

Міністерство освіти і науки України  
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту  
(повна назва)

Кафедра Інформатики  
(повна назва)

## КВАЛІФІКАЦІЙНА РОБОТА Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

**РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ТЕСТУВАННЯ ЗНАНЬ  
З МОВИ SQL З ВИКОРИСТАННЯМ LLM В РАМКАХ  
ЗАПРОПОНОВАНОГО МЕТОДУ АВТОМАТИЧНОГО  
ОЦІНЮВАННЯ ВІДПОВІДЕЙ СТУДЕНТІВ**

(тема)

Виконав:

студент 4 курсу, групи ІТІНФ-20-3

Науменко В.В.

(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки

(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика

(повна назва освітньої програми)

Керівник доц. Яковлева О. В.

(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри

\_\_\_\_\_

(підпис)

Кобилін О.А.

(прізвище, ініціали)

2024 р.

## Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту  
(повна назва)Кафедра Інформатики  
(повна назва)Рівень вищої освіти перший (бакалаврський)Спеціальність 122 Комп'ютерні науки  
(код і повна назва)Тип програми освітньо-професійнаОсвітня програма Інформатика  
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:

Зав. кафедри \_\_\_\_\_  
(підпис)

«\_\_\_\_» \_\_\_\_\_ 2024 р.

**ЗАВДАННЯ**  
НА КВАЛІФІКАЦІЙНУ РОБОТУстудентові Науменку Вадиму Віталійовичу  
(прізвище, ім'я, по батькові)1. Тема роботи Розробка вебзастосунку для тестування знань з мови SQL з використанням LLM в рамках запропонованого методу автоматичного оцінювання відповідей студентів

затверджена наказом університету від 20 травня 2024 року № 464 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 27 травня 2024 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, платформа розробки .Net, мова програмування C#, фреймворк Angular, система керування базами даних PostgreSQL, середовище розробки Microsoft Visual Studio Code.

4. Перелік питань, що потрібно опрацювати в роботі \_\_\_\_\_

1. Огляд існуючих сервісів для оцінювання.

2. Вивчення сучасних підходів та механізмів реалізації застосунку з методами автоматичного оцінювання.

3. Ознайомлення з великими мовними моделями.

4. Проектування архітектури застосунку.

5. Розробка алгоритмів для автоматичного оцінювання.

6. Розробка інтерфейсу користувача.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми автоматичного оцінювання та проведення навчального процесу, постановка задачі, діаграма архітектури застосунку, діаграма бази даних застосунку, зображення результатів тестування програми.

---



---



---



---

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

### КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.04.2024	
2	Аналіз завдання, підбір літератури	08.04.24-11.04.24	
3	Аналіз літератури з досліджуваної проблеми	12.04.24-14.04.24	
4	Аналіз технічних засобів	15.04.24-17.04.24	
5	Розробка алгоритмів	18.04.24-22.05.24	
6	Програмна реалізація	23.04.24-17.05.24	
7	Оформлення пояснювальної записки	18.05.24-2.05.24	
8	Перевірка на плагіат	27.05.24	
9	Рецензування	28.05.24	
10	Підготовка презентації та доповіді	29.05.24-02.06.24	
11	Занесення роботи в електронний архів	06.06.24	
12	Попередній захист кваліфікаційної роботи	06.06.24	

Дата видачі завдання 8 квітня 2024 р.

Студент \_\_\_\_\_  
(підпис)

Керівник роботи \_\_\_\_\_ доц. Яковлева О. В.  
(підпис) (посада, прізвище, ініціали)

## РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 70 с., 4 табл., 39 рис., 39 джерел.

LLM МОДЕЛІ, МЕТОДИ АВТОМАТИЧНОГО ОЦІНЮВАННЯ, GRPC, HTTPS, C#, ANGULAR.

Об'єктом роботи є питання тестування знань студентів з використанням методу автоматичного оцінювання відповідей студента.

Метою роботи є розробка застосунку для моніторингу знань студентів з мови SQL з використанням методу автоматичного оцінювання за допомогою LLM моделі.

Було створено вебзастосунок, де для реалізації клієнтської частини використано framework Angular, а серверна частина складається з декількох мікросервісів, написаних за допомогою платформи .NET, які спілкуються за допомогою протоколу GRPC та протоколу HTTPS. Для розробки методу автоматичного оцінювання була використана LLM GPT модель. Середовищами розробки були Visual Studio та Visual Studio Code. Для тестування серверної частини використовувались Postman та Swagger.

У результаті роботи здійснена програмна реалізація застосунку для перевірки знань з мови SQL з можливістю автоматичного оцінювання відповіді студента.

LLM MODELS, AUTOMATIC ASSESSMENT METHODS, GRPC, HTTPS, C#, ANGULAR.

The object of the work is the issue of testing students' knowledge using the method of automatic evaluation of student answers.

The purpose of the work is to develop an application for monitoring students' knowledge of SQL using the method of automatic assessment using the LLM model.

