

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА**

**Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»**

Кафедра інформаційних систем в економіці

**ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА
«Інформаційні управляючі системи і технології»**

галузь знань	12 Інформаційні технології
спеціальність	122 Комп'ютерні науки

Форма навчання: очна (денна)

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему: «Розробка вебплатформи для інтелектуального поширення навчальних матеріалів»

здобувача Терентьєв Євгеній Геннадійович
(ПІБ, підпис)

Науковий керівник: д.е.н., професор Устенко Станіслав Веніамінович
(науковий ступінь, учене звання, ПІБ)

(підпис)

**Робота допущена до захисту перед екзаменаційною
комісією з атестації здобувачів вищої освіти (ЕК)**

Завідувач кафедри: к.е.н., доцент Тішков Б.О. _____
(підпис)

Київ 2025

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА**

**Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці**

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА

«Інформаційні управляючі системи і технології»

галузь знань	12 Інформаційні технології
спеціальність	122 Комп'ютерні науки

ПОГОДЖЕНО:

Керівник проєктної групи (гарант)
освітньо-професійної програми

_____ Устенко С.В.
«___» _____ 202_ р.

ЗАТВЕРДЖУЮ:

Завідувач кафедри інформаційних
систем в економіці

_____ Тішков Б.О.
«___» _____ 202_ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти _____ Терентьєва Євгеній Геннадійовича

(прізвище, ім'я, по батькові)

_____ денної _____ форми навчання

очної (денної), заочної, дистанційної

на підготовку кваліфікаційної магістерської роботи

на тему: «Розробка вебплатформи для інтелектуального поширення навчальних матеріалів»

Тему затверджено наказом ректора Університету від «__» _____ 202_ р. № _____

Кваліфікаційна магістерська робота виконується на матеріалах

План кваліфікаційної магістерської роботи

Розділ I _____ Дослідження та аналіз підходів до створення ІУС предметної області

(назва розділу)

Розділ II _____ Характеристика ІУС та методи і моделі

(назва розділу)

Розділ III _____ Розроблення проєктних рішень та їх реалізація

(назва розділу)

Об'єкт дослідження: _____ процеси поширення, адаптації та споживання цифрового освітнього контенту в інформаційних системах

Предмет дослідження: _____ методи та засоби інтелектуального поширення навчальних матеріалів у контексті створення інформаційної управляючої системи

Мета кваліфікаційної магістерської роботи: _____ розробити інтелектуальну інформаційну систему для поширення навчальних матеріалів із підтримкою персоналізованих рекомендацій.

« » _____ 202 p.

Реферат

Кваліфікаційна магістерська робота містить 70 сторінок, 1 таблицю, __ 31 рисунок список використаних джерел з 51 найменування, 3 додатки.

«Розробка вебплатформи для інтелектуального поширення навчальних матеріалів»

Об'єктом дослідження кваліфікаційної магістерської роботи є процеси поширення, адаптації та споживання цифрового освітнього контенту в інформаційних системах

Предметом дослідження є методи та засоби інтелектуального поширення навчальних матеріалів у контексті створення інформаційної управляючої системи.

Мета і завдання дослідження. Метою є розробити інтелектуальну інформаційну систему для поширення навчальних матеріалів із підтримкою персоналізованих рекомендацій, гнучкого управління контентом і масштабованістю для використання як у навчальних закладах, так і в корпоративному середовищі.

Відповідно до поставленої мети визначені такі *завдання*:

- провести аналіз предметної області та визначити ключові об'єкти й процеси в ній;
- дослідити існуючі інформаційні управляючі системи в освітньому та корпоративному середовищі;
- обґрунтувати вибір концептуальних підходів і технологій для створення ІУС;
- розробити архітектурну та функціональну модель майбутньої системи;
- створити інфологічну та фізичну модель бази даних;
- реалізувати інтелектуальний модуль на основі нейронної моделі рекомендацій;
- здійснити розробку програмного забезпечення та інтерфейсу користувача;
- впровадити систему авторизації з урахуванням чотирьох рівнів доступу;
- провести тестування розробленої системи та підготувати керівництво користувача;
- забезпечити можливість контейнеризованого розгортання платформи з адаптацією під окремі підприємства та організації.

Теоретична, методична та практична значущість отриманих результатів.

Значущість отриманих результатів полягає в створенні універсального та гнучкого програмного рішення, що поєднує сучасні архітектурні принципи, адаптивну базу знань і моделі персоналізованих рекомендацій. Важливою перевагою є технологічна готовність платформи до індивідуального розгортання під потреби окремих замовників, що робить її застосування можливим як у великих освітніх установах, так і в межах внутрішнього навчання в компаніях.

Рік виконання кваліфікаційної магістерської роботи – 2025.

Рік захисту роботи – 2025.

Ключові слова: навчання, курси, рекомендації, нейронні мережі, вебплатформа.

В І Д Г У К

про кваліфікаційну магістерську роботу
здобувача навчально-наукового інституту «Інститут інформаційних технологій в економіці» освітньо-професійної програми «Інформаційні управляючі системи і технології»

Терентьєв Євгеній Геннадійович

**на тему «Розробка вебплатформи для інтелектуального поширення
навчальних матеріалів»**

1. Актуальність теми: дослідження визначається високим професійним рівнем створення цифрової вебплатформи, яка забезпечує зручну, масштабовану та персоналізовану взаємодію користувачів з навчальним контентом. У рамках реалізації цієї мети було досягнуто низку суттєвих результатів, які мають як теоретичну, так і прикладну значущість.

2. Позитивні риси кваліфікаційної магістерської роботи: Автор роботи розробив повноцінну функціональну структуру ІУС, що включає користувацький інтерфейс, серверну частину та базу даних. Побудовано діаграми класів, прецедентів, послідовностей та розгортання, які відображають взаємозв'язки між компонентами системи, поведінку під час виконання ключових сценаріїв та варіанти взаємодії з акторами. На базі побудованих моделей реалізовано бізнес-логіку системи із підтримкою базової та ролей розширеного рівня доступу..

3. Наявність і цінність самостійних розробок автора. У ході роботи спроектовано інфологічну та фізичну модель бази даних, яка реалізована у середовищі PostgreSQL з використанням ORM EF Core. Визначено структуру даних, описано логіку взаємозв'язків між сутностями, забезпечено відповідність нормальним формам, зокрема третій нормальній формі. Здійснено навантаження контрольним прикладом. Особливу увагу приділено реалізації інтелектуального аналізу даних: виконано підготовку та обробку набору даних з понад 10 000 курсів, реалізовано алгоритм побудови рекомендацій на базі латентних факторів, що враховує історію взаємодії користувачів з курсами. Побудовано базу знань системи, яка структурує правила, пріоритети та параметри моделей ШІ, та дозволяє легко масштабувати рекомендаційний механізм. Платформа пройшла процес апробації, результати будуть *продемонстровані на відеоролику при захисті КМР*.

4. Недоліки та зауваження: відсутність перспектив у вигляді рекомендації щодо подальшого доопрацювання, використання та масштабування платформи, а також визначення її перспектив в прикладних навчальних центрах.

5. Загальна оцінка кваліфікаційної магістерської роботи та її допущення до захисту перед ЕК: є завершеним проєктом, що поєднує теоретичну новизну, практичну значущість і відповідає вимогам освітньо-професійної програми спеціальності F3 «Комп'ютерні науки». Під час тестування платформи продемонстровано її працездатність. Визначено конфігурації технічного забезпечення, достатні для розгортання проєкту на навчальному або невеликому сервісі.

6. Наукові результати та практичну цінність платформи підтверджено участю у двох наукових конференціях, публікацією тез доповідей, які наведені в Додатках.

Отримані результати можуть бути використані в спеціалізованих навчальних центрах, для формування, демонстрації та використання навчального контенту за прикладним призначенням. КМР заслуговує **на високу оцінку**.

Науковий керівник д.е.н., професор С.В. Устенко

17.05.2025

Рецензія

на кваліфікаційну магістерську роботу
здобувача вищої освіти

Терентьєв Євгеній Геннадійович

(прізвище, ім'я, по батькові)

Тема Розробка вебплатформи для інтелектуального поширення навчальних матеріалів

Актуальність теми кваліфікаційної магістерської роботи і доцільність її розроблення Тема кваліфікаційної магістерської роботи є надзвичайно актуальною з огляду на зростання попиту на дистанційне навчання та потребу підприємств у швидкому поширенні та адаптації навчального контенту. Розробка вебплатформи з елементами штучного інтелекту дозволяє ефективно вирішувати завдання персоналізованого доступу до матеріалів, що є доцільним як для освітніх закладів, так і для корпоративного використання.

Якість проведеного дослідження Проведене дослідження вирізняється системністю, глибиною аналізу предметної області та комплексним підходом до оцінки сучасних технологій створення інформаційних управляючих систем. Автор виявив здатність самостійно аналізувати наукові джерела, порівнювати існуючі рішення та обґрунтовувати вибір методів і підходів, що позитивно впливає на загальну якість дослідження.

Позитивні риси Серед позитивних рис кваліфікаційної магістерської роботи Серед позитивних рис кваліфікаційної магістерської роботи варто відзначити практичну спрямованість розробки, використання сучасного технологічного стеку та впровадження інтелектуального модуля персоналізованих рекомендацій. Робота характеризується логічною структурою, чітким викладом матеріалу, наявністю UML-діаграм, обґрунтуванням вибору архітектури та технологій, а також увагою до масштабованості та впровадження у виробниче середовище.

Зауваження Єдине зауваження стосується можливості більш детального опису результатів впровадження у реальних умовах — зокрема, надання кількісних показників ефективності системи на підприємстві або в організації. Також доцільним було б доповнення розділу тестування автоматизованими тестами, що підвищило б рівень надійності та перевірки працездатності системи.

Практична значимість висновків і рекомендацій _____ Висновки та рекомендації, сформульовані у кваліфікаційній магістерській роботі, мають високу практичну значимість, оскільки орієнтовані на реальні потреби підприємств у сфері внутрішнього навчання персоналу. Запропоноване рішення є гнучким, легко розгортається під конкретного замовника, а також забезпечує інтелектуальну підтримку користувача через рекомендаційний механізм, що робить платформу ефективним інструментом для підвищення кваліфікації працівників. _____

Місце роботи та посада рецензента

Науковий ступінь, учене звання (за наявності) _____
(підпис, ПІБ)

Підпис засвідчую: _____
(посада, підпис)

Місце печатки організації, де працює рецензент

ЗМІСТ

ВСТУП	4
РОЗДІЛ 1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ПРЕДМЕТНОЇ ОБЛАСТІ	7
1.1 Аналіз існуючих інформаційних управляючих систем (предметної області)	7
1.1.1 Дослідження предметної області	7
1.1.2 Дослідження ринку інформаційних управляючих систем	8
1.2 Обґрунтування вибору підходів і технологій для створення інформаційної управляючої системи	14
РОЗДІЛ 2 ХАРАКТЕРИСТИКА ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ТА МЕТОДИ І МОДЕЛІ	19
2.1 Структура і характеристика системи	19
2.2 Методи та моделі в інформаційних управляючих системах і технологіях	34
РОЗДІЛ 3 РОЗРОБЛЕННЯ ПРОЕКТНИХ РІШЕНЬ ТА ЇХ РЕАЛІЗАЦІЯ	37
3.1 Проектування бази даних та/або сховища даних для інформаційної управляючої системи	37
3.1.1 Проектування інфологічної (логічної) моделі БД	37
3.1.2 Обґрунтування вибору СКБД та проектування фізичної моделі БД в середовищі вибраної СКБД	40
3.1.3 Контрольний приклад та обрахунок прогнозних обсягів БД	42
3.2 Проектування бази знань інформаційної управляючої системи та/або засоби інтелектуального аналізу даних	43
3.2.1 Організація та проведення інтелектуального аналізу даних з метою пошуку знань в базі даних	43
3.2.2 Проектування та реалізація бази знань	45
3.3. Програмне забезпечення	48
3.3.1 Вимоги до програмного забезпечення	48

	3
3.3.2 Архітектура програмного забезпечення	50
3.3.3 Інструментальні засоби розробки для створення прикладного програмного забезпечення інформаційних управляючих систем	53
3.3.4 Керівництво (інструкція) користувача	55
3.4. Технічне забезпечення	61
3.4.1 Обґрунтування вибору технічного забезпечення	61
3.4.2 Ресурсні вимоги до технічного забезпечення	63
3.5 Реалізація інформаційної управляючої системи	64
ВИСНОВКИ	68
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	70
ДОДАТКИ	2

ВСТУП

Актуальність теми даного дослідження обумовлена стрімким зростанням ролі онлайн-освіти та необхідністю персоналізованого підходу до навчання у сучасному інформаційному суспільстві. Традиційні освітні моделі вже не повною мірою відповідають запитам сучасних користувачів, які прагнуть отримувати доступ до якісного, актуального та адаптованого до індивідуальних потреб контенту. Це особливо актуально як для закладів освіти, так і для підприємств, що впроваджують корпоративне навчання. Впровадження інформаційних управляючих систем, що підтримують елементи інтелектуального аналізу даних, дозволяє вирішувати задачі адаптації контенту до потреб конкретного користувача та підвищення ефективності навчального процесу.

Аналіз останніх досліджень свідчить про значну увагу наукової спільноти до проблем штучного інтелекту в освіті, інтелектуального пошуку, рекомендаційних систем, адаптивного навчання. Зокрема, у працях зарубіжних і вітчизняних авторів висвітлено методи персоналізації контенту, використання машинного навчання для аналізу взаємодії з освітніми ресурсами, а також архітектурні підходи до побудови інформаційних систем підтримки навчання. Водночас, попри значну кількість напрацювань, зберігається потреба в інтегрованих рішеннях, які були б водночас гнучкими, технологічно простими у впровадженні, придатними як для освітніх установ, так і для внутрішнього навчання на підприємствах.

Наукова проблема полягає у створенні інформаційної системи, здатної не лише зберігати та поширювати навчальні матеріали, а й інтелектуально взаємодіяти з користувачем, пропонуючи персоналізовані освітні рекомендації на основі зібраної інформації про його поведінку. У роботі сформульовано підхід до розв'язання цієї проблеми на основі сучасного стеку технологій та методів штучного інтелекту.

Результати магістерського дослідження вже отримали наукове використання у формі тез, опублікованих здобувачем на науково-практичних конференціях (див. додатки Б і В). У публікаціях розкрито аспекти оцінки якості

освітнього контенту, а також методи просування навчальних матеріалів за допомогою ІІІ. Особистий внесок автора полягає у повній реалізації архітектури платформи, включно з розробкою моделі рекомендацій на основі ML.NET, побудовою бази знань та реалізацією програмного коду клієнтської і серверної частини системи.

Мета дослідження — розробити інтелектуальну інформаційну систему для поширення навчальних матеріалів із підтримкою персоналізованих рекомендацій, гнучкого управління контентом і масштабованістю для використання як у навчальних закладах, так і в корпоративному середовищі.

Завдання дослідження:

- провести аналіз предметної області та сучасних інформаційних систем,
- обґрунтувати вибір методів, технологій і архітектури майбутньої ІУС,
- розробити функціональну структуру системи,
- реалізувати базу даних та алгоритми персоналізованих рекомендацій,
- протестувати програмне забезпечення,
- оцінити економічну ефективність впровадження системи.

Об'єкт дослідження — процеси поширення, адаптації та споживання цифрового освітнього контенту в інформаційних системах.

Предмет дослідження — методи та засоби інтелектуального поширення навчальних матеріалів у контексті створення інформаційної управляючої системи.

Методи дослідження: структурно-функціональний аналіз, порівняльне моделювання, об'єктно-орієнтоване проектування, методи машинного навчання (Matrix Factorization), інструменти візуального моделювання (PlantUML), інструментарій CRISP-DM.

Наукова та практична значущість дослідження полягає у створенні комплексного рішення, яке поєднує технології штучного інтелекту, сучасний підхід до архітектури ПЗ та гнучку модель масштабування. Результати можуть бути адаптовані для потреб ВНЗ, курсів підвищення кваліфікації, внутрішнього

навчання персоналу тощо. Запропоновані підходи є узагальнюваними та можуть бути основою для подальших досліджень у галузі цифрової трансформації освіти.

Інформаційна база дослідження охоплює публікації українських і зарубіжних дослідників, зокрема статті, присвячені рекомендаційним системам, цифровому навчальному середовищу, використанню нейронних мереж, ресурси платформ Kaggle, Coursera, Udey, а також офіційні джерела даних, включаючи статистику з освітнього ринку.

Структура роботи. Загальний обсяг роботи становить ____ сторінок друкованого тексту, 31 рисунок на 24 сторінках, 1 таблиця на 2 сторінках, 3 додатки на 8 сторінках. Список використаних джерел налічує 51 найменування.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз існуючих інформаційних управляючих систем (предметної області)

1.1.1 Дослідження предметної області

У сучасних умовах цифрової трансформації освіти та професійного розвитку особливого значення набувають інформаційні управляючі системи, призначені для поширення та управління навчальними матеріалами. Предметна область, яка охоплює як заклади освіти, так і підприємства, що мають власні навчальні програми, є перспективною для впровадження інтелектуальних рішень. Сучасні тенденції свідчать про зростання потреби у швидкому, персоналізованому та масштабованому поширенні знань у цифровому середовищі.

Основними об'єктами у цій предметній області виступають: навчальні ресурси, користувачі (студенти, співробітники, викладачі), платформи поширення знань, аналітичні системи для відстеження прогресу, а також засоби адаптації та рекомендацій контенту [1]. Процеси охоплюють створення, збереження, обробку та доставку навчального контенту, адміністрування користувачів, а також прийняття рішень на основі даних щодо освітньої траєкторії кожного користувача.

Предметна область також включає поняття інформаційної персоналізації — здатності системи адаптувати навчальний процес до індивідуальних потреб. Значну роль тут відіграє застосування штучного інтелекту, зокрема нейромережових підходів, що дозволяють аналізувати вподобання користувача та пропонувати релевантні навчальні матеріали [2].

Особливістю є і той факт, що потреби підприємств у навчанні персоналу дедалі більше тяжіють до використання внутрішніх систем, які легко впроваджуються та адаптуються. Це створює попит на такі рішення, які не потребують значних витрат на налаштування і можуть бути швидко розгорнуті, наприклад, за допомогою контейнеризації (Docker, Kubernetes тощо).

Проблемними питаннями, що потребують вирішення, залишаються:

- організація адаптивної підтримки прийняття рішень щодо навчального процесу;
- забезпечення релевантності навчального контенту;
- оптимізація інтерфейсу для зручності як адміністраторів, так і користувачів;
- забезпечення масштабованості та інтеграції з іншими системами.

Необхідність розробки ІУС для цієї предметної області обумовлена потребою у централізованому, інтелектуальному інструменті управління освітнім контентом, який поєднує простоту впровадження, адаптивність і можливості аналізу та рекомендацій. Рішення повинне забезпечувати підтримку прийняття управлінських рішень як в академічному, так і в корпоративному середовищі [3, 4].

1.1.2 Дослідження ринку інформаційних управляючих систем

У ході проведення дослідження основна увага була зосереджена на детальному аналізі функціональних можливостей сучасних освітніх платформ, їх архітектурних рішень, технологічних підходів та способів інтеграції інтелектуальних компонентів. Значну увагу приділено вивченню рівня впровадження технологій штучного інтелекту (ШІ), можливостей персоналізації навчання, масштабованості систем, простоті їх впровадження в освітнє чи корпоративне середовище, а також доступності та відкритості для кастомізації. Особливо важливо було визначити сильні та слабкі сторони платформ з точки зору підтримки прийняття рішень, автоматизації навчального процесу та інтерфейсної зручності для користувача. Аналіз проводився із метою виявлення існуючих недоліків у реалізації функцій сучасних ІУС, що, у свою чергу, дозволило обґрунтувати необхідність створення нової інтелектуальної вебплатформи, здатної поєднувати переваги вже наявних рішень і усувати їхні обмеження.

Для глибокого розуміння сучасного стану ринку ІУС у сфері поширення навчальних матеріалів було розглянуто чотири репрезентативні платформи: Coursera, Moodle, EdApp та Prometheus. Кожна з них представляє окремий сегмент

освітніх рішень: міжнародна академічна співпраця, інституційне внутрішнє навчання, корпоративна освіта та відкриті масові онлайн-курси.

Coursera є однією з провідних глобальних онлайн-платформ, що надає доступ до навчальних курсів від престижних університетів та компаній [5]. Вона має складну технологічну інфраструктуру, орієнтовану на масштабоване навчання. Завдяки вбудованим алгоритмам машинного навчання Coursera формує індивідуальні навчальні траєкторії, прогнозує ймовірність завершення курсів, адаптує складність матеріалів відповідно до поведінки користувача, а також пропонує персоналізовані рекомендації на основі аналізу попередніх дій та вподобань. Крім того, платформа пропонує корпоративні рішення для бізнесу, однак її рівень кастомізації обмежений рамками SaaS-моделі.

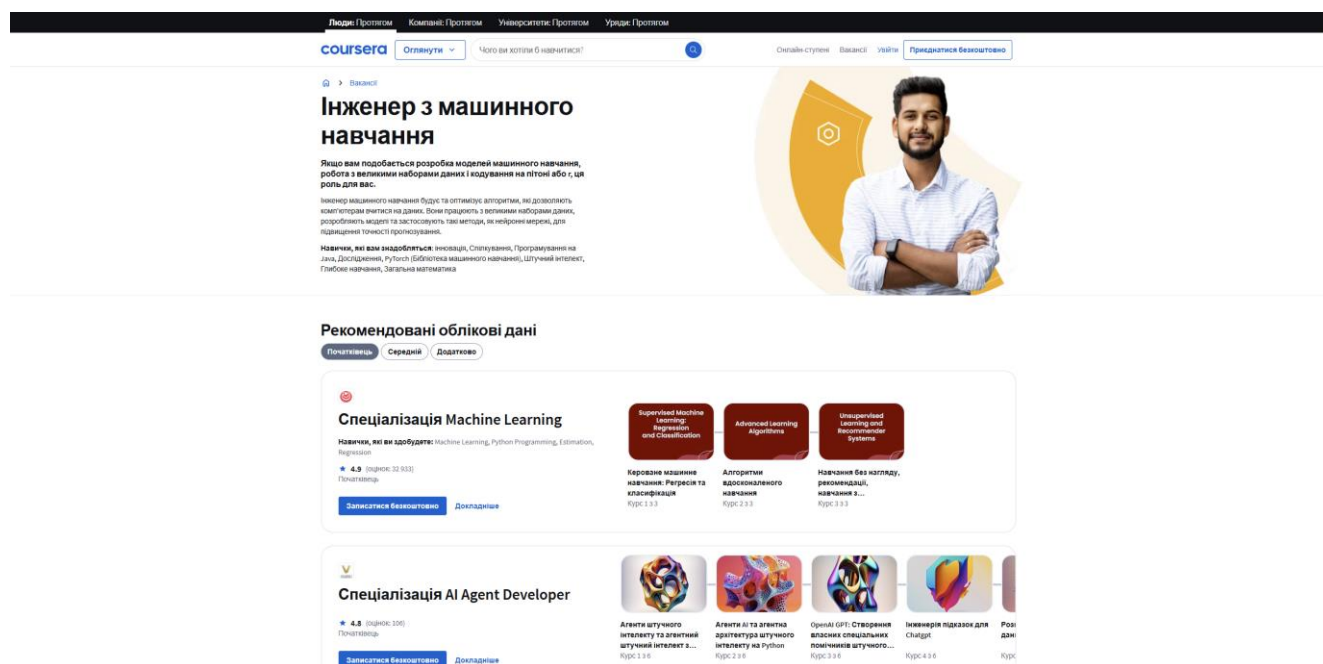


Рисунок 1.1 – користувацький інтерфейс платформи Coursera

Джерело: [5]

Moodle — це одна з найпоширеніших систем управління навчанням, яка має відкритий вихідний код і активно використовується навчальними закладами по всьому світу [6]. Сильними сторонами Moodle є її висока адаптивність, підтримка великої кількості плагінів і модулів, а також можливість повної кастомізації під потреби конкретної організації. Інтелектуальні функції в Moodle реалізуються через сторонні додатки: системи аналітики, адаптивного тестування, дашборди для

моніторингу прогресу тощо. Moodle вимагає технічних знань для налаштування та ефективного використання ШІ-компонентів, але відкритість коду дозволяє інтегрувати найсучасніші технології за бажанням адміністратора системи.

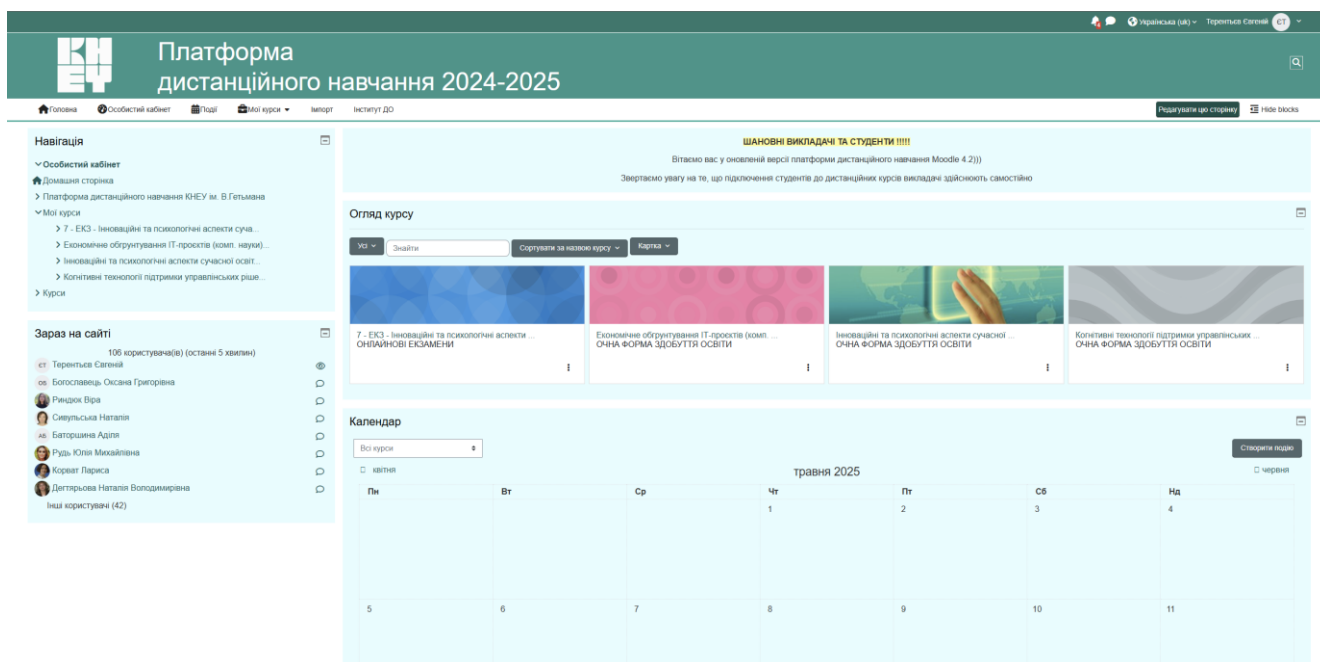


Рисунок 1.2 – користувацький інтерфейс системи Moodle

Джерело: [6]

EdApp — це сучасна SaaS-платформа, яка спеціалізується на мікронавчанні в корпоративному секторі [7]. Вона має чітку орієнтацію на користувацький досвід, інтерактивність та короткий цикл навчання. Платформа активно використовує алгоритми машинного навчання для формування персоналізованих курсів, оптимізації послідовності модулів на основі результатів проходження та залученості користувача. Серед додаткових переваг — підтримка гейміфікації, можливість швидкого оновлення вмісту, аналітика ефективності навчання та мобільна доступність. EdApp дозволяє легко інтегруватися з HRM-системами підприємств, що робить її зручною для широкого бізнес-використання.

