

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра інформатики

Кваліфікаційна робота

освітній ступінь – бакалавр

на тему: « **РОЗРОБКА ВЕБ-ПЛАТФОРМИ ДЛЯ ОНЛАЙН-НАВЧАННЯ**»

Виконала: студентка 4-го року
навчання,

Освітньої програми «Інженерія
програмного забезпечення», 121

Татарінова Яна Валеріївна

Керівник: Горборуков В.В.

Старший викладач, к.т.н.

Кваліфікаційна робота

захищена з оцінкою

Рецензент

(прізвище та ініціали)

Секретар ЕК

« ____ » _____ 2025р.

Київ – 2025

ЗМІСТ

<i>ЗМІСТ</i>	2
<i>КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ</i>	3
<i>АНОТАЦІЯ</i>	4
<i>ВСТУП</i>	5
<i>РОЗДІЛ 1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ОБІРУНТУВАННЯ НЕОБХІДНОСТІ РОЗРОБКИ СИСТЕМИ</i>	8
1.1 Приклади реалізації онлайн веб-платформ для навчання	8
1.1.1 Udemu	8
1.1.2 Coursera.....	10
1.1.3 Prometheus	11
1.1.4 Vseosvita.....	13
1.1.5 EdEra	14
1.2 Технології, що застосовуються в галузі	15
1.3 Формування вимог до системи на основі аналізу існуючих сервісів	17
<i>РОЗДІЛ 2. АРХІТЕКТУРНЕ ПРОЄКТУВАННЯ СИСТЕМИ</i>	20
2.1 Розробка архітектурної моделі веб-платформи	20
2.2 Вибір технологій та інструментів розробки	25
2.3 Розробка підходу інтеграції штучного інтелекту в систему	32
2.4 Архітектурні діаграми реалізованої системи	35
2.4.1 Діаграма зв'язків сутностей	35
2.4.2 Use-case diagram	37
<i>РОЗДІЛ 3. ОПИС РОЗРОБЛЕНОЇ СИСТЕМИ</i>	39
3.1 Користувацький функціонал веб-платформи	39
3.1.1 Реєстрація та початок роботи	39
3.1.2 Взаємодія студента з навчальним курсом	40
3.1.3 Взаємодія зі штучним інтелектом у системі.....	43
3.1.4 Функціонал вчителя	44
<i>ВИСНОВКИ</i>	49
<i>ДЖЕРЕЛА</i>	51

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапу	Дати виконання
1	Обрання теми на кваліфікаційну роботу	09.10.2024
2	Дослідження необхідних джерел, ресурсів для глибшого розуміння теми	15.10.2024 – 15.11.2024
3	Вивчення вже існуючих аналогів веб-платформи	17.11.2024 – 18.12.2024
4	Створення дизайну веб-платформи	19.12.2024 – 03.01.2025
5	Розробка веб-платформи	04.12.2025 – 05.05.2025
6	Створення текстової частини	10.04.2025 – 15.05.2025
7	Передзахист кваліфікаційної роботи	19.05.2025
8	Захист кваліфікаційної роботи	04.06.2025

Студентка Татарінова Яна Валеріївна

Керівник: Горборуков В.В.

« ____ » _____ 2025р.

АНОТАЦІЯ

Дана кваліфікаційна робота направлена на формування архітектурної моделі для навчальної веб-платформи та розробку програмного продукту, який забезпечує ефективний процес навчання для студентів і викладачів.

У ході роботи були дослідженні існуючі аналоги веб-платформ, для кращого розуміння недоліків і потреб користувачів та формування основних функціональних вимог для майбутнього застосунку.

Поставлене завдання було виконано з використанням React, JavaScript, хмарних технологій Firebase, та інших бібліотек для досягнення найкращої результативності. Розроблено модель рекомендаційної системи у вигляді AI-асистента, який використовує механізми контекстного формування запитів. Виконаний функціонал дозволяє студентам і викладачам зручно та ефективно взаємодіяти з платформою, отримувати персоналізовані рекомендації від AI-асистента, створювати, переглядати та проходити навчальні курси.

ВСТУП

Сьогодні онлайн навчання є як ніколи актуальною темою, все більше людей переходить на дистанційне навчання, адже це заощаджує кошти, час та є неймовірно зручним. Наприклад, згідно з дослідженням UNESCO, перехід до дистанційного навчання під час пандемії COVID-19 продемонстрував необхідність доступних та інклюзивних цифрових освітніх платформ для забезпечення безперервності навчального процесу [1]. Крім того, у деяких країнах, таких як Україна, онлайн навчання є вимушеним заходом безпеки.

Зараз на ринку існує велика кількість навчальних веб-платформ, проте користувачі часто стикаються з труднощами у їх використанні через заплутаний інтерфейс, недостатній рівень персоналізації навчання з використанням новітніх технологій, надмірну кількість функцій, проблеми з безпекою, швидкістю та інші.

Саме тому актуальним є створення простої та інтуїтивно зрозумілої веб-платформи для навчання, яка буде зручною, як для студентів, так і для викладачів і задовольнятиме всі їхні потреби. Дана платформа дозволить викладачам легко публікувати, редагувати і переглядати свої курси, а також переглядати учнів та взаємодіяти з ними для кращого розуміння їхніх потреб. У той самий час, платформа надасть учням можливість комфортно переглядати потрібні їм курси, слідкувати за своїм прогресом, звертатися до рекомендаційної системи у вигляді штучного інтелекту та спілкуватися з викладачами у разі виникнення додаткових питань.

Метою даного дослідження є проєктування та формування архітектури веб-платформи для онлайн-навчання, а також розробка програмного продукту, який забезпечить модульність, масштабованість, впровадження функцій на базі технологій штучного інтелекту, можливість подальшого функціонального

розширення, передбачається аналіз сучасних технологічних підходів до побудови подібних систем, аналіз присутніх на ринку аналогів та визначення основних вимог до освітніх онлайн-платформ, вибір оптимального стеку технологій та розробка програмного забезпечення і створення доступної веб-платформи якісної та ефективної освіти для широкого кола користувачів, зокрема у регіонах з відсутністю можливості відвідування офлайн-заняття.

Визначено три основні розділи роботи:

У першому розділі було досліджено застосунки для онлайн навчання та різноманітні інструменти для розробки, визначено критичні аспекти і сформовано функціональні вимоги до майбутнього рішення.

У другому розділі висвітлюється спроектована архітектурна модель веб-платформи з усіма необхідними компонентами, а також методи та інструменти розробки, що дозволили розробити підхід інтеграції AI-асистента в систему для забезпечення персоналізації навчання та рекомендаційної взаємодії, структурно-модульний підхід до проектування системи, сучасні методи фронтенд-розробки, зокрема такі інструменти, як React та JavaScript, а також хмарні технології Firebase для зберігання та обробки даних, що дозволили створити зручну, функціональну та ефективну веб-платформу для онлайн навчання, яка задовольняє вимоги користувачів, легко масштабується під потреби зростаючої аудиторії, має інтуїтивно-зрозумілий інтерфейс для користувачів навіть без технічної підготовки та забезпечує зручний доступ до навчальних матеріалів, інтерактивну взаємодію між викладачами й студентами та сприяє поширенню цифрової освіти для всіх охочих.

У третьому розділі описується технічні можливості та інтерфейс реалізованої конкурентоспроможної веб-платформи для дистанційного навчання для вчителів та учнів, яка забезпечує спрощений механізм публікації

курсів для викладачів, взаємодію з AI-асистентом для спрощення вибору курсів для навчання, зручний перегляд та прослідковування прогресу для студентів, сертифікати та легку взаємодію.

Під час виконання роботи, було проведено аналіз літературних джерел, статей, досліджень, а також сучасних практик у сфері веб-розробки, електронного навчання та дизайну інтерфейсу, щоб сформувавши повне бачення веб-платформи. Вивчені матеріали дозволили виділити головні засади розробки доступних, масштабованих онлайн-платформ.

РОЗДІЛ 1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ НЕОБХІДНОСТІ РОЗРОБКИ СИСТЕМИ

1.1 Приклади реалізації онлайн веб-платформ для навчання

Сьогодні, коли тема дистанційного навчання є популярною, як ніколи, у мережі Інтернет можна знайти велику кількість можливих реалізацій даного задуму. Ретельно дослідивши варіанти, їхній функціонал, структуру та особливості взаємодії з користувачами, технічні аспекти реалізації, так і вплив платформ на ефективність освітнього процесу, я виділила декілька основних платформ, які дозволили мені визначити основні підходи в даній сфері та головні інструменти, які використовуються для реалізації проектів у галузі цифрової освіти.

1.1.1 Udeemy

Першим у дослідженні був веб-сайт Udeemy [2]. Дана платформа наразі користується великою популярністю серед користувачів і є одною з наймасштабніших світових освітніх платформ, тому варто приділити їй значну увагу (рис. 1.1).

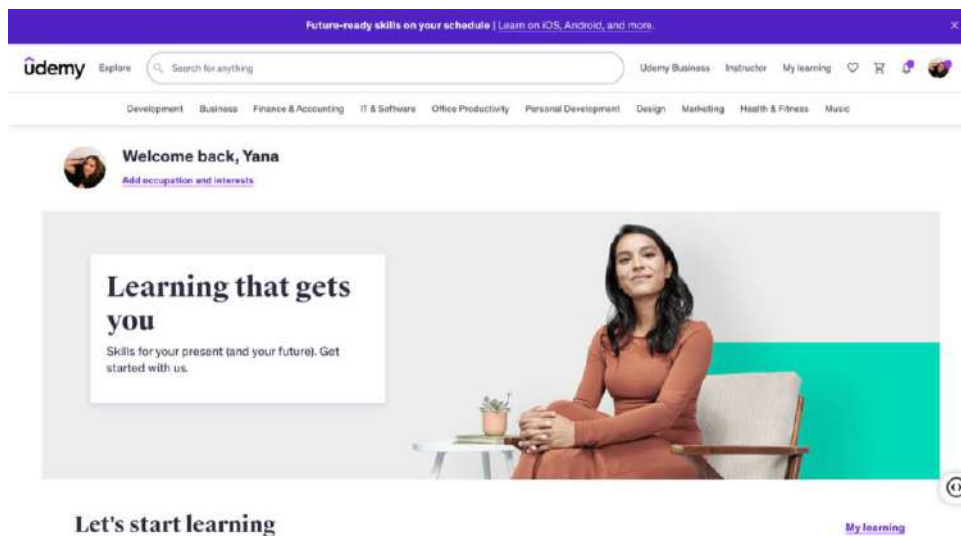


Рисунок 1.1 Скріншот головної сторінки веб-сайту Udeemy

Udeemy є прикладом ефективної реалізації платформи, яка поєднує великі обсяги навчального контенту з доступністю та простотою використання. Одною з основних переваг Udeemy є те, що вони надають можливість навчати і викладати свої курси всім охочим, що дає змогу людям поширювати і ділитись своїми знаннями не залежно від місця знаходження і професії. Також перевагою є те, що на платформі всі можуть залишати відгуки та оцінки курсам, що дає вчителям змогу краще зрозуміти, які покращення вони могли б внести до свого навчального матеріалу. Цей підхід може слугувати орієнтиром при розробці подібних рішень у сфері цифрової освіти.

Проте, звичайно, присутні й деякі недоліки, які можуть бути покращені. По-перше, велика кількість курсів на платформі є платними, що значно ускладнює доступ до знань, особливо у країнах з низьким рівнем доходу. По-друге більшість курсів є повністю асинхронними, а отже немає великої взаємодії учителя з учнем, це може стати проблемою, адже відсутність інтерактивності та живого контакту між викладачем і студентом є бар'єром для повноцінного навчального процесу. На курсах присутні лише коментарі, де

учні можуть залишити свої питання, але не завжди викладачі їх помічають та приділяють увагу для відповіді, тому це може стати на заваді розуміння того чи іншого курсу для учня і погіршує загальну взаємодію. Дослідження опубліковане в «Education and Information Technologies», наголошує, що зменшення взаємодії з викладачами та однолітками в онлайн-навчанні призводить до зниження залученості студентів та відчуття ізоляції [3]. Отже, ми бачимо, що ефективна координація від учителя є критично важливою у процесі навчання.

1.1.2 Coursera

Наступним прикладом успішної онлайн платформи для навчання, яку я взяла до уваги є Coursera [4], (рис. 1.2).