A web application has been created where the Angular framework is used to implement the client side, and the server side consists of several microservices written using the .NET platform that communicate using the GRPC protocol and the HTTPS protocol. The LLM GPT model was used to develop the automatic evaluation method. The development environments were Visual Studio and Visual Studio Code. Postman and Swagger were used to test the server side.

As a result of the work, the software implementation of the application for testing knowledge of SQL with the ability to automatically evaluate the student's answer was carried out.

## ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів .....	7
Вступ.....	8
1 Сучасний стан сервісів для онлайн навчання та тестування знань.....	9
1.1 Сучасні сервіси щодо онлайн навчання та перевірки знань.....	9
1.2 Сервіси онлайн навчання з використанням штучного інтелекту...	10
1.3 Популярність мови SQL та сервіси її для вивчення .....	11
1.4 Прогрес великих мовних моделей.....	12
1.5 Порівняння мовних моделей та вибір оптимальної .....	13
1.6 Підходи до проектування вебзастосунків.....	16
1.7 Програмне забезпечення для створення вебзастосунків.....	17
1.8 Постановка задачі .....	18
2 Проектування вебзастосунку для тестування знань студентів з мови SQL з використанням моделей GPT для автоматичного оцінювання відповідей .....	20
2.1 Специфікація вимог .....	20
2.2 Проектування архітектури застосунку .....	21
2.3 Проектування баз даних .....	22
2.4 Створення GUI .....	26
2.5 Розробка модулю для оцінювання відповідей студентів.....	32
2.5.1 Алгоритм оцінювання на основі евристичних правил.....	33
2.5.2 Використання моделей GPT для оцінювання відповідей .....	34
2.5.3 Розробка фінального алгоритму оцінювання відповідей.....	35
3 Розробка вебзастосунку для тестування знань мови SQL з використанням GPT моделі для автоматичного оцінювання відповідей .....	38
3.1 Вибір та налаштування програмного середовища.....	38
3.2 Ілюстрація роботи застосунка .....	40
3.2.1 Створення груп студентів.....	41
3.2.2 Створення тем, завдань та тестів.....	42
3.2.3 Проходження тестів та перегляд результатів.....	49

	6
3.3 Аналіз роботи алгоритму автоматичного оцінювання .....	51
Висновки .....	65
Перелік джерел посилання .....	67

## **ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ**

LLM – Large Language Model (велика мовна модель)

UI – User Interface (користувацький інтерфейс)

SQL – Structured Query Language (структурована мова запитів)

JSON – JavaScript Object Notation (запис об'єктів JavaScript)

API – Application Programming Interface (інтерфейс програмного застосування)

GPT - Generative Pre-trained Transformer (породжувальний попередньо натренований трансформер)

HTTP – HyperText Transfer Protocol (протокол передачі гіпертекстових документів)

SOAP – Simple Object Access Protocol (простий протокол доступу до об'єктів)

XML-RPC – Extensible Markup Language Remote Procedure Call (XML-виклик віддалених процедур)

REST – Representational State Transfer (передача репрезентативного стану)

## ВСТУП

У сучасному світі програмування SQL (Structured Query Language) стала невід'ємною частиною баз даних та систем управління базами даних. За допомогою SQL можна виконувати різноманітні операції з даними, такі як створення, редагування, видалення, а також вибірка необхідної інформації з однієї або декількох таблиць.

В свою чергу мова SQL є досить складною та може включати в себе деякі нюанси при її використанні, що неможливо запам'ятати та вивчити одразу без навчання та наявності певних навчальних матеріалів, що можуть бути не одразу зрозумілі для студентів.

Для того, щоб полегшити процес вивчення та оцінювання мови SQL, метою цієї дипломної роботи є розробка застосунку, який буде здатен моніторити та автоматично оцінювати знання студентів з мови SQL. Для цього буде використана LLM GPT модель.

Застосунок буде включати в себе дві складові: перша складова відповідатиме за процес тестування студентів, а друга – за оцінювання відповідей студентів та надання зрозумілих коментарів для покращення розуміння помилки студента при відповіді на певне запитання.

Розробка такого застосунку є цікавою як для практичного, так і для наукового співтовариства. Він може знайти своє застосування у навчальних закладах, допомагаючи студентам вивчати мову SQL та оцінювати їх рівень знань. Крім того, цей проєкт відкриває можливості для дослідження сучасних LLM моделей в розв'язанні реальних завдань.

# 1 СУЧАСНИЙ СТАН СЕРВІСІВ ДЛЯ ОНЛАЙН НАВЧАННЯ ТА ТЕСТУВАННЯ ЗНАНЬ

## 1.1 Сучасні сервіси щодо онлайн навчання та перевірки знань

У сучасному світі існує дуже багато сервісів для онлайн навчання, які дозволяють створювати різні типи тестів, проводити опитування в режимі реального часу та перевіряти знання. Ці сервіси з часом набули велику популярність, що почали співпрацювати з університетами та великими компаніями по всьому світі. Ось декілька прикладів таких сервісів:

– Coursera – це платформа, яка співпрацює з університетами та організаціями з усього світу, щоб надавати курси з різних предметів, включаючи інформатику та бази даних [1];

– edX – це платформа, яка пропонує безкоштовні онлайн-курси від університетів і організацій таких як MIT, Harvard, Microsoft тощо. Вони також мають можливість перевірки знань [2];

– Udeemy – це велика бібліотека, яка складається з різних курсів з різних тематик, включаючи SQL та бази даних. Вони надають перевірки знань через тести та завдання [3];

– Codecademy – ця платформа спеціалізується на програмуванні та надає курси з SQL та баз даних, включаючи можливість перевірки знань через інтерактивні вправи та тести [4].

