляє запит на сервер для миттєвої перевірки. Система відразу підсвічує результат (вірно чи невірно).

У разі надання неправильної відповіді платформа перевіряє рівень доступу користувача (наявність PRO-версії). Для власників активної преміум-підписки відкривається розширений функціонал роботи з помилками. Студент отримує панель інструментів, де може скористатися кнопкою «Показати правильну відповідь» для миттєвого перегляду вірного варіанту та детального пояснення граматичного правила. Також PRO-користувач може обрати опцію «Спробувати ще раз» для повторного проходження цього ж завдання, або натиснути «Наступне завдання», щоб пропустити його і продовжити урок.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

Користувачі без преміум-доступу (на безкоштовному тарифі) не мають доступу до миттєвих підказок і змушені шукати правильну відповідь самостійно. Такий підхід не лише забезпечує більш гнучкий процес навчання, але й слугує потужним інструментом монетизації, природно стимулюючи студентів переходити на PRO-версію платформи (рис. 3.6).

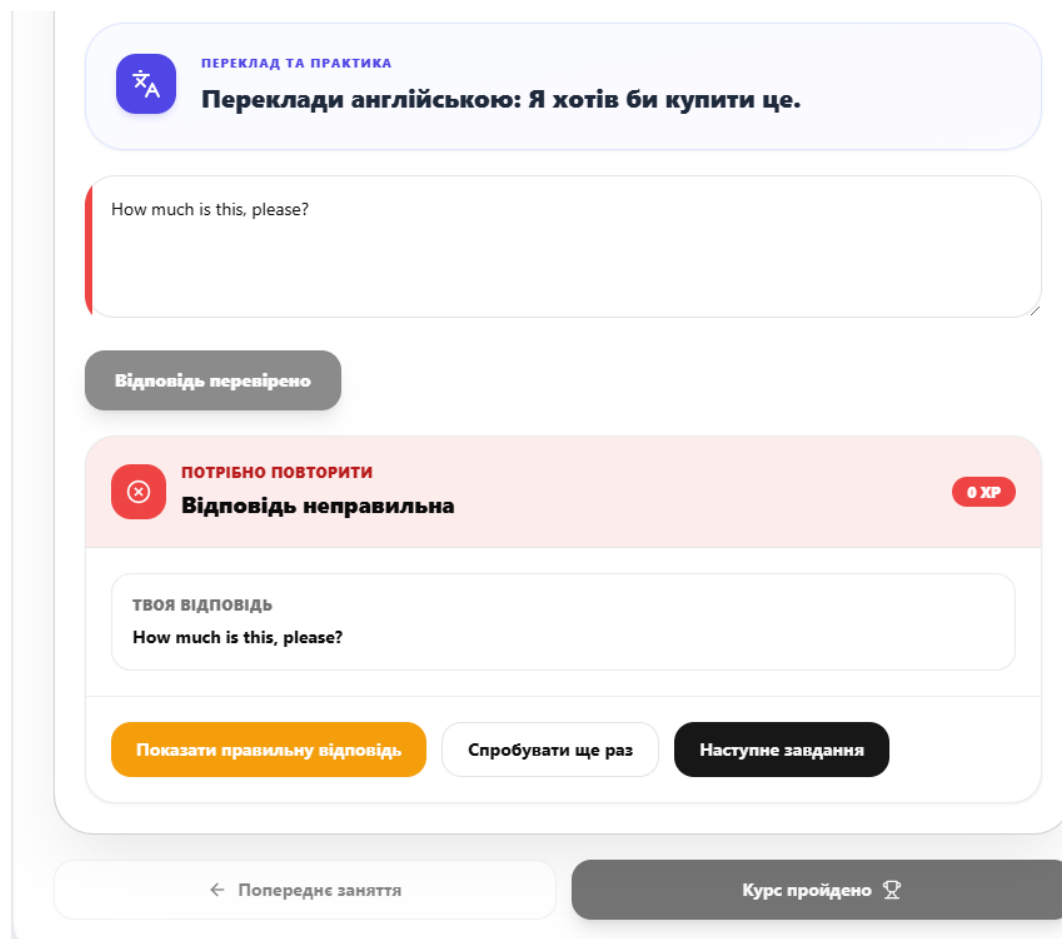


Рис. 3.6. Результат перевірки відповіді з преміум-можливостями

Особистий кабінет користувача (профіль) містить основну інформацію про учня, зокрема його аватар, ім'я, нікнейм (отриманий через Telegram), поточний статус преміум-підписки та дату реєстрації на платформі. Даний розділ є ключовим інструментом для відстеження власного навчального прогресу. Нижче в профілі розташовано детальну статистику: загальну кількість накопичених балів досвіду (XP), поточний стрік (кількість безперервних днів навчання) та історію проходження курсів у вигляді окремих карток. Кожна картка містить інформацію про поточний прогрес користувача на конкретному курсі. Завдяки

цьому учні можуть швидко повертатися до вивчення матеріалу та контролювати свої досягнення. Сторінка профілю користувача представлена на рис. 3.7.

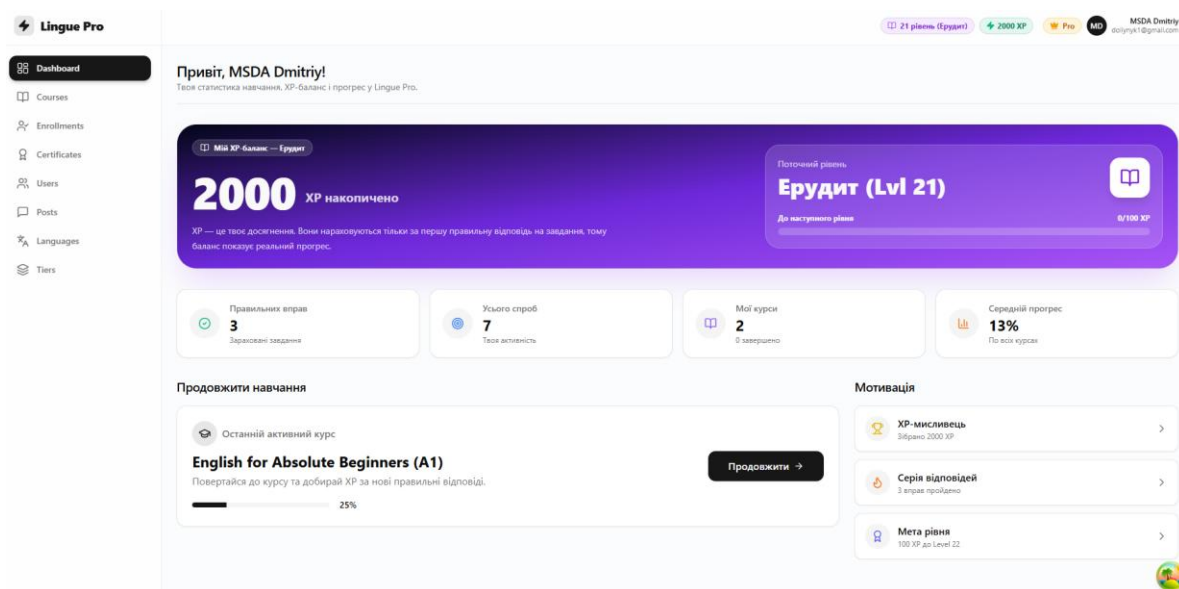


Рис. 3.7. Особистий кабінет користувача

Для отримання повного доступу до преміум-курсів та розширеного функціоналу системи (наприклад, детального аналізу помилок під час виконання вправ) передбачено можливість оформлення спеціалізованої підписки (Tier). Користувач може детально ознайомитися з переліком доступних тарифних планів і обрати найбільш оптимальний для себе період тарифікації наприклад, щомісячну підписку. Такий підхід гарантує високу гнучкість процесу навчання, дозволяючи студентам самостійно обирати найбільш вигідний і зручний формат взаємодії з платформою відповідно до своїх фінансових можливостей та навчальних цілей.