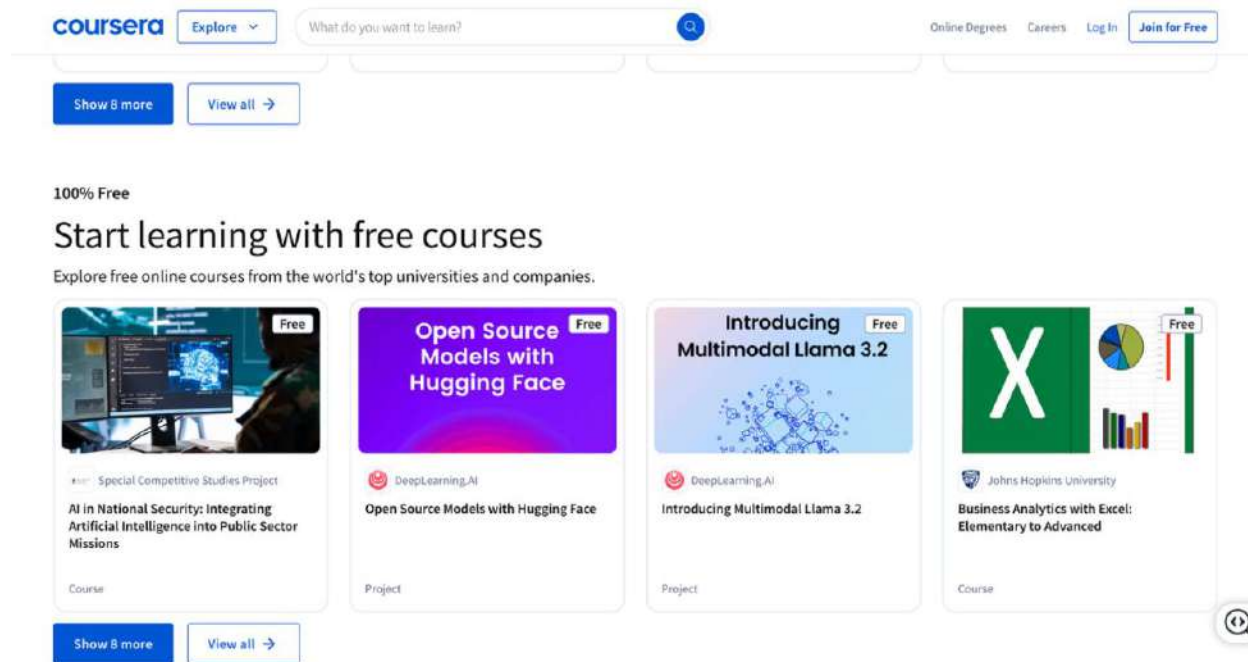


Рисунок 1.2 Скріншот головної сторінки веб-сайту Coursera

На веб-сайті Coursera, як і на Udemy, кожен охочий, може знайти предмет вивчення для себе. Coursera має низку переваг, які вирізняють її серед інших.

Перш за все це академічний підхід, адже структура курсів схожа на університетські, наявні тести і лекції. Крім того Coursera співпрацює із загальновідомими установами, такими як Stanford, Yale, Google. Це піднімає довіру користувачів, але в той же час змушує завищувати ціну. Coursera надає користувачам можливість навчатися у зручному для них режимі без встановлених дедлайнів, що дає змогу вивчати матеріал у власному темпі. Це особливо зручно для онлайн-формату навчання. Coursera використовує простий та зрозумілий дизайн, що дозволяє користувачам легко орієнтуватися на веб-сайті. Також наявна оптимізація на різних пристроях, персоналізовані рекомендації та фокус на користувацькому досвіді.

Проте, як і Udemy, Coursera пропонує обмежену кількість курсів безкоштовно, і через масштабність курси часто автоматизовані й не передбачають індивідуальної підтримки. Безкоштовні курси часто обмежені в функціоналі, наприклад відсутні сертифікати. Також деякі покращення можна внести стосовно дизайну. На головній сторінці наявна велика кількість інформації та різноманітних блоків, це може ускладнювати фокус користувача, новим користувачам може бути складно зорієнтуватися, тому краще було б залишити тільки основне, прибравши велику кількість непотрібних блоків і пропозицій. Крім того інтерфейс не завжди ефективно відображає поточний етап проходження курсу, що створює незручності для користувачів, особливо після відновлення навчання після перерви, адже часто при онлайн навчанні користувачі можуть робити паузу і поновлювати свою активність через певний час, тому така функція є критично важливою.

1.1.3 Prometheus

Ще одним не менш цікавим аналогом для вивчення стала веб-платформа Prometheus [5], що є провідним українським ресурсом з відкритими курсами.

Prometheus (рис. 1.3), пропонує своїм користувачам велику кількість безкоштовних і платних курсів.

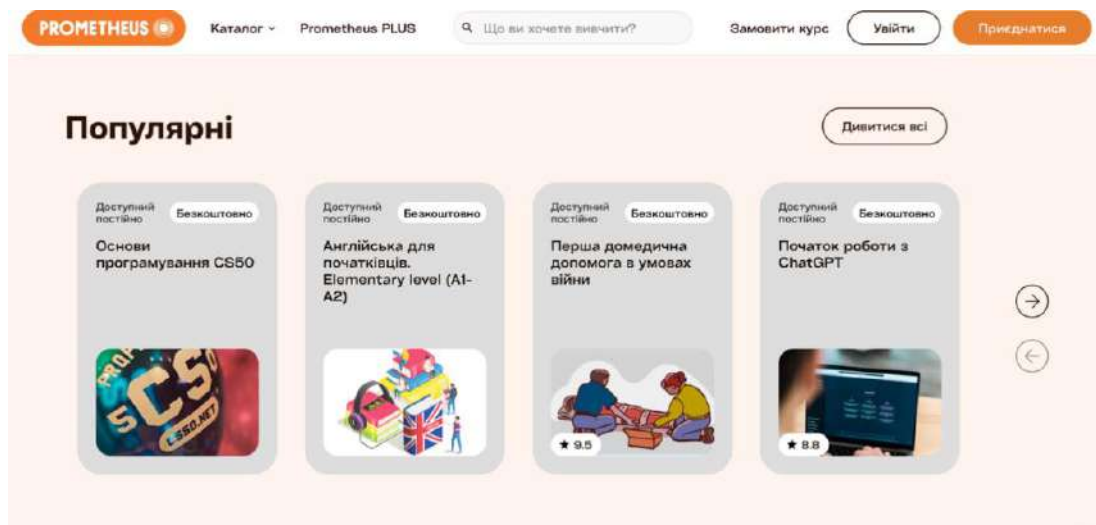


Рисунок 1.3 Скріншот головної сторінки веб-сайту Prometheus

Більшість курсів є безкоштовними і функціонал не обмежується в залежності від ціни. Після завершення курсів, користувачі Prometheus отримують іменні сертифікати, у якому зазначено всі головні деталі, що зміцнює довіру до результатів освітнього процесу та може слугувати цінним інструментом під час подання заявки на працевлаштування.

З недоліків Prometheus можна зазначити технічні обмеження та помилки, на які скаржаться немала кількість користувачів платформи. Зокрема інтерфейс застосунку хоч і зрозумілий, але візуально застарілий і не завжди інтуїтивний, наявна велика кількість кольорів і форм, що може заважати користувачу при вивченні матеріалу. Також наявні проблеми з мобільною версією веб-сайту, хоч вона і адаптована під мобільні пристрої, але не гарантує плавної навігації та якісного користувацького досвіду, який відповідає стандартам нативного застосунку. Деякі елементи інтерфейсу погано адаптовані, замалі або ж незручні для взаємодії.

1.1.4 Vseosvita

Наступним предметом для вивчення у сфері веб-платформ для навчання стала Vseosvita [6], (рис.1.4).

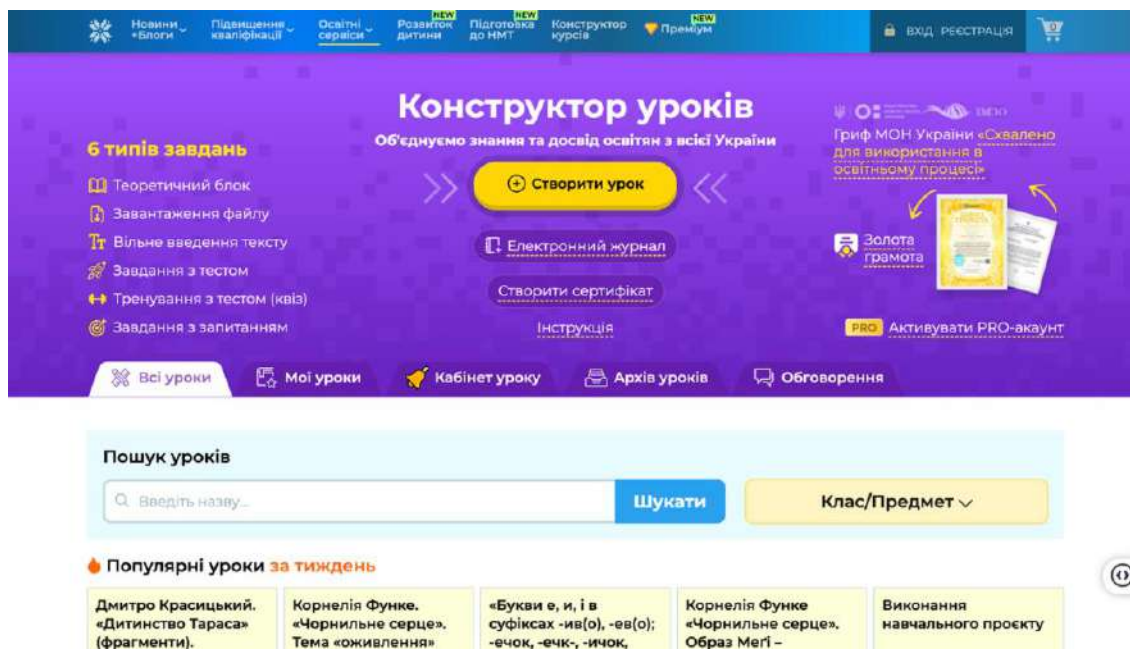


Рисунок 1.4 Скріншот головної сторінки веб-сайту Vseosvita

Ця українська платформа пропонує широкий спектр інструментів для навчання, таких як курси, тести, підвищення кваліфікації для вчителів, вебінари, конференції та інше. Ці інструменти стануть у пригоді користувачам, яким потрібна допомога у ефективному проведенню дистанційного навчання або підвищенню якості освітнього процесу, а також тим хто хоче пройти різноманітні курси для професійного зростання.

З цікавих можливостей Vseosvita можна виділити наявність методичних матеріалів та інноваційних педагогічних практик, а також онлайн вебінарів і практик, які користувачі можуть переглядати. Такі види ресурсів на веб-платформах є корисними.

Проте дизайн інтерфейсу платформи є застарілим та перевантаженим. Велика кількість елементів, випадаючих меню, банерів, кольорів та позначок, перенавантажують користувача і ускладнюють навігацію сайтом, що відразу заставляє відвідувача покинути веб-сайт. Також платформа пропонує невелику кількість інтерактивних елементів, таких як відео-лекції, у своїх онлайн курсах. Більшість інформації подається лише у вигляді тексту, що значно зменшує заохочення і мотивацію для користувачів при проходженні того чи іншого курсу. У дослідженні, яке було опубліковане в журналі «American Journal of Physics», зазначається, що студенти, які навчалися за допомогою мультимедійних відео, продемонстрували значно кращі результати як у миттєвому тестуванні, так і через два тижні після навчання, порівняно з тими, хто використовував лише текстові матеріали. Це показує, що при дистанційному навчанні подача інформації учню є одним з ключових моментів при вивченні.

1.1.5 EdEra

Наступним прикладом реалізації дистанційної платформи для навчання є EdEra [7], (рис.1.5).



Рисунок 1.5 Скріншот головної сторінки веб-сайту EdEra

Однією з явних переваг EdEra є приємний мінімалістичний дизайн, на головній сторінці наявна тільки основна інформація і немає великої кількості банерів і непотрібних кричущих елементів, а також весь веб-сайт виконаний у стриманих світлих тонах, що є зручним і легким у використанні.

З недоліків можна відмітити недостатню інформативність головної сторінки щодо функціоналу платформи та ролей користувачів. Користувач, який вперше заходить на сайт, не бачить одразу зрозумілих вказівок, що конкретно можна робити на платформі, які є формати курсів, і як зареєструватися і стати викладачем на платформі.

1.2 Технології, що застосовуються в галузі

Для ширшого розуміння та дослідження предметної області, було вирішено дослідити основні інструменти та технології, які використовують сучасні навчальні веб-платформи (табл.1.1).

Веб-платформа	Frontend	Backend	Бази даних	Хмарні сервіси	Інструменти розробки
Udemy	JavaScript, React	Python, Node.js	Redis, MySQL,	Amazon Web Services, Google Cloud Platform	GitHub, IntelliJ Idea, Docker
Coursera	JavaScript, React, Bootstrap, D3.js	Node.js, Django, Scala	MySQL, Amazon RDS	Amazon Web Services, Amazon EC2 Container Service	GitHub, Docker
Prometheus	HTML, CSS, JavaScript	Node.js, Django	MySQL, Cassandra	Amazon EC2, Amazon CloudFront	Kafka, GitHub,
Vseosvita	JavaScript, HTML, CSS	PHP, Laravel	MySQL	CloudFlare, Microsoft Azure	GitHub, IntelliJ Idea, nginx
EdEra	JavaScript, Bootstrap, Angular	PHP, TypeScript	PostgreSQL, MySQL	Amazon S3, Google Analytics UA	GitHub, nginx

Таблиця 1.1 Порівняльна таблиця використаних технологій на веб-платформах

Аналіз технологічних стеків провідних освітніх платформ засвідчує, що сучасна розробка у сфері цифрової освіти базується на використанні потужних, гнучких та масштабованих технологій як на клієнтській, так і на серверній частинах. У фронтенд розробці віддається перевага JavaScript-фреймворкам, HTML, CSS та Reac, щодо бекенду, то застосовуються

різноманітні мови і фреймворки, такі як Node.js, PHP, TypeScript, а також, як бачимо, надається перевага реляційним базам даних із додатковим використанням MySQL у випадках великих навантажень. Хмарні сервіси забезпечують необхідну інфраструктурну підтримку для високої доступності та масштабованості, багато веб-платформ використовують сервіси Amazon. Вибір даних інструментів розробки свідчить про орієнтацію на сучасні методології DevOps та Agile, що сприяють підвищенню якості і швидкості розробки. Загалом, досліджувані платформи демонструють комплексний підхід до побудови архітектури, такі як використання хмарних технологій і різноманітних фреймворків, і відповідають вимогам сучасного ринку цифрової освіти, тим самим вони підвищують безпеку, знижують помилки, спрощують масштабованість і підтримку в подальшому, що є важливим фактором при розробці повномасштабних систем на велику аудиторію.