Ці сервіси надають можливість людям з усього світу отримувати знання та сертифікати, спілкуватися з викладачами та однолітками з усього світу та отримувати доступ до матеріалів у будь-який час та з будь якого місця з Інтернетом.

## 1.2 Сервіси онлайн навчання з використанням штучного інтелекту

Сучасний швидкий темп життя і стрімкий розвиток технологій ставлять перед нами постійну необхідність навчання та пошуку нових знань. У такому контексті сервіси онлайн навчання, що використовують штучний інтелект, стають невід'ємною частиною освіти сучасного світу. Ці сервіси не лише допомагають у навчанні, а й адаптуються до потреб користувачів, створюючи персоналізовані навчальні програми та індивідуальні тестові завдання. Ось декілька онлайн сервісів для навчання, які використовують штучний інтелект:

– Quizlet – використовує штучний інтелект для створення індивідуальних навчальних програм та тестів залежно від потреб користувача [5];

– Duolingo – цей сервіс використовує штучний інтелект для персоналізованого навчання мов [6]. Він адаптується до швидкості та стилю навчання кожного користувача;

– Cerego – цей сервіс використовує адаптивні алгоритми машинного навчання для створення індивідуальних навчальних програм та персоналізованих тестів [7];

– SMARTeacher – цей сервіс використовує штучний інтелект для аналізу результатів тестів та рекомендацій щодо подальшого навчання [8].

Сервіси онлайн навчання з використанням штучного інтелекту перетворюють навчання на інтерактивний, ефективний та особисто зорієнтований процес. Вони сприяють постійному розвитку та набуттю нових навичок у будь-який час та в будь-якому місці, роблячи освіту доступною для всіх.

### 1.3 Популярність мови SQL та сервіси її для вивчення

Більша частина існуючих застосунків на сьогоднішній день базуються на реляційних базах даних [9]. Мова SQL була створена з метою надання програмістам можливості керувати структурою та вмістом баз даних.

Знання SQL стало однією з ключових вимог для молодих спеціалістів при пошуку роботи, оскільки вона використовується в різних системах управління базами даних, таких як MS SQL Server, MySQL, PostgreSQL [10] та інші. Для перевірки та покращення практичних навичок SQL можна скористатися різноманітними сервісами:

– LeetCode – це сервіс, який допомагає підготуватися до співбесід технічного рівня, розвивати алгоритмічне мислення та навички з програмування, розв’язуючи багато завдань з різних тем та рівнів складності [11]. Дозволяє обирати мови програмування, включаючи мову запитів SQL. Також LeetCode надає можливість переглянути рішення різних користувачів. Ваше рішення певної задачі перевіряється на швидкість виконання та продуктивність по використанню пам’яті;

– CodeWars – це сервіс дуже схожий на LeetCode, який так само допомагає вивчити та вдосконалювати власні навички в певній мові програмування за рахунок розв’язання задач різного рівня складності. Також CodeWars надає можливість для перегляду відповідей інших користувачів, оцінювання цих відповідей, а також обговорення задач разом на форумі для кожної задачі, що дозволяє не тільки покращувати свої власні знання, а й обмінюватися досвідом з людьми з різних країн нашої планети [12].

Незважаючи на популярність та велику кількість переваг таких сервісів, ці сервіси мають один великий недолік – це неможливість побудувати навчальний процес для однієї чи декількох груп студентів та налаштувати певний перелік завдань для певної теми під час її вивчення.

## 1.4 Прогрес великих мовних моделей

З різким розвитком штучного інтелекту у нашому світі різко почали розвиватись так звані великі мовні моделі (Large Language Models).

Великі мовні моделі у нас час є одними з найбільш вражаючих відкриттів у сфері штучного інтелекту протягом останніх років. Ці моделі, такі як GPT (Generative Pre-trained Transformer), BERT (Bidirectional Encoder Representations from Transformers) та їх варіації, спроектовані для розуміння та генерації природної мови [13]. Основним методом їх побудови є використання глибоких нейронних мереж, зокрема, трансформерних архітектур.