Оскільки на даному етапі розвитку проєкту платіжний функціонал реалізовано у вигляді програмного муляжу (симуляції), під час оформлення підписки користувач не вводить реальні банківські реквізити. Замість цього система демонструє роботу тестового платіжного сценарію, де користувач підтверджує симульовану транзакцію (рис. 3.8). Після успішного завершення тестової оплати інформація миттєво передається до системи, що забезпечує автоматичне оновлення статусу профілю до рівня «Premium» та відкриває доступ до закритого контенту. Закладена архітектура платформи дозволить у

		Олійник Д.С.			ІПЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

майбутньому легко замінити цей демонстраційний муляж на реальний платіжний шлюз (наприклад, Stripe або WayForPay).

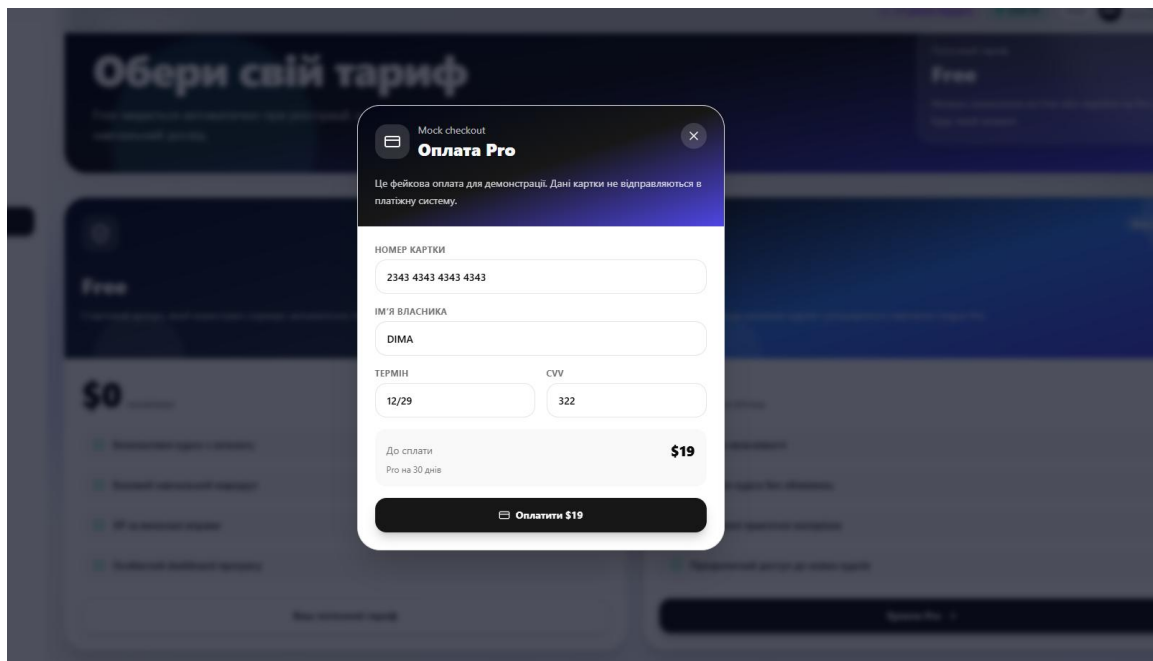


Рис. 3.8. Сторінка оформлення преміум-підписки

До моменту підтвердження симульованої транзакції система блокує можливість початку навчання на преміум-курсах, що запобігає несанкціонованому доступу до платного контенту. Одразу після підтвердження платежу статус користувача автоматично оновлюється без переходу на окремі сторінки, а інформація про активний рівень підписки зберігається у його особистому кабінеті.

У процесі навчання користувач може відстежувати свої досягнення. В окремому розділі «Мої сертифікати» міститься перелік усіх успішно пройдених мовних курсів. Для кожного курсу відображається основна інформація, зокрема дата його завершення та підтверджений рівень володіння мовою.

Наявність такого розділу не лише виконує функцію зручного архіву, але й слугує потужним мотиваційним фактором. Це дозволяє користувачу не тільки швидко знаходити власні документи та наочно контролювати загальний навчальний прогрес (рис. 3.9), але й за потреби демонструвати ці сертифікати потенційним роботодавцям або прикріплювати їх до професійних цифрових портфоліо.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		68

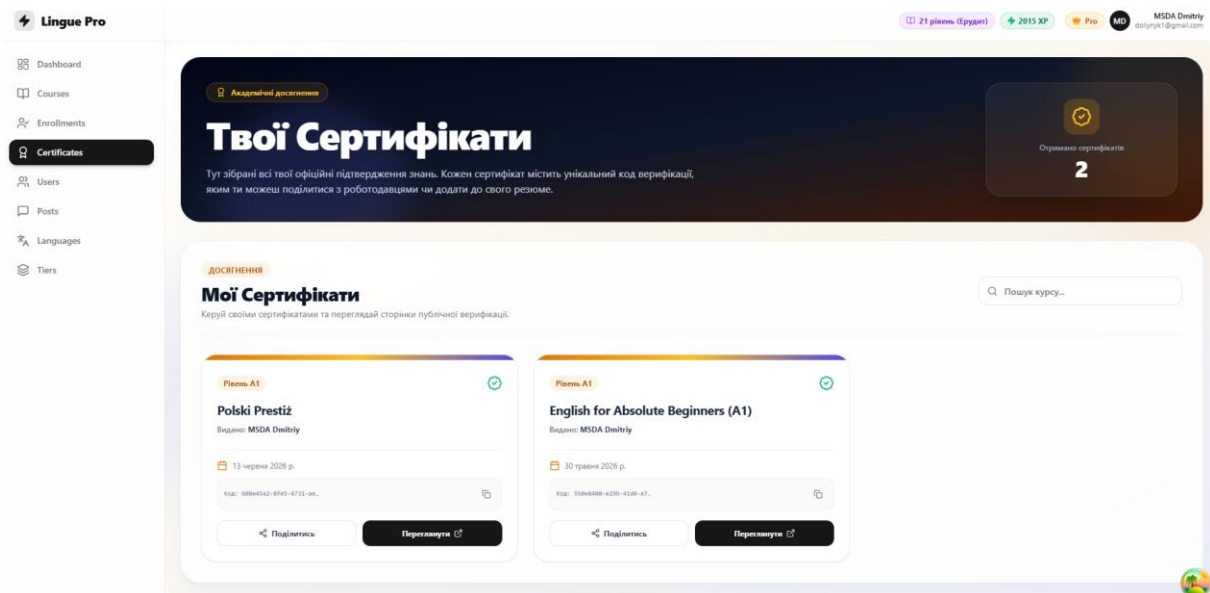


Рис. 3.9. Розділ досягнень та сертифікатів

Кожен отриманий цифровий сертифікат формується автоматично і містить інформацію про учня, назву курсу та унікальний ідентифікатор для підтвердження його автентичності (рис. 3.10).