1.3 Формування вимог до системи на основі аналізу існуючих сервісів

Отже, детально дослідивши основні навчальні веб-платформи, я однозначно сформувала такі основні підходи та функціональні рішення для врахування під час розробки власного проєкту:

- Зручний та простий у використанні дизайн інтерфейсу. Користувачі, які вперше заходять на веб-платформу не хочуть бачити велику кількість банерів, випадаючих меню та інших елементів, які присутні на деяких вищерозглянутих веб-сайтах, адже це відволікає увагу та часто може ускладнювати взаємодію для користувача. Натомість, дизайн повинен бути інтуїтивно зрозумілий, незавантажений та структурований, адже якщо користувач легко взаємодіє з платформою, це створює позитивне враження про весь продукт.

- Відеоконтент, як основний засіб подачі матеріалу. Як було зазначено раніше і як підтверджують дослідження, під час онлайн навчання учні найкраще засвоюють матеріал за допомогою відео, тому відео матеріали мають бути основним форматом контенту на веб-платформі.
- Адаптація функціоналу під роль викладача та учня. Деякі з платформ не мають чіткого розмежування між цими двома ролями, що ускладнює роботу для викладачів і спричиняє плутанину для учнів. Тому було визначено, що на веб-сайті повинні бути окремі панелі та інструменти для кожної категорії користувачів.
- Наявність взаємодії між викладачем та учнем. Вище згадані платформи не мають прямої взаємодії учня з викладачем, що значно погіршує доступність та якість навчання. На веб-платформі треба реалізувати доступ до прямої взаємодії, оскільки це сприяє кращому засвоєнню матеріалу, дозволяє учням отримувати персоналізовані пояснення та відповіді на свої питання в реальному часі.
- Використання AI-асистента для персоналізації навчання. Вище згадані платформи не використовують у своїх системах інтеграцію AI-асистента, хоч ця технологія і є зараз на піку популярності й використовується у багатьох сферах. Тому було вирішено розробити модель рекомендаційної системи на платформі та інтегрувати штучний інтелект для забезпечення персоналізації навчання та покращення якості навчання для користувача, тим самим допомогти студентам у виборі курсів для навчання з великого переліку наявних та покращити якість навчання.
- Зрозуміла навігація. Багато платформ використовують заплутані форми реєстрації. У моєму проєкті буде передбачено просту та явну

автентифікацію і чіткий маршрут користувача з моменту входу до старту навчання.

- Мобільна адаптивність. Так як онлайн навчання має передбачати можливий перегляд веб-платформи не тільки на комп'ютері, а й на мобільному пристрої та планшеті, інтерфейс буде реалізовано з урахуванням адаптивної верстки для покращення зручності користувачів платформи.
- Сертифікація після завершення курсів. Під час онлайн навчання, для додаткового заохочення необхідно реалізувати на платформі сертифікати, які користувач зможе отримати і завантажити після завершення того чи іншого курсу, адже це є важливим інструментом для поповнення професійного портфоліо.

Отже, проведений детальний аналіз існуючих навчальних веб-платформ дозволив виокремити низку суттєвих функціональних та користувацьких вимог, хоч на ринку й існує велика кількість навчальних платформ, не всі вони реалізують потреби користувачів, тому узагальнення вказаних вимог і недоліків існуючих платформ обґрунтовує потребу у розробленні архітектурної моделі та створенні функціонально-повноцінної та технологічно-інноваційної веб-платформи для дистанційного навчання із використанням штучного інтелекту для персоналізації навчання та різноманітних методів розробки, що врахують потреби різних категорій користувачів і підвищення якості навчання.

РОЗДІЛ 2. АРХІТЕКТУРНЕ ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Розробка архітектурної моделі веб-платформи

Дослідивши наявні на ринку аналоги, їхні недоліки та переваги, було визначено ключові інструменти та підходи, необхідні для ефективної реалізації власного проєкту. На основі цього було визначено розробити просту у використанні, інтуїтивно зрозумілу, сучасну веб-платформу для навчання, у якій будуть присутні реалізація взаємодії викладача з учнем, відеоматеріали як основна форма подачі контенту, наявність AI-асистента, до якого студент зможе звернутися за потреби для покращення і визначення навчальних цілей та курсів, зрозуміла навігація, мобільна адаптивність, сертифікація та інші функціональні рішення. Дана веб-платформа має стати чудовим інструментом онлайн навчання, як для учнів, так і для вчителів, передбачаючи та реалізуючи всі їхні потреби, тим самим сприяючи розвитку та популяризації якісної дистанційної освіти, що є надзвичайно актуальним у сучасному світі.

Архітектурна модель – це концептуальне представлення програмної системи, що описує її загальну структурну організацію. Дана модель повинна охоплювати всі основні компоненти програми, описуючи їх зовнішні характеристики та безпосередньо зв'язки між компонентами. Архітектура визначає фундаментальні рішення щодо організації системи, що в подальшому впливають на такі особливості, як масштабування, продуктивність та функціональність системи. Сьогодні існує багато підходів та типів архітектури програмного забезпечення, серед наймасштабніших : монолітна, сервіс-орієнтована, мікросервісна, клієнт-серверна, подійно-орієнтована архітектура, кожна з яких мають певні переваги та недоліки. З переваг монолітної

архітектури можна зазначити, простоту розгортання, адже усі компоненти будуть знаходитись в одному застосунку, з цього випливає і легкість у тестуванні. З її недоліків варто зазначити складність при збільшенні, адже код стає занадто великим і з'являється важкість у зміні елементів. Мікросервісна модель дозволяє нам масштабувати кожен сервіс окремо і використовувати в них різні технології, проте не є простою у розгортанні, ускладнює моніторинг та логування компонентів і часто спричиняє затримку. Подійно-орієнтована архітектура полягає в тому, що вона побудована навколо подій, які реагують на зміни в системі, легко додавати нові події та масштабувати, проте без надійного брокера подій, наприклад Kafka чи RabbitMQ є ризик втрати даних, а також ускладнене тестування. Сервіс-орієнтована архітектура подає нам модель, де частини це незалежно-представлені сервери. Вона вирізняється повторних використанням сервісів і гнучкістю, але в той же час може бути ускладнене управління, адже воно потребує координація між багатьма сервісами.

Клієнт-серверна архітектура з моменту свого впровадження зазнала низки вдосконалень, які сприяли покращенню співпраці та доступу до спільних ресурсів, хоч вона і містить деякі недоліки, проте вони не значно впливають на загальний огляд архітектури. Однією з ключових переваг цієї моделі є централізоване зберігання даних із можливістю віддаленого доступу до них з різних клієнтських пристроїв, незалежно від їхнього географічного розташування. Це означає, що користувачі можуть підключатися до серверних ресурсів з будь-якої точки світу. Такий підхід забезпечує високу масштабованість, адже до системи можна легко додавати нові ресурси, які доступні з будь-якого нам потрібного місця, тим самим розширюючи обчислювальні можливості архітектури та зменшуючи час обробки запитів.

Розроблена мною веб-платформа для навчання базується на класичній клієнт-серверній архітектурі, що передбачає чітке розмежування між клієнтською (фронтенд) та серверною (бекенд) частинами [8]. Фронтенд реалізований із використанням бібліотеки React, яка відповідає за відображення інтерфейсу, взаємодію з користувачем і надсилання запитів до серверної логіки. У ролі бекенду використовується хмарна платформа Firebase, що надає інфраструктуру типу BaaS (Backend as a Service). Вибір саме даного архітектурного підходу для розробки програмного забезпечення був зумовлений кількома вагомими факторами, а саме зручність і швидкість розгортання проєкту, що було забезпечене використанням інструментами хмарної платформи Firebase, масштабованість і адаптивність до змін, адже дана система відповідає модульному принципу побудови, де кожен функціональний блок (курси, AI-помічник, чат, система сертифікації, система прогресу) реалізовані як окремий логічний модуль, однак фізично всі вони працюють у рамках централізованої BaaS-інфраструктури, і, крім того, логічна модульність, яка дозволяє легко масштабувати окремі частини при зростанні навантаження або зміні вимог.

Для забезпечення масштабованості, інтерактивності та персоналізації навчального процесу було розроблено архітектурну модель веб-платформи, що поєднує переваги класичних систем управління навчанням (LMS) та в той же час базується на сучасних веб-технологіях та хмарних сервісах, які забезпечують не лише базовий функціонал доступу до навчального контенту, але й підтримку персоналізованого навчання, аналітики освітньої активності користувачів, і, що не менше важливо, інтерактивну взаємодію через вбудованого AI-асистента. Система управління навчанням (LMS) – це програмне забезпечення, що функціонує на базі веб-серверів, хмарних

обчислень або локальних комп'ютерів і забезпечує організацію навчального процесу в академічному чи неакадемічному середовищі без обмежень у часі та просторі [9]. Завдяки можливості доступу через веб-браузер або мобільний застосунок, більшість таких систем є доступними практично з будь-якого місця.

Оснoву розробленої архітектури складає модульний підхід побудови, що забезпечує гнучкість і можливість розширення функціоналу у майбутньому без порушення стійкості системи. Архітектура системи реалізована за принципом клієнт-серверної моделі з використанням сервісів Firebase, зокрема Firestore та Realtime Database, а також інтеграцією штучного інтелекту для рекомендацій користувача. Архітектура веб-платформи містить декілька основних компонентів, які складають її сутність (рис. 2.1) :

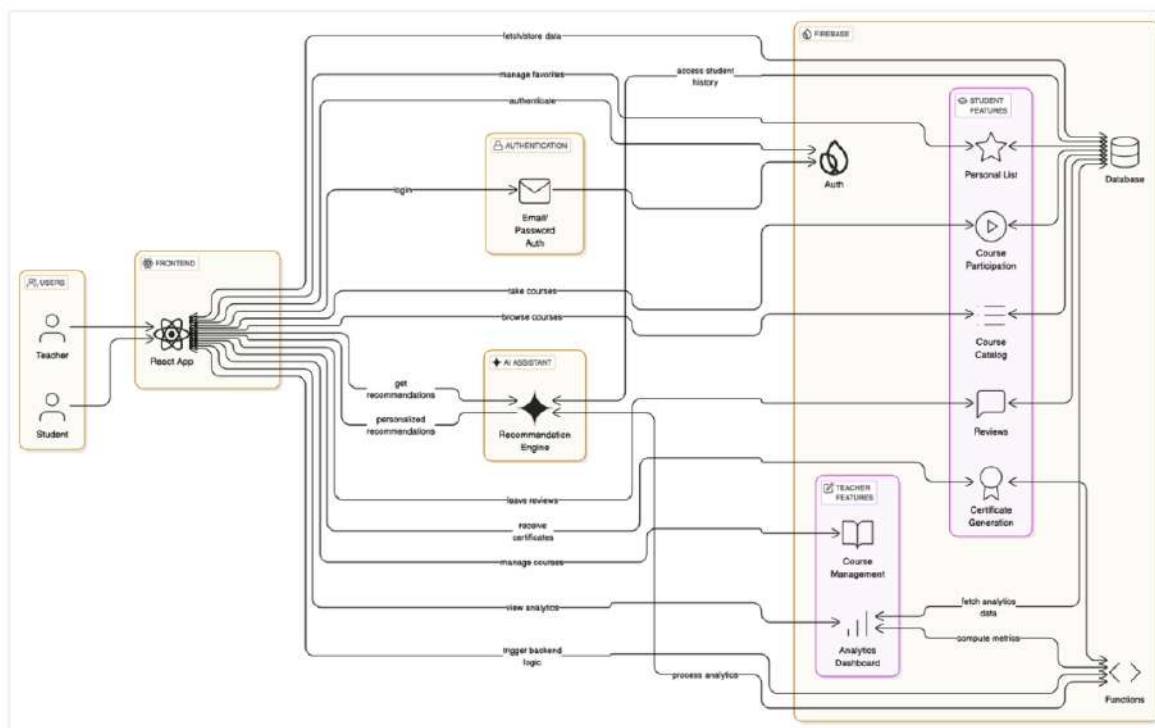


Рисунок 2. 1 Архітектура веб-платформи

1. Клієнтська частина застосунку реалізована з використанням фреймворку React. Вона забезпечує взаємодію користувача з платформою, динамічне відображення навчального контенту, управління курсами та комунікацію через чат. Для маршрутизації та захисту маршрутів використано React Router та механізми аутентифікації Firebase. React забезпечує неперервний досвід користувача, адаптивність та можливість розширення.
2. Серверна частина реалізована з використанням хмарної платформи Firebase, яка забезпечує аутентифікацію користувачів, зберігання та обробку даних у Firestore, а також реальну часову синхронізацію повідомлень у Realtime Database.
3. Модуль рекомендацій на основі штучного інтелекту (AI-асистент) функціонує як допоміжний сервіс на веб-платформі, який обробляє збережені промти (запити користувача) та формує індивідуальні рекомендації щодо вибору курсів, ґрунтуючись на попередньому досвіді навчання, пройдених курсів та наявних курсів на платформі, а також зазначених уподобаннях користувача та його цілей.
4. База даних представлена двома основними компонентами, а саме Cloud Firestore , що відповідає за збереження даних щодо відгуків, курсів з усіма параметрами, пройдені курси певного користувача, його сертифікати за завершення навчання, розділи, які викладач додає створюючи новий курс з описами до уроків та відеоматеріали. Крім того Firestore взаємодіє з AI-асистентом, зберігаючи інформацію з запитом та відповіддю на певне питання та надаючи йому список пройдених користувачем курсів та всіх наявних на платформі курсів Realtime Database використовується для чату між викладачем та студентом,

забезпечуючи миттєву синхронізацію повідомлень та високий рівень взаємодії.