Такий різкий прогрес LLM у сучасному світі вплинув на різні популярні світові галузі, а саме:

- мовне розуміння та генерація тексту – LLM можуть злегкістю розуміти та генерувати текст на природній мові з вражаючою точністю та креативністю. Це використовується від автоматичного підписування електронних листів до створення змісту для реклами, або медіа, також є випадки використання LLM під час написання книг та журналів;

- машинний переклад – великі мовні моделі можуть не тільки генерувати текст на природній мові, а й перекладати текст з однієї мови на іншу з великою точністю, що допомагає покращувати машинний переклад всіх нами відомих онлайн перекладачів, роблячи їх більш точними;

- пошук інформації – за рахунок великої інформаційної бази, що містить в собі LLM, вона може використовуватись для розуміння користувацьких запитів та покращення результатів пошуку, що робить результати пошуку більше релевантними та інформативними;

- генерація контенту – велика кількість компаній використовують великі мовні моделі для генерації тем та змісту нових сайтів, постів у соціальних мережах, бізнес-звітів та для багатьох інших цілей;

– медична сфера – LLM використовуються для аналізу медичних даних, діагностики, прогнозування захворювань та вивчення наукових публікацій [14, 15];

– освіта та дослідження – великі мовні моделі використовуються для створення нових інтерактивних методів навчання, тестування, оцінювання та дослідницьких інструментів [16].

Прогрес у сучасному світі охопив і прогрес великих мовних моделей. Вони швидко розвиваються та навчаються, що підвищує якість та різноманітність їх використання у різних галузях. Однак разом з цим розвитком з'являються етичні та соціальні питання, які пов'язані з використанням мовних моделей у певній галузі, ці питання потребують уважного розгляду та регулювання.

### 1.5 Порівняння мовних моделей та вибір оптимальної

Із стрімким розвитком мовних моделей, ми отримали вражаючий спектр інструментів, котрий здатен покращити та автоматизувати наше життя. Великі мовні моделі стали асистентами у повсякденному житті людей, надаючи широкий функціонал та підтримку користувачам. Від написання текстів і створення контенту до перекладу мов та аналізу даних, кожна модель має свої унікальні можливості та застосування. Ці моделі здатні навчатися та покращувати свої знання в різних людських сферах, стаючи добрими консультантами в різних людських питаннях. Ось приклади популярних великих мовних моделей:

- GPT-4 Turbo – нова версія великої мовної моделі GPT від OpenAI;
- GPT-3.5 Turbo – модель GPT третього покоління від OpenAI;
- Llama 3 Instruct (8B) – дотренована мовна модель від компанії Meta;
- Mistral Large – комерційна версія мовної моделі від Mistral;
- Claude 3 Sonnet – нове покоління мовної моделі від компанії Claude.

Всі вищеперераховані мовні моделі надають потужний функціонал для роботи з даними та аналізом тексту, проте необхідно обрати з них найоптимальнішу для використання при розробці застосунку для оцінювання знань студентів. Для вибору найоптимальнішої великої мовної моделі було виділено наступні критерії [17]:

- якість використання знань – об’єм знань, яким володіє мовна модель та якість їх використання при аналізі питання, котре задає користувач;
- швидкість – показник швидкості генерування токенів в секунду, для генерації відповіді користувачу;
- ціна – ціна, за місяць використання функціоналу мовної моделі.

Порівняння якості знань великих мовних моделей відображено в таблиці 1.1.

Таблиця 1.1 – Якість використання знань великих мовних моделей

<b>Назва мовної моделі</b>	<b>Якість використання знань у відсотках</b>
GPT-4 Turbo	86%
Mistral Large	81%
Claude 3 Sonnet	79%
GPT-3.5 Turbo	70%
Llama 3 Instruct (8B)	68%

За даними з таблиці можна зрозуміти, що найвища якість використання знань є у мовної моделі від OpenAI GPT-4 Turbo, також можна зазначити, що мовна модель минулого покоління GPT-3.5 Turbo не відстає по показникам від більш нових версій моделей від інших компаній, що показує те, що OpenAI приділяє велику увагу до навчання та покращення знань своїх мовних моделей протягом часу, що зможе надавати більш інформативну та конструктивну відповідь для користувача.

Порівняння швидкості мовних моделей зображено в таблиці 1.2.

Таблиця 1.2 – Порівняння швидкості мовних моделей

<b>Назва моделі</b>	<b>Швидкість генерування токенів на секунду при генерації відповіді</b>
GPT-4 Turbo	24
Mistral Large	30
Claude 3 Sonnet	62
GPT-3.5 Turbo	54
Llama 3 Instruct (8B)	121

У даному порівнянні можна визначати, що найшвидшою моделлю при генеруванні відповіді є мовна модель від компанії Meta – Llama 3 Instruct (8B), проте відсоткове співвідношення якості знань цієї моделі нижча від інших мовних моделей, що може означати, що така швидка відповідь може бути аргументована тим, що моделі потрібно менше часу для аналізу свого об'єму знань, що призводить до швидкої, проте не завжди якісної та конструктивної відповіді. У свою чергу мовна модель GPT-3.5 Turbo займає 3 місце по показнику швидкості, що можна аргументувати трохи більшим об'єм знань проте не таким високим як у інших моделей.

Порівняння ціни за місяць використання зображено у таблиці 1.3.