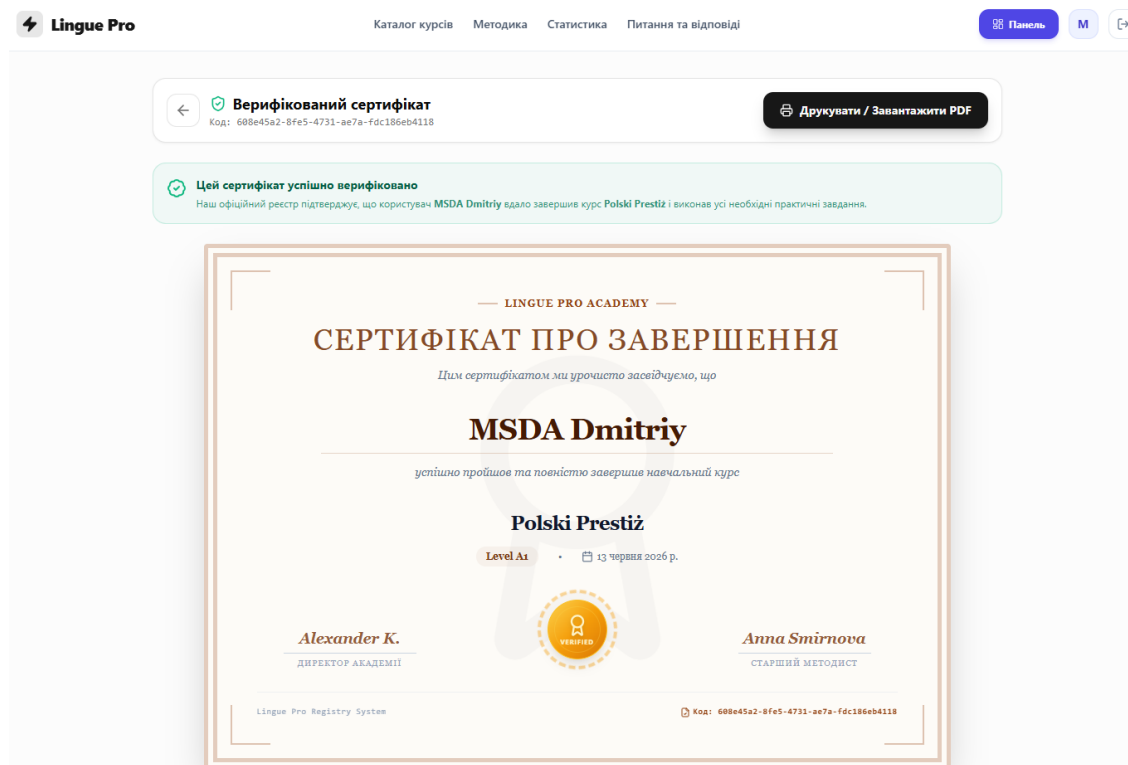


Рис. 3.10. Електронний сертифікат про завершення курсу

Окрім власної статистики, користувач має змогу відстежувати свою позицію в системі відносно інших учнів. На платформі реалізовано сторінку соціальної взаємодії, де відображається глобальний рейтинг (лідерборд) за

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

кількістю накопичених ігрових балів (XP) та список доданих друзів (рис. 3.11). Це значно підвищує змагальний дух та мотивацію до щоденного навчання.

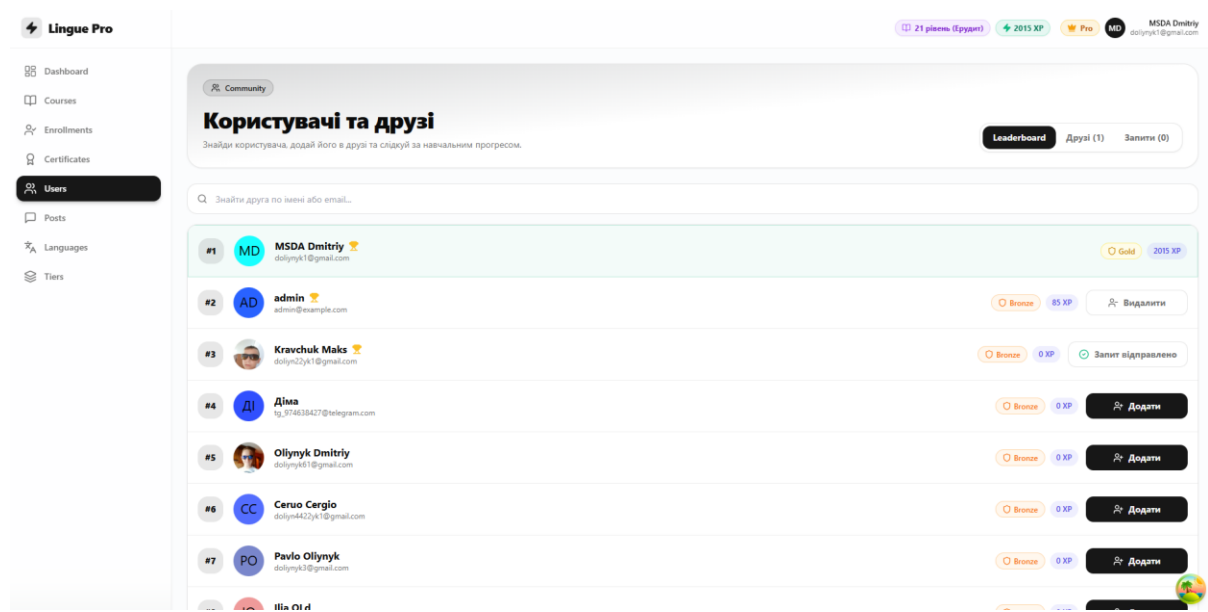


Рис. 3.11. Сторінка соціальної взаємодії та глобального рейтингу

Натиснувши кнопку «Налаштування», користувач переходить до сторінки керування особистим профілем. На сторінці налаштувань користувач може змінювати своє ім'я та прізвище, а також змінювати пароль для безпечного входу до системи. Додатково передбачено можливість завантаження або оновлення аватара профілю, що дозволяє персоналізувати обліковий запис та покращити його візуальне представлення для інших користувачів платформи (рис. 3.12).