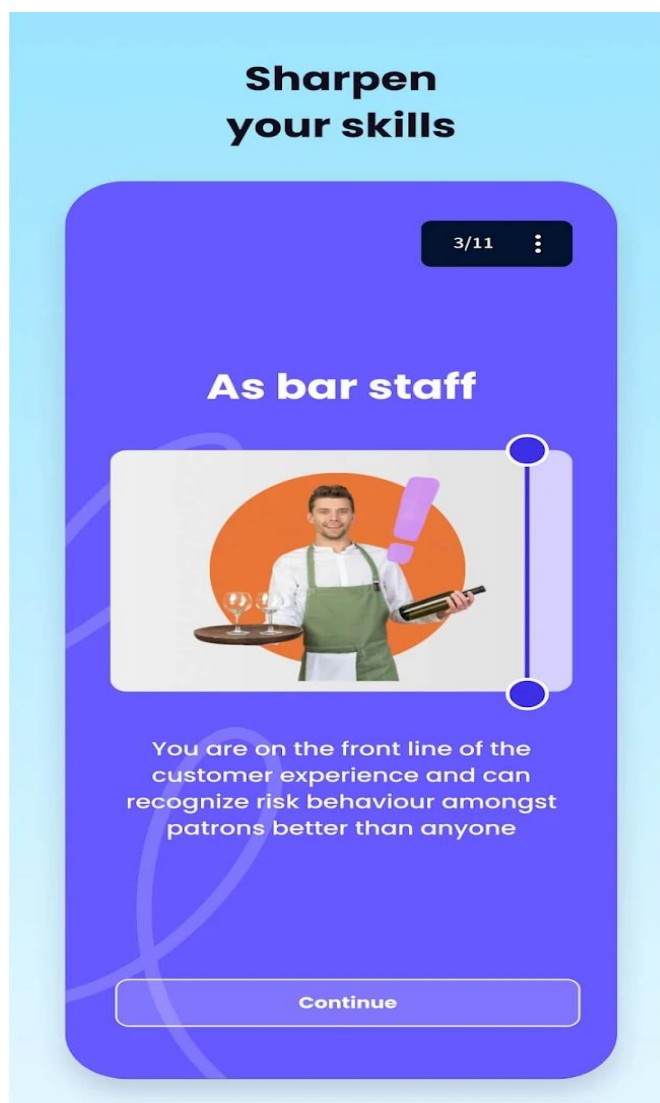


Рисунок 1.3 – користувацький інтерфейс EdApp

Джерело: [7]

Prometheus — українська платформа для масових відкритих онлайн-курсів (МООС), створена з метою підвищення доступу до якісної освіти [8]. Платформа в основному орієнтована на університетські курси, громадянську освіту та соціально важливі програми. Вона реалізована на базі Moodle, що дозволяє використовувати частину її функціоналу, але через обмежений бюджет і фокус на масовість та безоплатність, застосування ШІ у Prometheus практично відсутнє. Рекомендації курсів здійснюються вручну або за простими критеріями фільтрації. Це робить платформу менш адаптивною, але дуже доступною для широкого загалу користувачів.

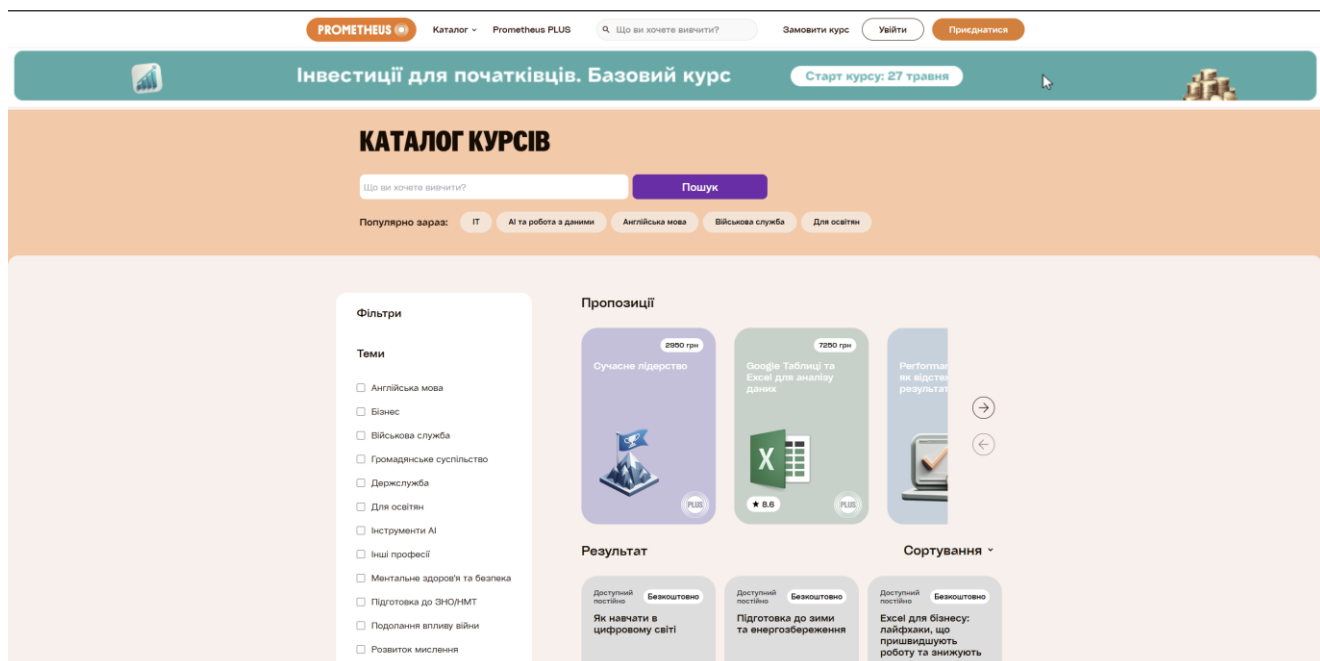


Рисунок 1.4 – Користувацький інтерфейс платформи Prometheus

Джерело: [8]

В таблиці 1.1 надано розширене порівняння платформ за ключовими параметрами.

Таблиця 1.1 – Порівняння існуючих рішень

	Coursera	Moodle	EdApp	Prometheus
Орієнтація	Університети, B2C	Освітні установи	Підприємства	Університети, B2C
Тип впровадження	SaaS	Self-hosted	SaaS	SaaS
Підтримка AI	Так	Обмежено	Так	Ні
Типи AI-функцій	Рекомендації, адаптація складності, прогнозування	Аналітика, дашборди, адаптивне тестування (через плагіни)	Персоналізація, гейміфікація, аналіз поведінки	-
Персоналізація	Низька	Висока	Середня	Низька
Підтримка підприємств	Частково	Можлива	Висока	Ні
Простота впровадження	Висока	Середня	Висока	Висока
Гнучкість масштабування	Висока	Висока	Висока	Середня

Продовження таблиці 1.1

	Coursera	Moodle	EdApp	Prometheus
Технологічна основа	Власна платформа, ML	PHP, MySQL, плагіни	SaaS, ML, мобільна підтримка	Moodle-based
Додаткові особливості	Сертифікація, інтеграція з бізнесом	Повна кастомізація, open-source	Інтеграція з HRM, швидке розгортання	Орієнтація на соціальні проєкти

Джерело: сформовано автором на основі виконаного дослідження

Результати проведеного аналізу показують, що кожна з досліджуваних платформ володіє окремими конкурентними перевагами. Coursera та EdApp виділяються як системи з глибокою інтеграцією інтелектуальних технологій та високим рівнем автоматизації освітнього процесу. Moodle, хоча і потребує налаштувань, залишається найбільш гнучкою для розгортання та адаптації. Prometheus забезпечує широкий доступ до знань, однак обмежений в аспектах персоналізації та ІІІ.

Незважаючи на значний прогрес у розвитку інформаційних управляючих систем, на ринку відчутно бракує комплексного рішення, яке б одночасно відповідало вимогам корпоративного сегменту, було б доступним для масштабування, забезпечувало високий рівень персоналізації та використовувало сучасні підходи штучного інтелекту. Це підтверджує актуальність і доцільність розробки нової вебплатформи, концептуальні засади якої закладено в межах цієї магістерської роботи.

1.2 Обґрунтування вибору підходів і технологій для створення інформаційної управляючої системи

Під час проєктування було обґрунтовано концептуальні підходи, інструменти та сучасні технології, які були застосовані для створення інформаційної управляючої системи інтелектуального поширення навчальних матеріалів. Особливу увагу було приділено критеріям вибору стеку технологій, підходам до розробки та архітектурним рішенням. Ці фактори визначалися на основі потреб у масштабованості, безпеці, високій продуктивності, простоті розгортання, використанні технологій штучного інтелекту, а також важливості зручного, інтуїтивного інтерфейсу для кінцевих користувачів — як в освітньому середовищі, так і в корпоративному секторі.

Було прийнято рішення організовувати процес розробки на базі ітеративної моделі, у поєднанні з принципами Rapid Application Development (RAD). Ітеративний підхід передбачає розробку системи у вигляді послідовності коротких циклів (ітерацій), кожна з яких включає аналіз, проєктування, реалізацію і тестування частини функціоналу [9]. Завдяки цьому існує можливість бачити результат на ранніх етапах і коригувати його. Це дозволяє гнучко адаптувати систему до змін у вимогах або зовнішньому середовищі.

Метод RAD орієнтований на швидку розробку робочого продукту з активним залученням користувачів у процес створення. Основною його перевагою є скорочення часу до появи функціонального прототипу, що дозволяє оперативно отримувати відгуки й на їх основі вдосконалювати систему. RAD базується на максимальному використанні повторного використання компонентів, візуальному програмуванню і безперервному тестуванні протягом всього життєвого циклу [10]. Це дозволяє швидко створювати робочі прототипи, поступово впроваджувати нові функції та адаптувати систему до зворотного зв'язку, що надходить від тестувальників та потенційних користувачів. Такий підхід забезпечує гнучкість та адаптивність у процесі розробки, дозволяє уникнути затримок та перевитрат ресурсів, а також сприяє підвищенню якості кінцевого продукту. Завдяки цьому

команда може оперативно реагувати на зміну вимог, оновлювати функціонал та враховувати нові виклики.

Крім того, було прийнято рішення використовувати Kanban-систему, реалізовану за допомогою GitHub Projects. Це дозволяє чітко відстежувати завдання, контролювати їхній пріоритет та забезпечує прозору взаємодію між розробниками, дизайнерами та аналітиками.

Kanban — це метод візуального управління процесом розробки, який використовує дошку з розділенням на колонки (наприклад, "заплановано", "в роботі", "готово") для відображення поточного статусу завдань [11]. Такий підхід дозволяє уникнути перевантаження команди, забезпечити безперервний потік робіт та своєчасне виявлення вузьких місць у розробці [12, 13].

З огляду на необхідність швидкого масштабування і стабільного функціонування в різних середовищах, ключовим організаційним рішенням стало використання контейнеризації. Одним з основних інструментів для реалізації цього підходу став Docker — програмне середовище, яке дозволяє створювати ізольовані контейнери для запуску додатків з усіма необхідними залежностями [14, 15]. Завдяки цьому розробники можуть гарантувати стабільну роботу додатка незалежно від середовища, в якому він розгортається. Docker дозволяє значно знизити витрати на підтримку інфраструктури, прискорює оновлення програмного забезпечення, а також сприяє стандартизації процесів розгортання. У межах розробки даної системи Docker використовується для створення шаблонних ізольованих інстансів, які можна швидко масштабувати або адаптувати під конкретного клієнта, що критично важливо для B2B-моделі платформи [16, 17].

Для хостингу було обрано Railway — хмарну платформу, яка спеціалізується на автоматизованому розгортанні та масштабуванні вебдодатків і баз даних. Railway є зручним інструментом для розробників, який дозволяє швидко деплоїти проекти без необхідності глибокого занурення в налаштування серверної інфраструктури [18]. Платформа підтримує автоматичне оновлення контейнерів, інтеграцію з GitHub, логування, моніторинг та налаштування середовищ розгортання [19]. Railway також пропонує зручні інтерфейси для керування

змінними середовища, резервного копіювання баз даних і перегляду логів. У контексті створення MVP та підтримки швидких ітераційних циклів розробки, ця платформа ідеально відповідає вимогам гнучкого, адаптивного підходу до хостингу інформаційної системи.

Штучний інтелект, що лежить в основі системи персоналізованих рекомендацій, реалізовано на базі бібліотеки ML.NET [20, 21]. Для побудови рекомендованої системи використано підхід Matrix Factorization — один із найпоширеніших методів у системах рекомендацій [22]. Matrix Factorization (матричний факторинг) — це метод машинного навчання, який передбачає розкладання великої матриці користувачів і об'єктів (у даному випадку курсів) на добуток двох менших матриць латентних ознак [23]. Це дозволяє виявити приховані залежності між користувачами та контентом, навіть у випадках, коли прямі зв'язки відсутні. Такий підхід є особливо ефективним у випадках з великою кількістю користувачів і обмеженим обсягом даних про кожного з них, оскільки дозволяє передбачати уподобання на основі схожості в поведінці інших користувачів [24].

Використання Matrix Factorization у поєднанні з ML.NET забезпечує ефективну реалізацію моделі рекомендацій з високою точністю, адаптивністю та швидкістю навчання [25]. Цей інструмент інтегрується з .NET, дозволяючи розробляти моделі машинного навчання без необхідності залучення сторонніх мов програмування, таких як Python. Завдяки цьому забезпечується єдність технологічного середовища, підвищується швидкість розробки та спрощується обслуговування системи.

Для тренування моделі було знайдено датасет "Online Courses" за авторством Khaled Shata, який доступний на платформі Kaggle та містить понад 10 000 записів про онлайн-курси з різних платформ [26]. Попри те, що ліцензія не була вказана автором, було прийнято рішення використовувати ці дані, але виключно в рамках науково-дослідницької та навчальної діяльності, що не порушує принципи академічної доброчесності.

Для реалізації клієнтської частини було обрано ReactJS у поєднанні з мовою TypeScript. ReactJS — це популярна JavaScript-бібліотека для створення інтерфейсів користувача, яка підтримується компанією Meta (колишній Facebook) [27]. Основною перевагою React є компонентний підхід до розробки, що дозволяє створювати багаторазово використовувані блоки інтерфейсу, забезпечуючи при цьому гнучкість і масштабованість застосунку [28]. Крім того, React має віртуальний DOM, що забезпечує високу продуктивність навіть у складних динамічних інтерфейсах [29].

TypeScript — це надмножина мови JavaScript, яка додає підтримку статичної типізації [30, 31]. Завдяки цьому розробники можуть виявляти помилки на етапі компіляції, а не під час виконання, що суттєво підвищує якість і надійність програмного забезпечення. TypeScript також полегшує роботу в команді, робить код більш читабельним та самодокументованим [32].

Для зборки проєкту було обрано інструмент Vite, який є сучасною альтернативою Webpack і створений з акцентом на надзвичайно швидкий старт розробки та миттєве оновлення змін (hot module replacement) [33]. Vite оптимізований під екосистему JavaScript/TypeScript і React, забезпечує низький час запуску навіть для великих проєктів та дозволяє ефективно працювати з сучасними модулями ES. Такий вибір обумовлений потребою у високій гнучкості інтерфейсу, простій інтеграції з REST API та швидкому запуску проєкту [34]. Також було розглянуто альтернативу React – Next.js [35, 36]. Але на відміну від Next.js, React дозволяє точніше контролювати маршрутизацію на клієнті, що є критично важливим для динамічних освітніх платформ [37-39]. Використання TypeScript забезпечує статичну типізацію, що значно зменшує кількість помилок на етапі компіляції та покращує супровід коду.

Для побудови UI було прийнято рішення застосувати бібліотеку Ant Design, яка включає великий набір адаптивних компонентів, що сприяє створенню сучасного, зручного та естетичного інтерфейсу. Проєкт збирається за допомогою інструменту Vite, який забезпечує високу швидкодію в режимі розробки та мінімізує час запуску.

Для управління асинхронними запитами, кешуванням та оновленням даних було вирішено використати бібліотеку TanStack Query, що дозволяє значно оптимізувати взаємодію клієнта з сервером і покращує користувацький досвід.

У якості основної бази даних обрано PostgreSQL — реляційну систему управління базами даних з відкритим кодом, яка добре зарекомендувала себе у масштабованих, продуктивних та надійних інформаційних системах [40]. PostgreSQL підтримує транзакції, індексацію, складні агрегатні запити та розширення, що дозволяє адаптувати її під специфіку освітньої платформи. Її використання забезпечує надійне зберігання даних, високу продуктивність та гнучкість при зростанні навантаження [41].

Загалом вибір зазначеного стеку технологій обґрунтовується не лише їхньою технічною перевагою, а й здатністю забезпечити стійкість системи в умовах зростаючих вимог до цифрового навчання. Поєднання сучасних підходів до розробки, використання відкритих стандартів, підтримка штучного інтелекту та зручний інтерфейс користувача дозволяє створити конкурентоспроможну платформу, яка відповідає актуальним вимогам галузі. Такий підхід робить запропоновану систему ефективною не лише як технічне рішення, а й як стратегічний інструмент цифрової трансформації навчального процесу як в академічному середовищі, так і у корпоративному секторі.

РОЗДІЛ 2 ХАРАКТЕРИСТИКА ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ТА МЕТОДИ І МОДЕЛІ

2.1 Структура і характеристика системи

Інформаційна управляюча система, що розробляється в рамках цієї роботи, призначена для інтелектуального поширення навчальних матеріалів серед користувачів різного типу: студентів, освітян, працівників підприємств. Головною метою є автоматизація процесів доступу до освітнього контенту, персоналізоване надання рекомендацій за допомогою технологій штучного інтелекту, управління навчальними курсами та аналітики освітньої активності користувачів.

Система реалізована у вигляді вебплатформи з чітко визначеною архітектурою, що включає клієнтську частину (інтерфейс користувача), серверну логіку (контролери, сервіси, моделі машинного навчання) та реляційну базу даних. Особливістю системи є її модульність, гнучкість масштабування за рахунок контейнеризації та використання сучасного технологічного стеку, орієнтованого на швидке розгортання та підтримку штучного інтелекту в режимі реального часу.

Функціональна структура системи передбачає реалізацію типових сценаріїв користувацької взаємодії: реєстрація та вхід, перегляд та пошук курсів, керування кошиком, оформлення замовлень, перегляд власної бібліотеки, оцінювання та коментування курсів, отримання персоналізованих рекомендацій. Для адміністраторів передбачено можливість публікації курсів та керування категоріями [42].

На рисунку 2.1 зображено діаграму класів, яка представляє сутнісний рівень інформаційної управляючої системи [43]. Цей рівень містить основні об'єкти предметної області, які відображають постійні або довгоживучі елементи даних системи. Усі класи наслідують абстрактний клас `AbstractEntity`, що містить уніфіковані поля для обліку моментів створення та останнього оновлення записів (`CreatedAt`, `UpdatedAt`), що є типовим для проєктів з використанням ORM-рішень, зокрема `Entity Framework` [44, 45].

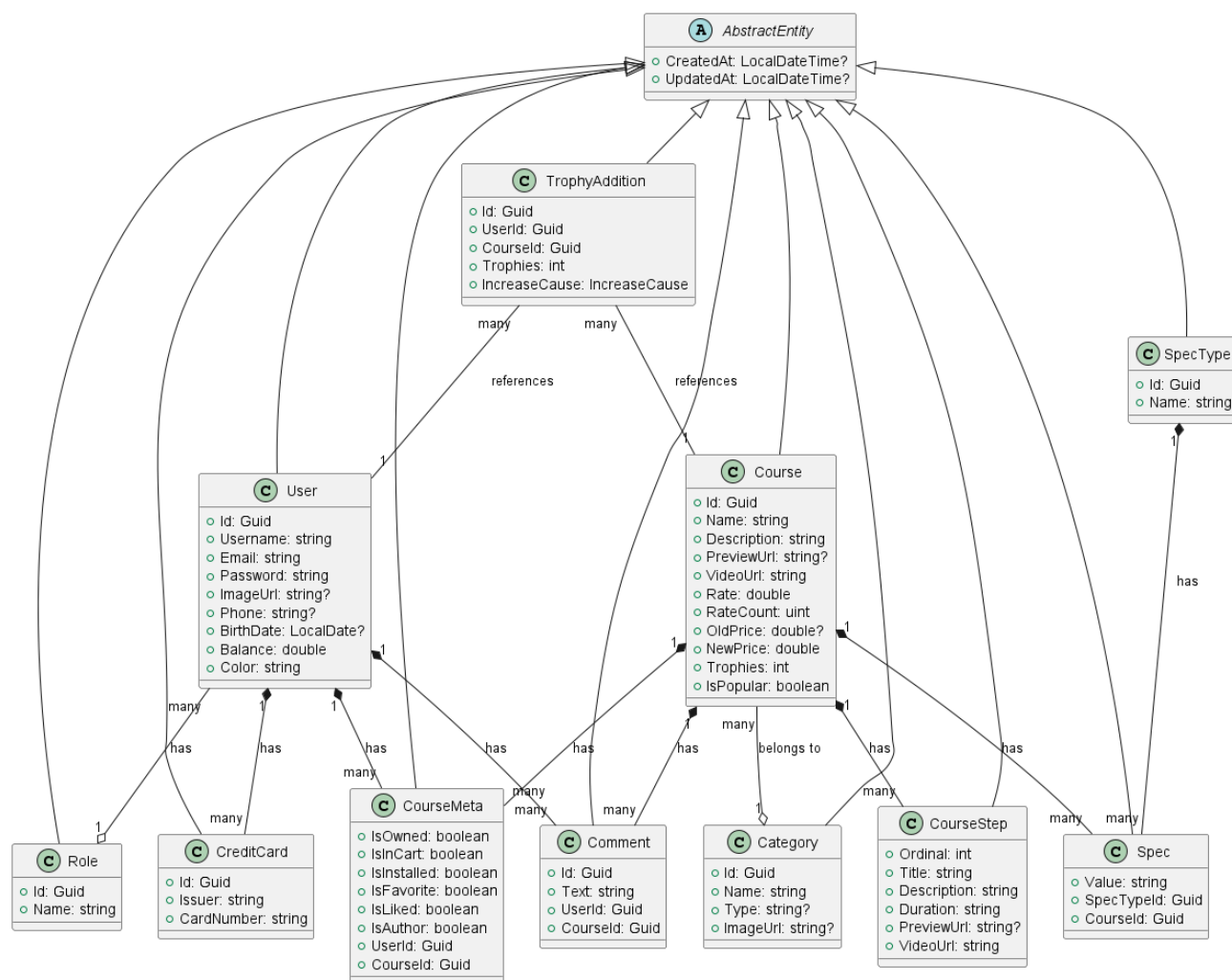


Рисунок 2.1 – діаграма класів сутностей

Джерело: розроблено автором під час проектування

Клас User представляє зареєстрованого користувача системи. Він містить як обов’язкові атрибути (Username, Email, Password, Balance), так і необов’язкові (ImageUrl, Phone, BirthDate, Color). Користувач може мати кілька ролей (Role), що забезпечує гнучкий механізм керування доступом до функціоналу системи.

Клас Course описує навчальний курс — основний продукт, що поширюється платформою. Він включає атрибути назви, опису, відеоматеріалів (PreviewUrl, VideoUrl), а також показники рейтингу (Rate, RateCount) та ціноутворення (OldPrice, NewPrice). Логічний атрибут IsPopular може використовуватись для побудови розділів “Рекомендоване” або “Популярне”.

Кожен курс належить до певної Category — категорії, яка також є сутністю з полями Name, Type, ImageUrl.

Клас `CourseMeta` реалізує метайнформацію про взаємодію конкретного користувача з курсом. Він охоплює такі стани, як володіння курсом (`IsOwned`), наявність у кошику (`IsInCart`), вподобання (`IsFavorite`, `IsLiked`), а також авторство (`IsAuthor`). Таким чином забезпечується реалізація поведінки системи без дублювання сутностей.

Система дозволяє залишати `Comment` до курсів, що зв'язує користувача з об'єктом навчання. Також підтримуються кроки курсу (`CourseStep`), які містять логічну структуру освітнього контенту. Структура кроків може бути використана для відстеження прогресу, виводу змісту тощо.

Платформа також передбачає додавання `Spec` — додаткових характеристик курсу, наприклад, рівня складності, типу тощо. Типи характеристик визначаються у класі `SpecType`, а конкретні значення — у відповідному `Spec`.

Особливістю системи є реалізація механізму “трофеїв” (`TrophyAddition`) — бонусів або очок, які користувач отримує за певну активність. Це відкриває можливості для впровадження гейміфікації навчального процесу.

Для забезпечення фінансової взаємодії передбачена модель `CreditCard`, яка може бути пов'язана з користувачем для подальших транзакцій (у разі реального впровадження платіжної системи).

Узагальнено, сутнісний рівень (рис. 2.1) відображає гнучку, розширювану модель даних, яка повністю охоплює навчальний процес, персоналізацію, взаємодію користувача з контентом, адміністративне управління й потенційно — платіжну інфраструктуру.

На рисунку 2.2 представлено структурну діаграму, яка відображає взаємозв'язки між ключовими компонентами логічної архітектури системи: контролерами (`Controller Layer`), сервісами (`Service Layer`), інтерфейсами, реалізаціями сервісів та зв'язком з базою даних (`Database Layer`).

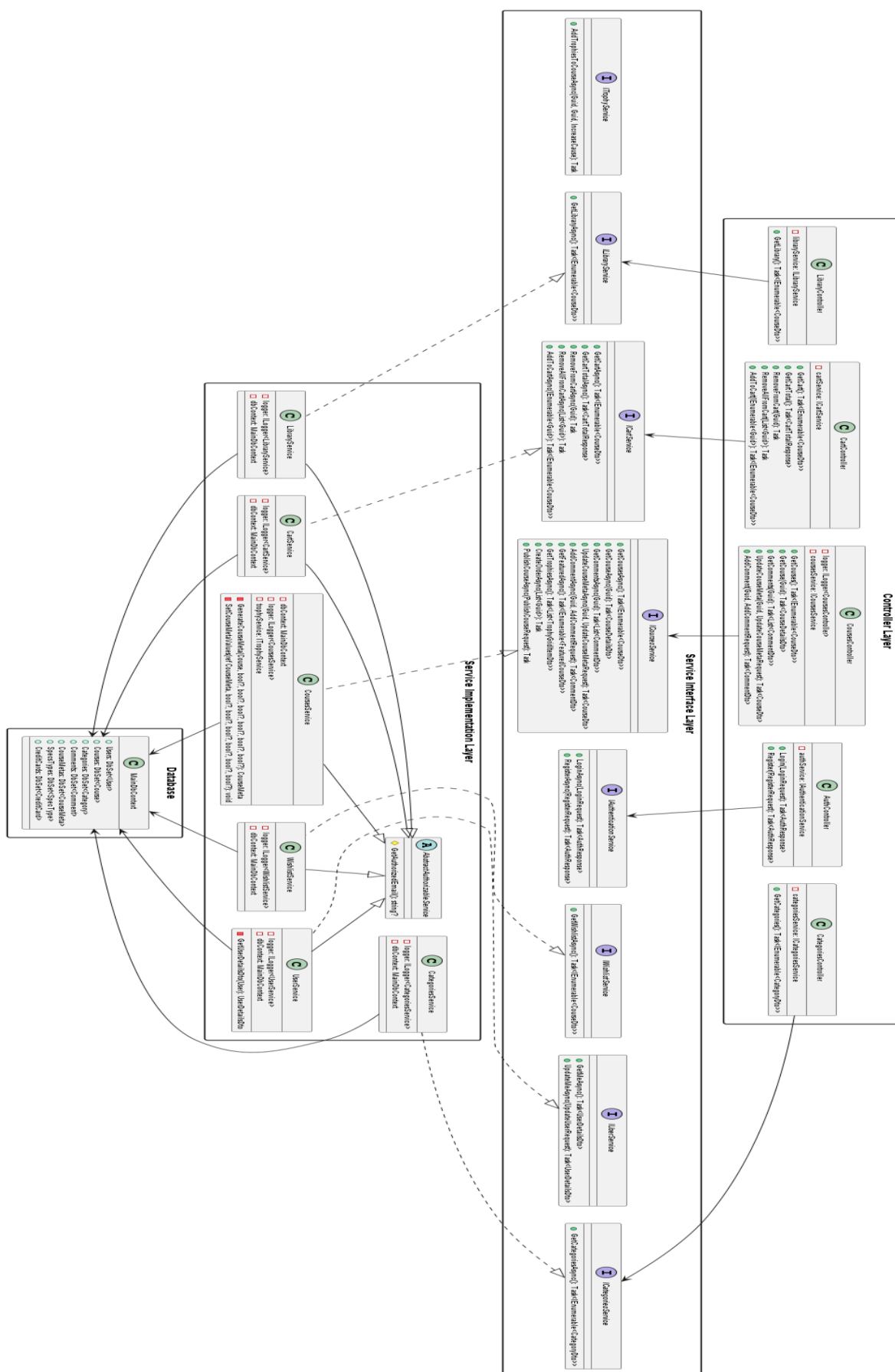


Рисунок 2.2 – загальна діаграма класів системи

Джерело: розроблено автором під час проєктування

Контролери (Controller Layer) є точкою входу в систему з боку клієнта. Вони приймають HTTP-запити від інтерфейсу користувача, передають дані в бізнес-логіку та повертають відповідь. Контролери є максимально “тонкими”, оскільки не містять бізнес-логіки, а делегують її відповідним сервісам. Наприклад:

- AuthController відповідає за автентифікацію користувачів;
- CoursesController обробляє запити на отримання курсів, оновлення метаінформації, публікацію коментарів тощо;
- CartController, LibraryController та CategoriesController працюють з відповідними підрозділами функціоналу системи.