2.2 Вибір технологій та інструментів розробки

Відповідно до поставленого завдання були визначені основні інструменти, які в подальшому були використані для розробки веб-платформи. Під час створення такого складного високонавантаженого застосунку, як веб-платформа для навчання, надзвичайно важливо спочатку продумати його архітектуру, оскільки саме вона значною мірою визначатиме рівень продуктивності системи.

Для середовища розробки було вирішено використовувати IntelliJ IDEA, адже це потужний інтелектуальний редактор для коду, що містить зручну навігацію та інтеграцію з фреймворками, що є критично важливим при розробці повноцінного продукту.

Для реалізації клієнтської частини було використано React. Даний фреймворк є максимально доцільною при розробці освітньої веб-платформи. React використовує компонентно-орієнтовану структуру, що дозволяє реалізовувати частини інтерфейсу у виглядів окремих компонентів (картки курсів, кнопки, сторінки), завдяки цьому можна легко використовувати код повторно та зменшити кількість написання дублікатів коду для тих самих функцій [11]. Повторне використання коду – це принцип проєктування програмних систем, який передбачає інтеграцію вже існуючих модулів, компонентів або фрагментів коду в нові програмні рішення. Такий підхід спрямований на підвищення ефективності розробки, зменшення обсягів ручного кодування та мінімізацію ймовірності помилок за рахунок використання перевірених рішень, саме цей підхід дозволяє нам використовувати React. Таким чином у своєму проєкті я буду використовувати,

наприклад, компонент `StudentLessonCard` для відображення курсів, як на головній сторінці, так і на сторінці вибраних курсів користувача (рис. 2.2). Зміни в одному елементі не будуть впливати на інші, що значно спрощує роботу і дозволяє нам розширювати додаток за потреби без порушення усієї структури та виникнення проблем.

```

return (
  <Slider {...settings}>
    {courses.map((course, index) => (
      <StudentLessonCard index={index} course={course} onClick={() => onCourseClick(course)} />
    ))}
  </Slider>
);

```

```

<div className="reading" id="reading-100000">
  {certificates.map((certificate, index, array) => {
    <div className="card" key={index}>
      <StudentLessonCard
        index={index}
        course={certificate}
        onClick={() =>
          navigate(`/course/${certificate.id}`)
        }
      />
    }
  )}
</div>

```

Рисунок 2. 2 Приклади використання одного компонента `StudentLessonCard` повторно

Також React працює з віртуальним DOM (Document Object Model) [12], що дозволить нам оновлювати лише ті частини сторінки, які реально змінилися. Функція `ReactDOM.render()` керує вмістом переданого вузла-контейнера, це означає, що саме під час першого виклику вона замінює всі наявні дочірні елементи всередині контейнера. А вже потім при його наступних викликах використовується алгоритм порівняння DOM, який дозволяє оновлювати лише змінені частини для підвищення ефективності та зменшення витрат. Також варто зазначити, що сам контейнер не змінюється, проте змінюється лише його вміст. При роботі з даними, такими як списки курсів чи учнів, технологія Document Object Model забезпечить нам бажану продуктивність при відображенні та оновленні інтерфейсу. А також усі інтерактивні елементи, такі як «Зарахувати на курс» будуть відображатися швидко і без затримок, через локальне оновлення.

Крім того у проєкті буде використано React Router (рис.2.3), що дозволить нам створювати Single Page Application [13], що забезпечить безперервність досвіду користувача, адже завдяки SPA наш застосунок буде відображатися плавно, без затримок, і у будь-який момент, оскільки він показує

користувачеві інший контент без оновлення сторінки веб-сайту у його браузері, що зробить застосунок максимально зручним для користувача.

```

<AppliedCourseProvider>
  <FavoritesProvider>
    <div>
      <Routes>
        <Route path="/" element={<Layout />} />
        <Route path="/login" element={<Login />} />
        {/} <Route path="/register" element={<Register />} />{/}
        <Route path="/password-reset" element={<ResetPswd />} /> {/}
        <Route path="*" element={<NotFound404 />} />
      </Routes>
      <Route element={<ProtectedRoute />} />
      {/} Student routes {/}
      <Route path="/" element={<StudentLayout />} />
      <Route index element={<LandingPage />} />
      <Route path="/student_courses" element={<StudentCourses />} />
      <Route path="/student_courses/:id" element={<ViewCoursePage />} />
      <Route path="/account" element={<AccountPage />} />
      <Route path="/course_details/:courseId" element={<CourseDetailsPage />} />
      <Route path="/student_chat" element={<StudentChat />} />
    </Route>
    {/} Teacher routes {/}
    <Route path="/" element={<LayoutContainer />} />
    <Route path="/dashboard" element={<Dashboard />} />
    <Route path="/courses" element={<Courses />} />
    <Route path="/users" element={<Users />} />
    <Route path="/course/new" element={<AddNewCourse />} />
    <Route path="/course/edit/:courseId" element={<AddNewCourse />} />
    <Route path="/chat" element={<Chat />} />
  </Route>
</Route>
</Routes>
</div>
</AppliedCourseProvider>

```

Рисунок 2. 3 Основний файл програми App.js

Було оформлено та реалізовано компонентну структуру та використано React Context API (AppliedCourseProvider, FavoritesProvider, ColorModeContext) [14], що сприяє поширенню даних через ієрархію компонентів, дозволяючи компонентам отримувати доступ до спільного стану без посередників та використовує застосування концепції управління глобальним станом із мінімальною кількістю пропсів та дозволяє підтримувати масштабовану та слабо зв'язану структуру, що є одним із принципів сучасного архітектурного дизайну у веб-розробці. Щоб додати дані до контексту та зробити їх доступними для дочірніх компонентів, використовується Provider, доступний в екземплярі Context. Provider – це компонент, який приймає одну властивість з назвою value, де вказані дані для

контексту, які можуть бути змінними, функціями або об'єктами. Передача функцій зворотного виклику дочірнім компонентам за допомогою Context API дозволяє дочірньому компоненту змінювати стан батьківського компонента.

React Router створює чітку схему маршрутів в програмі, що допомагає легко орієнтуватися в коді, робить його більш читабельним та спрощує масштабування проєкту. Також це дозволить нам зручно організувати окремі інтерфейси студента та викладача, визначивши чіткі компонування Layout для кожної групи сторінок. Також React Router дозволить користувачам без зайвих зусиль повертатися до сторінок, що були відвідані раніше. А динамічна маршрутизація в проєкті дозволить формувати універсальні маршрути, які адаптуються до конкретних запитів, дозволяючи, наприклад, відображати інформацію про будь-який курс або користувача без створення окремого маршруту для кожного з них [15]. Більш того, я використовую ProtectedRoute, так званий захищений маршрут для відображення окремих сторінок, таких як список курсів, лише для авторизованих користувачів, тим самим забезпечивши більший контроль безпеки та доступу до контенту веб-платформи.

Для зручності роботи з динамічними даними, використовую state management (рис. 2.4), а саме так звані React Hooks: useState, useEffect, useContext.

```
function Courses() {  
  const [courses, setCourses] = useState( initialState: [] );  
  const { fetchAllCourses, deleteCourse } = useTeacherCourse();  
  const [openDialog, setOpenDialog] = useState( initialState: false );  
  const [selectedCourseId, setSelectedCourseId] = useState( initialState: null );  
  const navigate = useNavigate();  
}
```

Рисунок 2.4 Приклад використання useState в файлі проєкту Courses

React Hooks – це так звані функції, які дозволяють нам «під'єднувати» можливості React, такі як стан, життєвий цикл та такі інші, до функціональних компонентів, які раніше не могли цього робити, тобто вони не працюють в середині класів, проте дають нам змогу використовувати React, якщо це потрібно, без класів. Це дає нам декілька суттєвих переваг, як от спрощення і краще читабельність коду, спрощення повторного використання коду та можливість створення власних хуків за потреби.

У функціональних компонентах React стан не визначено за замовчуванням. Для керування станом використовується особливий хук `useState`, який дозволяє нам зберігати та оновлювати стан протягом усього життєвого циклу компонента. Цей хук може бути викликаний кілька разів у межах одного компонента для створення незалежних фрагментів стану. Крім того, для управління глобальним станом або передавання даних між компонентами використовується за допомогою хуку `useContext`. Ці інструменти забезпечать просте та локальне керування станом, продуктивність, адже тільки обхідні компоненти будуть реагувати на зміни стану та легке відстежування зміни стану в наших компонентах [16].

React також забезпечує легку інтеграцію з різноманітними корисними у використанні бібліотеками та базою даних Firebase, що я буду використовувати в подальшому.

Ще одним ключовим інструментом під час розробки є JavaScript, динамічна, об'єктно орієнтована мова програмування, яка дозволяє створювати інтерактивні, динамічні та зручні у використанні веб інтерфейси та має велику кількість переваг, таких як швидкість, розширена функціональність, простота використання та універсальність. JavaScript допомагає з логікою фронтенду (рис. 2.5). Використовується для написання

логіки взаємодії на клієнтському боці, таких елементів, як обробки подій (клік, введення тексту), оновлення контенту сторінок. У проєкті я використовую сучасний JavaScript-синтаксис, що дозволяє зробити код більш читабельним, коротким та легшим для підтримки. А також за допомогою JavaScript здійснюється збереження даних у `localStorage`, що дозволяє сесії користувача утримуватись та зберігатись і після перезавантаження сторінки. Синхронно з React дозволяє створити розширений та сучасний веб-застосунок.

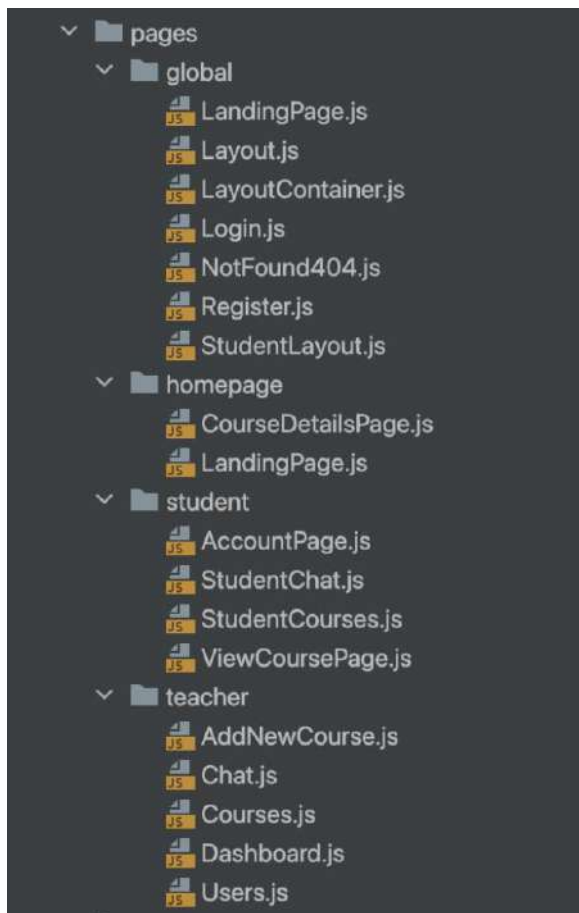


Рисунок 2. 5 Структура сторінок веб-платформи

Наступним не менш важливим елементом при розробці веб-сайту є база даних, її вибір також є важливим, особливо при наявності великої кількості різних даних для зберігання, як у нашому випадку курсів, студентів та прогресу користувача. Я використовую у своєму проєкті базу даних Firebase

Firestore [17] для зберігання всіх потрібних даних. На мою думку, дане рішення є ефективним при розробці веб-платформи для навчання у зв'язку з кількома вагомими факторами. По-перше, Firebase добре поєднується з React і дозволяє створити зручний інструмент аутентифікації в застосунку. Firebase Authentication, що також входить до системи мікросервісних інструментів Google Firebase, дозволить нам використати всі бажані методи реєстрації, створювати безпечні функції реєстрації та входу для користувачів без необхідності починати з нуля або налаштовувати сервер. Firebase Authentication також підтримує сучасні протоколи безпеки, такі як JWT(JSON Web Token) та OAuth 2.0, що забезпечує безпечне зберігання облікових даних користувачів веб-платформи. По-друге, Firebase має в доступі Realtime Database, що забезпечує автоматичне оновлення даних на наших сторінках застосунку, це є важливим у даному проєкті, адже у нас наявний чат з повідомленнями, що потребує швидких змін за потреби. Realtime Database забезпечує миттєву синхронізацію повідомлень між усіма користувачами в режимі реального часу, що є критично важливим для реалізації інтерактивної комунікації між вчителями і студентами. Крім того, Firebase надає нам хмарну базу даних із моментальним підключенням без налаштування серверу зі всіма потрібними елементами бази даних, що значно спростить і пришвидшить розробку, адже не потрібно буде налаштовувати сервер вручну, а натомість використовувати вже готові SDK.

З корисних бібліотек для забезпечення структурованого та сучасного інтерфейсу користувача, було використано бібліотеку компонентів для React – MUI (mui/material) [18]. Як було зазначено раніше, однією з цілей є досягти гарного у вигляді та зручного у використанні інтерфейсу, адже це одне з найголовніших вимог користувача. З цим допомагає бібліотека MUI, яка

значно пришвидшує розробку інтерфейсу, забезпечує адаптивний дизайн, підтримку світлих і темних тем застосунку, безпроблемне розширення та відповідає сучасним стандартам зручності та доступності [19].

Крім того, з допоміжних бібліотек для якісного збереження PDF файлів з сторінки, а саме сертифікатів після проходження курсів використовуються бібліотеки `jspdf` та `html2canvas`. Для створення слайдів з курсами використовується `React Slick`, перевагою є те, що `React Slick` забезпечує правильне відображення на різних пристроях. А також зручну бібліотеку `Yup` для створення схем перевірки полів при реєстрації та аутентифікації.