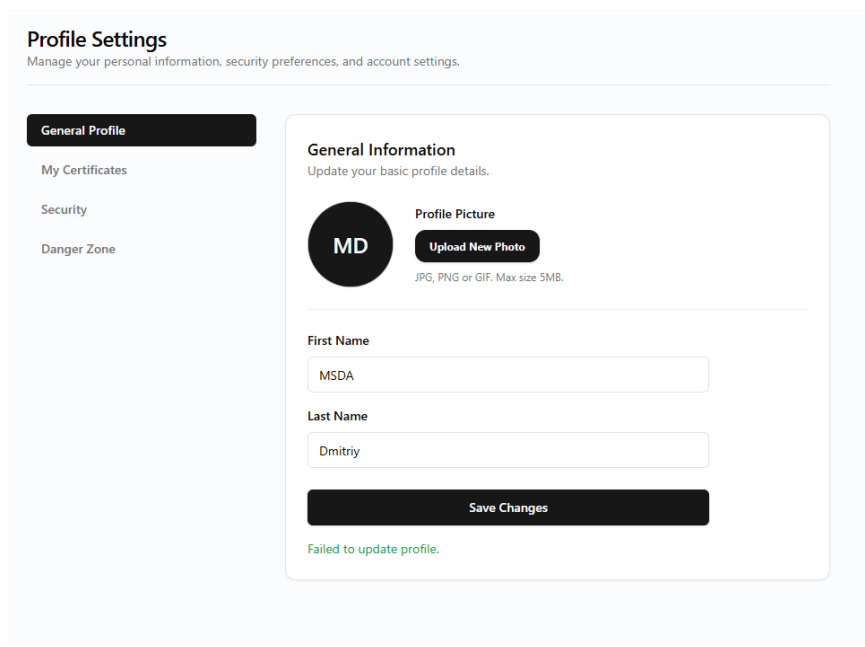


Рис. 3.12. Особистий кабінет користувача

У нижній частині сторінки налаштувань профілю розміщено блок «Зона небезпеки», який забезпечує користувачу повний контроль над своїми даними. У ньому реалізовано можливість безпечного виходу з системи (із деактивацією JWT-токенів), відв'язка інтегрованого Telegram-акаунта, а також функція повного видалення профілю. Дані функції гарантують дотримання принципів конфіденційності та дозволяють гнучко керувати доступом (рис. 3.13).

Рис. 3.13. Блок керування акаунтом

Оскільки платформа не передбачає ролі публічного організатора подій, весь процес створення та модерації навчального контенту винесено в окрему адміністративну панель, побудовану на базі бібліотеки FastCRUD. Даний розділ доступний виключно користувачам із правами адміністратора (is_superuser). У ньому зібрано ключові інструменти для керування курсами, уроками, користувачами та підписками, що дозволяє зручно адмініструвати проєкт. Інтерфейс панелі містить бокове навігаційне меню зі швидким доступом до всіх реляційних моделей бази даних (таких як User, Course, Lesson та Exercise). На головному екрані (Dashboard) виводиться загальна статистика записів за кожною таблицею, що забезпечує миттєвий моніторинг інформаційного наповнення системи (рис. 3.14).

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		71

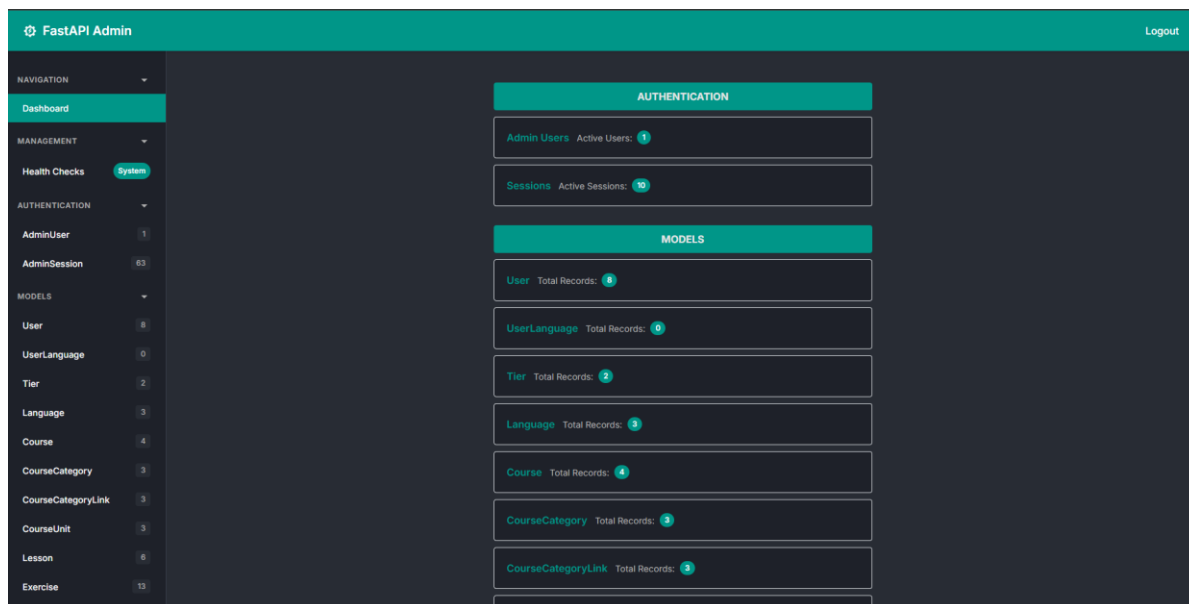


Рис. 3.14. Головна сторінка адміністративної панелі

Натиснувши кнопку «Створити» в модулі курсів, адміністратор переходить до форми додавання нової навчальної програми. Дана форма дозволяє задавати основні параметри курсу, зокрема назву, детальний опис, мову вивчення, рівень складності, орієнтовну кількість годин та статус (чернетка або опубліковано). Такий підхід забезпечує структуроване введення даних і дозволяє гнучко формувати навчальний план (рис. 3.15).

Рис. 3.15. Форма створення нового мовного курсу

Важливою конкурентною перевагою розробленої вебплатформи є її глибока інтеграція з месенджером Telegram у форматі Telegram Mini App. Це

		Олійник Д.С.			ПІЗ.КР.Б–121–26–ПЗ	Арк.
		Локтікова Т.М.				72
Змн.	Арк.	№ докум.	Підпис	Дата		

інноваційне рішення дозволяє студентам проходити навчання, виконувати вправи та перевіряти свій рейтинг безпосередньо у вікні месенджера, без необхідності відкривати сторонні мобільні браузері або завантажувати окремі застосунки з App Store чи Google Play.

Інтерфейс Telegram Mini App повністю адаптований під екрани мобільних пристроїв і візуально відчувається як нативна частина самого месенджера. Головною перевагою такого формату є безшовна авторизація: система автоматично перехоплює дані профілю (зокрема telegram_id та telegram_username) і виконує миттєвий вхід до системи без ручного введення логіна та пароля.

Через інтерфейс Mini App учень отримує швидкий доступ до повного функціоналу платформи — від перегляду каталогу курсів до проходження інтерактивних уроків і відстеження власного балансу XP (рис. 3.16). Таке архітектурне рішення значно знижує поріг входу для нових студентів та підвищує загальний показник їхнього утримання (Retention Rate), оскільки процес навчання стає максимально зручним і доступним у будь-який час.