Сервіси (Service Implementation Layer) реалізують основну логіку платформи. Усі класи сервісів реалізують відповідні інтерфейси (Service Interface Layer), що дозволяє дотримуватись принципів SOLID та спрощує написання модульних тестів. Більшість сервісів наслідують абстрактний клас AbstractAuthorizableService, який надає спільну функціональність, зокрема методи для отримання авторизованого користувача. Це дозволяє централізовано обробляти авторизацію та запити, що залежать від контексту користувача.

Кожен інтерфейс, наприклад ICartService або ICoursesService, визначає перелік операцій, які може виконати система в межах певного домену: отримання курсів, додавання в кошик, обробка замовлень, коментування, оновлення профілю, робота з трофеями тощо. У свою чергу, реалізації (CartService, CoursesService, UserService) мають залежності від MainDbContext та логерів, що відповідає кращим практикам сучасної розробки .NET-додатків.

Рівень бази даних (Database Layer) представлений класом MainDbContext, який інкапсулює зв'язок з реляційною базою даних PostgreSQL. Через цей клас забезпечується доступ до сутностей, зокрема Users, Courses, Categories, CourseMetas, Comments, CreditCards, SpecsTypes. Це дозволяє централізовано зберігати і взаємодіяти з усіма даними системи.

Загалом, архітектура системи, зображена на рисунку 2.2, демонструє чітке розділення обов'язків між шарами. Такий підхід сприяє масштабованості,

гнучкості у розширенні функціоналу, легшому супроводу проєкту та впровадженню нових підсистем, таких як система рекомендацій або аналітичний модуль.

На рисунку 2.3 зображено діаграму прецедентів, яка відображає основні функціональні сценарії взаємодії користувачів із системою. Діаграма ілюструє три типи акторів — Guest (гість), User (zareєстрований користувач) і Admin (адміністратор), а також відповідні їм варіанти використання системи.

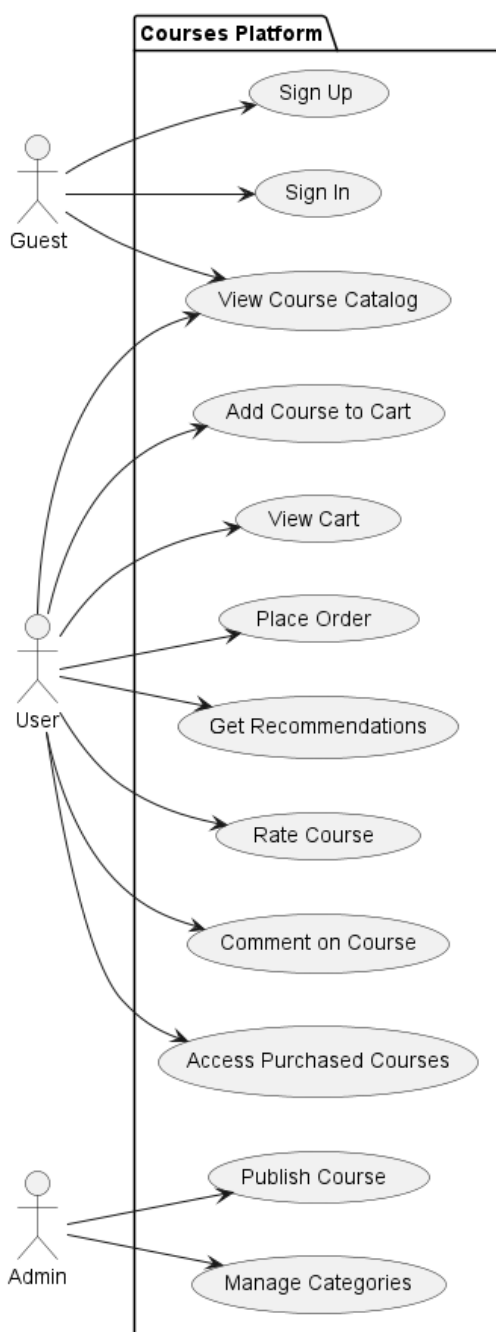


Рисунок 2.3 – діаграма прецедентів

Джерело: розроблено автором під час проєктування

Гість (Guest) має обмежені права доступу, проте може:

- створити обліковий запис (Sign Up),
- увійти до системи (Sign In),
- ознайомитися з доступними курсами через відкритий каталог (View Course Catalog).

Зареєстрований користувач (User) має ширший набір функцій і може:

- переглядати каталог курсів,
- додавати курси до кошика,
- переглядати вміст кошика,
- оформляти замовлення,
- отримувати персоналізовані рекомендації на основі ШІ-моделі,
- оцінювати пройдені курси,
- залишати текстові коментарі до курсів,
- переглядати придбані курси у своїй бібліотеці.

Адміністратор (Admin) має доступ до функціоналу, пов'язаного з керуванням навчальним контентом, зокрема:

- публікувати нові курси на платформу (Publish Course),
- керувати категоріями навчальних курсів (Manage Categories).

Зазначені сценарії охоплюють повний життєвий цикл взаємодії користувача з навчальними матеріалами: від ознайомлення з курсами до їх придбання, проходження, оцінювання й аналізу освітньої активності. Включення адміністративних можливостей демонструє підтримку різних ролей у системі та реалізацію принципів багаторівневого доступу.

Діаграма прецедентів (рис. 2.3) служить логічною основою для побудови діаграм послідовності кожного з варіантів використання, а також відображає вимоги до архітектури, інтерфейсів і поведінки системи загалом.

На рисунку 2.4 зображено діаграму послідовності, яка описує сценарій додавання коментаря до курсу. Користувач вводить текст повідомлення на сторінці перегляду курсу, після чого інтерфейс надсилає запит на серверний контролер

CoursesController. Контролер делегує обробку запиту сервісу CoursesService, передаючи дані про коментар у вигляді об'єкта AddCommentRequest.

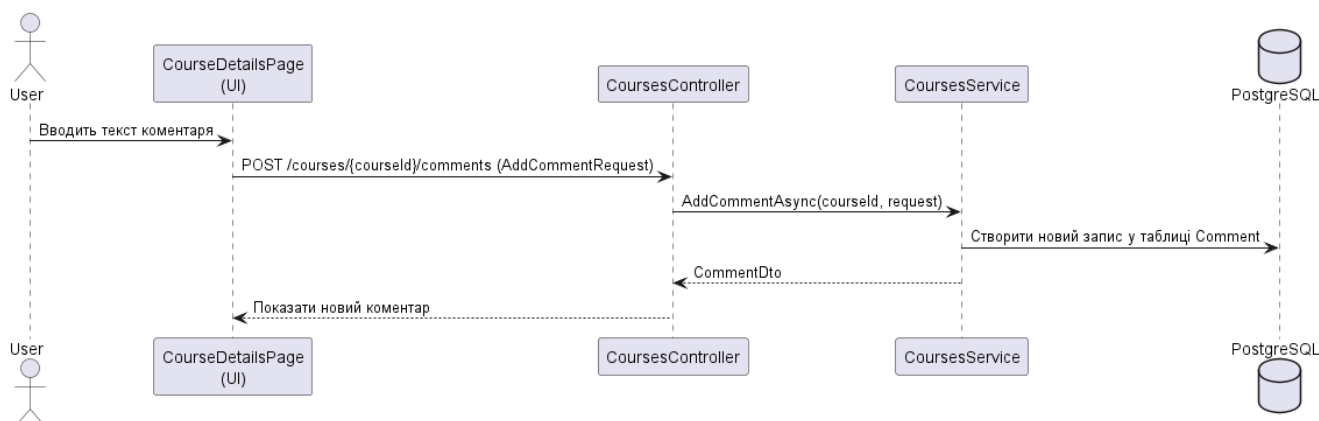


Рисунок 2.4 – діаграма послідовності для додавання коментаря

Джерело: розроблено автором під час проєктування

Сервіс створює відповідний запис у таблиці Comment бази даних PostgreSQL та повертає сформований об'єкт CommentDto, який містить інформацію про новий коментар. Цей об'єкт передається назад через контролер на UI, де відображається для користувача без перезавантаження сторінки.

На рисунку 2.5 показано сценарій додавання курсу до кошика. Користувач натискає відповідну кнопку на сторінці каталогу курсів, у результаті чого інтерфейс надсилає POST-запит на ендпоінт /cart з ідентифікаторами курсів.

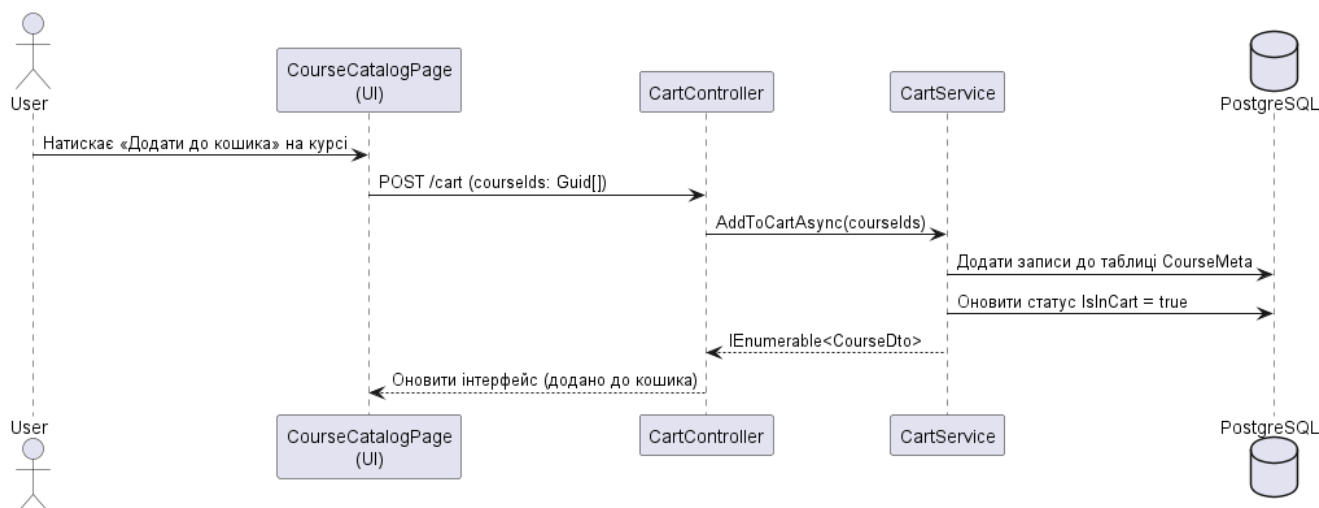


Рисунок 2.5 - діаграма послідовності для додавання курсу до кошика

Джерело: розроблено автором під час проєктування

Контролер `CartController` передає ці дані до методу `AddToCartAsync()` сервісу `CartService`. Сервіс взаємодіє з базою даних, створюючи або оновлюючи записи в таблиці `CourseMeta`, встановлюючи для них статус `IsInCart = true`. Після цього оновлена інформація повертається у вигляді списку об'єктів `CourseDto`, які відображаються в інтерфейсі користувача.

На рисунку 2.6 зображено діаграму послідовності для сценарію перегляду каталогу курсів. Користувач відкриває відповідну сторінку, після чого інтерфейс надсилає GET-запит на ендпоінт `/courses`.

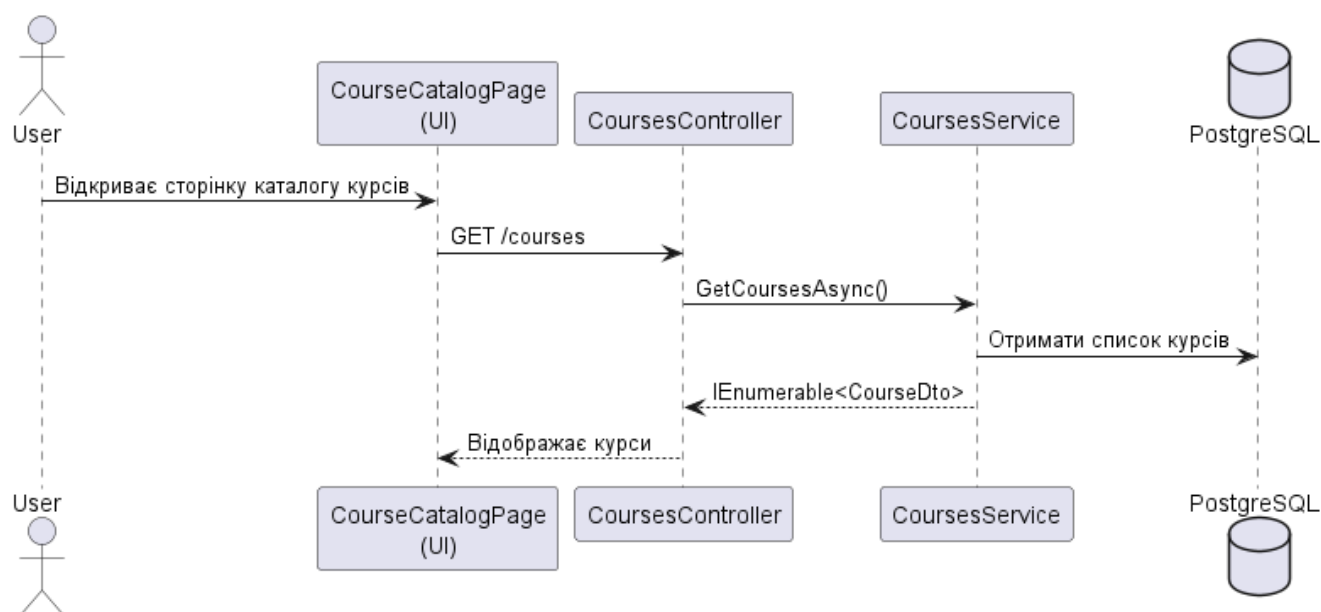


Рисунок 2.6 - діаграма послідовності для перегляду каталогу

Джерело: розроблено автором під час проектування

Контролер `CoursesController` викликає метод `GetCoursesAsync()` сервісу `CoursesService`, який звертається до бази даних для отримання актуального списку курсів. Після виконання запиту дані повертаються до інтерфейсу у вигляді списку об'єктів `CourseDto`, де вони відображаються користувачу.

На рисунку 2.7 представлено діаграму послідовності, яка описує сценарій перегляду особистої бібліотеки користувача. При відкритті відповідної сторінки інтерфейс надсилає GET-запит на ендпоінт `/library`.

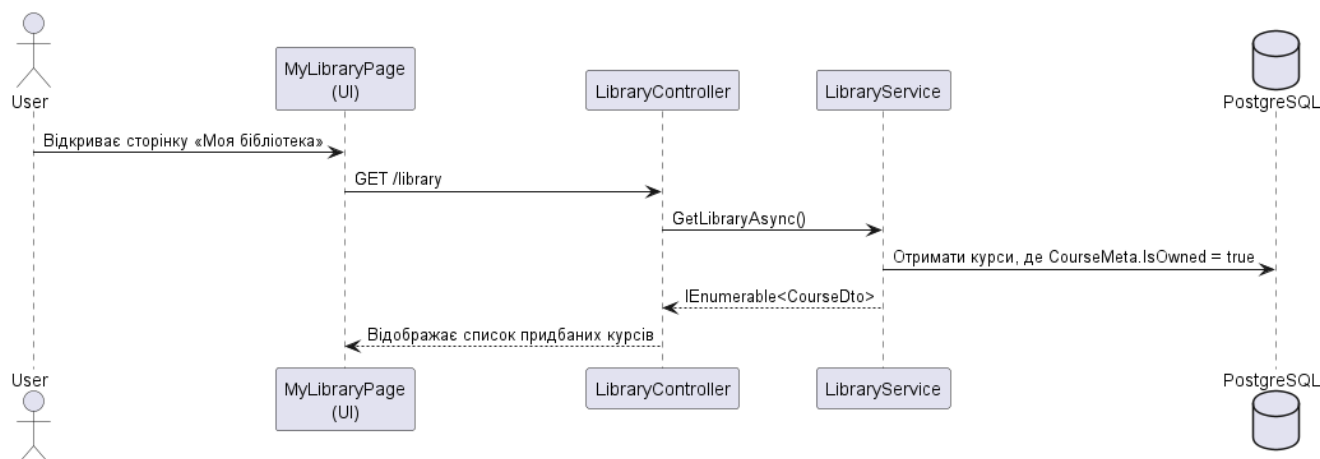


Рисунок 2.7 - діаграма послідовності для перегляду особистої бібліотеки

Джерело: розроблено автором під час проєктування

Контролер `LibraryController` викликає метод `GetLibraryAsync()` сервісу `LibraryService`, який здійснює вибірку курсів з бази даних, де у зв'язку `CourseMeta` встановлено прапорець `IsOwned = true`. Сформований список курсів повертається у вигляді об'єктів `CourseDto` й відображається в інтерфейсі.

На рисунку 2.8 наведено послідовність дій адміністратора під час керування категоріями курсів. При відкритті модуля категорій інтерфейс надсилає GET-запит до ендпоінта `/categories`, який обробляється контролером `CategoriesController`.

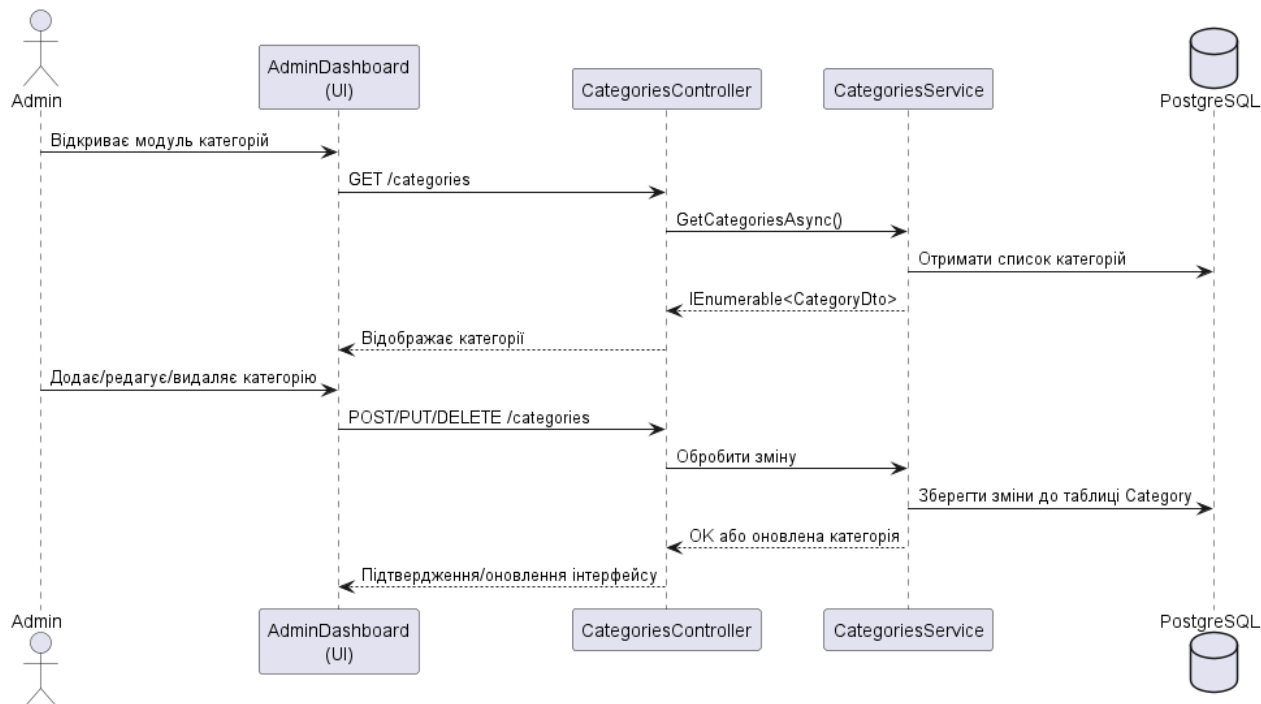


Рисунок 2.8 - діаграма послідовності для керування категоріями

Джерело: розроблено автором під час проєктування

Контролер викликає метод `GetCategoriesAsync()` сервісу `CategoriesService`, який звертається до бази даних для отримання списку доступних категорій. Після цього адміністратор може додавати, редагувати або видаляти категорії. Ці дії призводять до виконання відповідних запитів (POST, PUT або DELETE), які обробляються сервісом та спричиняють зміну даних у таблиці `Category`. У відповідь повертається оновлена інформація або підтвердження дії.

На рисунку 2.9 показано сценарій оформлення замовлення користувачем на один або кілька курсів. Після натискання кнопки «Оформити замовлення» інтерфейс надсилає POST-запит до ендпоінта `/courses/order` з масивом ідентифікаторів курсів.

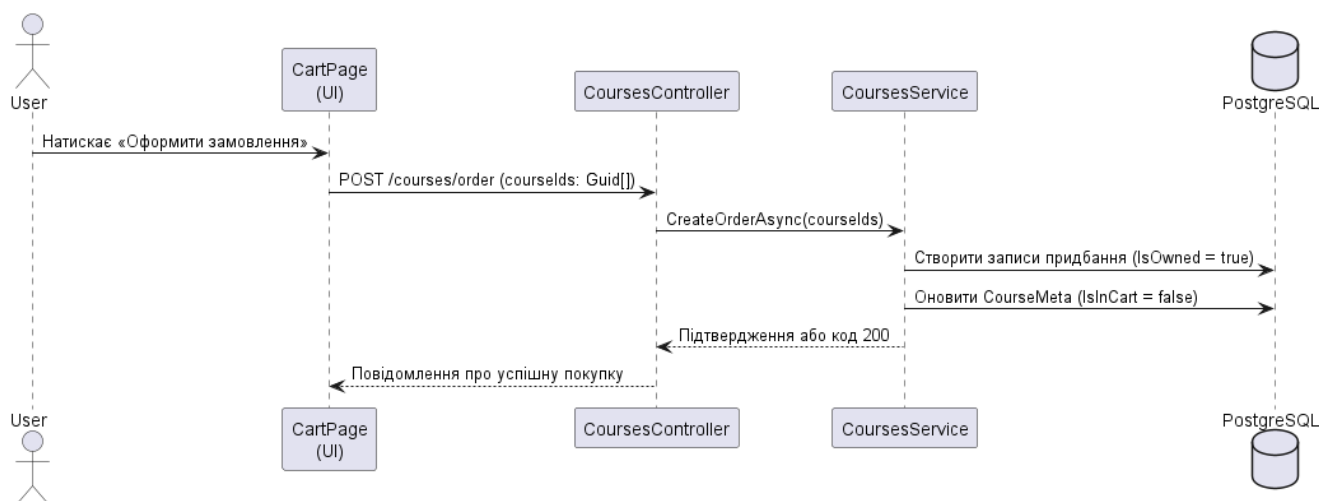


Рисунок 2.9 - діаграма послідовності для оформлення замовлення

Джерело: розроблено автором під час проєктування

Контролер `CoursesController` викликає метод `CreateOrderAsync()` сервісу `CoursesService`, який оновлює зв'язки `CourseMeta`, встановлюючи для вибраних курсів статус `IsOwned = true` та змінюючи `IsInCart = false`. У відповідь повертається підтвердження виконання запиту або код 200, після чого інтерфейс виводить повідомлення про успішну покупку.

На рисунку 2.10 зображено послідовність дій адміністратора під час публікації нового курсу. Після заповнення відповідної форми у панелі керування адміністратор надсилає POST-запит до ендпоінта `/courses/publish` з інформацією про курс.

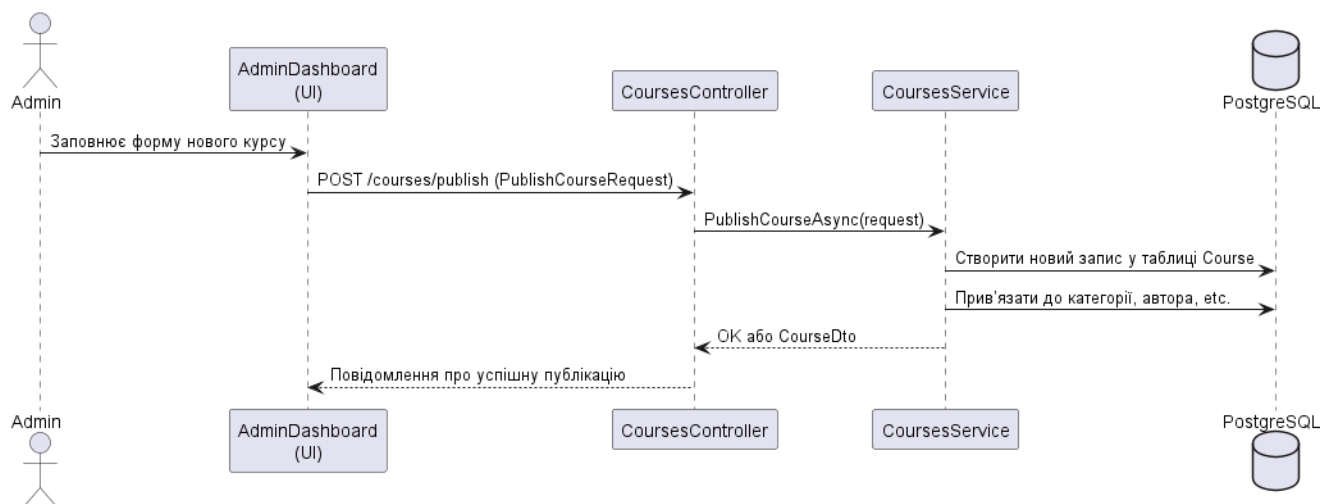


Рисунок 2.10 - діаграма послідовності для публікації нового курсу

Джерело: розроблено автором під час проєктування

Контролер `CoursesController` передає дані до методу `PublishCourseAsync()` сервісу `CoursesService`, який створює новий запис у таблиці `Course` бази даних. Окрім цього, курс зв'язується з відповідною категорією, автором та іншими метаданими. У відповідь повертається об'єкт `CourseDto` або підтвердження, після чого в інтерфейсі відображається повідомлення про успішну публікацію.

На рисунку 2.11 зображено діаграму послідовності, яка ілюструє процес оцінювання курсу користувачем. Після виставлення оцінки у вигляді зірочок інтерфейс надсилає PUT-запит на ендпоінт `/courses/meta/{courseId}` із об'єктом `UpdateCourseMetaRequest`.