2.3 Розробка підходу інтеграції штучного інтелекту в систему

Для забезпечення персоналізованості на веб-платформі, було вирішено інтегрувати штучний інтелект Gemini, як AI-асистент для реалізації рекомендаційної системи. У наш час технології швидко розвиваються і використання штучного інтелекту на онлайн платформах стає дедалі більш популярним і ефективно допомагає покращити та вдосконалити якість навчання, особливо в онлайн форматі [20], проте саме платформи для навчання ще не в повній мірі використовують потенціал цієї технології і тому ця ніша в навчальних платформах залишається незаповненою. Саме тому, було вирішено розробити модель рекомендаційної системи у вигляді AI-асистента на платформі з використанням штучного інтелекту під назвою Gemini.

Gemini – це чат бот зі штучним інтелектом від Google та використовує версію моделі 1.0 Pro, що вийшов у світ у 2024 році, тому технологія є явно новітньою. За заявами представників Google, модель штучного інтелекту Gemini є проривною розробкою, яка, за їхніми оцінками, здатна перевершити як GPT-4 від OpenAI, так і фахівців-людей. Її унікальність ґрунтується на двох

ключових характеристиках – це глибока мультимодальність та здатність до природної взаємодії [21].

Лише деякі з досліджених платформ використовують персоналізовані рекомендації, проте не реалізують спілкування для користувача на пряму, що значно погіршує якість персоналізованого навчання онлайн. На моїй платформі студент зможе прямо спілкуватися з AI-асистентом, вибравши одне з питань із запропонованого переліку, що зменшить час на формулювання питання, адже студенту буде достатньо просто вибрати вже готовий варіант. Також він повинен буде вказати свою ціль, рівень професіоналізму і сферу в якій хоче розвиватися. Після цього AI зробить запит до бази даних щодо вже пройдених даним користувачем курсів і наявних курсів на платформі і надасть користувачу відповідь у вигляді переліку можливих курсів для вивчення на основі всіх вище згаданих параметрів. Модель, що була реалізована, передбачає трикомпонентну взаємодію: користувач – AI-асистент – база даних. Після отримання запиту (через вибір шаблонного питання) AI-модуль виконує семантичний аналіз вхідного повідомлення, формує структурований запит до бази даних, інтерпретує відповідь і генерує адаптовану відповідь. Взаємодія відбувається в реальному часі, що забезпечує оперативність та персоналізацію відповіді (рис.2.5).

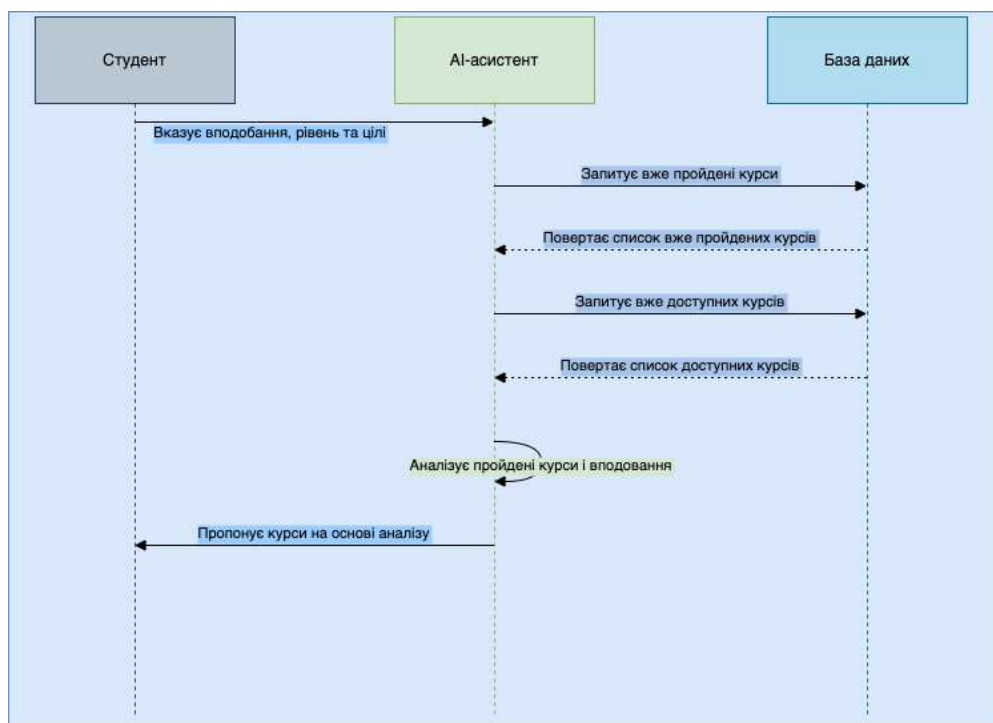


Рисунок 2. 5 Модель взаємодії AI-асистента на платформі

Використання штучного інтелекту на освітній платформі дозволяє значно підвищити якість навчального процесу завдяки персоналізованим рекомендаціям, швидкому доступу до знань та адаптивній підтримці користувача. Незважаючи на складність технології, її інтеграція виправдана з погляду ефективності та зручності для студентів. ШІ дозволяє адаптувати навчальний процес до індивідуальних потреб кожного студента [22].

У рамках реалізації персоналізованого навчального середовища на базі веб-платформи було впроваджено елементи механізму генерації природною мовою (Natural Language Generation) із використанням технологій штучного інтелекту. Основна мета це підвищення якості взаємодії користувача з системою через інтелектуального асистента, який адаптує рекомендації навчального контенту відповідно до індивідуальних цілей, уподобань і прогресу користувача.

Цей механізм базується на підходах prompt engineering [23] , тобто формування запитів до мовної моделі в структурованій і контекстно-залежній формі. Зокрема, при взаємодії з AI-асистентом система:

- аналізує пройдені користувачем курси, що збережені в базі даних
- враховує введені параметри: освітню мету, обрану галузь навчання та бажаний рівень складності;
- формує запит (prompt) у вигляді синтаксично правильного тексту природною мовою з включенням ключових параметрів;
- надсилає його до мовної моделі, яка генерує релевантні рекомендації щодо наступних курсів;
- відображає відповідь у зручному для користувача форматі, у вигляді переліку курсів, які користувач може знайти на платформі та додати до списку проходження.

Збереження промтів та відповідей AI здійснюється у Firebase Firestore.

2.4 Архітектурні діаграми реалізованої системи

2.4.1 Діаграма зв'язків сутностей

Для виразнішої реалізації зв'язків сутностей на платформі SkillLab, пропонуємо розглянути Entity-Relationship diagram, яка наочно демонструє та висвітлює взаємозв'язки сутностей в базі даних (рис. 2.6).

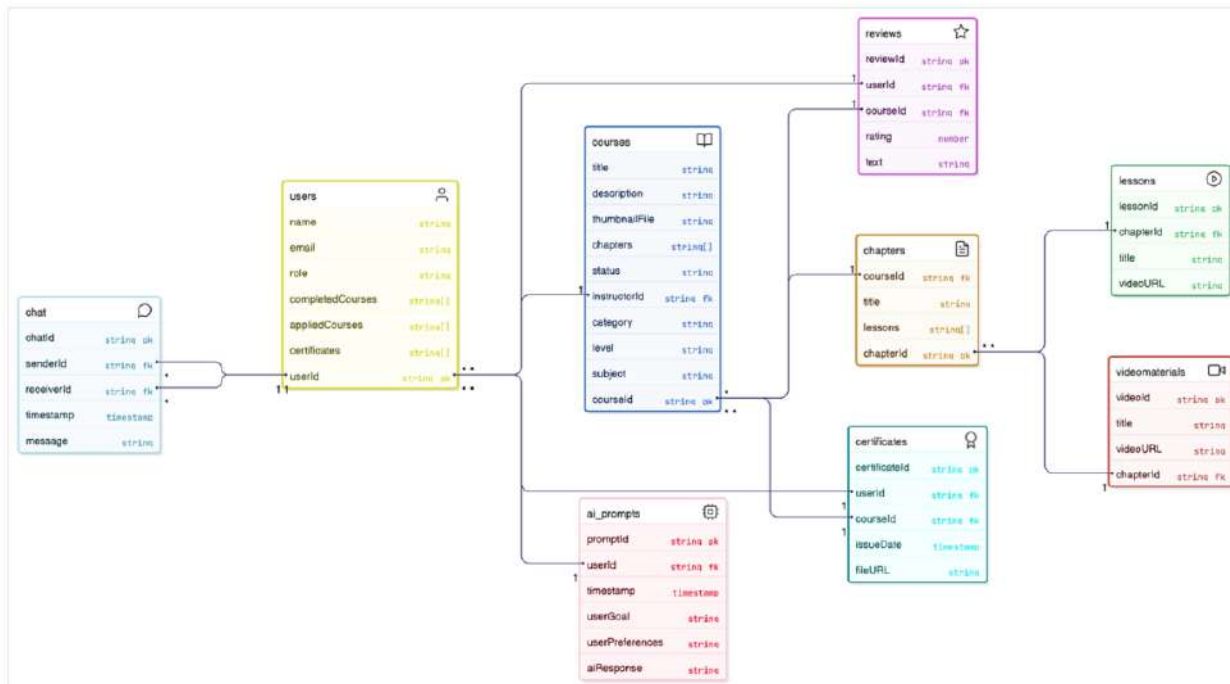


Рисунок 2.6 Діаграма зв'язків сутностей веб-платформи

Основною сутністю є користувач, він має поля ім'я, прізвище, роль, пройдені курси, отримані сертифікати, та id. Для кожного користувача в базі даних зберігається список вже отриманих ним сертифікатів, та відгуків, які він залишив до певних курсів на платформі. Також наявна сутність курси, що має в собі такі поля як, назва, опис, зображення, статус, id інструктора, рівень, сфера та id. Для кожного курсу зберігається список розділів, які викладач повинен додати при створенні курсу, а також список самих уроків з id, назвою та id розділу. Крім того зберігаються самі відео матеріали, які присутні у кожному розділі курсу. Для збереження повідомлень із чату між викладачем та студентом використовується Realtime Database, що забезпечує моментальне обмінювання повідомленнями для кращої взаємодії. Також в базі даних зберігаються так звані ai промти, тобто запити користувачів до AI-асистента та його відповіді.

2.4.2 Use-case diagram

Для кращого розуміння дії платформи SkillLab з різних сторін користувачів, пропонуємо розглянути Use-case diagram (діаграма прецедентів або варіантів використання), що показує, які дії можуть виконувати користувачі на платформі (рис. 2.7).

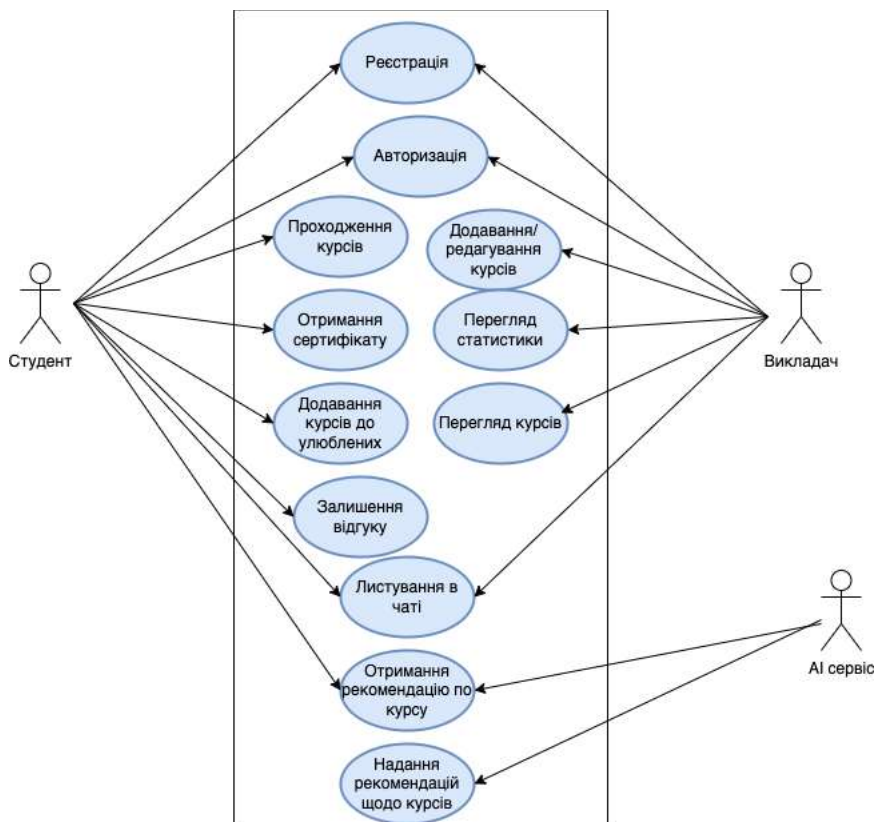


Рисунок 2.7 Use-case diagram веб-платформи

На даній діаграмі зображено три основних «учасника» веб-платформи, а саме студент, викладач і AI-сервер і детально показано дії, які вони можуть виконувати та як саме вони взаємодіють один з одним, таке представлення краще відображає можливі дії на платформі. Обоє студент та викладач можуть реєструватися та авторизуватися на платформі, а також обмінюватися повідомленнями один з одним в чаті для кращої взаємодії.

Викладач має змогу створити новий курс, добавивши всі необхідні параметри, а також переглядати, видаляти та редагувати його. Також він має доступ до перегляду статистики щодо своїх курсів, скільки людей пройшло, скільки годин прослухано.

Студент має змогу переглядати всі доступні на веб-платформі курси. Також добавляти курси в перелік своїх курсів або до улюблених і проходити курси. Після завершення проходження курсу, студент має змогу переглянути та завантажити сертифікат, а також залишити відгук і рецензію на курс для інших користувачів.