Таблиця 1.3 – Порівняння ціни за місяць використання моделлю

<b>Назва моделі</b>	<b>Ціна за місяць використання</b>
GPT-4 Turbo	30
Mistral Large	24
Claude 3 Sonnet	15
GPT-3.5 Turbo	1,5
Llama 3 Instruct (8B)	0,2

За даними з таблиці, можна зрозуміти, що ціни за місяць використання дуже різні при використанні різної моделі, модель GPT-4 Turbo є

найдорожчою у порівнянні з іншими моделями, що можна аргументувати високим показником якості знань, тому що OpenAI приділяє дуже багато уваги навчанню своїх моделей для того щоб надавати користувачу максимально зрозумілу та конструктивну відповідь.

Після порівняння кожної моделі за визначеними вище критеріями можна зробити висновок, що оптимальнішою моделлю у відношенні ціна/швидкість/якість знань є модель від OpenAI GPT-3.5 Turbo.

## 1.6 Підходи до проектування вебзастосунків

Вебзастосунок – це програми, що надають певні можливості через Інтернет за допомогою стандартних протоколів, таких як HTTP, SOAP, XML-RPC, REST і т. д. Вони можуть використовуватися для різних цілей, таких як обмін даними, інтеграція систем, автоматизація процесів бізнесу, а також для надання віддалених сервісів. Розробка вебзастосунку вимагає різних підходів методологій, залежно від типу та складності сервісу, його архітектури, використовуваних технологій та інструментів розробки. Ось деякі приклади:

- модельно-орієнтований підхід (Model-Driven Approach). Це підхід, який передбачає використання різних типів абстракції, за допомогою написання моделей для аналізу та опису вимог, структури та поведінки вебзастосунку. Це моделювання дозволяє зробити легшим розуміння так званої «складної» реальності, а також забезпечити надійний зв'язок між технічною реалізацією та логікою сервісу зі сторони бізнесу. Для моделювання вебзастосунку за цим підходом можна використовувати різні мови та стандарти серед них є такі відомі як UML (Unified Modeling Language), WSDL (Web Services Description Language), BPMN (Business Process Model and Notation) тощо;

- сервісний підхід. Це підхід, який представляє у собі організацію вебзастосунку, як набору різної кількості сервісів, які можуть «спілкуватися» між собою та бути перевикористані для різних задач та потреб користувачів.

Такий підхід дозволяє гнучко, масштабовано та адаптивно розробити вебзастосунок. Для реалізації такого підходу можна використовувати різні стилі та патерни (шаблони);

– інтеративний та інкрементний підхід. Цей підхід поєднує в собі концепції гнучкої розробки, об'єктно-орієнтованого програмування та інкременталізму, щоб допомогти командам працювати над великими складними проблемами, не застрягаючи в громіздких проєктах, які можуть стати неактуальними, коли вони нарешті будуть готові до реалізації. Інкрементні модульні проєкти додають один модуль за іншим, наче кладуть цеглу для фундаменту будинку. Оцінка ефективності перших кроків може допомогти спланувати наступні кроки і визначити, чи потрібно коригувати план проєкту.

Існує велика кількість підходів при розробці вебзастосунків. Звісно кожні з них мають свої переваги і недоліки. Використання певного підходу при розробці залежить конкретно від проєкту, а також від самої команди розробників, тому що у кожного може бути свій певний досвід з деяким підходом до розробки. Тому для того, щоб не уповільнювати процес розробки необхідно вибрати найбільш зручний для команди підхід.

## 1.7 Програмне забезпечення для створення вебзастосунків

При проєктуванні та розробці вебзастосунку необхідно використовувати різноманітні технології та програмні забезпечення, які залежать від архітектури та складності застосунку для того, щоб побудувати правильний робочий процес у команді [18]. Ось декілька прикладів:

– інструменти для спілкування – це інструменти, які дозволяють комунікувати різним членам команди та зацікавленими у проєкті людьми для обговорення певних задач та актуалізації процесів роботи;

– інструменти для розробки – це інструменти, які дозволяють програмістам розробляти необхідну логіку застосунку;

– інструменти для моделювання – це інструменти, які дозволяють описувати та аналізувати вимог на рахунок певної задачі, для того щоб правильно побудувати процес та логіку певної задачі, що дозволить бачити цю задачу як правильний набір логічних інструкцій;

– інструменти для створення задач та моніторингу їх статусу – це інструменти, які дозволяють бачити та розуміти над якою задачею працюють певні члени команди, скільки часу потрібно на виконання певної задачі, або для якої задачі потрібно більше інформації чи вимог;

– інструменти для зберігання актуальної версії проекту – це інструменти, які дозволяють зберігати останню актуальну версію вебзастосунку та надає можливість для покращення та оновлення проекту до нової версії;

– інструменти для розгортання вебзастосунку – це інструменти, які дозволяють запуснути і підтримувати вебзастосунок на певному сервісі.

Вибір інструментів для створення вебзастосунку є одним з найважливіших аспектів при розробці додатку.

## 1.8 Постановка задачі

На сьогоднішній день мова SQL є важливою складовою у сучасному світі інформаційних технологій та відіграє ключову роль в інформаційному суспільстві. Проте відсутність відповідного сервісу для створення навчального процесу з її вивчення та недостатня увага до контролю та моніторингу знань студентів у цій галузі залишаються актуальними проблемами.