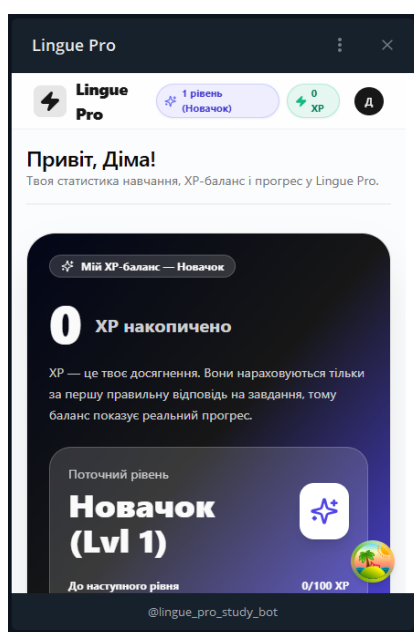


Рис. 3.16. Інтерфейс платформи у форматі Telegram Mini App

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				73
Змн.	Арк.	№ докум.	Підпис	Дата		

3.3 Тестування роботи програмного продукту

Після завершення розробки вебплатформи було проведено функціональне тестування основних можливостей системи. Метою тестування було перевірити коректність роботи всіх модулів, відповідність реалізованого функціоналу заявленим вимогам, стабільність взаємодії між клієнтською та серверною частинами, а також інтеграцію із зовнішніми сервісами (Telegram, Cloudflare R2). Тестування виконувалося вручну в локальному середовищі (React dev server + FastAPI Uvicorn) та в ізольованому контейнеризованому середовищі (за допомогою Docker Compose), що дозволило повноцінно оцінити працездатність системи та коректність взаємодії всіх її компонентів.

Перевірка здійснювалася за сценаріями користувача (use case testing): для кожного тесту фіксувалися вхідні дані, очікуваний результат, фактичний результат і статус виконання. Окремо перевірялися позитивні сценарії (успішне виконання операцій) та негативні (відхилення при некоректних даних, спроба доступу без авторизації або порушення бізнес-правил платформи). У разі виявлення багів виконувалося їхнє усунення з подальшим повторним проходженням відповідних тестів.

Особливу увагу було приділено перевірці двох методів авторизації (через Email із генерацією JWT-токенів та безшовного входу через Telegram Mini App), захисту приватних маршрутів (ProtectedRoute), а також процесу реєстрації на мовні курси. Детально протестовано логіку гейміфікації: правильність розрахунку балів досвіду (XP) за виконані вправи, роботу таймерів під час тестувань, безперебійне оновлення щоденних стріків та генерацію електронних сертифікатів.

Окремо перевірено роботу механізмів доступу до платного контенту (оформлення преміум-підписок Tiers), соціальних функцій (рейтинг лідерборду, коментарі та лайки в новинній стрічці), а також працездатність адміністративної панелі на базі FastCRUD (створення курсів, додавання медіафайлів через URL,

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		74

модерація користувачів). Результати проведеного тестування наведено у табл. 3.1.

Результати проведеного тестування наведено у табл. 3.1.

Таблиця 3.1

Результати тестування програмного забезпечення

Назва тесту	Вхідні дані	Очікуваний результат	Фактичний результат	Статус
Авторизація через Email	Email та пароль	Успішний вхід, видача JWT токенів	Токени згенеровано, відкрито профіль	Пройдено
Авторизація Telegram	Запуск Mini App	Автентифікація за telegram_id	Користувача авторизовано миттєво	Пройдено
Захист маршрутів	Перехід гостем на захищену сторінку	Перенаправлення на сторінку входу	Користувач перенаправлений на /login	Пройдено
Каталог курсів	Завантаження головної сторінки	Відображення списку доступних курсів	Картки курсів успішно завантажено	Пройдено
Фільтрація курсів	Вибір мови та рівня складності (напр., B2)	Оновлений список відповідно до фільтра	Фільтрація працює коректно	Пройдено
Реєстрація на курс	Натискання «Почати навчання» (безкоштовний курс)	Створення запису CourseEnrollment	Запис створено, урок відкрито	Пройдено
Доступ до преміум-курсу (негативний)	Спроба доступу без активної підписки	Блокування доступу, пропозиція оплати	Доступ відхилено, показано попап	Пройдено

Продовження табл. 3.1

Оформлення підписки	Тестова оплата (симуляція) тарифного плану	Зміна статусу на "Premium"	Статус оновлено, контент відкрито	Пройдено
Виконання вправи	Правильна відповідь на тестове запитання	Нарахування балів (XP) та пояснення	Бали додано до бази, показано зелений маркер	Пройдено
Таймер вправи	Вичерпання ліміту часу (time_limit_seconds)	Зарахування помилки	Вправу зупинено, показано правильну відповідь	Пройдено
Оновлення стріка	Перша виконана вправа за поточний день	Лічильник стріка збільшено на +1	Стрік оновлено, показано анімацію	Пройдено
Соціальна активність	Натискання лайка під постом	Збереження лайка, оновлення лічильника	Лайк додано (post_likes оновлено)	Пройдено
Глобальний рейтинг	Перехід до лідерборду	Сортування учнів за кількістю XP	Рейтинг побудовано коректно	Пройдено
Генерація сертифіката	Завершення 100% уроків курсу	Створення унікального сертифіката	Сертифікат додано до профілю студента	Пройдено
Створення курсу	Заповнення форми в FastCRUD адміністратором	Збереження курсу в базі даних	Курс успішно створено та опубліковано	Пройдено
Модерація студентів	Блокування акаунта адміністратором	Заборона входу для даного користувача	Користувач не може увійти в систему	Пройдено

За результатами тестування всі розроблені сценарії виконано успішно.
Система коректно реалізує ключові бізнес-процеси: автентифікацію (включно з

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

Telegram), взаємодію з навчальним контентом, перевірку вправ, гейміфікацію (нарахування XP і стріків), розмежування доступу до платного контенту та адміністрування платформи.

Висновки до розділу 3

У третьому розділі розглянуто структуру інтерфейсу та порядок взаємодії користувача з освітньою платформою. Проаналізовано архітектуру розгортання та наведено діаграму фізичної взаємодії між клієнтом (React SPA), бекендом (FastAPI), базами даних (PostgreSQL, Redis), хмарним сховищем Cloudflare R2 та Telegram API.

У межах розділу описано основні екрани користувацького інтерфейсу, включаючи головну сторінку з каталогом навчальних курсів і системою фільтрації, сторінку детального перегляду курсу та особистий кабінет студента. Окрему увагу приділено функціоналу інтерактивного навчання, де користувач виконує практичні вправи з урахуванням ліміту часу, отримує ігрові бали досвіду (XP) та підтримує щоденні стріки активності. Також детально розглянуто механізм оформлення преміум-підписок (Tiers), який забезпечує доступ до закритого контенту та розширеного функціоналу (миттєві підказки під час помилок).

Представлено порядок встановлення та налаштування системи, включаючи процес розгортання клієнтської та серверної частин у контейнеризованому Docker-середовищі. Проведено ретельне тестування основних функцій вебзастосунку та його інтеграції у форматі Telegram Mini App, що підтвердило коректну роботу всіх ключових навчальних сценаріїв, високу швидкість обробки запитів та загальну стабільність платформи.