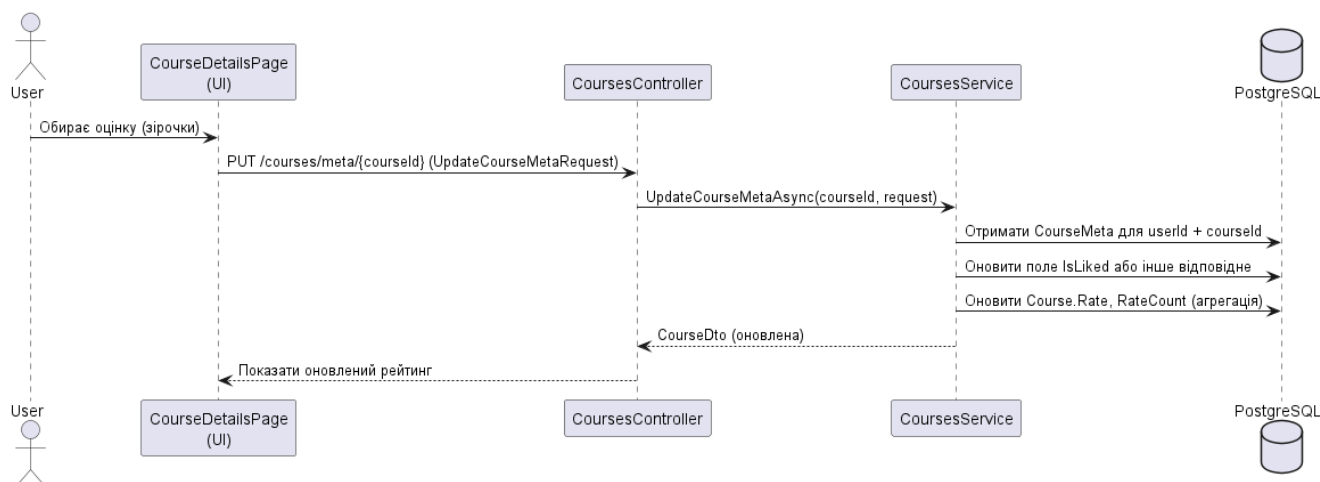


Рисунок 2.11 - діаграма послідовності для оцінки курсу користувачем

Джерело: розроблено автором під час проєктування

Контролер `CoursesController` викликає метод `UpdateCourseMetaAsync()` сервісу `CoursesService`, який звертається до бази даних для отримання об'єкта `CourseMeta` та оновлює відповідні поля (наприклад, `IsLiked`). Після цього перераховується середнє значення оцінки (`Course.Rate`, `RateCount`), і оновлений об'єкт `CourseDto` повертається для оновлення інтерфейсу.

На рисунку 2.12 представлено процес автентифікації користувача під час входу до системи. Гість вводить свої облікові дані на сторінці авторизації, після чого інтерфейс надсилає POST-запит на ендпоінт `/login` з об'єктом `LoginRequest`.

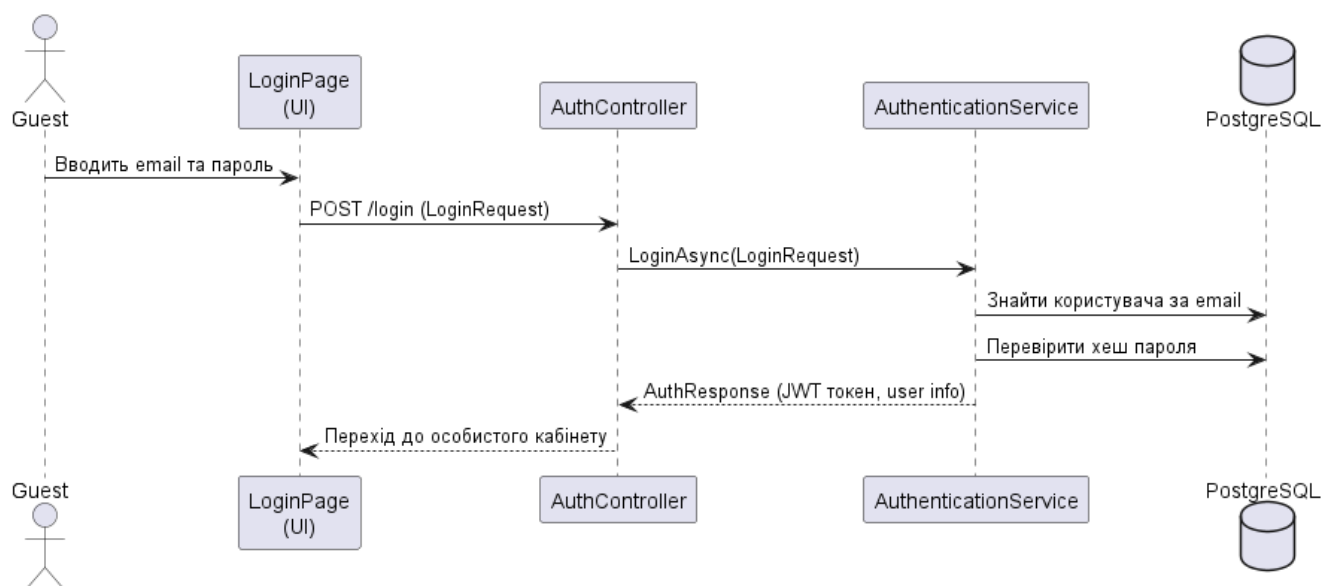


Рисунок 2.12 - діаграма послідовності для процесу авторизації

Джерело: розроблено автором під час проектування

Контролер `AuthController` викликає метод `LoginAsync()` сервісу `AuthenticationService`, який звертається до бази даних, перевіряючи відповідність email і хешу пароля. У разі успіху формується об'єкт `AuthResponse`, що містить JWT-токен та інформацію про користувача. Ці дані повертаються до інтерфейсу, який виконує перенаправлення до особистого кабінету.

На рисунку 2.13 представлено послідовність взаємодії під час реєстрації нового користувача. Гість заповнює реєстраційну форму, після чого інтерфейс надсилає POST-запит на ендпоінт `/register` із заповненим об'єктом `RegisterRequest`.

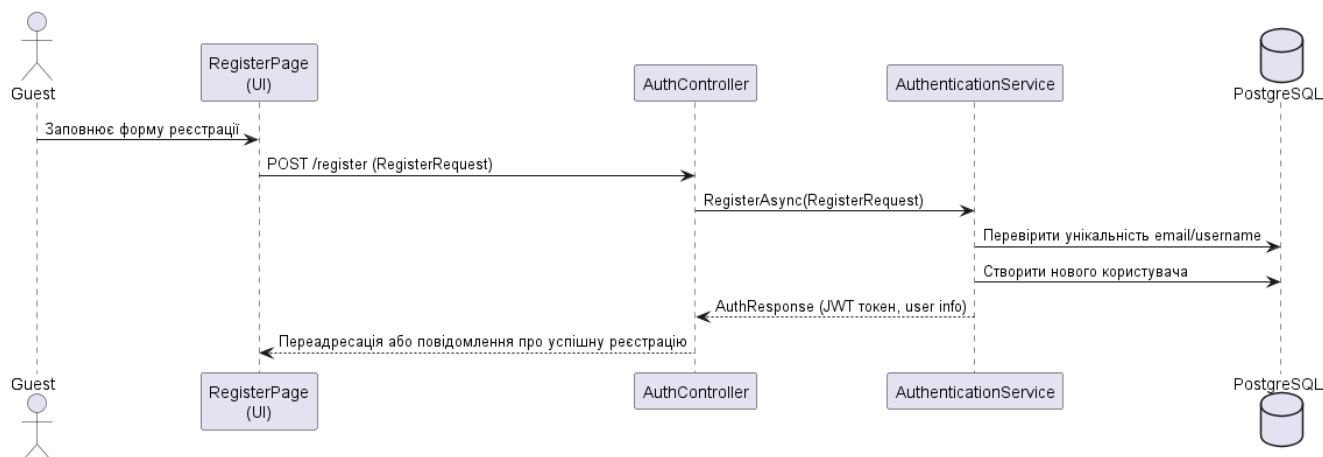


Рисунок 2.13 - діаграма послідовності для процесу реєстрації

Джерело: розроблено автором під час проєктування

Контролер `AuthController` викликає метод `RegisterAsync()` сервісу `AuthenticationService`, який перевіряє унікальність введеного email та імені користувача, а потім створює новий запис у таблиці користувачів бази даних. У відповідь формується об'єкт `AuthResponse`, який містить JWT-токен та дані нового користувача. Після цього інтерфейс відображає повідомлення про успішну реєстрацію або виконує перенаправлення до особистого кабінету.

На рисунку 2.14 представлено діаграму послідовності для сценарію отримання персоналізованих рекомендацій. Користувач відкриває відповідну сторінку, що ініціює GET-запит до ендпоінта `/suggestions`.

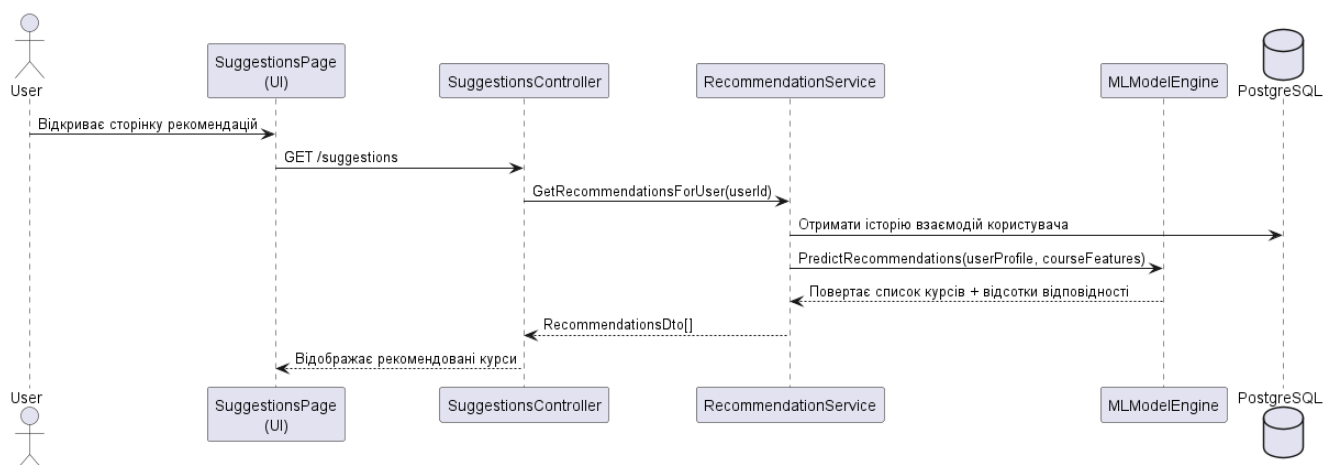


Рисунок 2.14 - діаграма послідовності для отримання рекомендацій

Джерело: розроблено автором під час проєктування

Контролер `SuggestionsController` викликає метод `GetRecommendationsForUser()` сервісу `RecommendationService`, який спочатку звертається до бази даних для збору історії взаємодій користувача. На основі цих даних формується вхідний профіль користувача, який разом із характеристиками доступних курсів передається до компонента `MLModelEngine`.

Модель машинного навчання повертає список рекомендованих курсів разом із відсотками відповідності. Результати упаковуються у вигляді об'єктів `RecommendationsDto[]` та передаються назад до інтерфейсу для відображення користувачу.

На рисунку 2.15 представлено послідовність дій при перегляді сторінки кошика користувача. Після переходу на відповідну сторінку інтерфейс надсилає GET-запит на ендпоінт `/cart`.

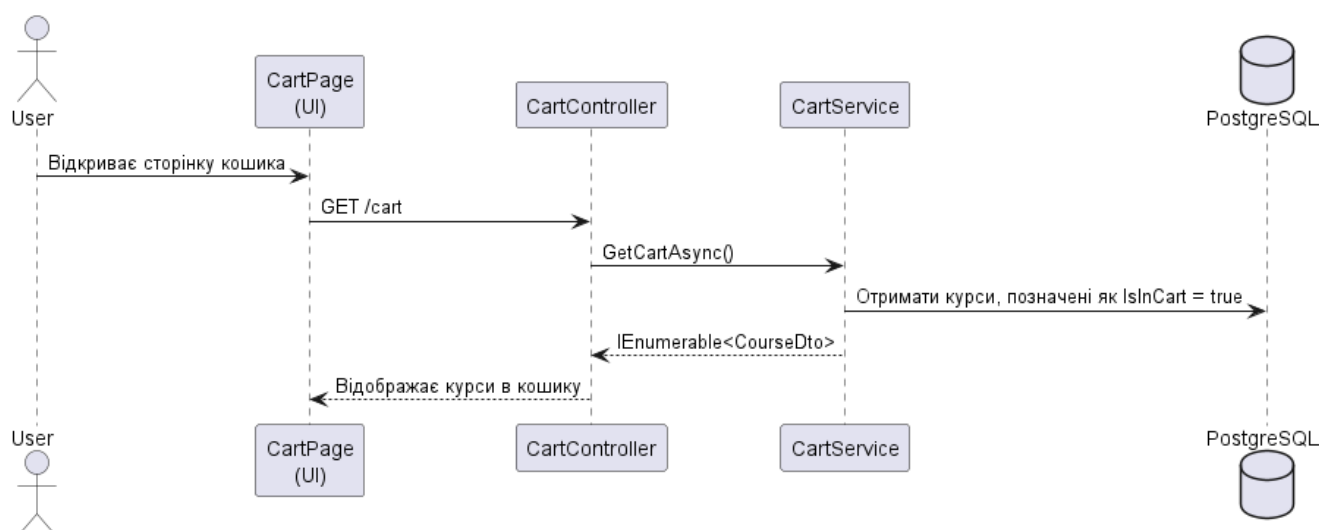


Рисунок 2.15 - діаграма послідовності для перегляду кошика

Джерело: розроблено автором під час проєктування

Контролер `CartController` викликає метод `GetCartAsync()` сервісу `CartService`, який звертається до бази даних для вибірки курсів, для яких у `CourseMeta` встановлено прапорець `IsInCart = true`. У результаті отриманий список об'єктів `CourseDto` повертається на інтерфейс користувача для відображення у кошику.

Було детально змодельовано функціональну поведінку ключових сценаріїв взаємодії користувача та адміністратора з інформаційною системою. Для цього було побудовано діаграму прецедентів (рис. 2.3), яка окреслює загальну

функціональність системи, діаграму класів (рис. 2.1-2.2), що описує структурні взаємозв'язки між основними сутностями, а також одинадцять діаграм послідовності (рис. 2.4–2.15), що ілюструють логіку взаємодії між користувачем, інтерфейсом, контролерами, сервісами та базою даних у різних сценаріях використання.

Побудовані моделі дозволили виокремити основні компоненти системи, механізми обробки запитів, принципи взаємодії між рівнями архітектури, а також зони відповідальності кожного модуля ІУС. Це створює міцний фундамент для подальшого логічного та фізичного проєктування системи в наступних етапах розробки. Було детально змодельовано функціональну поведінку ключових сценаріїв взаємодії користувача та адміністратора з інформаційною системою. Побудовані діаграми послідовності дозволили виокремити основні компоненти та механізми обробки запитів у системі. Це створює фундамент для подальшого проєктування логіки обробки даних, визначення ролей та зон відповідальності кожного модуля ІУС.

2.2 Методи та моделі в інформаційних управляючих системах і технологіях

У сучасних умовах функціонування інформаційних систем важливим завданням є здатність підтримувати прийняття рішень у складних, динамічних та багатофакторних середовищах. В контексті розробки вебплатформи для інтелектуального поширення навчальних матеріалів постає проблема ефективного підбору персоналізованих освітніх курсів для кожного користувача. Це завдання ускладнюється розмаїттям наявних навчальних матеріалів, відсутністю чітких формалізованих залежностей між інтересами користувача та курсами, а також варіативністю поведінки користувачів. Таким чином, постає необхідність у застосуванні методів і моделей, здатних працювати в умовах слабкоструктурованості та неповноти інформації.

Основна задача, яку вирішує інтелектуальна підсистема вебплатформи, — це надання рекомендацій щодо вибору курсів. З погляду математичного моделювання,

це є типовою задачею рекомендаційної системи, що має ознаки багатокритеріального та багатоцільового прийняття рішень. Користувачі мають різні уподобання, рівень підготовки, інтереси й цілі, тому система має забезпечити персоналізований підхід на основі попередніх дій: переглядів, вподобань, коментарів, покупок тощо. У моделі враховується велика кількість змінних, значна частина яких є неявними, що ще раз підтверджує необхідність використання сучасних інтелектуальних методів.

У рамках реалізації рекомендаційної системи було обрано Matrix Factorization (MF) — один з найбільш поширених методів колаборативної фільтрації, який добре зарекомендував себе для задач, що працюють з великими розрідженими матрицями взаємодій. Цей метод дозволяє виявити приховані патерни в даних, проектуючи користувачів та курси у спільний латентний простір, де їхня близькість вказує на ймовірність релевантності. Для реалізації моделі було використано бібліотеку ML.NET, яка надає зручні засоби для побудови, тренування та використання моделей машинного навчання без необхідності переходу на інші технологічні платформи, що особливо важливо для цілісності архітектури системи [46].

Задача рекомендації курсів у цій системі належить до слабкоструктурованих задач, адже лише частина інформації є явно формалізованою, а значна частина — латентною, виведеною з поведінки користувачів. Таким чином, використання моделей штучного інтелекту, зокрема нейронних мереж, є виправданим. Система може бути потенційно розширена до гібридної моделі, що поєднує контентну фільтрацію з колаборативною, а також доповнена нечіткою логікою для адаптації до нових сценаріїв, таких як корпоративне використання платформи [47].

Окрім Matrix Factorization, були розглянуті альтернативні підходи:

- Rule-based системи — прості у впровадженні, але не гнучкі до змін поведінки користувача;
- Content-based фільтрація — використовує метадані курсів, однак не враховує соціальні патерни;

- Гібридні підходи — поєднують переваги різних методів, зокрема корисні для реалізації в корпоративних середовищах з обмеженим числом користувачів.

У перспективі можлива інтеграція генетичних алгоритмів для пошуку оптимального навчального шляху, а також впровадження нейро-нечітких моделей, які забезпечують більш гнучке реагування на зміни у перевагах користувача. Це дозволяє побудувати систему, що не тільки адаптується до поведінки користувача, але й забезпечує стійкість у випадку неповних або суперечливих даних [48].

Таким чином, застосування методу Matrix Factorization в рамках ML.NET забезпечує ефективне вирішення задачі персоналізованої рекомендації курсів. Обраний підхід відповідає сучасним вимогам до систем підтримки прийняття рішень в умовах слабоструктурованої інформації та демонструє гнучкість для подальшого масштабування і вдосконалення через використання елементів штучного інтелекту в майбутніх ітераціях розвитку системи.

РОЗДІЛ 3 РОЗРОБЛЕННЯ ПРОЕКТНИХ РІШЕНЬ ТА ЇХ РЕАЛІЗАЦІЯ

3.1 Проектування бази даних та/або сховища даних для інформаційної управляючої системи

3.1.1 Проектування інфологічної (логічної) моделі БД

У рамках розроблення інформаційної управляючої системи для інтелектуального поширення навчальних матеріалів було спроектовано реляційну базу даних, реалізовану за допомогою системи керування базами даних PostgreSQL. На етапі інфологічного проектування було виокремлено основні інформаційні об'єкти предметної області: користувачі, курси, категорії, специфікації курсів, взаємодії користувачів із курсами, коментарі, трофеї та платіжні засоби. Для кожного об'єкта визначено атрибути та зв'язки між ними. Наприклад, кожен курс може мати багато коментарів і належати до певної категорії, а кожен користувач може додавати коментарі, оцінювати курси та взаємодіяти з ними через систему метаданих CourseMeta.

Інфологічна модель бази даних (рис. 3.1) побудована з урахуванням нормалізації до третьої нормальної форми (3NF) [49], що забезпечує цілісність і узгодженість даних. Єдиним винятком є збереження вкладеної структури кроків курсу (CourseStep) у полі типу jsonb, що є спрощеним способом представлення слабоструктурованих даних і не потребує винесення у окрему таблицю. Інфологічну модель реалізовано в середовищі CASE-засобів на основі діаграм зв'язків «сутність-зв'язок». Нижче подано детальний інфологічний опис полів основних таблиць бази даних.

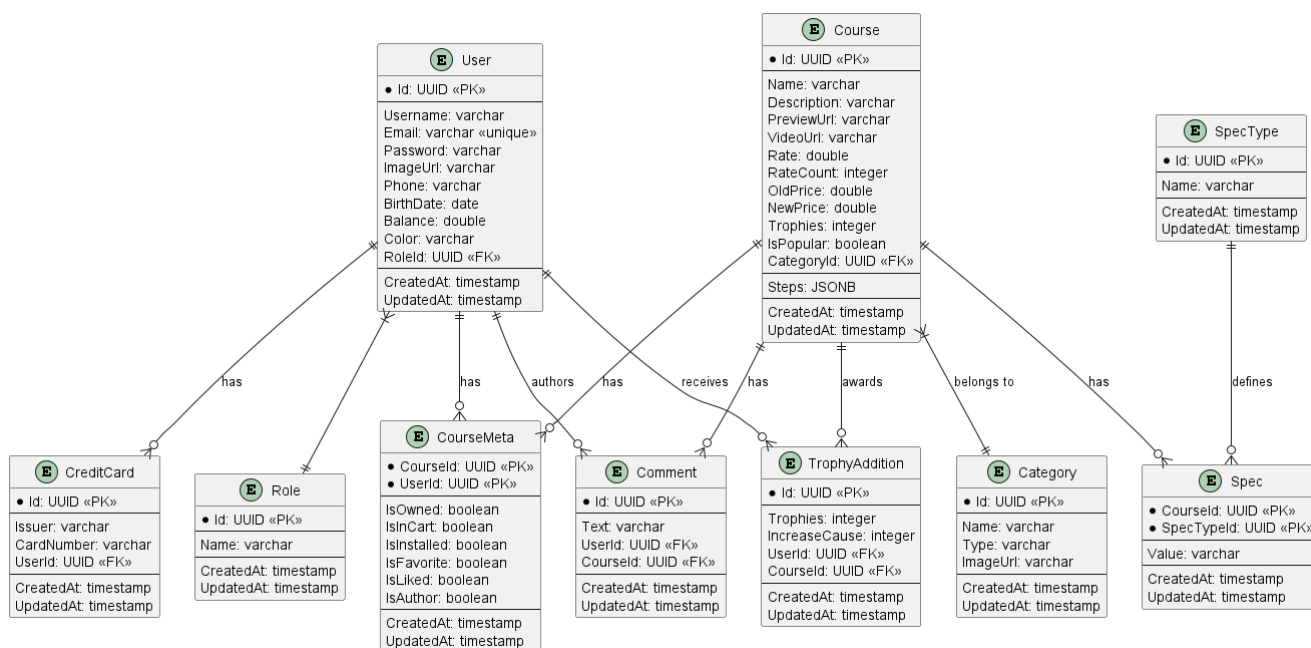


Рисунок 3.1 – Інфологічна модель БД

Джерело: розроблено автором під час проєктування

User — користувач системи:

- Id: унікальний ідентифікатор (PK),
- Username, Email, Password: облікові дані користувача, Email є унікальним,
- ImageUrl, Phone, BirthDate: персональна інформація користувача,
- Balance: грошовий баланс користувача на платформі,
- Color: індивідуальна кольорова схема інтерфейсу,
- RoleId: зовнішній ключ, що вказує на роль користувача,
- CreatedAt, UpdatedAt: автоматичні часові мітки створення та зміни запису.

Role — роль користувача:

- Id: унікальний ідентифікатор (PK),
- Name: текстове позначення ролі (наприклад, "Admin", "User"),
- CreatedAt, UpdatedAt: часові мітки.

CreditCard — платіжна картка користувача:

- Id: унікальний ідентифікатор картки,

- Issuer: назва банку або системи (Visa, MasterCard тощо),
- CardNumber: номер картки,
- UserId: зовнішній ключ до користувача,
- CreatedAt, UpdatedAt: часові мітки.

Course — навчальний курс:

- Id: унікальний ідентифікатор курсу,
- Name, Description: назва та короткий опис курсу,
- PreviewUrl, VideoUrl: посилання на мультимедійні матеріали,
- Rate, RateCount: середній рейтинг та кількість оцінок,
- OldPrice, NewPrice: попередня і поточна ціна курсу,
- Trophies: кількість трофеїв, які можна здобути за проходження курсу,
- IsPopular: логічний прапорець популярності,
- CategoryId: зовнішній ключ до категорії,
- Steps: JSONB-поле зі структурою навчальних кроків,
- CreatedAt, UpdatedAt: часові мітки.

Category — категорія курсу:

- Id: ідентифікатор,
- Name: назва категорії,
- Type: тип або підкатегорія,
- ImageUrl: посилання на зображення,
- CreatedAt, UpdatedAt: часові мітки.

CourseMeta — метадані взаємодії користувача з курсом:

- CourseId, UserId: складений первинний ключ,
- IsOwned, IsInCart, IsInstalled, IsFavorite, IsLiked, IsAuthor: булеві ознаки,
- CreatedAt, UpdatedAt: часові мітки.
- SpecType — тип характеристики:
- Id: унікальний ідентифікатор,
- Name: назва типу (наприклад, "Рівень", "Мова"),

- CreatedAt, UpdatedAt: часові мітки.

Spec — характеристика курсу:

- CourseId, SpecTypeId: складений ключ,
- Value: значення характеристики,
- CreatedAt, UpdatedAt: часові мітки.

Comment — коментар до курсу:

- Id: унікальний ідентифікатор,
- Text: вміст коментаря,
- UserId, CourseId: зовнішні ключі,
- CreatedAt, UpdatedAt: часові мітки.

TrophyAddition — нарахування трофеїв:

- Id: ідентифікатор запису,
- Trophies: кількість трофеїв,
- IncreaseCause: причина нарахування (енум),
- UserId, CourseId: зв'язки,
- CreatedAt, UpdatedAt: часові мітки.

3.1.2 Обґрунтування вибору СКБД та проєктування фізичної моделі БД в середовищі вибраної СКБД

Для реалізації фізичної моделі бази даних обрано PostgreSQL як надійну та потужну систему управління базами даних, яка відповідає критеріям ACID, підтримує складні транзакції та має широку підтримку з боку .NET екосистеми. PostgreSQL також має розвинуті можливості розширення, включаючи підтримку індексів, типів даних jsonb, гнучку схему без обов'язкової жорсткої типізації, що важливо для збереження структурованих і напівструктурованих даних.

Фізична модель (рис 3.2) реалізована за допомогою ORM-бібліотеки Entity Framework Core, яка інтегрується з .NET [44]. Кожна таблиця представлена у вигляді РОСО (Plain Old CLR Object) класів, а всі зв'язки між сутностями реалізовано з використанням FluentAPI. Такий підхід дозволяє точно контролювати

схему бази даних, зберігаючи при цьому читабельність і гнучкість коду [45]. Приклади моделей та конфігурацій доступні у додатках до цієї роботи (див. додаток А).

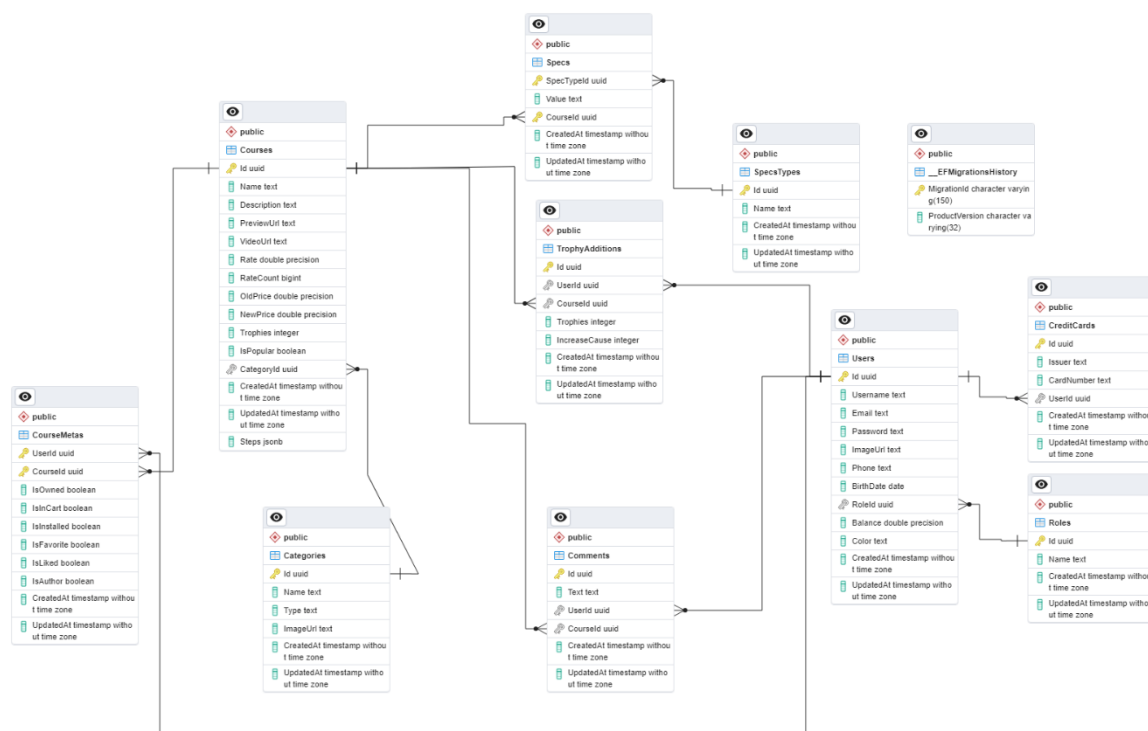


Рисунок 3.2 – Фізична модель БД

Джерело: розроблено автором під час проєктування

Таблиця Users — облікові дані користувачів.