AI-сервіс, що реалізований на платформі, взаємодіє з користувачем, а саме студентом. Студент надає такі параметри, як ціль, сфера вивчення і рівень складності, після чого AI-сервіс на основі цього, пройдених користувачем курсів і вже наявних на платформі курсів, надає користувачеві відповідь у вигляді списку курсів, які він може пройти за бажанням.

РОЗДІЛ 3. ОПИС РОЗРОБЛЕНОЇ СИСТЕМИ

3.1 Користувацький функціонал веб-платформи

3.1.1 Реєстрація та початок роботи

На початку користувач повинен зареєструватися на веб-платформі, або ввійти, якщо користувач вже має аккаунт, ввівши свій email та пароль (рис. 3.1).

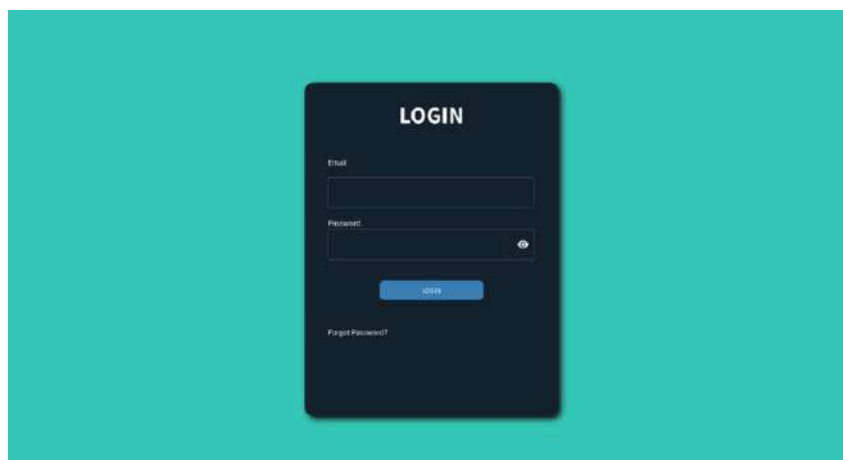


Рисунок 3.1 Сторінка реєстрації веб-платформи

Після входу в систему, користувач потрапляє на головну сторінку застосунку (рис. 3.2), де він може переглядати інформацію про платформу, всі наявні на ній курси та відгуки користувачів.

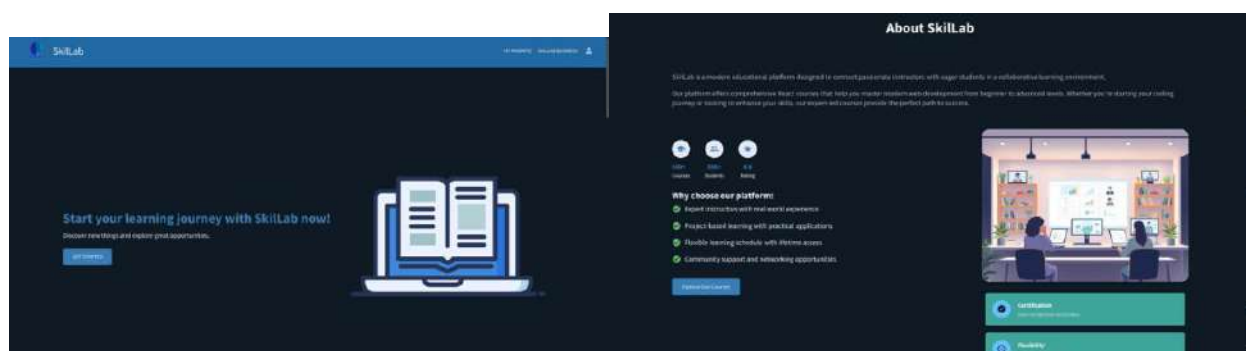


Рисунок 3.2 Головна сторінка веб-платформи

У правому верхньому куті веб-сайту користувач може переглядати панель користувача з основним функціоналом, а саме панель для вчителя, акаунт, обрані курси, повідомлення, AI-асистент та вихід (рис. 3.3).

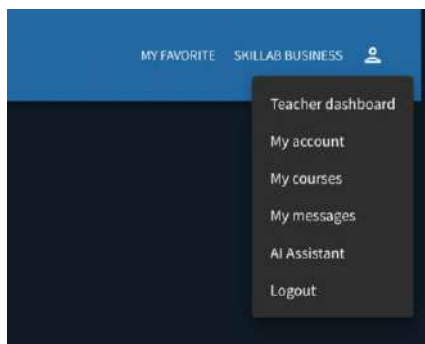


Рисунок 3.3 Панель користувача

3.1.2 Взаємодія студента з навчальним курсом

Для того, щоб добавитись на курс, студент повинен перейти до каруселі курсів на головній і обравши курс, натиснути на нього, тим самим перейшовши до сторінки даного курсу (рис. 3.4).

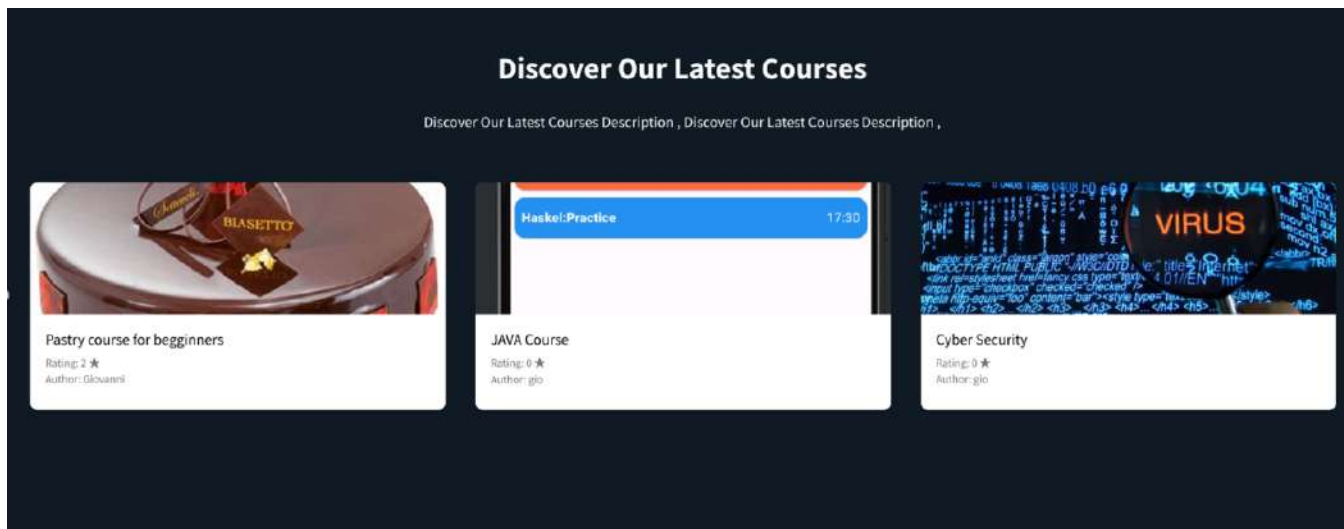


Рисунок 3.4 Карусель курсів на головній сторінці

Щоб додати курс до своїх курсів, користувач повинен натиснути кнопку «Apply now», а щоб додати курс до улюблених - «Add to favorites» (рис. 3.5)

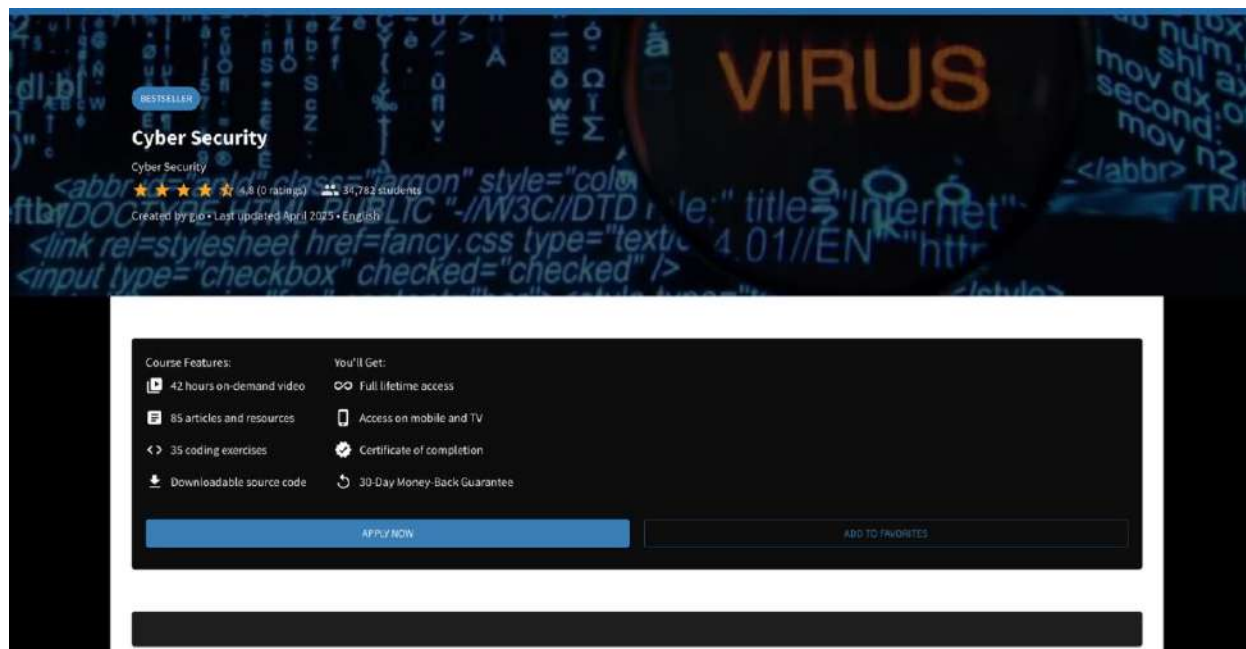


Рисунок 3.5 Сторінка курсу

Після цього користувач може переглядати і проходити курс, який він додав, перейшовши на панелі управління до «My courses», там користувач може побачити його курси, які він додав вже раніше, улюблені курси, які він хоче пройти пізніше та сертифікати уже пройдених курсів. Щоб почати проходження курсу, студент повинен натиснути на потрібний курс (рис. 3.6).

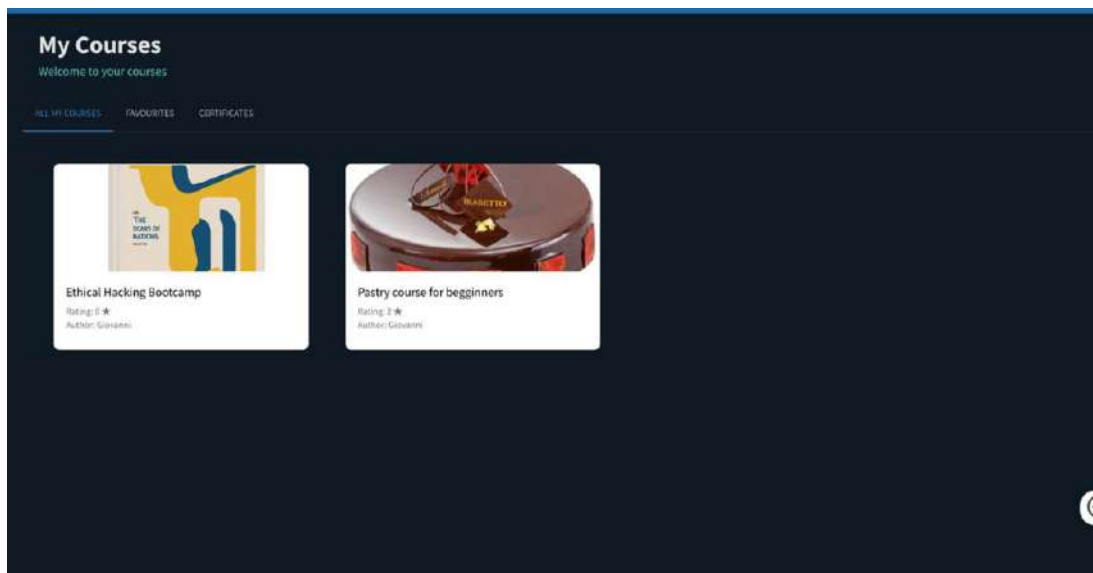


Рисунок 3.6 Сторінка вибраних користувач курсів

На сторінці курсу, студент може побачити відео матеріали до курсу, уроки, опис від викладача та рев'ю. А також він може залишати нотатки під час перегляду відео та залишати оцінку курсу під відео (рис. 3.7).