Таким чином, питання щодо розробки вебзастосунку для моніторингу знань студентів з мови SQL з автоматичним методом оцінювання є актуальною

задачею на сьогоднішній день. Вирішення цього питання допоможе організувати навчальний процес для вивчення технологій та дисциплін, що пов'язані з базами даних.

Об'єктом роботи є питання тестування знань студентів з використанням метода автоматичного оцінювання відповідей студента.

Метою роботи є розробка застосунку для моніторингу знань студентів з мови SQL з використанням методу автоматичного оцінювання за допомогою LLM моделі.

Для досягнення цієї мети було сформульовано наступні задачі:

- розглянути сучасний стан питання щодо розробки застосунку для тестування мови SQL з методом автоматичного оцінювання з використанням штучного інтелекту;
- розглянути сучасні застосунки для вивчення мови SQL;
- ознайомитись з різними LLM моделями та підібрати найбільш оптимальну для вирішення цього питання;
- навчитись генерувати запити до LLM моделі для отримання найбільш бажаного оцінювання щодо відповіді студента;
- спроектувати архітектуру вебзастосунку;
- спроектувати бази даних для вебзастосунку;
- спроектувати модуль тестування знань з мови SQL для таких типів користувачів: адміністратор, вчитель, студент, який буде забезпечувати автоматичну перевірку відповідей;
- спроектувати модуль оцінювання відповідей студента для отримання оцінки та коментарів щодо помилок студента;
- розробити інтерфейс користувача, який буде зручним та простим у використанні;
- реалізувати спроектований модуль тестування;
- провести тестування розробленого застосунку та провести оцінювання роботи модулів вебзастосунку.

## 2 ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ ДЛЯ ТЕСТУВАННЯ ЗНАТЬ СТУДЕНТІВ З МОВИ SQL З ВИКОРИСТАННЯМ МОДЕЛЕЙ GPT ДЛЯ АВТОМАТИЧНОГО ОЦІНЮВАННЯ ВІДПОВІДЕЙ

### 2.1 Специфікація вимог

Вебзастосунок для тестування знань студентів з мови SQL – це система з методом автоматичним оцінювання, з використанням GPT моделі, яка допоможе відслідковувати знання студентів з мови SQL, та налагоджувати навчальний процес.

Система повинна надавати підтримку для 4 типів користувачів:

– «Викладач». Користувач з даною роллю повинен мати можливість створювати питання, тести, назначати час проходження тестів, а також створювати користувачів з роллю Студент. Також переглядати результати студента по певному тесту. Окрім того користувач з роллю Викладач повинен мати можливість виправити оцінку з певного питання, якщо система невпевнена оцінці студента;

– «Адміністратор». Цей тип користувача повинен мати всі можливості, доступні користувачу з роллю «Викладач», і може створювати, редагувати та видаляти користувачів будь-якої ролі, крім адміністраторів;

– «Студент». Цей тип користувача повинен мати можливості проходити тести, за рахунок написання SQL-скриптів в якості відповіді на конкретне питання тесту, а також переглядати результати своїх тестів та оцінки, які були виставлені «Викладачем» або «Адміністратором» та системою;

– «Звичайний користувач». Цей тип користувача повинен мати можливість надавати відповіді на тести, які виставлені у публічний доступ «Викладачем» або «Адміністратором» (ці тести не можуть бути використанні для тестування «Студентів»).

## 2.2 Проєктування архітектури застосунку

Вебзастосунок – це програма із клієнт-серверною архітектурою, клієнт якої розроблений за допомогою фреймворку, який підтримується Google, Angular, а серверна частина – поділена на декілька сервісів, які спілкуються між собою за допомогою протоколу GRPC та HTTPS. Сервіси написані за допомогою мов C# та Python. Архітектура застосунку зображена на рисунку 2.1.