Таблиця Roles — ролі доступу.

Таблиця Courses — опис курсів, включаючи мультимедійні ресурси, оцінки, ціни та структуру кроків.

Таблиця Categories — категоризація курсів.

Таблиця Comments — збереження текстових відгуків.

Таблиця CourseMeta — метайнформація про взаємодію користувачів з курсами.

Таблиця SpecType і Spec — реалізація характеристики курсів.

Таблиця CreditCards — моделювання платіжних засобів.

Таблиця TrophyAdditions — заохочення за активність.

Усі таблиці мають Guid як первинний ключ. Інформація про час створення і зміни зберігається у CreatedAt і UpdatedAt. Було реалізовано унікальний індекс для поля Email у таблиці Users, що забезпечує однозначність облікових записів. Для інших полів, таких як Username чи CardNumber, унікальність підтримується на рівні бізнес-логіки застосунку. Також у таблицях із складеними ключами, зокрема CourseMeta, Spec, використовується комбіноване первинне ключове обмеження (UserId + CourseId або CourseId + SpecTypeId). Реалізація додаткових індексів здійснюватиметься за результатами аналізу продуктивності після впровадження системи., тому їх створення доцільно виконувати на етапі оптимізації продуктивності після запуску системи.

3.1.3 Контрольний приклад та обрахунок прогнозних обсягів БД

Для перевірки працездатності структури бази даних було створено контрольний приклад, що включає завантаження тестових даних до всіх таблиць. Наприклад, система містить: 200 курсів, 1000 користувачів, 3000 записів взаємодії у CourseMeta, 800 коментарів та 1500 записів у TrophyAdditions.

На основі контрольного прикладу виконано розрахунок прогнозного обсягу бази даних. Наприклад, середня довжина запису в таблиці Courses складає 1.2 КБ, у таблиці Users — 0.8 КБ. За умови щомісячного зростання на 5% кількості записів у ключових таблицях, прогнозований приріст за 6 місяців складе понад 30% від початкового обсягу.

Розрахунки проводились вручну та із застосуванням інструментів PostgreSQL. З урахуванням структури індексів, прогнозований обсяг БД за півроку складає близько 120 МБ. Це відповідає цільовому навантаженню для типової навчальної платформи малого та середнього масштабу, із запасом на подальше масштабування.

3.2 Проектування бази знань інформаційної управляючої системи та/або засоби інтелектуального аналізу даних

3.2.1 Організація та проведення інтелектуального аналізу даних з метою пошуку знань в базі даних

У межах реалізації інтелектуального компонента системи поширення навчальних матеріалів було здійснено процес інтелектуального аналізу поведінкових даних користувачів з метою побудови системи рекомендацій. В основу було покладено задачу персоналізованого підбору курсів на базі історії взаємодій користувача з платформою. До таких взаємодій належать: додавання курсів до обраного, придбання, оцінювання, а також інші маркери інтересу. Зважаючи на обсяг і частоту оновлення подібних даних, їх можна класифікувати як ознаки великих даних. Це обумовлює необхідність у застосуванні алгоритмів машинного навчання, здатних виявити латентні закономірності в цих взаємодіях.

Реалізацію інтелектуального аналізу даних було здійснено згідно зі структурою міжгалузевого стандарту CRISP-DM (The Cross-Industry Standard Process for Data Mining). Бізнес-мета проекту полягає у покращенні користувацького досвіду за рахунок надання персоналізованих рекомендацій курсів. На етапі розуміння даних було проаналізовано таблицю CourseMeta, що зберігає детальну інформацію про взаємодії користувачів із курсами. Дані трансформовано у формат матриці взаємодій «користувач – курс», яка є базовою структурою для рекомендаційних моделей.

Для реалізації моделі було обрано бібліотеку ML.NET, яка дозволяє інтегрувати компоненти машинного навчання безпосередньо в .NET-додаток. Використано вбудований алгоритм MatrixFactorizationTrainer, що реалізує метод матричної факторизації — один із найбільш ефективних підходів для побудови рекомендацій у випадках, коли дані мають розріджену структуру. Метод дозволяє проектувати як користувачів, так і курси у спільний латентний простір, у якому їхня близькість інтерпретується як рівень відповідності.

Попередньо дані проходять етап обробки. З таблиці CourseMeta було вибрано лише ті записи, що мають достатній рівень інформативності: курси, які були оцінені або додані до улюбленого. Кожному користувачу та курсу присвоєно числовий ідентифікатор, а взаємодії приведено до шкали від 1 до 5 на основі типу активності. У результаті створено датасет, сумісний із вимогами алгоритму факторизації матриць. На основі цього датасету була виконана побудова навчального та тестового наборів, а також проведене тренування моделі.

Налаштування гіперпараметрів моделі відбувалося експериментальним шляхом. Було протестовано кілька варіантів значень параметрів `NumberOfIterations`, `ApproximationRank`, `LearningRate` та інших. Найкращі результати були досягнуті при використанні 30 ітерацій і рангів латентного простору на рівні 64. Тренування моделі виконано на локальній машині, після чого модель було серіалізовано для подальшого використання в API-сервісі рекомендацій.

Оцінка якості моделі проводилася за допомогою метрик RMSE (корінь середньоквадратичної помилки) та MAE (середня абсолютна похибка). Отримані значення помилок продемонстрували прийнятну точність — достатню для використання в продуктивному середовищі. Результати моделі вбудовано у REST API системи, зокрема в ендпоінт `/suggestions`, який повертає список рекомендованих курсів разом з відсотковим показником відповідності кожного курсу до профілю користувача. Ці рекомендації інтегруються в інтерфейс користувача у відповідному розділі платформи.

Всі етапи роботи над моделлю реалізовано в рамках єдиного пайплайну машинного навчання з використанням ML.NET. Було дотримано принципів повторюваності експериментів і структурованості процесу. У додатках до роботи наведено приклади коду підготовки даних, побудови моделі, її навчання та приклади відповіді API.

3.2.2 Проєктування та реалізація бази знань

У межах реалізації інтелектуального компонента системи поширення навчальних матеріалів було здійснено процес інтелектуального аналізу поведінкових даних користувачів з метою побудови системи рекомендацій. В основу було покладено задачу персоналізованого підбору курсів на базі історії взаємодій користувача з платформою. До таких взаємодій належать: додавання курсів до обраного, придбання, оцінювання, а також інші маркери інтересу. Зважаючи на обсяг і частоту оновлення подібних даних, їх можна класифікувати як ознаки великих даних. Це обумовлює необхідність у застосуванні алгоритмів машинного навчання, здатних виявити латентні закономірності в цих взаємодіях.

Реалізацію інтелектуального аналізу даних було здійснено згідно зі структурою міжгалузевого стандарту CRISP-DM (The Cross-Industry Standard Process for Data Mining). Бізнес-мета проєкту полягає у покращенні користувацького досвіду за рахунок надання персоналізованих рекомендацій курсів. На етапі розуміння даних було проаналізовано таблицю `CourseMeta`, що зберігає детальну інформацію про взаємодії користувачів із курсами. Дані трансформовано у формат матриці взаємодій «користувач – курс», яка є базовою структурою для рекомендаційних моделей.

Для реалізації моделі було обрано бібліотеку ML.NET, яка дозволяє інтегрувати компоненти машинного навчання безпосередньо в .NET-додаток. Використано вбудований алгоритм `MatrixFactorizationTrainer`, що реалізує метод матричної факторизації — один із найбільш ефективних підходів для побудови рекомендацій у випадках, коли дані мають розріджену структуру. Метод дозволяє проєктувати як користувачів, так і курси у спільний латентний простір, у якому їхня близькість інтерпретується як рівень відповідності.

Попередньо дані проходять етап обробки. З таблиці `CourseMeta` було вибрано лише ті записи, що мають достатній рівень інформативності: курси, які були оцінені або додані до улюбленого. Кожному користувачу та курсу присвоєно числовий ідентифікатор, а взаємодії приведено до шкали від 1 до 5 на основі типу

активності. У результаті створено датасет, сумісний із вимогами алгоритму факторизації матриць. На основі цього датасету була виконана побудова навчального та тестового наборів, а також проведене тренування моделі.

Налаштування гіперпараметрів моделі відбувалося експериментальним шляхом. Було протестовано кілька варіантів значень параметрів `'NumberOfIterations'`, `'ApproximationRank'`, `'LearningRate'` та інших. Найкращі результати були досягнуті при використанні 30 ітерацій і рангів латентного простору на рівні 64. Тренування моделі виконано на локальній машині, після чого модель було серіалізовано для подальшого використання в API-сервісі рекомендацій.

Оцінка якості моделі проводилася за допомогою метрик RMSE (корінь середньоквадратичної помилки) та MAE (середня абсолютна похибка). Отримані значення помилок продемонстрували прийнятну точність — достатню для використання в продуктивному середовищі. Результати моделі вбудовано у REST API системи, зокрема в ендпоінт `'/suggestions'`, який повертає список рекомендованих курсів разом з відсотковим показником відповідності кожного курсу до профілю користувача. Ці рекомендації інтегруються в інтерфейс користувача у відповідному розділі платформи.

Всі етапи роботи над моделлю реалізовано в рамках єдиного пайплайну машинного навчання з використанням ML.NET. Було дотримано принципів повторюваності експериментів і структурованості процесу. У додатках до роботи наведено приклади коду підготовки даних, побудови моделі, її навчання та приклади відповіді API.

Таким чином, реалізований процес інтелектуального аналізу даних дозволив отримати набір знань про користувацькі переваги, який безпосередньо використовується для побудови рекомендацій. Це формує основу для подальшого наповнення та актуалізації бази знань системи, що дозволяє платформі адаптуватися до інтересів користувача в динамічному режимі.

На основі отриманих у процесі аналізу знань було реалізовано базу знань системи у вигляді серіалізованої рекомендаційної моделі, що є частиною сервісу

'RecommendationService'. Проектування бази знань виконано за класичними принципами, адаптованими до архітектури інтелектуального застосунку. На етапі ідентифікації було встановлено, що головним завданням є рекомендація користувачеві релевантних курсів. Це задача з ознаками слабкої структурованості, яка базується на латентних характеристиках. Джерелом знань стали результати матричної факторизації — зважені коефіцієнти подібності між користувачами та курсами.

Концептуалізація бази знань полягала у виокремленні таких понять, як «користувач», «курс» та «рівень релевантності». Взаємозв'язки між ними формуються через числову оцінку відповідності (score), що розраховується моделлю. Знання, що формують базу, не є жорстко структурованими правилами, як у класичних експертних системах, а представлені у вигляді прогнозованих показників, які постійно оновлюються відповідно до динаміки користувацької поведінки.

Формалізація бази знань реалізована шляхом подання результатів у структурі (UserId, CourseId, Score), яка доступна для запитів API. Механізм логічного висновку полягає у ранжуванні рекомендацій за отриманим score. Інтерпретація здійснюється інтерфейсом користувача або через бізнес-логіку, що визначає критерії відображення. Сам механізм оновлення бази знань передбачає перенавчання моделі та її повторну серіалізацію.

Реалізація бази знань здійснена у вигляді компонента, що підвантажується в оперативну пам'ять сервісу при запуску системи. Завдяки цьому забезпечується висока швидкодія відповіді на запити '/suggestions'. На етапі тестування було перевірено відповідність рекомендацій реальним інтересам користувачів — на основі зворотного зв'язку та експертного оцінювання результатів. Дослідна експлуатація довела придатність системи до використання в реальних умовах, і на її основі можливе подальше вдосконалення — зокрема за рахунок багатоканального навчання, впровадження контентних фільтрів чи гібридних моделей.

3.3. Програмне забезпечення

3.3.1 Вимоги до програмного забезпечення

Проектування програмного забезпечення системи інтелектуального поширення навчальних матеріалів передбачало створення багаторівневої архітектури, орієнтованої на забезпечення масштабованості, модульності, легкості розгортання та підтримки. Основною метою програмного забезпечення було забезпечити ефективну взаємодію між користувачем і навчальним контентом, автоматизувати процес рекомендацій курсів на основі поведінкових характеристик та надати зручний інтерфейс для керування контентом як кінцевим користувачем, так і адміністраторам системи.

У процесі розробки було обрано ітеративну модель із застосуванням Kanban-дошки GitHub Projects для візуального контролю прогресу. Також активно застосовувався підхід Rapid Application Development (RAD), що дозволив швидко створити прототип і здійснити ранню валідацію архітектурних рішень. У процесі розробки використовувалися автоматизовані CI/CD-процеси, що значно пришвидшило виведення нових функцій у продуктивне середовище.

Серверна частина системи реалізована мовою C# із використанням фреймворку ASP.NET Core. Для моделювання та доступу до реляційної бази даних PostgreSQL використовувалася ORM-бібліотека EF Core, що забезпечує ефективну роботу з даними та реалізацію бізнес-логіки. Компонент для побудови персоналізованих рекомендацій розроблено з використанням ML.NET і реалізовано як окремий сервіс. Клієнтська частина реалізована на основі ReactJS з використанням TypeScript, Vite як інструменту збірки, TanStack Query для роботи з API, а також бібліотеки компонентів Ant Design. Такий стек забезпечує гнучку, швидко та реактивну побудову інтерфейсу користувача.

У рамках архітектури серверної частини реалізовано шаблон Controller-Service-Repository, що дозволяє чітко розмежувати відповідальність між рівнями логіки. DTO-класи використовуються для уніфікації даних, що передаються між клієнтом та сервером. Для автентифікації застосовано JWT токени, що забезпечує

безпечну передачу ідентифікаційної інформації між клієнтом та сервером. Рекомендаційний сервіс інкапсулює логіку роботи з навченою моделлю ML.NET та є доступним через окрему кінцеву точку API /suggestions.

Програмне забезпечення передбачає підтримку наступних функціональних блоків: реєстрація та авторизація користувача, перегляд каталогу курсів, додавання курсів до кошика, оформлення замовлень, формування бібліотеки придбаних курсів, можливість залишати коментарі та оцінки, управління категоріями курсів (для адміністратора), а також модуль рекомендацій курсів користувачам.

У розгортанні системи (рис 3.3) використовується контейнеризація за допомогою Docker. Завдяки цьому кожен інстанс системи (наприклад, для конкретного підприємства) може бути розгорнутий окремо. Хмарна платформа Railway обрана як сервіс для хостингу, оскільки вона надає інструменти CI/CD, логування та керування ресурсами без зайвого конфігурування. Це дозволяє мінімізувати витрати на підтримку інфраструктури й пришвидшує масштабування системи. Розгортання системи в Railway наведено на рисунку 3.4.

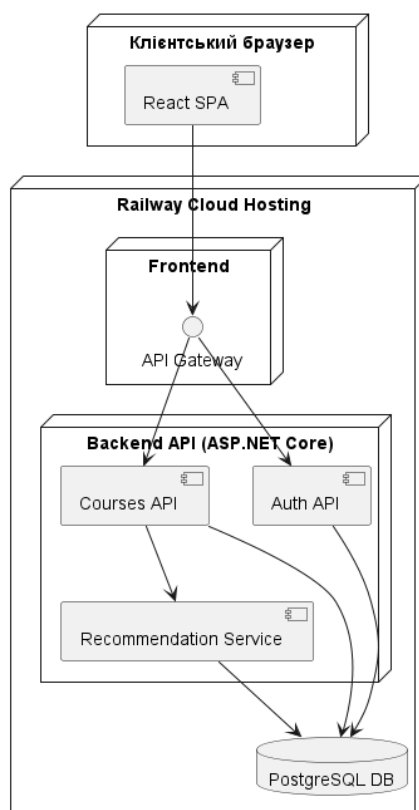


Рисунок 3.3 – діаграма розгортання проєкту

Джерело: розроблено автором під час проєктування

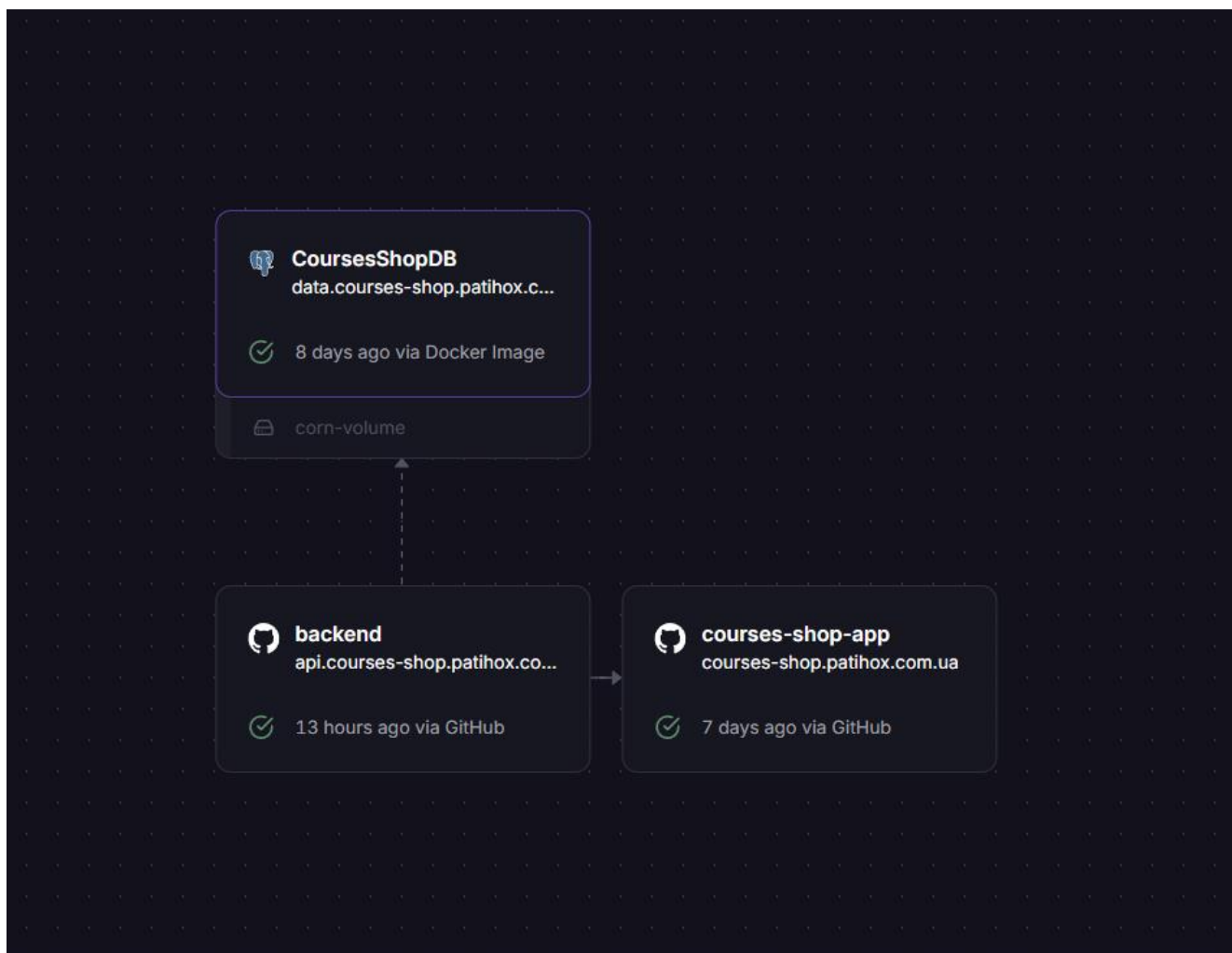


Рисунок 3.4 – розгортання системи в Railway

Джерело: створено автором під час розробки

Загальна структура ПЗ побудована на принципах модульності та гнучкості. Кожен компонент легко розширюється або замінюється, що відповідає принципам підтримуваності та адаптивності сучасних інформаційних систем. Структура дозволяє масштабувати окремі модулі незалежно один від одного, наприклад, запускати кілька екземплярів Recommendation Service або переносити клієнтську частину на окремий домен.

3.3.2 Архітектура програмного забезпечення

Вибір архітектури програмного забезпечення став одним із найважливіших рішень у процесі проєктування та реалізації системи інтелектуального поширення навчальних матеріалів. Це рішення безпосередньо вплинуло на подальший

розвиток системи, її функціональні можливості, гнучкість та масштабованість. Враховуючи ключові потреби проєкту — швидкість реалізації MVP-версії, легкість розгортання, можливість автономної роботи інстансів системи та низький поріг входження для технічної підтримки — було прийнято рішення про використання монолітної архітектури з чіткою внутрішньою модуляцією. Такий підхід дозволив централізовано управляти всіма компонентами, зменшити кількість зовнішніх інтеграцій та підвищити стабільність роботи системи.

Монолітна архітектура в даному випадку не означає повну відсутність гнучкості: завдяки контейнеризації із застосуванням технології Docker було досягнуто високого рівня ізоляції середовищ. Це дозволило розгорнути окремі версії системи для різних підприємств або підрозділів, реалізуючи принцип багатокористувацької мультиінстансної моделі. Застосування хмарного хостингу Railway забезпечує автоматичну обробку CI/CD, масштабування, моніторинг стану системи, ведення журналів та конфігурування середовищ без необхідності глибоких технічних знань зі сторони обслуговуючого персоналу. Таким чином, платформа зберігає гнучкість розгортання і підтримує просте масштабування при зростанні кількості користувачів.

Система підтримує механізм автентифікації, побудований на основі JWT (JSON Web Token). Цей стандарт де-факто є найпопулярнішим способом передачі ідентифікаційних даних у вебзастосунках [50]. Після успішної аутентифікації користувач отримує підписаний токен, який зберігається на стороні клієнта та передається з кожним запитом до API. JWT містить лише мінімальні дані: унікальний ідентифікатор користувача та дату закінчення дії токена. Важливо зазначити, що в JWT не зберігається інформація про роль користувача. Це пов'язано з тим, що права доступу можуть бути змінені в будь-який момент, і тому інформація про роль динамічно отримується з бази даних під час кожного запиту. Це дозволяє системі зберігати централізований контроль над правами доступу без потреби перевипуску токенів.

Механізм авторизації реалізовано у вигляді перевірок на сервері, де кожен запит спочатку обробляється на предмет наявності та валідності токена. Далі з бази

даних витягується актуальна інформація про користувача, зокрема його роль, яка й використовується для подальшої логіки доступу. Такий підхід дозволяє швидко адаптувати систему під змінні політики безпеки підприємства та оперативно реагувати на необхідність зміни прав користувачів. Реалізована перевірка прав базується на сервісному шарі бізнес-логіки, що дозволяє не лише перевіряти ролі, а й враховувати додаткові умови (наприклад, статус облікового запису, час останньої активності тощо) [51].

Крім ідентифікації та авторизації, значну увагу приділено захисту даних під час їх передавання. Усі запити між клієнтом і сервером відбуваються виключно через захищений протокол HTTPS, який реалізовано на базі стандарту SSL/TLS. Сертифікація і підтримка захищеного з'єднання виконується автоматично завдяки можливостям Railway або проксі-серверів (наприклад, Cloudflare). Це гарантує цілісність та конфіденційність усіх даних, які передаються між клієнтом і сервером, включаючи паролі, особисту інформацію та маркери автентифікації.

Профілювання користувачів реалізовано через систему ролей, яка зберігається в базі даних та оновлюється адміністративним шляхом. У поточній реалізації підтримуються чотири рівні доступу: Guest (неzareєстровані користувачі), User (авторизовані користувачі платформи), Developer (користувачі, які створюють і публікують курси) та Admin (адміністратори системи, що мають повний доступ до управління контентом і користувачами). Всі перевірки прав доступу виконуються в ручному режимі на рівні API або сервісного шару, що забезпечує повний контроль і гнучкість у визначенні політик доступу.

У підсумку, обрана архітектура демонструє вдалий компроміс між простотою реалізації та підтримки і потенціалом для масштабування. Такий підхід особливо ефективний для систем типу MVP або внутрішньокорпоративних рішень, де критичними є швидкий час виходу на ринок, мінімізація витрат на інфраструктуру та висока стабільність. Водночас, модульна побудова дозволяє у майбутньому поступово перейти до гібридної або мікросервісної архітектури без кардинальної перебудови системи.

3.3.3 Інструментальні засоби розробки для створення прикладного програмного забезпечення інформаційних управляючих систем

У процесі реалізації інформаційної управляючої системи було застосовано сучасний повноцінний стек технологій, що охоплює інструменти як для серверної, так і клієнтської частини застосунку. Такий комплексний підхід дозволив досягти високої швидкості розробки, надійності функціонування, гнучкої масштабованості та простоти подальшого супроводу. Кожен компонент інструментарію обирався на основі низки факторів: продуктивність, активність і підтримка спільноти, легкість у вивченні та інтеграції, наявність документації та зручність розгортання на практиці. Додатково враховувались особистий досвід розробника та відповідність конкретним вимогам, продиктованим архітектурою та функціональністю системи.

Серверна частина програмного забезпечення була реалізована з використанням мови програмування C#, яка поєднує типобезпечність, високу продуктивність і підтримку широкого спектра парадигм програмування. У поєднанні з ASP.NET Core — кросплатформним, модульним та гнучким фреймворком від Microsoft — це забезпечило можливість створення масштабованого та швидкого RESTful API, придатного до розширення і повторного використання. Особливістю використаного фреймворку є підтримка middleware, dependency injection з коробки, а також гнучка система маршрутизації. Для зв'язку з реляційною базою даних PostgreSQL було обрано Entity Framework Core (EF Core), який виконує роль ORM і дозволяє працювати з БД через високорівневі запити LINQ. Сутності в системі реалізовані у вигляді POCO-класів, що полегшує підтримку чистої архітектури. Зв'язки між сутностями реалізовано через FluentAPI, що забезпечує гнучкість у конфігуруванні моделей без потреби в атрибутах.

Керування підключеннями до бази даних реалізовано автоматично через DbContext, який підтримує пулінг підключень, дозволяючи повторно використовувати з'єднання та зменшувати накладні витрати на підключення. Усі запити до БД реалізовано асинхронно з використанням async/await, що є

стандартом у сучасних .NET-застосунках і дозволяє ефективно обробляти одночасні запити без блокування потоків. Крім того, структура бізнес-логіки чітко поділена на сервіси, які відповідають за реалізацію конкретних функціональностей, що значно полегшує тестування, рефакторинг і масштабування системи.

Клієнтська частина застосунку створена з використанням бібліотеки ReactJS — однієї з найпопулярніших бібліотек для побудови користувацьких інтерфейсів. React дозволяє будувати динамічні, реактивні інтерфейси, що легко масштабуються, і відмінно підходить для SPA (Single Page Applications). У зв'язці з мовою TypeScript, яка забезпечує статичну типізацію, це дозволяє уникати багатьох помилок ще на етапі компіляції, полегшує рефакторинг та значно підвищує надійність коду. Для збору і запуску проєкту застосовано Vite — високопродуктивний інструмент нового покоління для frontend-розробки, що забезпечує миттєву перезбірку та гаряче оновлення під час розробки.

У клієнтському застосунку реалізовано ефективну взаємодію з сервером завдяки бібліотеці TanStack Query, яка відповідає за виконання HTTP-запитів, кешування даних, автоматичну інвалідацію кешу, фонове оновлення та обробку станів завантаження. Це дозволяє значно зменшити кількість дубльованих запитів, оптимізувати трафік та покращити користувацький досвід. Інтерфейс побудований на основі бібліотеки компонентів Ant Design, яка надає готові елементи керування та дозволяє реалізувати сучасний, зручний та візуально уніфікований UI. Структура компонентів дотримується принципів розділення відповідальності, логіка винесена в окремі хуки, що спрощує повторне використання та тестування логіки відображення.