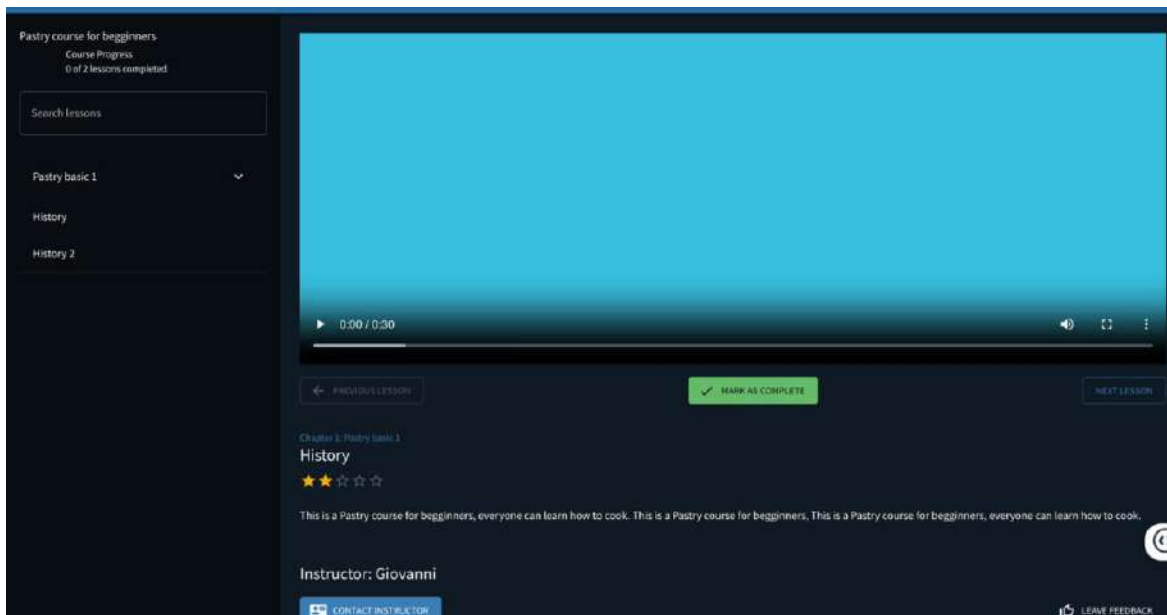


Рисунок 3.7 Сторінка курсу

Для того, щоб переглянути або завантажити певний сертифікат, студент має перейти до вкладки сертифікати і тоді він зможе обрати потрібний сертифікат для подальших дій (рис. 3.8).



Рисунок 3.8 Сторінка сертифікату після завершення курсу

3.1.3 Взаємодія зі штучним інтелектом у системі

Якщо студент захоче звернутися до AI-асистента, він повинен перейти до вкладки AI-assistant на панелі управління. Після чого студент має ввести потрібні параметри, а саме рівень професіоналізму, сфера та предмет навчання, а також свою навчальну ціль (наприклад, стати програмістом). Далі, користувач може вибрати один із запропонованих для нього запитів і надіслати його до AI-асистента. Після обробки запиту та уроків з бази даних, AI-асистент надасть користувачу список курсів, які він може пройти на платформі для досягнення своїх цілей (рис. 3.9).

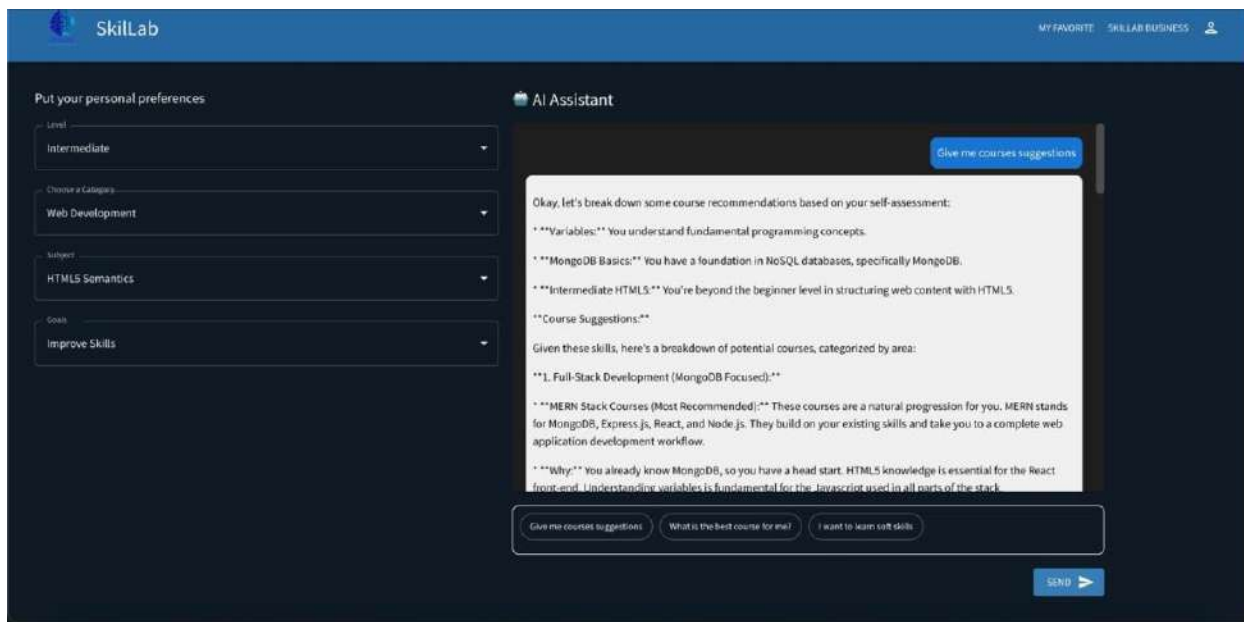


Рисунок 3.9 Сторінка взаємодії з AI-асистентом

3.1.4 Функціонал вчителя

Для переходу у режим вчителя, користувач повинен натиснути на «Teacher dashboard» на головній панелі. Після цього користувач опиниться на панелі вчителя зі всіма необхідними функціями (рис. 3.10).

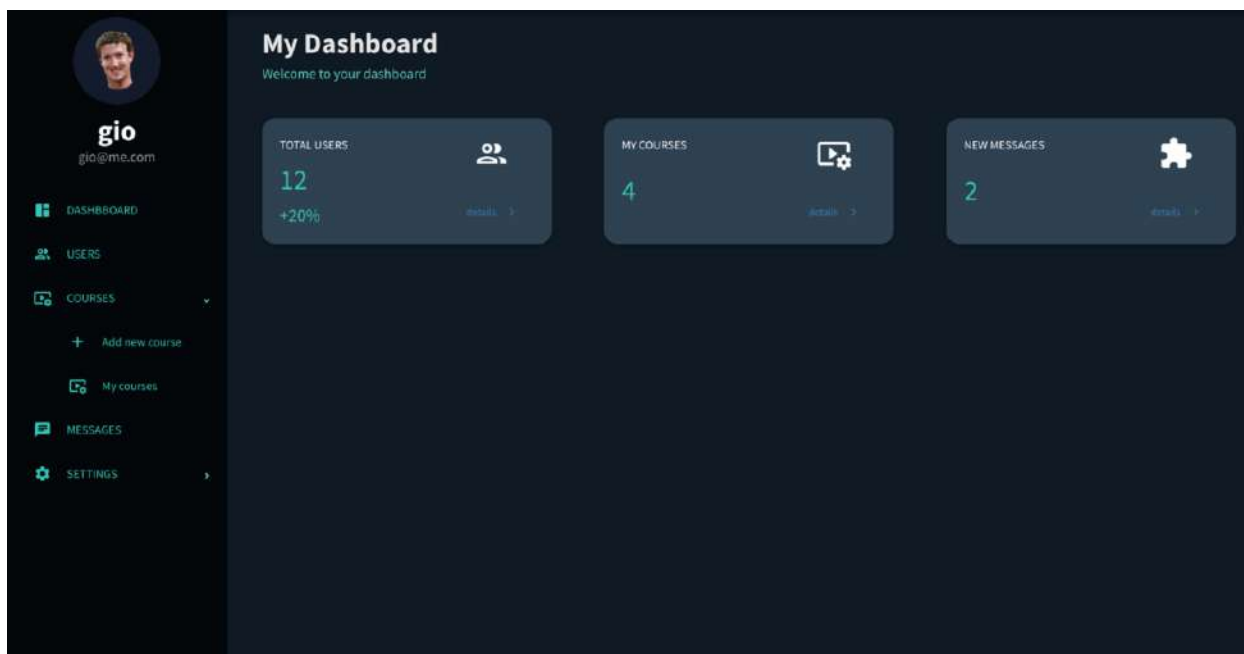


Рисунок 3.10 Панель учителя

Для того, щоб додати новий курс до вже наявних, вчителю необхідно натиснути «Add new course», після чого додати всі необхідні параметри, опис, назви уроків та розділів і додати відеоматеріали, після чого натиснути «Save» (рис. 3.11).

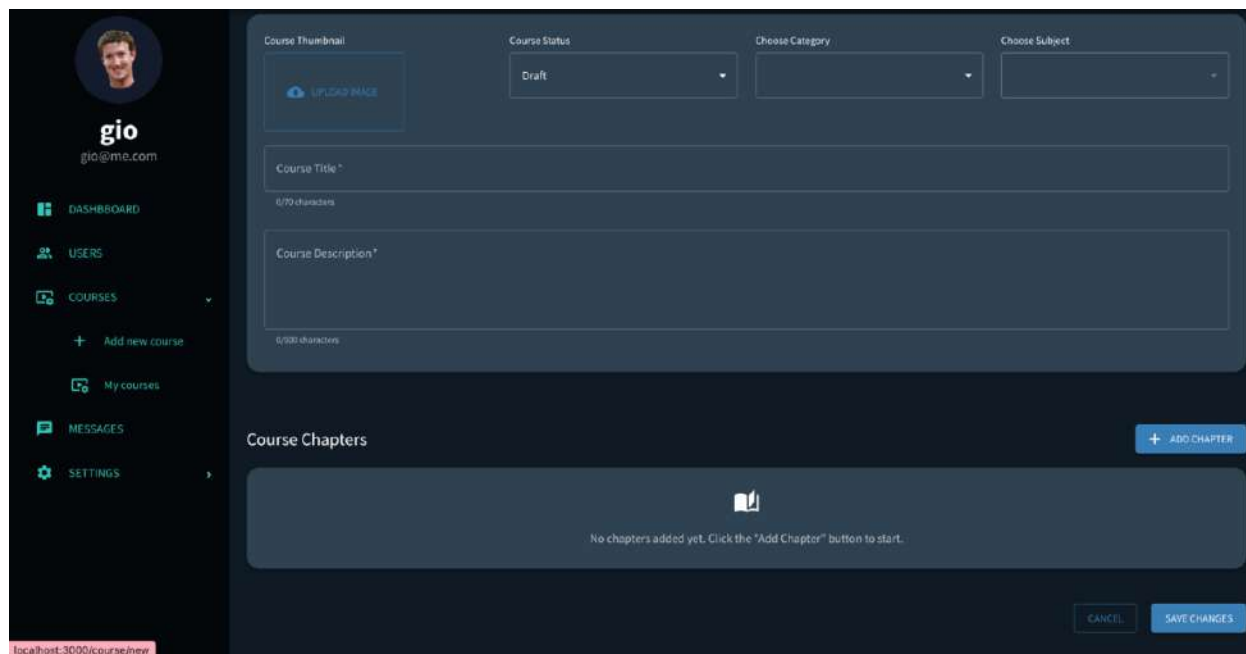


Рисунок 3.11 Сторінка додавання нового курсу

Також вчитель може переглядати, видаляти та редагувати всі свої курси з вкладки «My courses» (рис. 3.12).

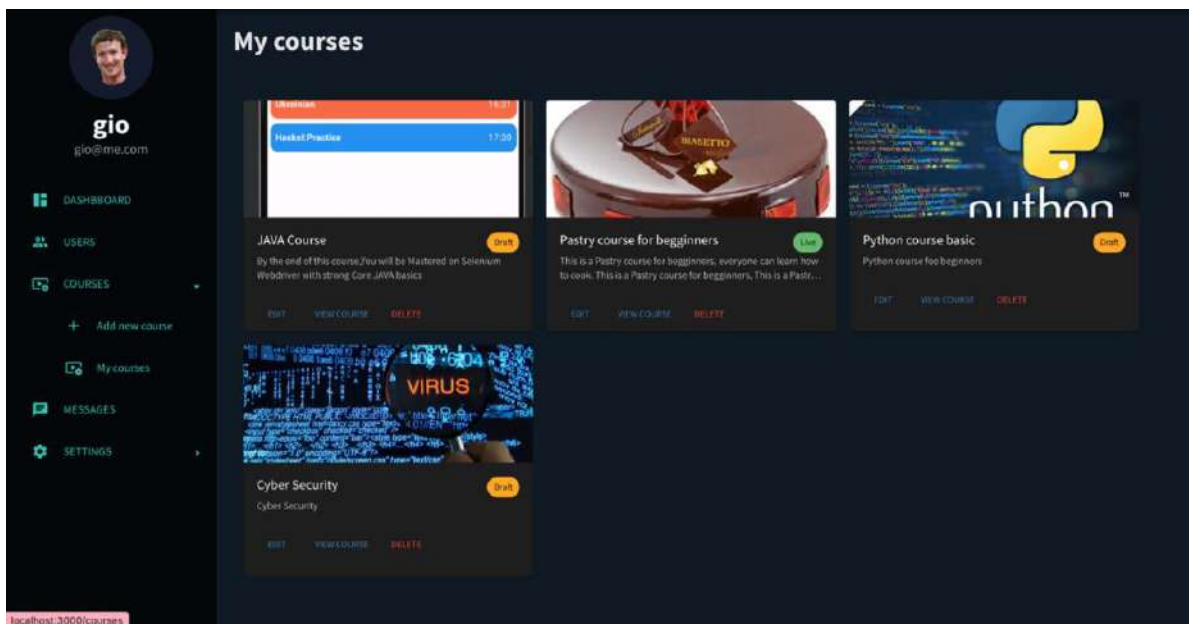


Рисунок 3.12 Сторінка доданих курсів викладача

Для спілкування та взаємодії один з одним, викладач та учень можуть використовувати вкладку «Messages», там вони знайдуть усі повідомлення, які надсилали раніше, та зможуть відправити нові (рис. 3.13).