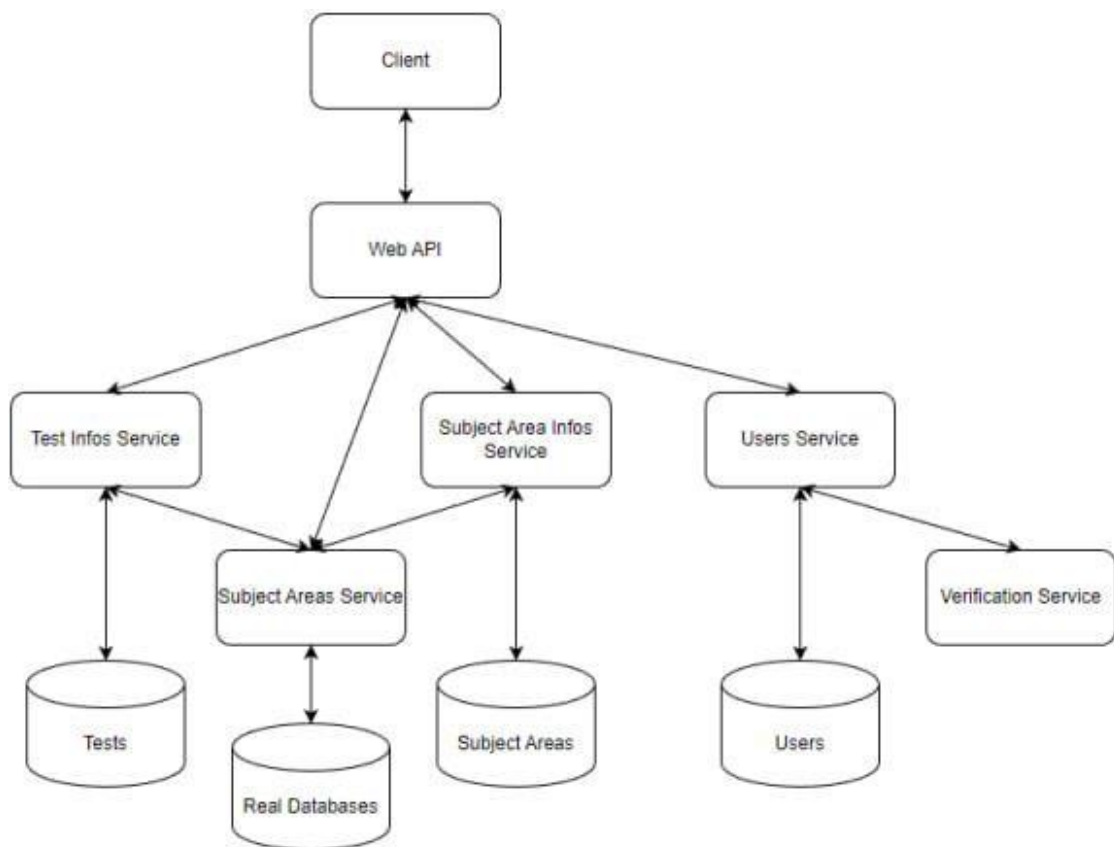


Рисунок 2.1 – Діаграма архітектури застосунку

Client – лицева частина вебзастосунку, котру бачить користувач.

Web API – сервер який виконує роль оркестратора, котрий перепрямовує запит з клієнтської частини на необхідний мікросервіс, для виконання певного блоку логіки в кодї.

Subject Area Infos Service відповідає за обробку запитів щодо маніпулювання даними в наявних базах даних. Він зберігає інформацію про всі бази даних, їх таблиці та атрибути цих таблиць.

Users Service відповідає за управління даними користувачів, а також здійснює авторизацію та реєстрацію користувачів, а також верифікацію обличчя студента.

Subject Areas Service здійснює запити на створення або видалення баз даних, а також проводить маніпуляції з існуючими базами даних. Крім того, він виконує SQL-скрипти студентів та повертає результат стосовно певного запиту.

Test Infos Service відповідає за обробку запитів щодо створення, редагування та видалення тестів та питань, а також за надання відповідей на питання від студентів. Крім того, цей сервіс також відповідає за оцінювання відповідей студентів на запитання тесту, забезпечуючи процес оцінки та повернення результатів студентам з певними коментарями з приводу їх відповіді.

### 2.3 Проектування баз даних

У мікросервісній архітектурі кожний сервіс має власну базу даних, що вимагає більш детального планування та дотримання правил нормалізації баз даних, при проектуванні бази даних для кожного з вебсерверів, з яких складається вебзастосунок.

Users Service працює з базою даних користувачів, яка складається з наступних таблиць:

- Users – таблиця, яка зберігає інформацію про користувача, складається з унікального ідентифікатора, електронної адреса, ім'я користувача, зашифрований пароль, інформація про роль та групу користувача, якщо користувач має роль Студент;

– Roles – таблиця, яка зберігає інформацію про ролі користувачів у системі, складається з імені ролі та унікального ідентифікатора;

– Groups – таблиця, яка зберігає інформацію про групи студентів, складається з унікального ідентифікатора групи та назви групи у системі.

Схема бази даних для Users Service зображена на рисунку 2.2.