База даних побудована на основі PostgreSQL — потужної об'єктно-реляційної СКБД, яка відзначається стабільністю, масштабованістю та високою продуктивністю. Для адміністрування та проєктування структури бази даних активно використовувався інструмент pgAdmin — офіційне графічне середовище для роботи з PostgreSQL. Завдяки pgAdmin розробник мав можливість зручно аналізувати структуру таблиць, створювати зв'язки, генерувати SQL-запити, переглядати дані та керувати транзакціями. PostgreSQL підтримує складні типи

даних, включаючи масиви, JSON та JSONB. У даній системі особливу роль відіграє поле Steps у таблиці Course, що реалізовано у вигляді JSONB-масиву, що містить структуру курсу. Це дозволяє зберігати структуровані вкладені об'єкти без потреби створення окремої таблиці, зберігаючи при цьому можливість ефективного запиту, фільтрації та сортування. Зв'язки між таблицями реалізовано за допомогою зовнішніх ключів, що гарантує цілісність даних. У складних випадках використано складені ключі для представлення зв'язків типу «багато-до-багатьох».

Вся розробка системи — як серверної, так і клієнтської частини — здійснювалася виключно у Visual Studio Code, що є потужним, розширюваним редактором коду з підтримкою великої кількості плагінів. Для розмежування середовищ розробки використовувалися окремі робочі простори (workspaces) для backend і frontend частин. Git був основним засобом контролю версій, а GitHub — хостингом репозиторію та засобом для організації командної роботи, pull request-ів та перевірок змін. Автоматизація розгортання реалізована за допомогою Railway — платформи для CI/CD та хостингу, що дозволяє безперешкодно деплоїти як backend, так і frontend після злиття змін у основну гілку. Railway забезпечує також базові засоби моніторингу, журналювання та конфігурації змінних середовища.

Застосований стек технологій дозволяє реалізовувати як мінімально життєздатну версію продукту (MVP), так і масштабовані рішення, які в перспективі можуть бути адаптовані до мікросервісної архітектури. Наприклад, у разі зростання навантаження або збільшення кількості користувачів можна інтегрувати Redis для кешування найбільш затребуваних запитів або винести сервіс рекомендацій у окремий сервіс з подієвою комунікацією через RabbitMQ чи Apache Kafka. Завдяки обраному інструментарію, проєкт є гнучким, підтримуваним та готовим до подальшого розвитку у відповідності до потреб користувачів і замовника.

3.3.4 Керівництво (інструкція) користувача

У процесі розробки та впровадження інформаційної управляючої системи одним з найважливіших етапів стало тестування програмного забезпечення. Його метою є забезпечення працездатності системи, виявлення можливих помилок і

дефектів, оцінка відповідності технічним вимогам, а також формування впевненості у надійності та якості кінцевого продукту. Для досягнення цієї мети було проведено ручне тестування з елементами модульного, інтеграційного та системного рівнів, використовуючи сучасні інструменти тестування REST API. Результати тестування оформлені у вигляді тест-кейсів та проілюстровані скріншотами у додатках.

Проте для забезпечення ефективної експлуатації системи кінцевими користувачами також необхідно надати чітке і зрозуміле керівництво з користування вебплатформою. Це дає змогу забезпечити коректне використання всіх можливостей системи без додаткової технічної підтримки та зменшити поріг входу для нових користувачів.

Після відкриття головної сторінки платформи гість може ознайомитись із каталогом доступних курсів (рис. 3.5).

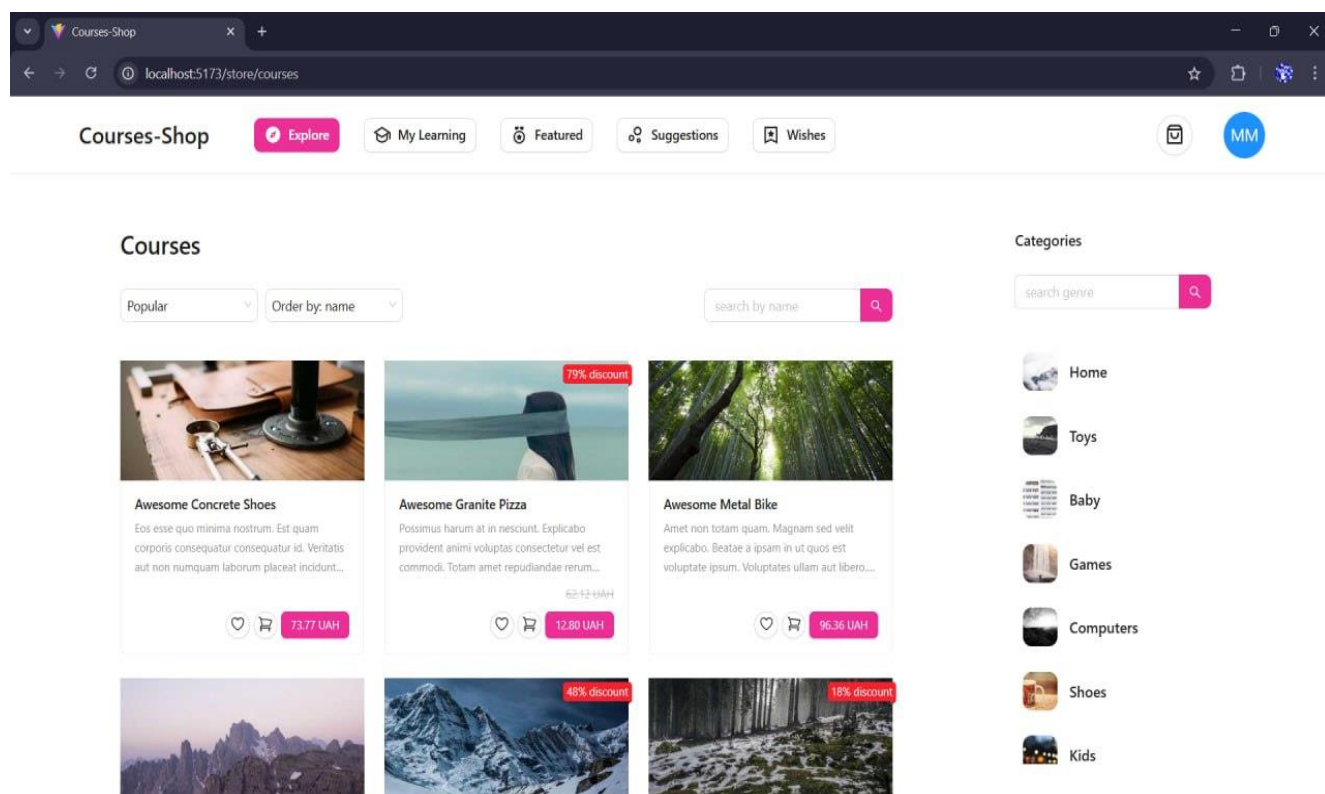


Рисунок 3.5 – Головна сторінка системи

Джерело: створено автором в результаті розробки

Для доступу до функціональності, пов'язаної з персоналізованим навчанням, необхідно зареєструватись або увійти до системи.

Реєстрація виконується шляхом заповнення простої форми (рис. 3.6) з ім'ям користувача, email та паролем. Після цього система автоматично виконує вхід і видає JWT токен, що дозволяє надалі ідентифікувати користувача в системі.

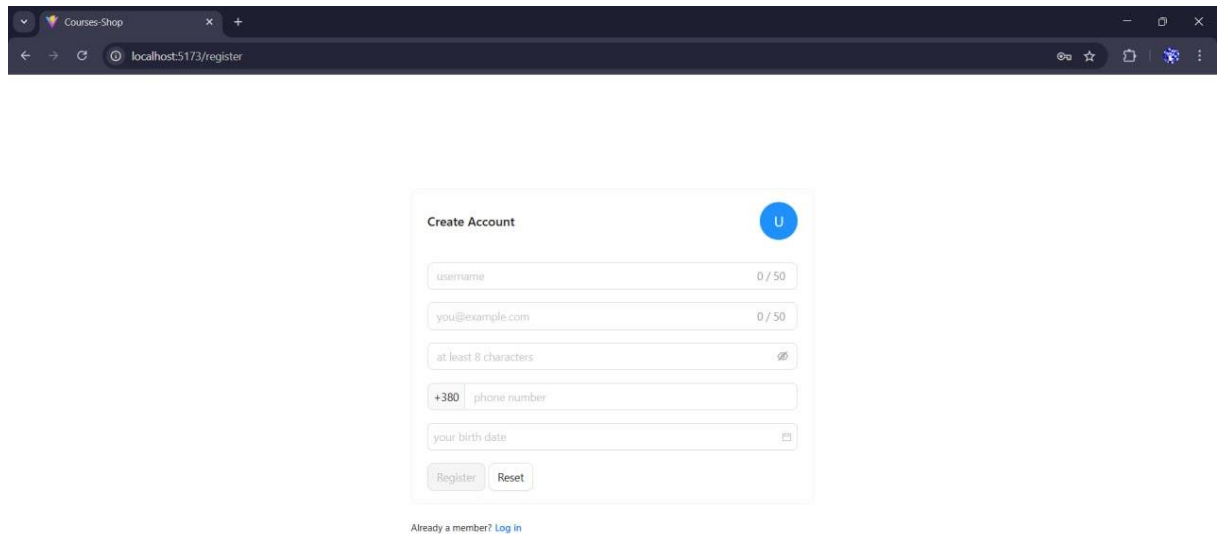


Рисунок 3.6 – Сторінка реєстрації

Джерело: створено автором в результаті розробки

Також на формі реєстрації користувач може персоналізувати власне зображення (рис. 3.7)

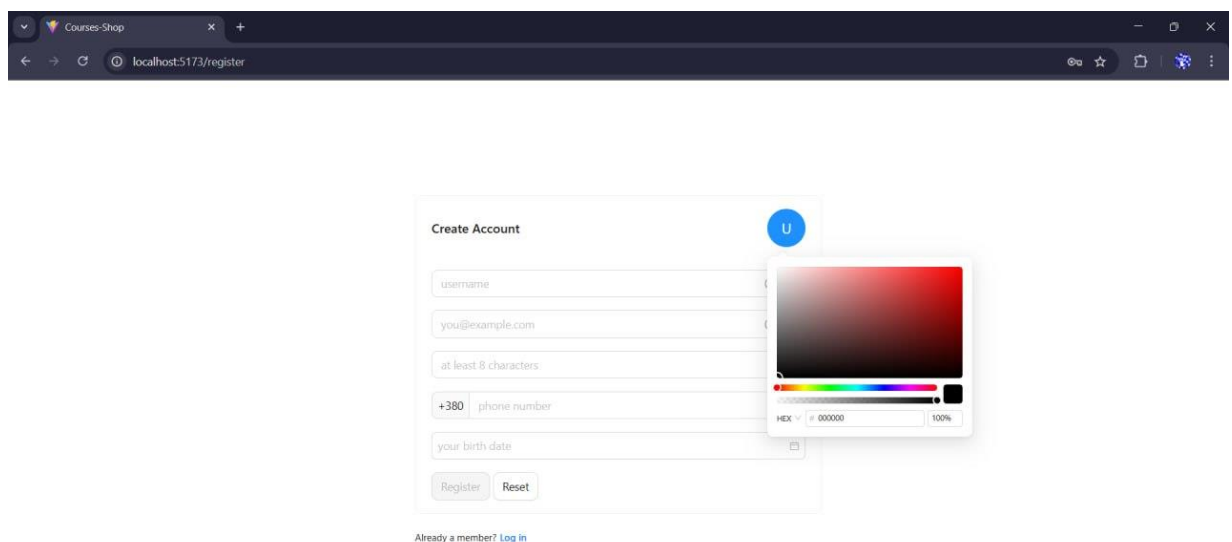


Рисунок 3.7 – Модуль вибору кольору

Джерело: створено автором в результаті розробки

Якщо користувач вже має акаунт, він може пройти авторизацію (рис 3.8), від час якої йому треба буде вказати адресу електронної пошти від існуючого профіля і коректний пароль

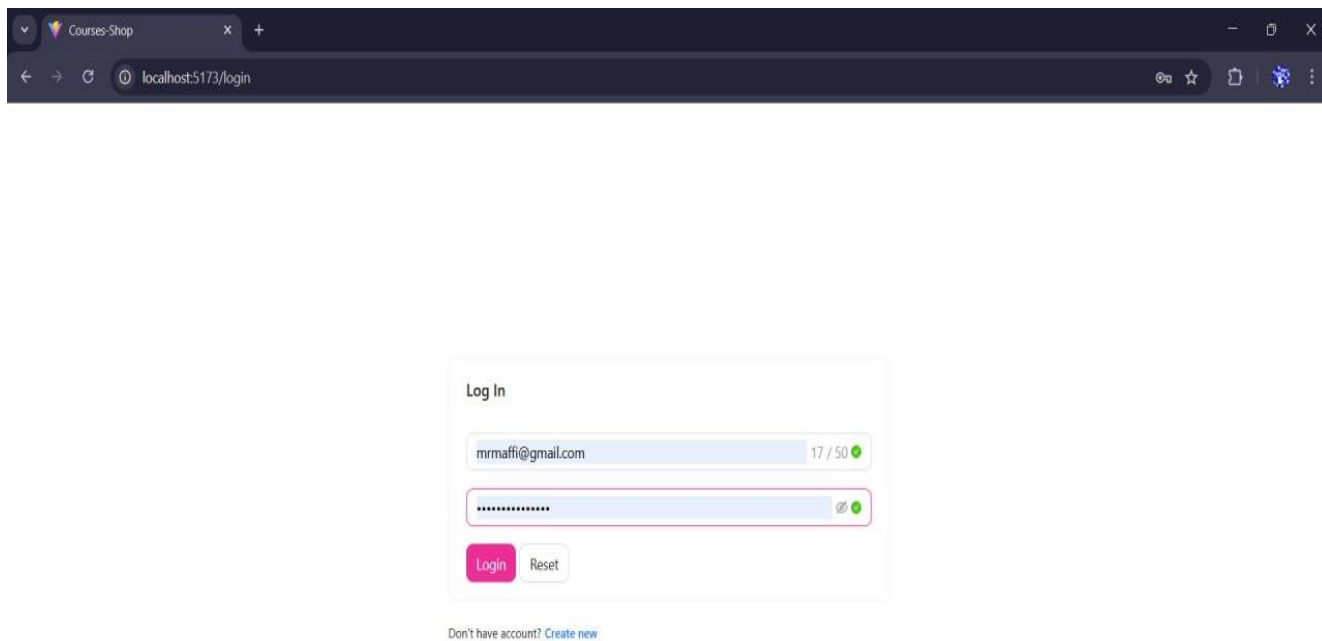


Рисунок 3.8 – Форма авторизації

Джерело: створено автором в результаті розробки

Після авторизації користувач отримує доступ до додаткових можливостей: додавання курсів до кошика, перегляду особистої бібліотеки придбаних курсів, коментування курсів, а також отримання персоналізованих рекомендацій.

Кожен курс містить сторінку з детальним описом, попереднім переглядом відео та відгуками інших користувачів. Для взаємодії з курсом передбачено функціональність лайків, додавання до обраного, оцінки у вигляді зірок тощо (рис. 3.9).

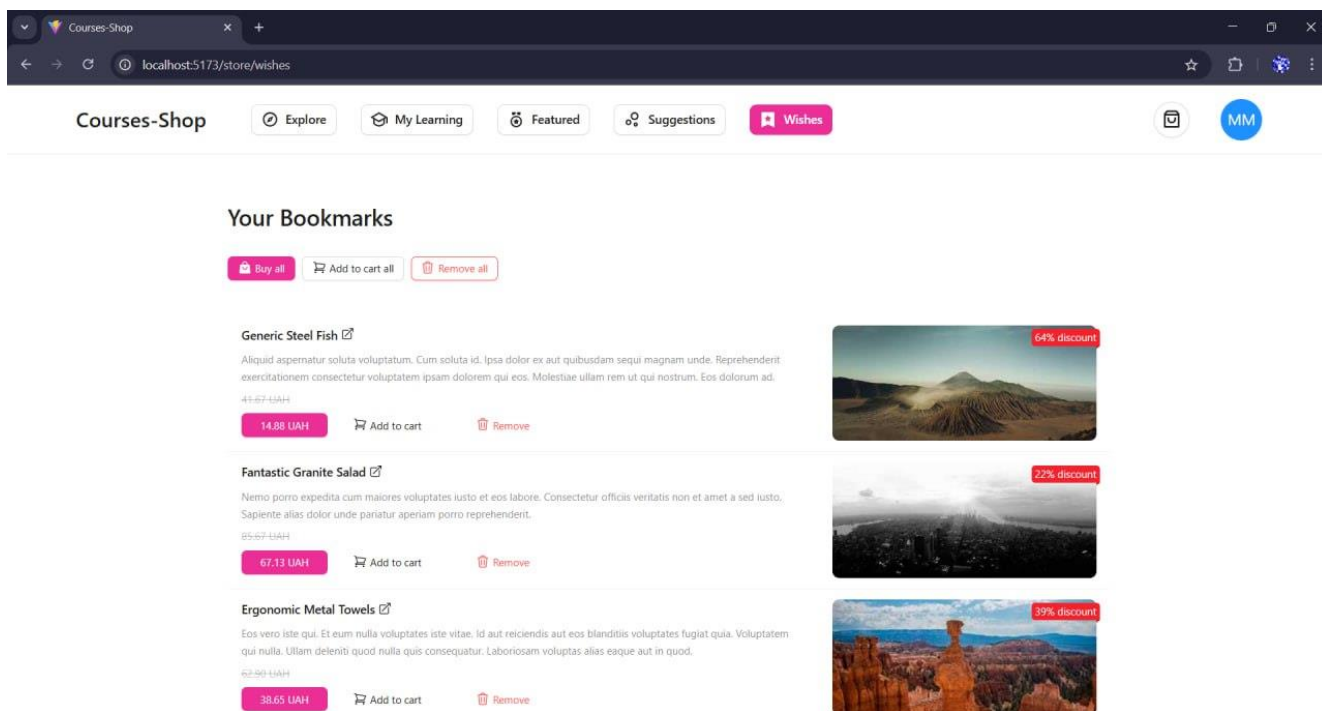


Рисунок 3.9 – Перелік вподобаних курсів

Джерело: створено автором в результаті розробки

Щоб придбати курс, користувач має додати його до кошика (рис. 3.11).

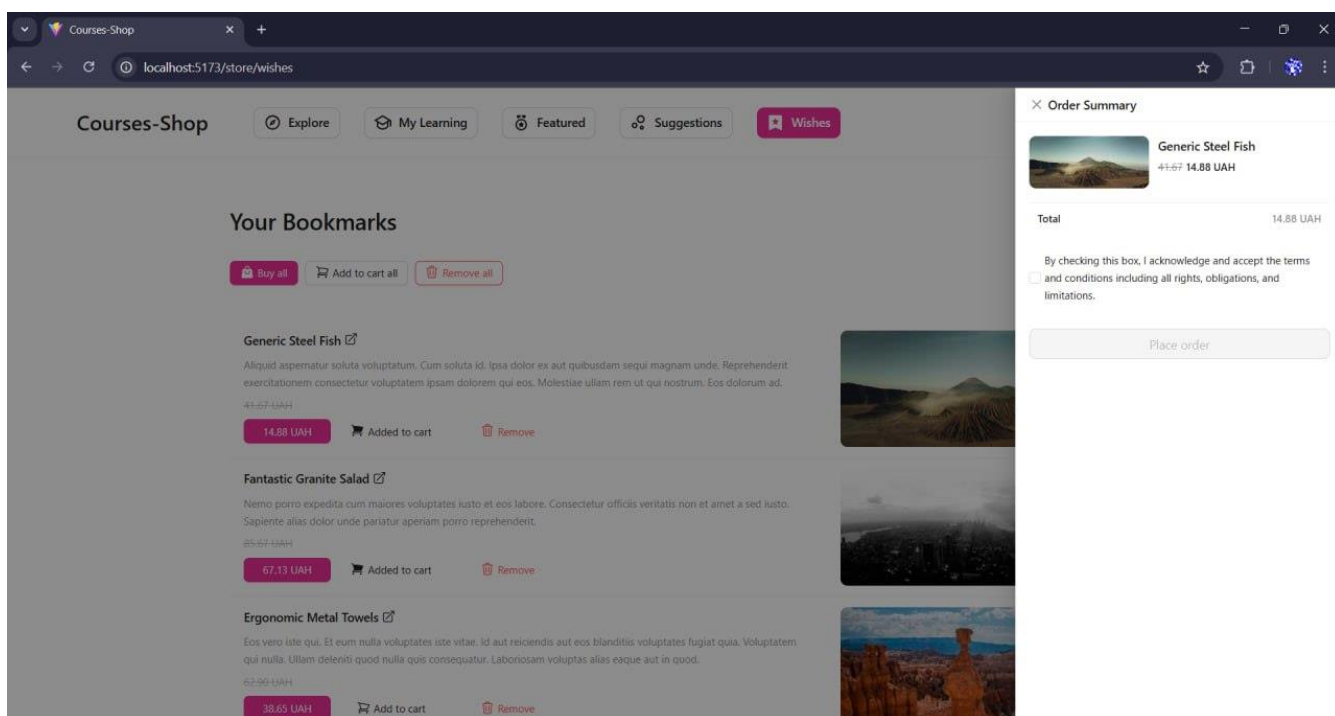


Рисунок 3.11 – Додавання курсу до кошику

Джерело: створено автором в результаті розробки

Після додавання потрібних курсів в кошик, користувач повинен натиснути кнопку оформлення замовлення (рис 3.12).

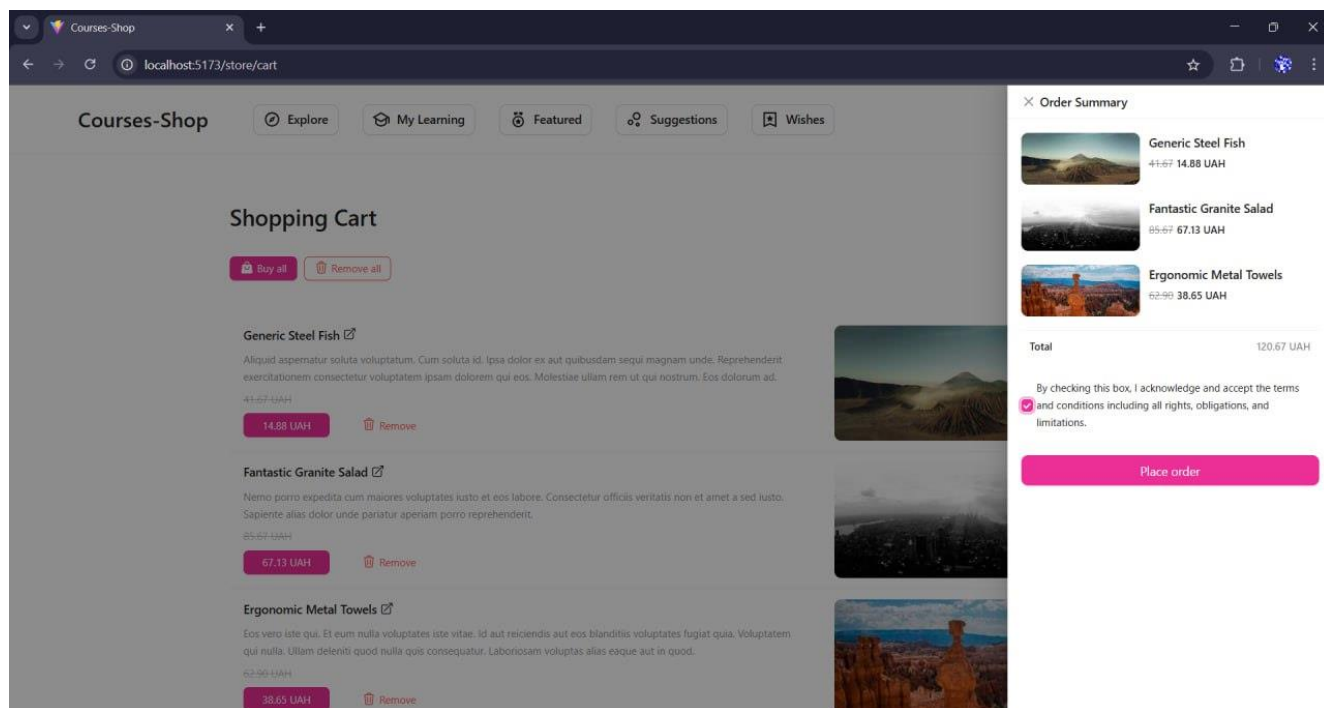


Рисунок 3.12 – Підтвердження замовлення

Джерело: створено автором в результаті розробки

Після підтвердження — курс з'являється у розділі «Моя бібліотека», і стає доступним для перегляду без обмежень (рис. 3.13).

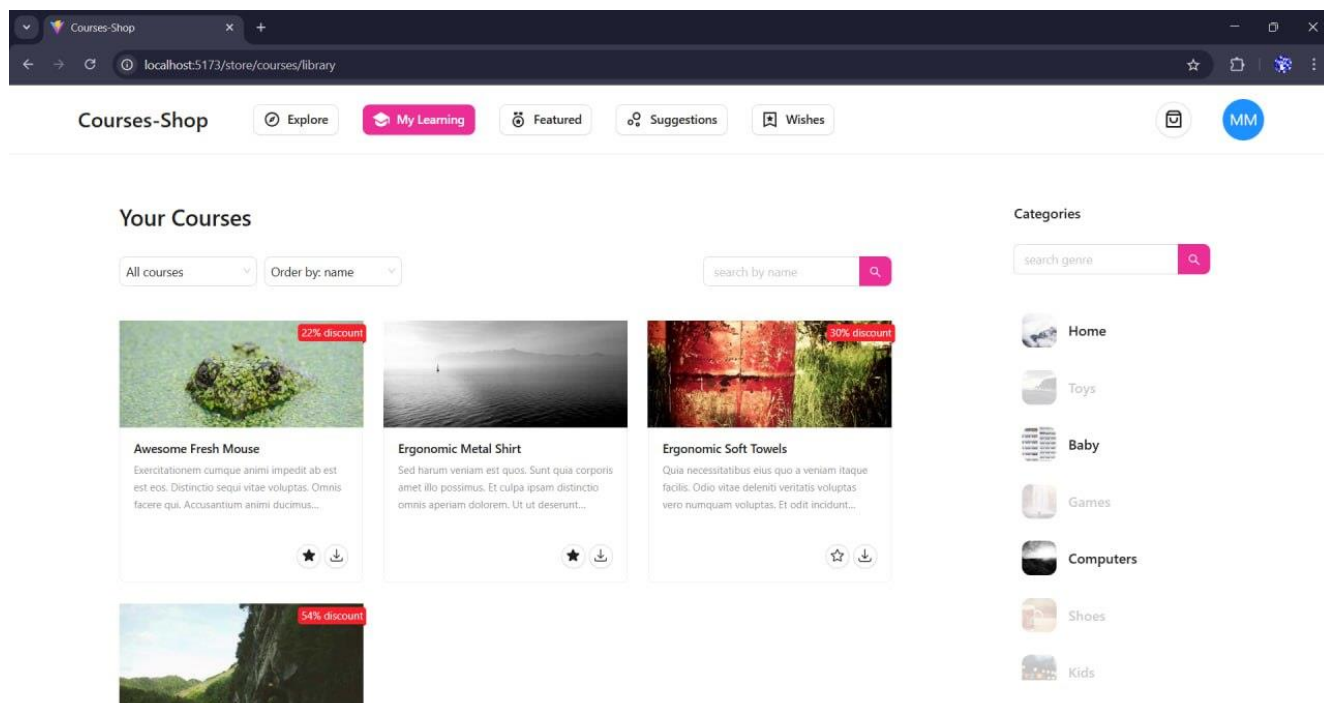


Рисунок 3.13 – Перелік власних курсів користувача

Джерело: створено автором в результаті розробки

У розділі рекомендацій («Рекомендоване») користувач отримує персональні поради щодо того, які курси можуть бути йому цікаві. Система формує ці рекомендації на основі вподобань, минулих дій, рейтингу курсів та нейромережевої моделі, натренованої на великому датасеті.