Отже, система повністю відповідає вимогам: вона забезпечує гейміфікований інтерфейс для студентів, підтримує монетизацію за підпискою та надає адміністраторам панель FastCRUD для управління контентом. Це робить платформу повністю готовою до комерційного використання.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

ВИСНОВКИ

У межах виконання даної кваліфікаційної роботи було розроблено повноцінну вебплатформу для інтерактивного вивчення іноземних мов. Проєкт охоплює повний цикл створення сучасного освітнього вебсервісу: від аналізу вимог і проєктування архітектури до безпосередньої реалізації функціоналу, інтеграції зовнішніх сервісів та фінального тестування системи.

Основною метою роботи було створення зручної, гейміфікованої платформи, яка дозволяє студентам переглядати каталог навчальних програм, виконувати практичні вправи, заробляти бали досвіду (XP), підтримувати щоденну активність (стріки) та змагатися у глобальному рейтингу. Для адміністраторів проєкту передбачено багатofункціональну панель керування, що дозволяє структурувати курси, додавати медіаматеріали та контролювати доступ користувачів.

У системі реалізовано клієнт-серверну архітектуру з використанням бібліотеки React на стороні клієнта та високопродуктивного фреймворку FastAPI на сервері. Для надійного зберігання даних використовується реляційна база PostgreSQL, механізми лімітування запитів та кешування побудовано на базі Redis, а всі медіафайли зберігаються в об'єктному хмарному сховищі Cloudflare R2. Автентифікація користувачів базується на технології JWT-токенів та тісно інтегрована з Telegram API для забезпечення безшовного входу у форматі Telegram Mini App.

Окрему увагу приділено проєктуванню модуля електронної комерції для обробки преміум-підписок (Tiers). Оскільки проєкт має навчальний характер, платіжний функціонал реалізовано у вигляді програмного муляжу (симуляції). Це дозволяє безпечно демонструвати логіку купівлі підписки без проведення реальних банківських транзакцій. Після успішної імітації оплати система автоматично надає користувачеві розширений доступ до закритого навчального контенту. Додатково було реалізовано повноцінний соціальний модуль, який

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

дозволяє адміністраторам публікувати новини, а студентам — залишати вподобання та коментарі.

Інтерфейс створеної системи є максимально адаптивним, що гарантує зручність використання як на десктопних, так і на мобільних пристроях. Проведене тестування підтвердило коректність роботи всіх розроблених модулів, включаючи алгоритми гейміфікації. Реалізований вебзастосунок забезпечує ефективну та прозору взаємодію між учнями та адміністраторами освітнього сервісу.

Отриманий результат є сучасною базовою платформою для вивчення мов, яка має потужний потенціал до подальшого масштабування. У майбутньому закладена архітектура дозволить легко замінити муляж оплат на реальний платіжний шлюз (наприклад, Stripe або WayForPay), а також інтегрувати розгорнуті аналітичні дашборди для викладачів і впровадити алгоритми розумних рекомендацій на базі штучного інтелекту.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Олійник Д.С., Локтікова Т.М. Вебплатформа для вивчення іноземних мов з підтримкою Телеграм-бота: тези Всеукраїнської науково-практичної on-line конференції здобувачів вищої освіти і молодих учених, присвяченої Дню науки, м. Житомир, 11-15 травня 2026 р. Житомир: «Житомирська політехніка», 2026. С. 86-87.
2. REST API Tutorial. URL: <https://restfulapi.net/> (дата звернення: 03.06.2026).
3. Duolingo: The world's best way to learn a language. URL: <https://www.duolingo.com/> (дата звернення: 03.06.2026).
4. Babbel: Language Learning Platform. URL: <https://www.babbel.com/> (дата звернення: 03.06.2026).
5. Memrise: Learn a new language. URL: <https://www.memrise.com/> (дата звернення: 03.06.2026).
6. OWASP Top Ten Web Application Security Risks / OWASP Foundation. URL: <https://owasp.org/www-project-top-ten/> (дата звернення: 03.06.2026).
7. Python 3.13.3 Documentation / Python Software Foundation. URL: <https://docs.python.org/3/> (дата звернення: 03.06.2026).
8. FastAPI Documentation / Sebastián Ramírez. URL: <https://fastapi.tiangolo.com/> (дата звернення: 03.06.2026).
9. JSON Web Tokens (JWT). Introduction and Documentation. URL: <https://auth0.com/docs/secure/tokens/json-web-tokens> (дата звернення: 03.06.2026).
10. Cloudflare R2 / Cloudflare. URL: <https://www.cloudflare.com/products/r2/> (дата звернення: 03.06.2026).
11. FastCRUD Documentation / Igor Benav. URL: <https://fastcrud.igorbenav.com/> (дата звернення: 03.06.2026).
12. PostgreSQL Documentation / PostgreSQL Global Development Group. URL: <https://www.postgresql.org/docs/> (дата звернення: 03.06.2026).
13. Neon Database / Neon Inc. URL: <https://neon.tech/> (дата звернення: 03.06.2026).

		Олійник Д.С.			ПЗ.КР.Б-121-26-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

- 14.SQLAlchemy Documentation. URL: <https://docs.sqlalchemy.org/> (дата звернення: 03.06.2026).
- 15.Telegram Mini Apps Documentation / Telegram. URL: <https://core.telegram.org/bots/webapps> (дата звернення: 03.06.2026).
- 16.Redis Documentation / Redis Ltd. URL: <https://redis.io/docs/> (дата звернення: 03.06.2026).
- 17.Uvicorn: The lightning-fast ASGI server. URL: <https://www.uvicorn.org/> (дата звернення: 03.06.2026)
- 18.Pydantic Documentation. URL: <https://docs.pydantic.dev/> (дата звернення: 03.06.2026).
- 19.React Documentation / Meta Open Source. URL: <https://react.dev/> (дата звернення: 03.06.2026).
- 20.Vercel Documentation / Vercel Inc. URL: <https://vercel.com/docs> (дата звернення: 03.06.2026).
- 21.Fly.io Documentation. URL: <https://fly.io/docs/> (дата звернення: 03.06.2026).
- 22.Docker Documentation / Docker Inc. URL: <https://docs.docker.com/> (дата звернення: 03.06.2026).
- 23.Vite: Next Generation Frontend Tooling. URL: <https://vitejs.dev/guide/> (дата звернення: 03.06.2026).
- 24.asyncpg Documentation / MagicStack. URL: <https://magicstack.github.io/asyncpg/current/> (дата звернення: 03.06.2026).
- 25.Dirgantara D. P., Kusumo D. S., Utomo R. G. Docker-Based Monolithic and Microservices Architecture Performance Comparison. URL: <https://jutif.if.unsoed.ac.id/index.php/jurnal/article/view/1338> (дата звернення: 03.06.2026).
- 26.Cloudflare Developers Documentation / Cloudflare. URL: <https://developers.cloudflare.com/> (дата звернення: 03.06.2026).
- 27.Fowler M. Patterns of Enterprise Application Architecture. Boston : Addison-Wesley Professional, 2002.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		81

28. Richards M., Ford N. Fundamentals of Software Architecture: An Engineering Approach. Sebastopol : O'Reilly Media, 2020.
29. Richardson L., Amundsen M. RESTful Web APIs: Services for a Changing World. Sebastopol : O'Reilly Media, 2013. 406 c.
30. Lutz M. Learning Python. Sebastopol : O'Reilly Media, 2020. 1552 c.