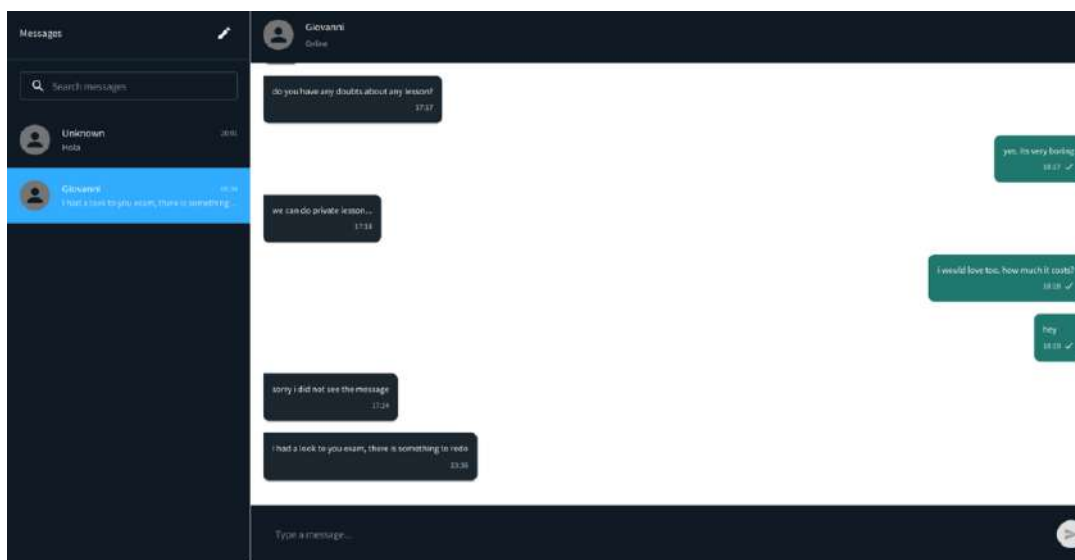


Рисунок 3.13 Сторінка повідомлень

Також користувачі можуть переглядати свій профіль, змінюючи та зберігаючи інформацію про себе (рис. 3.14).

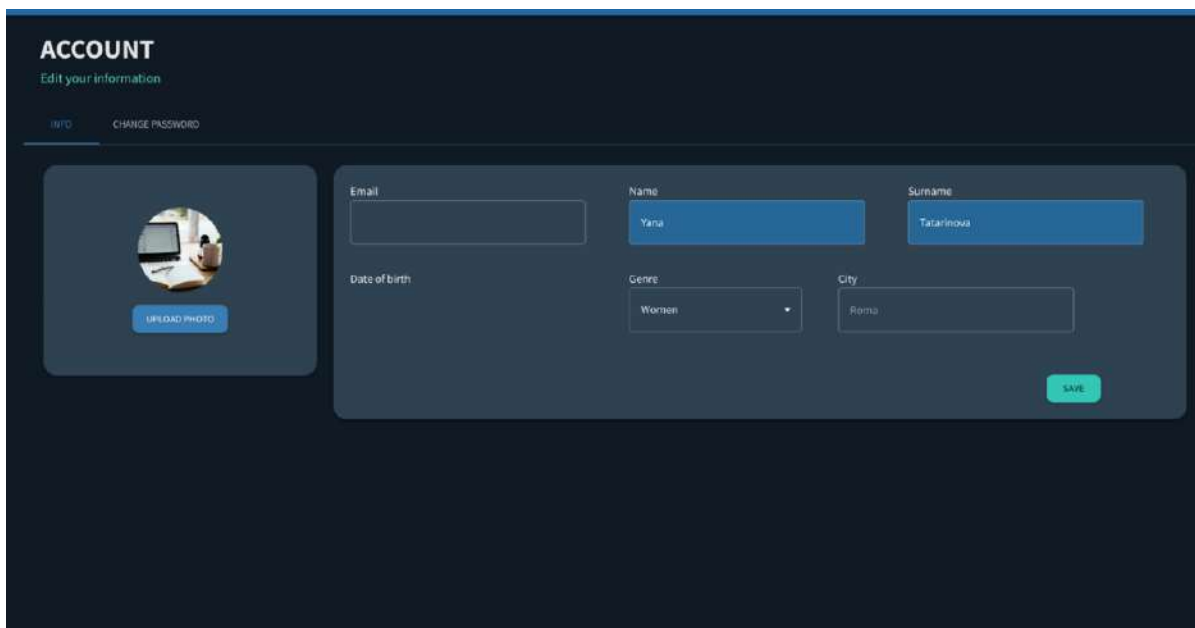


Рисунок 3.14 Сторінка профілю користувача

Крім того, платформа підтримує різні тем інтерфейсу для зручності користувача, а саме темну та світлу (рис. 3.15).

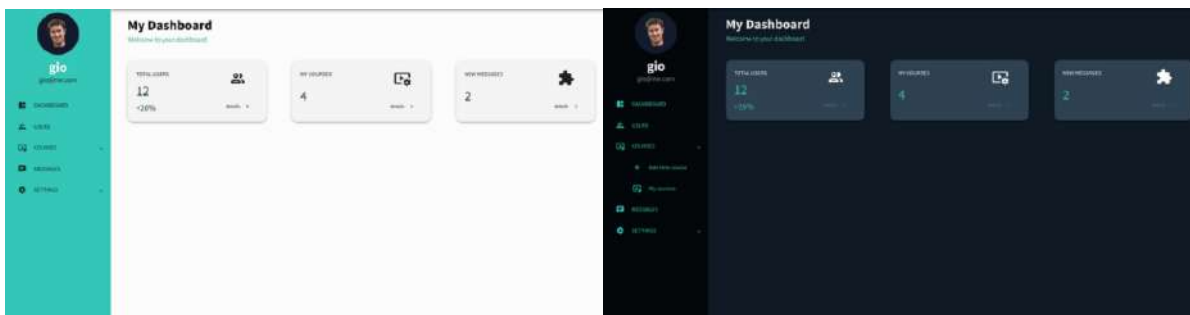


Рисунок 3.15 Демонстрація підтримки різних тем інтерфейсу

Отже, як наглядно продемонстровано після формування архітектурної моделі платформи, визначення основних функціональних вимог при аналізі аналогів, та розробки підходу інтеграції штучного інтелекту в систему, було реалізовано багатофункціональний та зручний веб-застосунок для онлайн-навчання. Його основні переваги та особливості, які варто зазначити:

- Інтегрований AI-асистент, що надає рекомендації для навчання на основі цілей, уподобань та попередніх курсів.
- Реалізація динамічного інтерфейсу, адже застосування фреймворку React дозволило нам створювати динамічний UI з миттєвим відгуком на дії користувача.
- Чат у реальному часі, що забезпечує ефективну взаємодію між студентами та викладачами.
- Інтерактивні курси, які містять в собі покрокову структуру, збереження прогресу, відеоматеріали, як основний спосіб подачі матеріалу.
- Система сертифікації, що дозволяє автоматичну генерацію сертифікатів після завершення курсу та можливість завантаження даних сертифікатів при потребі. Адаптивний дизайн, що зручно працює як на десктопах, так і на мобільних пристроях.
- Можливості для викладачів, а саме створення, редагування курсів, перегляд статистики проходження, отримання зворотного зв'язку.
- Підтримка різних тем (темна, світла)
- Використання хмарних сервісів (Firebase), що забезпечує надійність, швидкість доступу до даних.

ВИСНОВКИ

У ході дослідження було виконано аналіз та оформлено основні функціональні вимоги до онлайн-платформ для навчання. Було запропоновано структуровану архітектуру навчальної веб-платформи, яка поєднує елементи переваг класичних систем управління навчанням (LMS), таких як чітка організація курсів, рольова модель користувачів, система відгуків та аналітика, із гнучкістю та адаптивністю сучасних веб-застосунків, архітектурна клієнт-серверна модель містить у собі всі необхідні компоненти, такі як, клієнтська та серверна частина, модуль рекомендацій на основі штучного інтелекту та база даних.

Також було досліджено та реалізовано модульно орієнтований підхід, де кожна сутність (користувачі, курси, розділи) відокремлена структурно та функціонально, що забезпечує гнучкість системи. Побудовано логічну модель бази даних, яка описує сутності та їхні взаємозв'язки: курси, користувачі, розділи, відеоматеріали, рецензії, повідомлення. Модель оптимізована для потреб навчального процесу та розширення функціональності, застосовано хмарні технології Firebase для швидкого доступу до даних.

Розроблено модель рекомендаційної системи для студентів у вигляді інтеграції у систему AI-асистента, що формує персоналізовані рекомендації курсів на основі вибраних навчальних цілей, історії навчання й рівня користувача, із застосуванням механізмів генерації природною мовою та контекстного формування запитів (prompt engineering) та інтегровано дану модель в навчальну веб-платформу.

Реалізовано функціональний веб-застосунок для навчання, який включає в собі такі компоненти:

- Система управління навчальним контентом, яка надає можливість викладачам створювати, редагувати та видаляти навчальні курси, структурувати їх на розділи, додавати відеоматеріали.
- Інтелектуальний модуль рекомендацій, AI-асистент, що генерує персоналізовані рекомендації щодо навчальних курсів на основі освітніх цілей, історії проходження та поточного рівня користувача.
- Підсистема взаємодії в реальному часі, чат між студентами та викладачами реалізовано за допомогою Firebase Realtime Database, що забезпечує двосторонню асинхронну комунікацію.
- Інтерфейс для студента, із забезпеченням всіх необхідних функцій для проходження навчальних курсів.
- Адаптивний веб-інтерфейс, що забезпечує коректну роботу на різних типах пристроїв та забезпечує різні теми для інтерфейсу.

ДЖЕРЕЛА

1. UNESCO's education response to COVID-19. URL: <https://www.unesco.org/en/covid-19/education-response/initiatives?hub=800>
2. Udemy: website. URL: <https://www.udemy.com/>
3. Catherine Nabiem Akpen, Stephen Asaolu, Sunday Atobatele, Hilary Okagbue, Sidney Sampson. Impact of online learning on student's performance and engagement: a systematic review, 2024. URL: <https://link.springer.com/article/10.1007/s44217-024-00253-0?>
4. Coursera: website. URL: <https://www.coursera.org/>
5. Prometheus: website. URL: <https://prometheus.org.ua/>
6. Vseosvita: website. URL: <https://vseosvita.ua/>
7. EdEra: website. URL: <https://ed-era.com/>
8. Mr. Geoffrey Mwamba Nyabuto, Mr. Victor Mony, Professor Samuel Mbugua. Architectural Review of Client-Server Models, 2024. URL: https://ijsret.com/wp-content/uploads/2024/01/IJSRET_V10_issue1_125.pdf
9. Zilong Pan, Lauren Biegley, Allen Taylor, Hua Zheng. A Systematic Review of Learning Analytics–Incorporated Instructional Interventions on Learning Management Systems 2023. URL: <https://www.learning-analytics.info/index.php/JLA/article/view/8093/7791>
10. Timothy Stelzer, Gary Gladding, Jos'e Mestre, and David T. Comparing the efficacy of multimedia modules with traditional textbooks for learning introductory physics content Brookes Department of Physics; University of Illinois at Urbana-Champaign; Urbana, IL 61801 (Dated: October 22, 2018). URL : <https://arxiv.org/pdf/0806.0405>

11. Dr. Zoe Wood, Computer Science Department. Research Senior Project Final Report Teaching Intro to Web Development with React 2022. URL : <https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=1213&context=cscsp>
12. Sidorov Denys Volodymyrovych. Analyzing Virtual DOM Performance in Front-End Frameworks. URL: [https://doi.org/10.52058/2786-6025-2024-8\(36\)-840-858](https://doi.org/10.52058/2786-6025-2024-8(36)-840-858)
13. Veronica Gavrilă, Lidia Bajenaru, Ciprian Dobre. Studies in Informatics and Control, Modern Single Page Application Architecture: A Case Study. 2019. URL: <https://doi.org/10.24846/v28i2y201911>
14. David Johansson. Building maintainable web applications using React – An evaluation of architectural patterns conducted on Canvas LMS. Department of Computer and Information, Datateknik 2020. URL: <https://www.diva-portal.org/smash/get/diva2:1415320/FULLTEXT01.pdf>
15. Д. В. Кисюк, Г. А. Турко. Аналіз фреймворку React як інструмента для створення динамічних сайтів 2024. URL: <https://conferences.vntu.edu.ua/index.php/mn/mn2024/paper/viewFile/19897/16478>
16. Narender Reddy Karka. State Management in Large-Scale React Applications: A Comprehensive Analysis. International Journal of Advanced Research in Engineering and Technology (IJARET), 2025. URL: https://doi.org/10.34218/IJARET_16_02_007
17. Firebase: documentation. URL: <https://firebase.google.com/docs/auth>
18. Material UI: overview. URL: <https://mui.com/material-ui/getting-started/>
19. Jakob Nielsen. 10 Usability Heuristics for User Interface Design 2024. URL: <https://www.nngroup.com/articles/ten-usability-heuristics/>

20. Krystian Tuczynski. The Use of Artificial Intelligence in Distance Education. Journal of Modern Science, 2024. URL:<https://www.jomswsge.com/pdf-197010-119938?filename=The%20Use%20of%20Artificial.pdf>
21. Gemini : overview. URL: <https://deepmind.google/models/gemini/>
22. Тронь Т.В., Макатер С.В., Перетяга Л.Є., Коновалов О.Ю. Інформаційно-комунікаційні технології в освіті. Інтеграція штучного інтелекту в освітню та наукову діяльність, 2024. URL: <http://www.innovpedagogy.od.ua/archives/2024/77/59.pdf>
23. John Berryman, Albert Z. Prompt Engineering for LLMs The Art and Science of Building Large Language Model–Based Applications 2025. URL : <https://zncd.ir/wp-content/uploads/2025/01/John-Berryman-Albert-Ziegler-Prompt-Engineering-for-LLMs -The-Art-and-Science-of-Building-Large-Language-Model-Based-Applications-2024-OReilly-Media-libgen.li .pdf>