Адміністративна панель доступна тільки користувачам із відповідним рівнем доступу (роль Admin або Developer). Через неї можна керувати категоріями курсів, публікувати нові матеріали, а також переглядати статистику активності.

Таким чином, керівництво користувача охоплює повний цикл взаємодії із системою: від першого входу на платформу до активного використання навчальних можливостей. Інтерфейс спроектовано з урахуванням принципів простоти та інтуїтивності, що дає змогу користувачам швидко адаптуватись до роботи з системою та ефективно використовувати її функціонал.

3.4. Технічне забезпечення

3.4.1 Обґрунтування вибору технічного забезпечення

У рамках реалізації інформаційної управляючої системи для розповсюдження навчальних курсів особливу увагу було приділено вибору технічного забезпечення, яке є критично важливим для стабільного функціонування, високої доступності та гнучкої масштабованості програмного продукту. Зважаючи на вимоги до розгортання, супроводу та оновлення системи, було прийнято рішення реалізувати її як веборієнтовану хмарну платформу. Такий підхід дозволяє уникнути витрат на підтримку фізичних серверів, підвищує мобільність обслуговування та дозволяє оперативно реагувати на зміну навантаження.

Основним середовищем для розміщення системи обрано Railway — сучасну хмарну платформу, яка надає вбудовану підтримку повного життєвого циклу розгортання вебдодатків. Railway забезпечує зручне середовище для виконання серверної логіки, обробки HTTP-запитів, розміщення та обслуговування бази даних PostgreSQL, а також безперебійну інтеграцію з системами контролю версій

(GitHub) та CI/CD. Це суттєво прискорює цикл розробки, полегшує автоматизацію релізів та значно спрощує адміністрування.

Railway було обрано серед ряду аналогів — Heroku, Render, Netlify, Vercel — після аналізу функціональних можливостей, стабільності та зручності використання. Однією з ключових переваг Railway є інтеграція серверної, клієнтської частин і бази даних у межах одного середовища, що усуває потребу в окремому керуванні різними компонентами на сторонніх провайдерах. Крім цього, Railway пропонує інструменти для зручного моніторингу в реальному часі, гнучке налаштування змінних середовища, масштабування ресурсів за потреби, а також резервне копіювання баз даних. Ці фактори роблять Railway оптимальним вибором для вебрішень із середнім навантаженням, таких як освітні платформи.

Фактична конфігурація Railway для розміщення системи передбачає розгортання сервісу в регіоні EU West (Амстердам, Нідерланди). Для виконання основної репліки використовується потужна інфраструктура Metal (New). Станом на момент написання роботи кількість активних реплік становить одну, що є достатнім для підтримки платформи у звичайному режимі експлуатації. При цьому встановлені ресурсні обмеження досягають максимально допустимих в межах тарифного плану:

- 8 віртуальних процесорів (vCPU),
- 8 гігабайтів оперативної пам'яті (RAM).

На стороні клієнта не передбачається використання спеціального програмного забезпечення. Всі користувачі взаємодіють із системою через сучасні веббраузери, які підтримують стандарти HTML5, CSS3 і JavaScript (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari тощо). Такий підхід дозволяє забезпечити широке охоплення користувачів без додаткових вимог до їхніх пристроїв. Для розробки та супроводу програмного забезпечення було використано текстовий редактор Visual Studio Code, що надає зручні інструменти для роботи з кодом, інтеграцію з системами контролю версій, налагодженням, а також плагіни для роботи з C#, TypeScript, Docker та PostgreSQL.

3.4.2 Ресурсні вимоги до технічного забезпечення

Ресурсні вимоги до стабільної роботи інформаційної системи визначались з урахуванням типових навантажень, кількості одночасних користувачів, складності обробки запитів, а також потреб у збереженні та обслуговуванні навчального контенту. За результатами проведеного аналізу, було визначено, що для серверної частини — ASP.NET Core додатку — достатньо середовища зі такими мінімальними параметрами:

- 2 віртуальні ядра процесора (vCPU),
- 2–4 гігабайти оперативної пам'яті (RAM),
- 5–10 гігабайтів простору для бази даних PostgreSQL.

Клієнтська частина — побудована як SPA-додаток — не потребує окремого серверного середовища. Після збирання з використанням Vite, статичні файли фронтенду розміщуються на Railway Web сервісі або CDN, що забезпечує швидке завантаження інтерфейсу, мінімізацію часу відгуку та покращену продуктивність при великій кількості користувачів. За рахунок цієї архітектури фронтенд легко масштабується та є практично незалежним від апаратних ресурсів.

Особливу увагу приділено надійності та збереженню даних. База даних PostgreSQL, яка є ключовим елементом системи, функціонує у середовищі Railway з активованим щоденним резервним копіюванням, що гарантує відновлення у разі збою або втрати доступу. Крім того, Railway дозволяє масштабувати ресурси бази даних або переносити її до окремого керованого інстансу без необхідності втручання в код додатку.

У разі прогнозованого зростання навантаження або розширення функціоналу система готова до масштабування як вертикально (збільшення ресурсів реплік), так і горизонтально (додавання нових реплік). Така гнучкість забезпечує довготривалу працездатність проєкту без значних витрат на переоснащення інфраструктури.

Таким чином, вибране технічне забезпечення повністю відповідає сучасним вимогам до побудови гнучких, масштабованих та надійних інформаційних систем. Воно дозволяє не лише забезпечити ефективне поточне функціонування системи,

але й створює основу для подальшого її розвитку та адаптації до нових вимог і умов експлуатації.

3.5 Реалізація інформаційної управляючої системи

У результаті виконання комплексу теоретичних, аналітичних і проєктно-реалізаційних заходів була успішно створена та впроваджена повнофункціональна вебплатформа для інтелектуального поширення навчальних матеріалів. Реалізація інформаційної управляючої системи охопила широкий спектр компонентів, зокрема модулі керування користувачами, каталогом курсів, придбанням освітніх продуктів, а також рекомендаційний механізм на базі штучного інтелекту. У процесі розроблення було застосовано науково обґрунтовані підходи до побудови систем, що враховують сучасні вимоги до безпеки, масштабованості, адаптивності та доступності. Розроблена система забезпечує гнучке управління навчальним процесом, дає змогу динамічно оновлювати контент, підтримує зворотний зв'язок із користувачами та забезпечує аналітику успішності проходження курсів. Усі функціональні та технологічні рішення були інтегровані в єдину логічно пов'язану систему, що дозволяє ефективно підтримувати життєвий цикл освітнього контенту в межах цифрової платформи.

Наукова новизна розробленої системи полягає у побудові унікального рішення рекомендаційного механізму, яке реалізовано на основі Matrix Factorization за допомогою бібліотеки ML.NET. Цей тип моделей, який є частиною класу латентних факторних моделей, дозволяє з високою точністю прогнозувати ймовірні інтереси користувача, базуючись на історії його взаємодії з курсами, зокрема лайками, покупками, додаванням у вибране тощо. Модель проходила тренування на великому датасеті, що містить понад 10 000 курсів із відкритих онлайн-платформ. Рішення було реалізоване без використання готових інструментів сторонніх платформ, що підкреслює самостійність та індивідуальність підходу до створення ПЗ. Система також використовує концепцію контейнеризації для досягнення максимальної мобільності, гнучкості розгортання та масштабованості: за допомогою Docker можна швидко розгорнути інстанс

системи для окремої організації, що робить її універсальним рішенням як для вищих навчальних закладів, так і для корпоративного навчання.

Особливої уваги заслуговує реалізація гнучкої системи ролей та авторизації, яка дозволяє розмежовувати права доступу між неавторизованими користувачами, звичайними користувачами, розробниками курсів і адміністраторами. Ця модель підвищує безпеку, дозволяє делегувати обов'язки управління контентом без доступу до адміністративного функціоналу, та є важливим фактором адаптації системи під різні організаційні структури.

Практична значимість отриманих результатів підтверджується участю здобувача в науково-практичних конференціях, а також публікацією відповідних тез. У Додатку Б наведено тези "Оцінка якості та верифікація навчальних матеріалів, поширених через онлайн-платформи", в яких висвітлено підходи до перевірки надійності освітнього контенту. У Додатку В розміщено тези "Інноваційні підходи для таргетованого просування навчального контенту на основі ІІІ", що демонструють прикладний аспект використання ІІІ в освітньому середовищі. Наявність таких публікацій підтверджує як наукову вагу результатів КМР, так і їхню прикладну значущість.

Проведене техніко-економічне обґрунтування продемонструвало високу економічну ефективність впровадженого рішення. Було змодельовано типову ситуацію використання системи в середньому підприємстві з чисельністю 100 співробітників, які проходять щоквартальне навчання. У класичній моделі навчання витрати на одного працівника становили орієнтовно 400 грн за модуль, включаючи витрати на лекторів, оренду приміщень, навчальні матеріали, логістику та інші супутні витрати. Сумарні витрати за квартал досягали близько 40 000 грн. У моделі з використанням розробленої ІУС, витрати на інфраструктуру, включаючи хостинг, підтримку й оновлення контенту, складали орієнтовно 24 000 грн. Різниця становить близько 40%, що є значною економією для підприємств.

До складу аналізу входили: порівняння витрат на традиційне та цифрове навчання, моделювання обсягу використання платформи, урахування витрат на електронні ресурси, зменшення затрат часу викладачів, підвищення доступності

матеріалів у будь-який момент та автоматичне формування звітності. Економічний ефект був підтверджений низкою контрольних сценаріїв та типових кейсів з варіативним числом працівників, різною інтенсивністю використання платформи та з урахуванням витрат на технічну підтримку. Було також враховано перспективу збільшення продуктивності завдяки більш зручному формату подачі знань та постійному доступу до контенту, що знижує ймовірність помилок через неповне засвоєння матеріалу.

Результати аналізу ступеня готовності розробленої платформи до впровадження свідчать про її високу завершеність і технічну надійність. Система пройшла повний цикл тестування в розробницькому середовищі, зокрема юніт-тести, функціональне тестування, ручну перевірку бізнес-логіки та стабільності серверної частини. Реалізовано усі ключові функції: реєстрація та авторизація користувачів, перегляд і фільтрація каталогу курсів, механізми оцінювання, система коментарів, а також індивідуальні рекомендації на базі ШІ. Завдяки використанню архітектурних принципів модульності та розділення відповідальності між компонентами, система легко адаптується під нові вимоги, що робить її придатною для подальшого розширення та масштабування.

У контексті перспектив подальших досліджень і вдосконалення, розроблена система демонструє значний потенціал до розвитку. Планується розширення модуля аналітики, інтеграція з системами LTI для обміну навчальним контентом, а також впровадження розумних інтерфейсів на базі великих мовних моделей для покращення комунікації з користувачем. Додатково, передбачається розробка мобільного додатка з адаптивним інтерфейсом, який дозволить користувачам взаємодіяти з системою на різних типах пристроїв. У довгостроковій перспективі планується реалізація функціоналу для аналізу успішності в реальному часі, що дасть змогу викладачам або адміністраторам отримувати актуальну інформацію щодо прогресу студентів та вчасно коригувати навчальний процес.

Таким чином, реалізована інформаційна управляюча система демонструє не лише високий рівень готовності до широкого впровадження, але й значний потенціал до адаптації, модернізації та інтеграції в сучасні цифрові освітні

екосистеми. Її гнучка архітектура, ефективне використання сучасних технологій, підтверджена практична цінність і наукова новизна створюють міцну основу для її подальшого масштабного застосування в різних галузях освітньої та професійної підготовки.

ВИСНОВКИ

У ході виконання кваліфікаційної магістерської роботи було здійснено комплексне дослідження, розроблення та впровадження інформаційної управляючої системи для інтелектуального поширення навчальних матеріалів. Основна мета роботи полягала у створенні цифрової вебплатформи, яка забезпечує зручну, масштабовану та персоналізовану взаємодію користувачів з навчальним контентом. У рамках реалізації цієї мети було досягнуто низку суттєвих результатів, які мають як теоретичну, так і прикладну значущість.

Було здійснено глибокий аналіз предметної області, у тому числі особливостей функціонування онлайн-освіти як у контексті навчальних закладів, так і у корпоративному середовищі. Проаналізовано сучасні інформаційні системи з подібною функціональністю: Coursera, Udemy, Prometheus, OpenEdu, і на основі критичного порівняння обґрунтовано актуальність розробки нової системи з акцентом на простоту впровадження, гнучкість та підтримку інтелектуального компоненту рекомендацій.

Проведено систематизацію існуючих підходів до створення ІУС, зокрема висвітлено переваги ітеративного підходу, використання методології Rapid Application Development, а також гнучкої організації проєктної діяльності за допомогою Kanban. Було обґрунтовано вибір сучасного стеку технологій, який включає ASP.NET, EF Core, PostgreSQL, ReactJS, Docker, ML.NET та інші. Детально проаналізовано архітектурні та проєктні рішення, серед яких особливу увагу приділено впровадженню системи персоналізованих рекомендацій на базі моделей факторизації матриць.

Було розроблено повноцінну функціональну структуру ІУС, що включає користувацький інтерфейс, серверну частину та базу даних. Побудовано діаграми класів, прецедентів, послідовностей та розгортання, які відображають взаємозв'язки між компонентами системи, поведінку під час виконання ключових сценаріїв та варіанти взаємодії з акторами. На базі побудованих моделей

реалізовано бізнес-логіку системи із підтримкою базової та ролей розширеного рівня доступу.

У ході роботи спроектовано інфологічну та фізичну модель бази даних, яка реалізована у середовищі PostgreSQL з використанням ORM EF Core. Визначено структуру даних, описано логіку взаємозв'язків між сутностями, забезпечено відповідність нормальним формам, зокрема третій нормальній формі. Здійснено навантаження контрольним прикладом і виконано прогностні розрахунки щодо обсягу даних у перспективі росту аудиторії.

Особливу увагу приділено реалізації інтелектуального аналізу даних: виконано підготовку та обробку набору даних з понад 10 000 курсів, реалізовано алгоритм побудови рекомендацій на базі латентних факторів, що враховує історію взаємодії користувачів з курсами. Побудовано базу знань системи, яка структурує правила, пріоритети та параметри моделей ШІ, та дозволяє легко масштабувати рекомендаційний механізм.

У процесі реалізації програмного забезпечення було розроблено інтерфейс користувача з підтримкою реактивної взаємодії, забезпечено надійне серверне API, впроваджено механізми авторизації за допомогою JWT. Система повністю контейнеризована і розгорнута на Railway-хостингу. Проведено тестування ключових компонентів системи, підготовлено керівництво користувача та забезпечено повний цикл документації.

Платформа пройшла апробацію в розробницькому середовищі. Практичну цінність та результативність КМР підтверджено участю у двох наукових конференціях, публікацією тез доповідей.

Таким чином, поставлені в роботі цілі та завдання було досягнуто. Результати КМР свідчать про здатність здобувача самостійно вирішувати складні прикладні задачі в галузі створення інформаційних управляючих систем з використанням сучасних технологій та методів штучного інтелекту. Робота має як наукову новизну, так і вагоме прикладне значення, що відкриває перспективи для подальших досліджень і впровадження платформи в реальні умови навчання та професійної підготовки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Quality Matters Program. (n.d.). Higher Education Rubric. <https://www.google.com/search?q=https://www.qualitymatters.org/qa-resources/rubrics-standards>
2. Generative AI in Education: Shaping the Future of Learning URL: Generative AI in Education: Opportunities and Challenges (ryangroup.org)
3. Як штучний інтелект змінює шкільну освіту: результати дослідження Малої академії наук і Projector Institute URL: <https://man.gov.ua/about/news/yak-shtuchnij-intelekt-zminyuye-shkilsnu-osvitu-rezulstati-doslidzhennya-maloyi-akademiyi-nauk-i-projector-institute>
4. Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців URL: 2020-Slipchyshyn-USING THE AGILE APPROACH IN EDUCATION-181-183-PB - -Slipchyshyn.pdf (npu.edu.ua)
5. Coursera. Machine Learning Engineer. URL: <https://www.coursera.org/career-academy/roles/machine-learning-engineer> (дата звернення: 16.05.2025).
6. KHEU Moodle. URL: <https://ido-m.kneu.edu.ua/moodle/my/> (дата звернення: 16.05.2025).
7. Google Play Store. EdApp. URL: <https://play.google.com/store/apps/details?id=com.cld.ed&hl=uk> (дата звернення: 16.05.2025).
8. Prometheus. Каталог/Безплатні. URL: <https://prometheus.org.ua/courses-catalog?price=0> (дата звернення: 16.05.2025).
9. Wikipedia. Iterative and incremental development. URL: https://en.wikipedia.org/wiki/Iterative_and_incremental_development (дата звернення: 16.05.2025).
10. Wikipedia. Rapid application development. URL: https://en.wikipedia.org/wiki/Rapid_application_development (дата звернення: 16.05.2025).
11. Wikipedia. Канбан. URL: <https://uk.wikipedia.org/wiki/%D0%9A%D0%B0%D0%BD%D0%B1%D0%B0%D0%BD> (дата звернення: 16.05.2025).
12. Bhaskar S. What is Kanban?. Nimble. 30.04.2024. URL: <https://www.nimblework.com/kanban/what-is-kanban/> (дата звернення: 16.05.2025).
13. Radigan D. What is kanban?. Atlassian. URL: <https://www.atlassian.com/en/agile/kanban> (дата звернення: 16.05.2025).
14. Docker. Wikipedia. URL: <https://uk.wikipedia.org/wiki/Docker> (дата звернення: 16.05.2025).
15. What is Docker?. Docker. URL: <https://docs.docker.com/get-started/docker-overview/> (дата звернення: 16.05.2025).
16. Use containers to Build, Share and Run your applications. Docker. URL: <https://www.docker.com/resources/what-container/> (дата звернення: 16.05.2025).

17. Containerization (computing). Wikipedia. URL: [https://en.wikipedia.org/wiki/Containerization_\(computing\)](https://en.wikipedia.org/wiki/Containerization_(computing)) (дата звернення: 16.05.2025).
18. Railway. URL: <https://railway.com/> (дата звернення: 16.05.2025).
19. Deploying to Railway. Vendure. 09.05.2025. URL: <https://docs.vendure.io/guides/deployment/deploy-to-railway/> (дата звернення: 16.05.2025).
20. ML.NET. Microsoft. URL: <https://dotnet.microsoft.com/en-us/apps/ai/ml-dotnet> (дата звернення: 16.05.2025).
21. ML.NET. Wikipedia. URL: <https://en.wikipedia.org/wiki/ML.NET> (дата звернення: 16.05.2025).
22. Matrix factorization (recommender systems). Wikipedia. URL: [https://en.wikipedia.org/wiki/Matrix_factorization_\(recommender_systems\)](https://en.wikipedia.org/wiki/Matrix_factorization_(recommender_systems)) (дата звернення: 16.05.2025).
23. Zonoozi A. What Is Matrix Factorization?. Builtin. 25.09.2024. URL: <https://builtin.com/articles/matrix-factorization> (дата звернення: 16.05.2025).
24. Matrix Factorization. Dremio. URL: <https://www.dremio.com/wiki/matrix-factorization/> (дата звернення: 16.05.2025).
25. Tutorial: Build a movie recommender using matrix factorization with ML.NET. Microsoft. 29.11.2021. URL: <https://learn.microsoft.com/en-us/dotnet/machine-learning/tutorials/movie-recommendation> (дата звернення: 16.05.2025).
26. Shata K. Online Courses. Kaggle. URL: <https://www.kaggle.com/datasets/khaledatef1/online-courses> (дата звернення: 16.05.2025).
27. React. URL: <https://react.dev/> (дата звернення: 16.05.2025).
28. React. Wikipedia. URL: <https://uk.wikipedia.org/wiki/React> (дата звернення: 16.05.2025).
29. Що таке React JS і для чого він потрібен?. DAN.IT. 18.10.2024. URL: <https://dan-it.com.ua/uk/blog/chto-takoe-react-js-i-dlja-chego-on-nuzhen/> (дата звернення: 16.05.2025).
30. TypeScript is JavaScript with syntax for types.. TypeScript. URL: <https://www.typescriptlang.org/> (дата звернення: 16.05.2025).
31. TypeScript. Wikipedia. URL: <https://uk.wikipedia.org/wiki/TypeScript> (дата звернення: 16.05.2025).
32. TypeScript: Переваги та Використання в Україні. Galaktica. 23.01.2024. URL: <https://galaktica.io/blog/typescript-tse/> (дата звернення: 16.05.2025).
33. Vite. URL: <https://vite.dev/> (дата звернення: 16.05.2025).
34. Vite (software). Wikipedia. URL: [https://en.wikipedia.org/wiki/Vite_\(software\)](https://en.wikipedia.org/wiki/Vite_(software)) (дата звернення: 16.05.2025).
35. The React Framework for the Web. NextJS. URL: <https://nextjs.org/> (дата звернення: 16.05.2025).
36. Next.js. Wikipedia. URL: <https://uk.wikipedia.org/wiki/Next.js> (дата звернення: 16.05.2025).

37. About React and Next.js. NextJS. URL: <https://nextjs.org/learn/react-foundations/what-is-react-and-nextjs> (дата звернення: 16.05.2025).
38. NextJs vs React. Reddit. URL: https://www.reddit.com/r/react/comments/18uxapg/nextjs_vs_react/ (дата звернення: 16.05.2025).
39. Okoro E. N. Next.js vs React – Differences and How to Choose the Right One for Your Project. Freecodecamp. 04.12.2024. URL: <https://www.freecodecamp.org/news/nextjs-vs-react-differences/> (дата звернення: 16.05.2025).
40. PostgreSQL. Wikipedia. URL: <https://uk.wikipedia.org/wiki/PostgreSQL> (дата звернення: 16.05.2025).
41. Dhruv S. PostgreSQL Advantages and Disadvantages. *Aalpha*. 12.07.2024. URL: <https://www.aalpha.net/blog/pros-and-cons-of-using-postgresql-for-application-development/> (дата звернення: 16.05.2025).
42. Top 21 AI use cases in eCommerce (the rise of AI in eCommerce). Instant. 28.01.2025. URL: <https://instant.so/blog/ai-use-cases-in-ecommerce> (дата звернення: 16.05.2025).
43. Moisinho E. Generating Class Diagrams for .Net Core. Medium. 08.04.2022. URL: <https://medium.com/better-programming/generating-class-diagrams-for-net-core-c4913db9398b> (дата звернення: 16.05.2025).
44. Entity Framework. Wikipedia. URL: https://uk.wikipedia.org/wiki/Entity_Framework (дата звернення: 16.05.2025).
45. Entity Framework documentation hub. Microsoft. URL: <https://learn.microsoft.com/en-us/ef/> (дата звернення: 16.05.2025).
46. Elie A. Deep Dive into Matrix Factorization for Recommender Systems: From Basics to Implementation. Medium. 31.08.2024. URL: <https://medium.com/@eliasah/deep-dive-into-matrix-factorization-for-recommender-systems-from-basics-to-implementation-79e4f1ea1660> (дата звернення: 16.05.2025).
47. Matrix factorization. Developers Google. URL: <https://developers.google.com/machine-learning/recommendation/collaborative/matrix>.
48. НЕЙРО-НЕЧІТКА МОДЕЛЬ БАГАТОКРИТЕРІАЛЬНОГО ОЦІНЮВАННЯ / М. М. Маляр та ін. УжНУ. 11.10.2019. URL: <https://dspace.uzhnu.edu.ua/jspui/bitstream/lib/28785/1/83-91.pdf> (дата звернення: 16.05.2025).
49. Нормальні форми бази даних. *JavaRush*. URL: <https://javarush.com/ua/quests/lectures/ua.questhibernate.level17.lecture02> (дата звернення: 16.05.2025).
50. API Authentication and Authorization: 6 Methods and Tips for Success. Frontegg. 05.07.2023. URL: <https://frontegg.com/guides/api-authentication-api-authorization> (дата звернення: 16.05.2025).
51. RFC 7235. Hypertext Transfer Protocol (HTTP/1.1): Authentication. На заміну RFC 2617, RFC 2616 ; чинний від 01.06.2014. RFC Editor, 2014. 19 с. URL: <https://www.rfc-editor.org/info/rfc7235> (дата звернення: 16.05.2025).

ДОДАТКИ

Додаток А

MainDbContext.cs

```
using CoursesShop.Entities;
using Microsoft.EntityFrameworkCore;
using NodaTime;
using NodaTime.Extensions;

namespace CoursesShop.Database;

public class MainDbContext(IConfiguration configuration) : DbContext
{
    #region Dependencies

    private readonly IConfiguration _configuration = configuration;

    #endregion

    #region Constants

    private const string GenerateUuidFunction = "gen_random_uuid()";

    #endregion

    #region DbSets

    public DbSet<CreditCard> CreditCards { get; set; }
    public DbSet<Course> Courses { get; set; }
    public DbSet<CourseMeta> CourseMetas { get; set; }
    public DbSet<Category> Categories { get; set; }
    public DbSet<Role> Roles { get; set; }
    public DbSet<Spec> Specs { get; set; }
    public DbSet<SpecType> SpecsTypes { get; set; }
    public DbSet<User> Users { get; set; }
    public DbSet<Comment> Comments { get; set; }
    public DbSet<TrophyAddition> TrophyAdditions { get; set; }

    #endregion

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        optionsBuilder.UseNpgsql(_configuration.GetConnectionString("DefaultConnection"),
            o => o.UseNodaTime());
    }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
    }
```

```

modelBuilder.Entity<User>()
    .HasIndex(user => user.Email)
    .IsUnique();

modelBuilder.Entity<CreditCard>()
    .Property(cc => cc.Id)
    .HasDefaultValueSql(GenerateUuidFunction);
modelBuilder.Entity<Course>()
    .Property(g => g.Id)
    .HasDefaultValueSql(GenerateUuidFunction);
modelBuilder.Entity<Course>()
    .OwnsMany(c => c.Steps, s =>
    {
        s.ToJson("Steps");
        s.Property(s => s.Ordinal)
            .ValueGeneratedOnAdd();
    });
modelBuilder.Entity<Category>()
    .Property(g => g.Id)
    .HasDefaultValueSql(GenerateUuidFunction);
modelBuilder.Entity<Role>()
    .Property(r => r.Id)
    .HasDefaultValueSql(GenerateUuidFunction);
modelBuilder.Entity<SpecType>()
    .Property(st => st.Id)
    .HasDefaultValueSql(GenerateUuidFunction);
modelBuilder.Entity<User>()
    .Property(u => u.Id)
    .HasDefaultValueSql(GenerateUuidFunction);
modelBuilder.Entity<Comment>()
    .Property(c => c.Id)
    .HasDefaultValueSql(GenerateUuidFunction);
modelBuilder.Entity<TrophyAddition>()
    .Property(ta => ta.Id)
    .HasDefaultValueSql(GenerateUuidFunction);

modelBuilder.Entity<CourseMeta>()
    .HasKey(gm => new { gm.CourseId, gm.UserId });

modelBuilder.Entity<Spec>()
    .HasKey(s => new { s.CourseId, s.SpecTypeId });
}

public override int SaveChanges()
{
    AddTimestamps();
    return base.SaveChanges();
}

public override async Task<int> SaveChangesAsync(CancellationToken
cancellation_token = default)

```

```

    {
        AddTimestamps();
        return await base.SaveChangesAsync(cancellationToken);
    }

    private void AddTimestamps()
    {
        var entities = ChangeTracker.Entries()
            .Where(x => x is { Entity: AbstractEntity, State: EntityState.Added or
EntityState.Modified });

        foreach (var entity in entities)
        {
            if (entity.State == EntityState.Added)
            {
                ((AbstractEntity)entity.Entity).CreatedAt =
SystemClock.Instance.InUtc().GetCurrentLocalDateTime();
            }
            ((AbstractEntity)entity.Entity).UpdatedAt =
SystemClock.Instance.InUtc().GetCurrentLocalDateTime();
        }
    }
}

```

Course.cs

```

namespace CoursesShop.Entities;

public class Course : AbstractEntity
{
    public Guid Id { get; set; }
    public string Name { get; set; } = string.Empty;
    public string Description { get; set; } = string.Empty;
    public string? PreviewUrl { get; set; }
    public string VideoUrl { get; set; } = string.Empty;
    public double Rate { get; set; } = 0;
    public uint RateCount { get; set; } = 0;
    public double? OldPrice { get; set; }
    public double NewPrice { get; set; } = 0;
    public int Trophies { get; set; } = 0;
    public bool IsPopular { get; set; }

    public ICollection<CourseStep> Steps { get; set; } = [];

    public ICollection<Spec> Specs { get; set; } = [];

    public Category Category { get; set; } = new();

    public ICollection<CourseMeta> CourseMetas { get; set; } = [];
}

```

```
    public ICollection<Comment> Comments { get; set; } = [];  
}
```

Category.cs

```
namespace CoursesShop.Entities;  
  
public class Category : AbstractEntity  
{  
    public Guid Id { get; set; }  
    public string Name { get; set; } = string.Empty;  
    public string? Type { get; set; }  
    public string? ImageUrl { get; set; }  
  
    public ICollection<Course> Courses { get; set; } = [];  
}
```

CourseStepDto.cs

```
namespace CoursesShop.Models.DTO;  
  
public class CourseStepDto  
{  
    public string Title { get; set; } = string.Empty;  
    public string Description { get; set; } = string.Empty;  
    public string Duration { get; set; } = string.Empty;  
    public string? PreviewUrl { get; set; }  
    public string VideoUrl { get; set; } = string.Empty;  
}
```

CourseDetailsDto.cs

```
namespace CoursesShop.Models.DTO;  
  
public class CourseDetailsDto  
{  
    public Guid Id { get; set; }  
    public string Name { get; set; } = string.Empty;  
    public string Description { get; set; } = string.Empty;  
    public string? PreviewUrl { get; set; }  
    public string Video { get; set; } = string.Empty;  
    public CategoryDto Category { get; set; } = new();  
    public double Rate { get; set; }  
    public double NewPrice { get; set; }  
    public double? OldPrice { get; set; }  
    public int Trophies { get; set; }  
    public CourseMetaDto? Meta { get; set; }  
    public IEnumerable<SpecDto> Specs { get; set; } = [];  
    public IEnumerable<CourseStepDto> Steps { get; set; } = [];  
}
```

*Устенко С.В., д.е.н., професор
Терентьєв Є.Г., магістр
Київський національний економічний
університет імені Вадима Гетьмана
stasustenko@ukr.net
123j123t@gmail.com*

ІННОВАЦІЙНІ ПІДХОДИ ДЛЯ ТАРГЕТОВАНОГО ПРОСУВАННЯ НАВЧАЛЬНОГО КОНТЕНТУ НА ОСНОВІ ШІ

Наразі штучний інтелект впливає на освіту у різних країнах, таких як США, Індія та Великобританія та ін.. Він допомагає персоналізувати процес навчання, автоматизувати оцінювання та створювати інтерактивні навчальні матеріали.