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

ДОДАТКИ

		Олійник Д.С.			ІІЗ.КР.Б–121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

Лайки та Коментарі під новинними публікаціями

Лістинг А.1 Поставити/видалити лайк:

```

@router.post("/{id}/like", response_model=dict[str, Any])
async def toggle_like_post(
    id: int,
    current_user: Annotated[dict, Depends(get_current_user)],
    db: Annotated[AsyncSession, Depends(async_get_db)],
) -> dict[str, Any]:
    db_post = await db.scalar(select(Post).where(Post.id == id, Post.is_deleted ==
False))
    if db_post is None:
        raise NotFoundException("Post not found")

    is_teacher = current_user.get("account_type") == "teacher"

    # Перевірка, чи користувач вже лайкнув публікацію
    stmt = select(PostLike).where(PostLike.post_id == id)
    if is_teacher:
        stmt = stmt.where(PostLike.teacher_id == current_user["id"])
    else:
        stmt = stmt.where(PostLike.user_id == current_user["id"])

    like_obj = await db.scalar(stmt)

    if like_obj:
        # Видалення лайку (Unlike)
        await db.delete(like_obj)
        await db.commit()

        liked = False

```

		Олійник Д.С.			ІІЗ.КР.Б-121-26-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

else:

Додавання нового лайку (Like)

new_like = PostLike(

post_id=id,

user_id=None if is_teacher else current_user["id"],

teacher_id=current_user["id"] if is_teacher else None,

)

db.add(new_like)

await db.commit()

liked = True

Отримання оновленої кількості лайків для клієнта

likes_count_stmt =

select(func.count()).select_from(PostLike).where(PostLike.post_id == id)

likes_count = await db.scalar(likes_count_stmt) or 0

return {

"message": "Like status updated",

"liked": liked,

"likes_count": likes_count

}

Лістинг А.2 Створення коментарів під публікацією::

@router.post("/{id}/comments", response_model=PostCommentRead,
status_code=201)

async def create_post_comment(

id: int,

comment: PostCommentCreate,

current_user: Annotated[dict, Depends(get_current_user)],

		Олійник Д.С.			ПІЗ.КР.Б-121-26-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

db: Annotated[AsyncSession, Depends(async_get_db)],
) -> dict[str, Any]:
    db_post = await db.scalar(select(Post).where(Post.id == id, Post.is_deleted ==
False))
    if db_post is None:
        raise NotFoundException("Post not found")

    is_teacher = current_user.get("account_type") == "teacher"
    is_superuser = current_user.get("is_superuser", False)

    # Збереження нового коментаря в базу даних
    new_comment = PostComment(
        post_id=id,
        created_by_user_id=None if is_teacher else current_user["id"],
        created_by_teacher_id=current_user["id"] if is_teacher else None,
        text=comment.text,
    )
    db.add(new_comment)
    await db.commit()
    await db.refresh(new_comment)

    # Формування інформації про автора коментаря
    author_info = PostCommentAuthor(
        name=current_user.get("name"),
        surname=current_user.get("surname"),
        profile_image_url=current_user.get("profile_image_url"),
        account_type="admin" if is_superuser else current_user.get("account_type"),
    )

    return {

```

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

    "id": new_comment.id,
    "post_id": new_comment.post_id,
    "created_by_user_id": new_comment.created_by_user_id,
    "created_by_teacher_id": new_comment.created_by_teacher_id,
    "text": new_comment.text,
    "created_at": new_comment.created_at,
    "author": author_info
}

```

Лістинг А.3 Видалення коментаря:

```

from fastapi import Response

@router.delete("/{post_id}/comments/{comment_id}", status_code=204)
async def delete_post_comment(
    post_id: int,
    comment_id: int,
    current_user: Annotated[dict, Depends(get_current_user)],
    db: Annotated[AsyncSession, Depends(async_get_db)],
):
    # Знаходимо коментар у базі даних
    db_comment = await db.scalar(
        select(PostComment).where(
            PostComment.id == comment_id,
            PostComment.post_id == post_id
        )
    )

    if db_comment is None:
        raise NotFoundException("Коментар не знайдено")

```

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		87

```

is_teacher = current_user.get("account_type") == "teacher"
is_superuser = current_user.get("is_superuser", False)

# Перевірка прав доступу: видалити може лише автор або адміністратор
authorized = False
if is_superuser:
    authorized = True
elif is_teacher and db_comment.created_by_teacher_id == current_user["id"]:
    authorized = True
elif not is_teacher and db_comment.created_by_user_id == current_user["id"]:
    authorized = True

if not authorized:
    raise ForbiddenException("У вас немає прав для видалення цього коментаря")

# Видалення коментаря з бази
await db.delete(db_comment)
await db.commit()

return Response(status_code=204)

```

Управління зарахуваннями та підписками

Лістинг Б.1 Ініціалізація навчання та зарахування на курс:

```

@router.post("/{course_id}/enroll", response_model=MessageOut)
async def enroll_in_course(
    request: Request,
    course_id: int,
    current_user: Annotated[dict[str, Any], Depends(get_current_user)],
    db: Annotated[AsyncSession, Depends(async_get_db)],
) -> dict[str, str]:
    if current_user.get("account_type") != "student":
        raise ForbiddenException("Only students can enroll in courses")

    course_result = await db.execute(
        select(Course)
        .where(Course.id == course_id)
        .where(Course.status == CourseStatus.PUBLISHED)
    )
    course = course_result.scalar_one_or_none()
    if course is None:
        raise NotFoundException("Course not found")

    enrollment_result = await db.execute(
        select(CourseEnrollment)
        .where(CourseEnrollment.user_id == current_user["id"])
        .where(CourseEnrollment.course_id == course_id)
    )
    enrollment = enrollment_result.scalar_one_or_none()

```

Розрахунок поточного прогресу навчання

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

course_exercises_res = await db.execute(
    select(Exercise.id)
    .join(Lesson, Lesson.id == Exercise.lesson_id)
    .where(Lesson.course_id == course_id)
)
all_course_exercise_ids = {r for r in course_exercises_res.scalars().all()}

completed_res = await db.execute(
    select(ExerciseAttempt.exercise_id)
    .join(Exercise, Exercise.id == ExerciseAttempt.exercise_id)
    .join(Lesson, Lesson.id == Exercise.lesson_id)
    .where(
        Lesson.course_id == course_id,
        ExerciseAttempt.user_id == current_user["id"],
        ExerciseAttempt.is_correct.is_(True)
    )
)
completed_exercise_ids = {r for r in completed_res.scalars().all()}

if all_course_exercise_ids:
    progress_percent = int((len(completed_exercise_ids & all_course_exercise_ids)
/ len(all_course_exercise_ids)) * 100)
else:
    progress_percent = 100

status = EnrollmentStatus.COMPLETED if progress_percent == 100 else
EnrollmentStatus.ENROLLED

# Створення нового запису про зарахування
if enrollment is None:

```

		Олійник Д.С.			ІІЗ.КР.Б-121-26-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		90

```

db.add(
    CourseEnrollment(
        user_id=current_user["id"],
        course_id=course_id,
        status=status,
        progress_percent=progress_percent,
    )
)
await db.commit()
return {"message": "Successfully enrolled in the course"}

```

```

enrollment.status = status
enrollment.progress_percent = progress_percent
await db.commit()
return {"message": "Course enrollment updated"}

```

Лістинг Б.2 Оформлення підписки та розблокування преміум-доступу:

```
@router.post("/tiers/{name}/purchase", response_model=TierPurchaseRead)
```

```
async def purchase_tier(
```

```
    request: Request,
```

```
    name: str,
```

```
    purchase: TierPurchaseRequest,
```

```
    current_user: Annotated[dict, Depends(get_current_user)],
```

```
    db: Annotated[AsyncSession, Depends(async_get_db)],
```

```
) -> dict[str, Any]:
```

```
    db_tier = await crud_tiers.get(db=db, name=name, schema_to_select=TierRead)
```

```
    if db_tier is None:
```

```
        raise NotFoundException("Tier not found")
```

```
    now = datetime.now(UTC)
```

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		91

```
expires_at = now + timedelta(days=30)
```

```
# Створення транзакції (муляжна оплата)
```

```
subscription = UserTierSubscription(  
    user_id=current_user["id"],  
    tier_id=db_tier["id"],  
    status=SubscriptionStatus.ACTIVE,  
    payment_provider="mock",  
    payment_reference=f"mock_{uuid4().hex}",  
    amount=19.0 if db_tier["name"].lower() in {"pro", "premium"} else 0.0,  
    currency="USD",  
    started_at=now,  
    expires_at=expires_at,  
)
```

```
db.add(subscription)
```

```
# Автоматичне оновлення статусу профілю користувача
```

```
await crud_users.update(db=db, object={"tier_id": db_tier["id"]},  
id=current_user["id"])
```

```
await db.commit()
```

```
await db.refresh(subscription)
```

```
return {
```

```
    "message": f"Mock payment via {purchase.payment_method} completed",
```

```
    "subscription": UserTierSubscriptionRead(  
        id=subscription.id,
```

```
        user_id=subscription.user_id,
```

```
        tier_id=subscription.tier_id,
```

```
    )
```

		Олійник Д.С.			ІІЗ.КР.Б-121-26-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		92

```

        tier_name=db_tier["name"],
        status=subscription.status,
        payment_provider=subscription.payment_provider,
        payment_reference=subscription.payment_reference,
        amount=subscription.amount,
        currency=subscription.currency,
        started_at=subscription.started_at,
        expires_at=subscription.expires_at,
        created_at=subscription.created_at,
    ),
}

```

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		93

Адміністрування контенту (FastCRUD)

Лістинг В.1 Ініціалізація CRUD-інтерфейсу для роботи з курсами:

```

from fastcrud import FastCRUD
from ..models.course import Course
from ..schemas.course import CourseCreateInternal, CourseDelete, CourseRead,
CourseUpdate, CourseUpdateInternal

# Автоматична генерація методів для роботи з БД
CRUDCourse = FastCRUD[Course, CourseCreateInternal, CourseUpdate,
CourseUpdateInternal, CourseDelete, CourseRead]
crud_courses = CRUDCourse(Course)

```

Лістинг В.2 Захищені адміністративні маршрути (Створення та редагування):

```

@router.post("", response_model=CourseRead, status_code=201,
dependencies=[Depends(get_current_superuser)])
async def write_course(
    request: Request,
    course: CourseCreate,
    db: Annotated[AsyncSession, Depends(async_get_db)],
) -> dict[str, Any]:
    created_course = await crud_courses.create(
        db=db,
        object=CourseCreateInternal(**course.model_dump()),
        schema_to_select=CourseRead,
    )
    if created_course is None:
        raise NotFoundException("Failed to create course")
    return created_course

```

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		94

```

@router.patch("/{course_id}", dependencies=[Depends(get_current_superuser)])
async def patch_course(
    request: Request,
    course_id: int,
    values: CourseUpdate,
    db: Annotated[AsyncSession, Depends(async_get_db)],
) -> dict[str, str]:
    db_course = await crud_courses.get(db=db, id=course_id,
    schema_to_select=CourseRead)
    if db_course is None:
        raise NotFoundException("Course not found")

    await crud_courses.update(db=db, object=values, id=course_id)
    return {"message": "Course updated"}

```

		Олійник Д.С.			ІІЗ.КР.Б –121–26–ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

Алгоритми гейміфікації та нарахування ХР

Лістинг Г.1 Обробка спроби виконання вправи та нарахування балів:

```

@router.post("/exercises/{exercise_id}/attempts",
response_model=ExerciseAttemptRead)
async def submit_exercise_attempt(
    request: Request,
    exercise_id: int,
    values: ExerciseAttemptSubmit,
    db: Annotated[AsyncSession, Depends(async_get_db)],
    current_user: Annotated[dict, Depends(get_current_user)],
) -> ExerciseAttemptRead:
    db_exercise = await crud_exercises.get(db=db, id=exercise_id,
schema_to_select=ExerciseRead)
    if db_exercise is None:
        raise NotFoundException("Exercise not found")

    correct_answer = db_exercise.get("correct_answer")
    is_correct = bool(correct_answer and _normalize_answer(values.answer) ==
_normalize_answer(correct_answer))

    # Перевірка, чи не була ця вправа пройдена правильно раніше
    previous_correct_result = await db.execute(
        select(ExerciseAttempt.id)
        .where(ExerciseAttempt.user_id == current_user["id"])
        .where(ExerciseAttempt.exercise_id == exercise_id)
        .where(ExerciseAttempt.is_correct.is_(True))
        .limit(1)
    )
    already_completed = previous_correct_result.scalar_one_or_none() is not None

    # Нархування ХР тільки за перше правильне проходження
    awarded_xp = int(db_exercise.get("points") or 0) if is_correct and not
already_completed else 0

    attempt = ExerciseAttemptCreateInternal(
        user_id=current_user["id"],
        exercise_id=exercise_id,
        answer=values.answer,
        is_correct=is_correct,
        score=awarded_xp,
        time_spent_seconds=values.time_spent_seconds,
    )

```

		Олійник Д.С.			ПІЗ.КР.Б-121-26-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		96

```

    created_attempt = await crud_exercise_attempts.create(db=db, object=attempt,
schema_to_select=ExerciseAttemptRead)

```

```

    # Оновлення загального балансу XP користувача
    xp_balance_result = await
db.execute(select(UserXpBalance).where(UserXpBalance.user_id ==
current_user["id"]))
    xp_balance = xp_balance_result.scalar_one_or_none()

```

```

    if xp_balance is None:
        xp_balance = UserXpBalance(user_id=current_user["id"], total_xp=0)
        db.add(xp_balance)
        await db.flush()

```

```

    if awarded_xp > 0:
        xp_balance.total_xp += awarded_xp
        await db.commit()
        await db.refresh(xp_balance)

```

```

    attempt_payload =
ExerciseAttemptRead.model_validate(created_attempt).model_dump()
    return ExerciseAttemptRead(
        **attempt_payload,
        awarded_xp=awarded_xp,
        user_total_xp=xp_balance.total_xp,
    )

```

		Олійник Д.С.			ІІЗ.КР.Б-121-26-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		97