Персоналізоване навчання допомагає створювати індивідуальні навчальні програми, які враховують потреби та здібності кожного учня. Такі програми дозволяють учням здобувати знання більш ефективно та у своєму власному темпі.

Автоматизоване оцінювання може використовуватися для автоматичного оцінювання завдань, тестів та інших робіт. Це зменшує навантаження на вчителів і дозволяє їм більше часу приділяти індивідуальному навчанню, що також позбавляє процес оцінювання від людського фактору, таких як дискримінація та шахрайство з боку учнів.

Віртуальні асистенти: Штучний Інтелект може виступати в ролі віртуальних асистентів, які відповідають на запитання, що також знижує загальне напруження на вчителів, та дозволяє учням самостійно шукати та переперевіряти інформацію, що є критичною навичкою під час навчання та в подальшому житті. [1]

Розглядаючи важливість цифрової трансформації в сфері освіти, метою цього дослідження є аналіз та узагальнення останніх тенденцій щодо просування навчального контенту на основі штучного інтелекту. Світові тренди розвитку цифрових технологій постійно вивчаються та обговорюються на різних міжнародних заходах, таких як «Цифрова Європа». Україна приєдналася до Програми «Цифрова Європа», яка сприяє розвитку цифрових технологій в країні. Це відкриває нові можливості для покращення навчального процесу та забезпечення якісної освіти. [2]

Україна також активно досліджує можливості впровадження штучного інтелекту в освітній сектор. Однак існують певні складнощі:

1. **Технічні обмеження:** Недостатня розвиненість інфраструктури та доступу до сучасних технологій може ускладнити впровадження ШІ в навчальних закладах;
2. **Кадрові питання:** Необхідно підготувати вчителів та адміністраторів до роботи з ШІ, а це вимагає часу та навчання;
3. **Етичні аспекти:** Використання ШІ повинно дотримуватися етичних норм та забезпечувати конфіденційність даних учнів.

Незважаючи на ці виклики, впровадження ШІ в освіту може покращити якість навчання та зробити його більш доступним для всіх

В Україні впровадження штучного інтелекту у навчальний процес має свої складнощі та вимагає уваги з боку законодавства. Ось деякі аспекти, які варто враховувати:

Нормативно-правова база: Закони України “Про освіту” та “Про вищу освіту” визначають загальні принципи та засади функціонування системи освіти. Однак, вони не завжди враховують специфіку впровадження ШІ. Необхідно розробити додаткові нормативно-правові акти, які б врегулювали використання ШІ в навчальних закладах.

Конфіденційність та безпека даних: Використання ШІ повинно дотримуватися принципів конфіденційності та захисту особистих даних учнів. Законодавство повинно визначити правила обробки та збереження даних.

Етичні аспекти: Закони мають враховувати етичні питання, пов'язані з використанням ШІ. Наприклад, визначити, які завдання можуть бути автоматизовані, а які вимагають присутності вчителя.

Підготовка вчителів: Законодавство повинно передбачити навчання вчителів використанню ШІ та підтримку їх у процесі впровадження.

Закони України вже визначають загальні принципи, але деталізація та адаптація до впровадження ШІ вимагає додаткових зусиль. Наразі це актуальне питання, і законодавці повинні враховувати ці аспекти для успішного впровадження ШІ в навчальний процес. [3]

Ось деякі інноваційні підходи, які можуть бути використані в Україні:

Персоналізоване навчання:

Використання ШІ дозволяє створювати індивідуальні навчальні програми для кожного учня. Адаптація контенту до потреб та рівня знань допомагає підвищити ефективність навчання та підвищить мотивованість студентів, що позитивно відібразиться на загальному рівні освіти в країні.

Аналіз даних для вдосконалення навчання:

ШІ може обробляти великі обсяги даних, щоб виявити слабкі місця в навчальному процесі та запропонувати вдосконалення.

Віртуальна реальність та розширена реальність:

Використання ШІ для створення іммерсивних навчальних досвідів, наприклад, віртуальних екскурсій або тренажерів, таких як VR Chat або Meta Verse.

Автоматизоване оцінювання та зворотний зв'язок:

ШІ може автоматично оцінювати завдання та надавати індивідуальний зворотний зв'язок учням у будь який час, що позитивно вплине на загальну продуктивність та допоможе студентам вкладатись у строки здачі робіт.

Гейміфікація навчання:

Використання елементів гри в навчальному процесі зацікавлює учнів та стимулює активність. Це чудово підійде для заохочення дітей до навчання ще з малого віку, та допоможе подати навчання як цікавий та веселий досвід, а не лише нудні лекції та\або уроки.

Ці підходи можуть сприяти покращенню якості навчання та зробити його більш цікавим та ефективним. [4]

Використання штучного інтелекту в освіті має великий потенціал, але також вимагає уважного вивчення етичних та правових питань. Незважаючи на виклики, ШІ може допомогти створити більш інноваційне та доступне навчання для всіх. Впровадження штучного інтелекту в освіту полягає не лише в застосуванні нових інструментів; йдеться про переосмислення того, як ми навчаємось і навчаємо. Використовуючи генеративний ШІ, ми можемо створити майбутнє, де освіта буде динамічною, захоплюючою та адаптованою до індивідуальних потреб.

Список використаних джерел:

1. Generative AI in Education: Shaping the Future of Learning URL: [Generative AI in Education: Opportunities and Challenges \(ryangroup.org\)](https://ryangroup.org/generative-ai-in-education-opportunities-and-challenges)
2. Україна приєдналася до програми «Цифрова Європа». URL: <https://www.nas.gov.ua/UA/Messages/Pages/View.aspx?MessageID=9444>
3. Як штучний інтелект змінює шкільну освіту: результати дослідження Малої академії наук і Projector Institute URL: <https://man.gov.ua/about/news/yak-shtuchnij-intelekt-zminyuye-shkilsnu-osvitu-rezulstaty-doslidzhennya-maloyi-akademiyi-nauk-i-projector-institute>
4. Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців URL: [2020-Slipchyshyn-USING THE AGILE APPROACH IN EDUCATION-181-183-PB - -Slipchyshyn.pdf \(npu.edu.ua\)](https://npu.edu.ua/2020-Slipchyshyn-USING%20THE%20AGILE%20APPROACH%20IN%20EDUCATION-181-183-PB--Slipchyshyn.pdf)

Додаток В

Устенко С.В., д.е.н., професор

Терентьев Є.Г., магістр

*Київський національний економічний
університет імені Вадима Гетьмана*

stasustenko@ukr.net

123j123t@gmail.com

ОЦІНКА ЯКОСТІ ТА ВЕРИФІКАЦІЯ НАВЧАЛЬНИХ МАТЕРІАЛІВ, ПОШИРЕНИХ ЧЕРЕЗ ОНЛАЙН-ПЛАТФОРМИ

У сучасній освітній парадигмі онлайн-платформи стали потужним інструментом для поширення навчальних матеріалів, забезпечуючи гнучкість, доступність та потенціал для інтерактивного навчання. Однак, разом зі значними перевагами, зростання обсягу освітнього контенту в онлайн-середовищі висуває на перший план критично важливу проблему – оцінку якості та верифікацію цих матеріалів. Відсутність належних механізмів контролю може призвести до поширення недостовірної, застарілої або неефективної інформації, що негативно впливає на якість освіти та навчальні результати студентів.

Проблема оцінки якості онлайн-навчальних матеріалів є багатогранною та охоплює різні аспекти. По-перше, це змістова якість, яка включає актуальність, точність, повноту та наукову обґрунтованість представленої інформації. В умовах швидкого розвитку знань та технологій, навчальні матеріали потребують постійного оновлення та перевірки на відповідність сучасним науковим досягненням. По-друге, важливим є педагогічний дизайн матеріалів, що визначає їхню здатність ефективно сприяти навчанню. Це охоплює чіткість навчальних цілей, логічну структуру, використання різноманітних методів навчання, наявність інтерактивних елементів та можливостей для самоконтролю. По-третє, не менш значущою є технічна якість, яка включає зручність навігації, якість мультимедійних елементів (відео, аудіо, графіка), доступність для осіб з особливими потребами та стабільність роботи платформи.

Верифікація навчальних матеріалів, поширених онлайн, є складним завданням через децентралізований характер інтернету та велику кількість авторів і платформ. Традиційні методи контролю якості, що застосовуються до друкованих видань (рецензування, редагування, експертна оцінка), часто є недостатніми або не повністю адаптованими до онлайн-середовища. Відсутність єдиних стандартів та централізованих органів контролю створює ризики поширення неякісного контенту.

Однак, існують та розвиваються різні підходи до оцінки якості та верифікації онлайн-навчальних матеріалів. Одним з ключових є експертна оцінка, яка передбачає залучення фахівців у відповідній галузі знань та педагогічних дизайнерів для рецензування контенту на предмет його змістової, методичної та технічної якості. Цей процес може бути формальним (наприклад, перед розміщенням матеріалів на авторитетних платформах) або неформальним (наприклад, відгуки колег та студентів).

Іншим важливим механізмом є системи рейтингу та відгуків користувачів. Студенти та викладачі, які використовують певні навчальні матеріали, можуть залишати свої коментарі та оцінки, що формує певну репутацію контенту та допомагає іншим користувачам орієнтуватися у його якості. Хоча цей підхід є корисним, він може бути суб'єктивним та потребує механізмів запобігання маніпуляціям.

Розвиток автоматизованих інструментів оцінки також є перспективним напрямком. За допомогою технологій штучного інтелекту можна аналізувати тексти на предмет граматичних

та стилістичних помилок, перевіряти на плагіат, оцінювати читабельність та навіть виявляти потенційно недостовірну інформацію. Однак, такі інструменти поки що не можуть повністю замінити експертну оцінку, особливо у питаннях змістовної якості та педагогічної ефективності.

Важливу роль у забезпеченні якості онлайн-навчальних матеріалів відіграють критерії оцінки, які мають бути чіткими, об'єктивними та охоплювати різні аспекти якості. Ці критерії можуть розроблятися на рівні окремих навчальних закладів, професійних асоціацій або навіть на державному рівні. Вони повинні враховувати специфіку предметної області, цільову аудиторію та формат навчальних матеріалів.

Перспективи подальшого розвитку систем оцінки якості та верифікації онлайн-навчальних матеріалів пов'язані з комбінуванням різних підходів. Ефективна система може включати експертну оцінку для первинної верифікації, системи рейтингу та відгуків для постійного моніторингу якості, а також використання автоматизованих інструментів для виявлення потенційних проблем. Крім того, важливим є підвищення цифрової грамотності як викладачів, так і студентів, щоб вони могли критично оцінювати інформацію, знайдену в онлайн-середовищі.

Забезпечення високої якості та надійності навчальних матеріалів, поширених через онлайн-платформи, є ключовим фактором успішної цифрової трансформації освіти. Розробка та впровадження ефективних механізмів оцінки та верифікації вимагає спільних зусиль науковців, педагогів, розробників освітніх платформ та державних регуляторів. Лише таким чином можна гарантувати, що онлайн-навчання сприятиме отриманню якісної та актуальної освіти.

Список використаних джерел

1. Quality Matters Program. (n.d.). *Higher Education Rubric*.
<https://www.google.com/search?q=https://www.qualitymatters.org/qa-resources/rubrics-standards>
2. Hodges, C. B., Moore, S., Lockee, B. B., Trust, T., & Bond, M. A. (2020). The difference between emergency remote teaching and online learning. *EDUCAUSE Review*, 27

Науковий керівник: Устенко С.В., д.е.н., професор



Звіт подібності

метадані

Назва організації

Kyiv National Economic University named after Vadym Hetman KNEU

Заголовок

Розробка вебплатформи для інтелектуального поширення навчальних матеріалів

Автор

Науковий керівник / Експерт

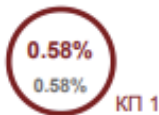
Терентьєв Євгеній ГеннадійовичУстенко С.В.

Підрозділ

кафедра інформаційних систем в економіці

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1

25

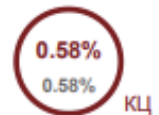
Довжина фрази для коефіцієнта подібності 2



КП 2

12052

Кількість слів



КЦ

98886

Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		0
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)		0

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Копію тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://www.studocu.com/uk/document/kiivs%D1%8Ckiy-natsional%D1%8Cniy-ekonomichniy-universitet-imeni-vadima-get%D1%8Cman/finansi-bankivs%D1%8Cka-sprava-ta-strakhuvannya/individual%D1%8Cni-investori-na-fondovomu-rinku/78336831	21 0.17 %
2	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	20 0.17 %

3	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	10 0.08 %
4	Формування та реалізація публічної політики сталого розвитку територій в Україні 5/7/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра національної економіки та публічного управління)	8 0.07 %
5	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	6 0.05 %
6	Формування та реалізація публічної політики сталого розвитку територій в Україні 5/7/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра національної економіки та публічного управління)	5 0.04 %

з бази даних RefBooks (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
------------------	-----------	--

з домашньої бази даних (0.41 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	36 (3) 0.30 %
2	Формування та реалізація публічної політики сталого розвитку територій в Україні 5/7/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра національної економіки та публічного управління)	13 (2) 0.11 %

з програми обміну базами даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
------------------	-----------	--

з Інтернету (0.17 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://www.studocu.com/uk/document/kiivs%D1%8Ckiy-natsional%D1%8Cniy-ekonomichniy-universitet-imeni-vadima-get%D1%8Cman/finans-bankivs%D1%8Cka-sprava-ta-strakhuvannya/individual%D1%8Cni-investori-na-fondovomu-rinku/78336831	21 (1) 0.17 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)
------------------	-------	---------------------------------------

Устенко С.В., д.е.н., професор
Терентьев Є.Г., магістр
*Київський національний економічний
університет імені Вадима Гетьмана*
stasustenko@ukr.net
123j123t@gmail.com

ОЦІНКА ЯКОСТІ ТА ВЕРИФІКАЦІЯ НАВЧАЛЬНИХ МАТЕРІАЛІВ, ПОШИРЕНИХ ЧЕРЕЗ ОНЛАЙН-ПЛАТФОРМИ

У сучасній освітній парадигмі онлайн-платформи стали потужним інструментом для поширення навчальних матеріалів, забезпечуючи гнучкість, доступність та потенціал для інтерактивного навчання. Однак, разом зі значними перевагами, зростання обсягу освітнього контенту в онлайн-середовищі висуває на перший план критично важливу проблему – оцінку якості та верифікацію цих матеріалів. Відсутність належних механізмів контролю може призвести до поширення недостовірної, застарілої або неефективної інформації, що негативно впливає на якість освіти та навчальні результати студентів.

Проблема оцінки якості онлайн-навчальних матеріалів є багатогранною та охоплює різні аспекти. По-перше, це змістова якість, яка включає актуальність, точність, повноту та наукову обґрунтованість представленої інформації. В умовах швидкого розвитку знань та технологій, навчальні матеріали потребують постійного оновлення та перевірки на відповідність сучасним науковим досягненням. По-друге, важливим є педагогічний дизайн матеріалів, що визначає їхню здатність ефективно сприяти навчанню. Це охоплює чіткість навчальних цілей, логічну структуру, використання різноманітних методів навчання, наявність інтерактивних елементів та можливостей для самоконтролю. По-третє, не менш значущою є технічна якість, яка включає зручність навігації, якість мультимедійних елементів (відео, аудіо, графіка), доступність для осіб з особливими потребами та стабільність роботи платформи.

Верифікація навчальних матеріалів, поширених онлайн, є складним завданням через децентралізований характер інтернету та велику кількість авторів і платформ. Традиційні методи контролю якості, що застосовуються до друкованих видань (рецензування, редагування, експертна оцінка), часто є недостатніми або не повністю адаптованими до онлайн-середовища. Відсутність єдиних стандартів та централізованих органів контролю створює ризики поширення неякісного контенту.

Однак, існують та розвиваються різні підходи до оцінки якості та верифікації онлайн-навчальних матеріалів. Одним з ключових є експертна оцінка, яка передбачає залучення фахівців у відповідній галузі знань та педагогічних дизайнерів для рецензування контенту на предмет його змістової, методичної та технічної якості. Цей процес може бути формальним (наприклад, перед розміщенням матеріалів на авторитетних платформах) або неформальним (наприклад, відгуки колег та студентів).

Іншим важливим механізмом є системи рейтингу та відгуків користувачів. Студенти та викладачі, які використовують певні навчальні матеріали, можуть залишати свої коментарі та оцінки, що формує певну репутацію контенту та допомагає іншим користувачам орієнтуватися у його якості. Хоча цей підхід є корисним, він може бути суб'єктивним та потребує механізмів запобігання маніпуляціям.

Розвиток автоматизованих інструментів оцінки також є перспективним напрямком. За допомогою технологій штучного інтелекту можна аналізувати тексти на предмет граматичних та стилістичних помилок, перевіряти на плагіат, оцінювати читабельність та навіть виявляти потенційно недостовірну інформацію. Однак, такі інструменти поки що не можуть повністю

замінити експертну оцінку, особливо у питаннях змістовної якості та педагогічної ефективності.

Важливу роль у забезпеченні якості онлайн-навчальних матеріалів відіграють критерії оцінки, які мають бути чіткими, об'єктивними та охоплювати різні аспекти якості. Ці критерії можуть розроблятися на рівні окремих навчальних закладів, професійних асоціацій або навіть на державному рівні. Вони повинні враховувати специфіку предметної області, цільову аудиторію та формат навчальних матеріалів.

Перспективи подальшого розвитку систем оцінки якості та верифікації онлайн-навчальних матеріалів пов'язані з комбінуванням різних підходів. Ефективна система може включати експертну оцінку для первинної верифікації, системи рейтингу та відгуків для постійного моніторингу якості, а також використання автоматизованих інструментів для виявлення потенційних проблем. Крім того, важливим є підвищення цифрової грамотності як викладачів, так і студентів, щоб вони могли критично оцінювати інформацію, знайдену в онлайн-середовищі.

Забезпечення високої якості та надійності навчальних матеріалів, поширених через онлайн-платформи, є ключовим фактором успішної цифрової трансформації освіти. Розробка та впровадження ефективних механізмів оцінки та верифікації вимагає спільних зусиль науковців, педагогів, розробників освітніх платформ та державних регуляторів. Лише таким чином можна гарантувати, що онлайн-навчання сприятиме отриманню якісної та актуальної освіти.

Список використаних джерел

1. Quality Matters Program. (n.d.). *Higher Education Rubric*.
<https://www.google.com/search?q=https://www.qualitymatters.org/qa-resources/rubrics-standards>
2. Hodges, C. B., Moore, S., Lockee, B. B., Trust, T., & Bond, M. A. (2020). The difference between emergency remote teaching and online learning. *EDUCAUSE Review*, 27

Науковий керівник: Устенко С.В., д.е.н., професор

*Устенко С.В., д.е.н., професор
Терентьєв Є.Г., магістр
Київський національний економічний
університет імені Вадима Гетьмана
stasustenko@ukr.net
123j123t@gmail.com*

ІННОВАЦІЙНІ ПІДХОДИ ДЛЯ ТАРГЕТОВАНОГО ПРОСУВАННЯ НАВЧАЛЬНОГО КОНТЕНТУ НА ОСНОВІ ШІ

Наразі штучний інтелект впливає на освіту у різних країнах, таких як США, Індія та Великобританія та ін.. Він допомагає персоналізувати процес навчання, автоматизувати оцінювання та створювати інтерактивні навчальні матеріали.

Персоналізоване навчання допомагає створювати індивідуальні навчальні програми, які враховують потреби та здібності кожного учня. Такі програми дозволяють учням здобувати знання більш ефективно та у своєму власному темпі.

Автоматизоване оцінювання може використовуватися для автоматичного оцінювання завдань, тестів та інших робіт. Це зменшує навантаження на вчителів і дозволяє їм більше часу приділяти індивідуальному навчанню, що також позбавляє процес оцінювання від людського фактору, таких як дискримінація та шахрайство з боку учнів.

Віртуальні асистенти: Штучний Інтелект може виступати в ролі віртуальних асистентів, які відповідають на запитання, що також знижує загальне напруження на вчителів, та дозволяє учням самостійно шукати та переперевіряти інформацію, що є критичною навичкою під час навчання та в подальшому житті. [1]

Розглядаючи важливість цифрової трансформації в сфері освіти, метою цього дослідження є аналіз та узагальнення останніх тенденцій щодо просування навчального контенту на основі штучного інтелекту. Світові тренди розвитку цифрових технологій постійно вивчаються та обговорюються на різних міжнародних заходах, таких як «Цифрова Європа». Україна приєдналася до Програми «Цифрова Європа», яка сприяє розвитку цифрових технологій в країні. Це відкриває нові можливості для покращення навчального процесу та забезпечення якісної освіти. [2]

Україна також активно досліджує можливості впровадження штучного інтелекту в освітній сектор. Однак існують певні складнощі:

4. **Технічні обмеження:** Недостатня розвиненість інфраструктури та доступу до сучасних технологій може ускладнити впровадження ШІ в навчальних закладах;

5. **Кадрові питання:** Необхідно підготувати вчителів та адміністраторів до роботи з ШІ, а це вимагає часу та навчання;

6. **Етичні аспекти:** Використання ШІ повинно дотримуватися етичних норм та забезпечувати конфіденційність даних учнів.

Незважаючи на ці виклики, впровадження ШІ в освіту може покращити якість навчання та зробити його більш доступним для всіх

В Україні впровадження штучного інтелекту у навчальний процес має свої складнощі та вимагає уваги з боку законодавства. Ось деякі аспекти, які варто враховувати:

Нормативно-правова база: Закони України “Про освіту” та “Про вищу освіту” визначають загальні принципи та засади функціонування системи освіти. Однак, вони не завжди враховують специфіку впровадження ШІ. Необхідно розробити додаткові нормативно-правові акти, які б врегулювали використання ШІ в навчальних закладах.

Конфіденційність та безпека даних: Використання ШІ повинно дотримуватися принципів конфіденційності та захисту особистих даних учнів. Законодавство повинно визначити правила обробки та збереження даних.

Етичні аспекти: Закони мають враховувати етичні питання, пов'язані з використанням ІІІ. Наприклад, визначити, які завдання можуть бути автоматизовані, а які вимагають присутності вчителя.

Підготовка вчителів: Законодавство повинно передбачити навчання вчителів використанню ІІІ та підтримку їх у процесі впровадження.

Закони України вже визначають загальні принципи, але деталізація та адаптація до впровадження ІІІ вимагає додаткових зусиль. Наразі це актуальне питання, і законодавці повинні враховувати ці аспекти для успішного впровадження ІІІ в навчальний процес. [3]

Ось деякі інноваційні підходи, які можуть бути використані в Україні:

Персоналізоване навчання:

Використання ІІІ дозволяє створювати індивідуальні навчальні програми для кожного учня. Адаптація контенту до потреб та рівня знань допомагає підвищити ефективність навчання та підвищить мотивованість студентів, що позитивно відібразиться на загальному рівні освіти в країні.

Аналіз даних для вдосконалення навчання:

ІІІ може обробляти великі обсяги даних, щоб виявити слабкі місця в навчальному процесі та запропонувати вдосконалення.

Віртуальна реальність та розширена реальність:

Використання ІІІ для створення іммерсивних навчальних досвідів, наприклад, віртуальних екскурсій або тренажерів, таких як VR Chat або Meta Verse.

Автоматизоване оцінювання та зворотний зв'язок:

ІІІ може автоматично оцінювати завдання та надавати індивідуальний зворотний зв'язок учням у будь який час, що позитивно вплине на загальну продуктивність та допоможе студентам вкладатись у строки здачі робіт.

Гейміфікація навчання:

Використання елементів гри в навчальному процесі зацікавлює учнів та стимулює активність. Це чудово підійде для заохочення дітей до навчання ще з малого віку, та допоможе подати навчання як цікавий та веселий досвід, а не лише нудні лекції та/або уроки.

Ці підходи можуть сприяти покращенню якості навчання та зробити його більш цікавим та ефективним. [4]

Використання штучного інтелекту в освіті має великий потенціал, але також вимагає уважного вивчення етичних та правових питань. Незважаючи на виклики, ІІІ може допомогти створити більш інноваційне та доступне навчання для всіх. Впровадження штучного інтелекту в освіту полягає не лише в застосуванні нових інструментів; йдеться про переосмислення того, як ми навчаємось і навчаємо. Використовуючи генеративний ІІІ, ми можемо створити майбутнє, де освіта буде динамічною, захоплюючою та адаптованою до індивідуальних потреб.

Список використаних джерел:

5. Generative AI in Education: Shaping the Future of Learning URL: [Generative AI in Education: Opportunities and Challenges \(ryangroup.org\)](https://ryangroup.org/generative-ai-in-education)

6. Україна приєдналася до програми «Цифрова Європа». URL: <https://www.nas.gov.ua/UA/Messages/Pages/View.aspx?MessageID=9444>

7. Як штучний інтелект змінює шкільну освіту: результати дослідження Малої академії наук і Projector Institute URL: <https://man.gov.ua/about/news/yak-shtuchnij-intelekt-zminyuye-shkilsnu-osvitu-rezulstati-doslidzhennya-maloyi-akademii-nauk-i-projector-institute>

8. Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців URL: [2020-Slipchyshyn-USING THE AGILE APPROACH IN EDUCATION-181-183-PB - -Slipchyshyn.pdf \(npu.edu.ua\)](https://slipchyshyn-npu.edu.ua/2020-Slipchyshyn-USING%20THE%20AGILE%20APPROACH%20IN%20EDUCATION-181-183-PB--Slipchyshyn.pdf)