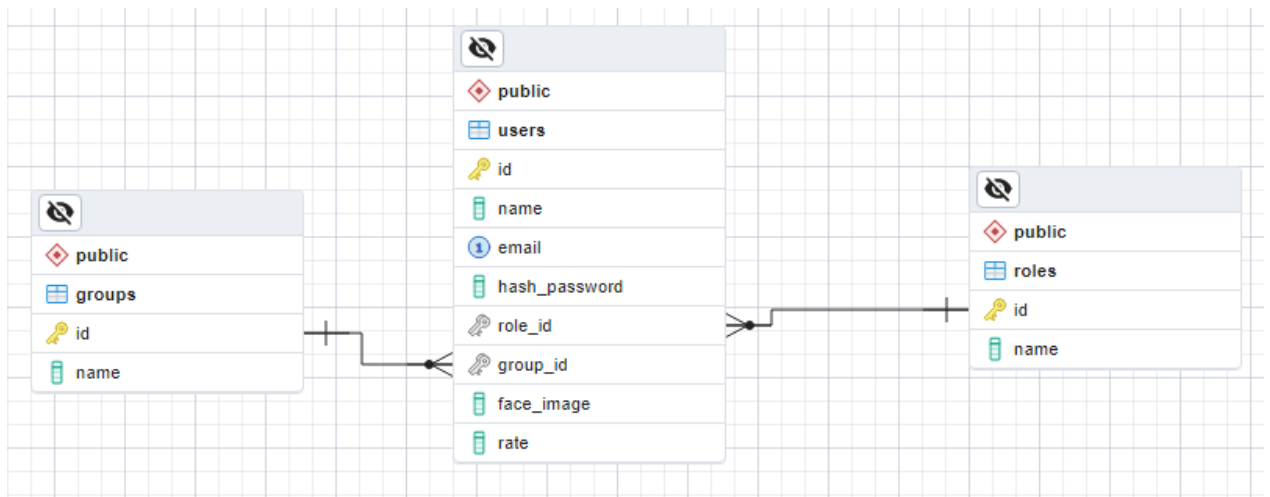


Рисунок 2.2 – Схема бази даних для Users Service

Test Infos Service працює з базою даних тестів, яка складається з наступних таблиць:

– Topics – таблиця, яка відповідає за зберігання тем для питань, складається з унікального ідентифікатора теми та її назви;

– Questions – таблиця, яка відповідає за зберігання питань, складається з унікального ідентифікатора питання, тіла питання та складності цього питання;

– Correct Answers – таблиця, яка відповідає за зберігання правильних відповідей певного запитання, складається з унікального ідентифікатора відповіді, тіла відповіді та інформації до якого питання належить відповідь;

– Tests – таблиця, яка відповідає за зберігання тестів, складається з унікального ідентифікатора тесту, назви тесту та дати створення тесту;

– Test Questions – проміжна таблиця між таблицями Questions та Tests, котра зберігає відношення питань до певного тесту;

– Schedule Tests – таблиця, яка відповідає за зберігання інформації про те який тестів, складається з унікального ідентифікатора розкладу, дати початку тесту, дати закінчення тесту та інформацію який це саме тест;

– User Schedule Tests – таблиця, яка відповідає за зберігання відношення який тест по розкладу призначений якому студенту, складається з унікального ідентифікатора відношення, інформації про тест та інформації про студента;

– User Answers – таблиця, яка відповідає за зберігання інформації про відповіді студента на певне питання певного тесту за розкладом, складається з інформації про студента, інформації про тест за розкладом, інформації про питання, відповіді студента, оцінки від системи та оцінки від керівника.

Схема бази даних для Test Infos Service зображена на рисунку 2.3.

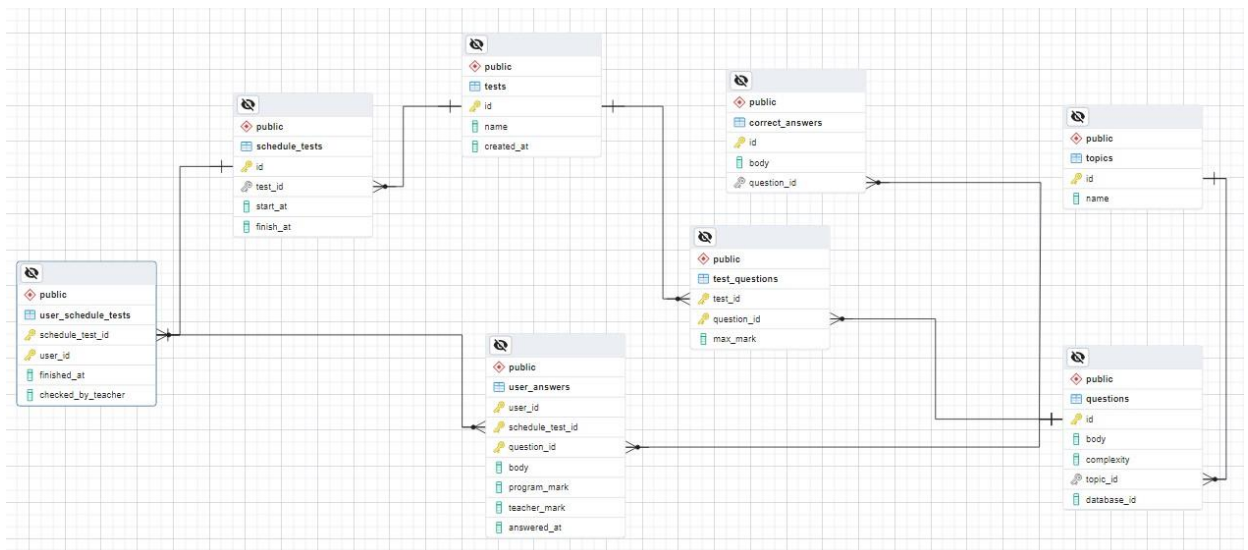


Рисунок 2.3 – Схема бази даних для Test Infos Service

Subject Area Infos Service працює з базою даних, яка зберігає інформацію про створені бази даних, у рамках який проводяться тести для студентів, яка складається з наступних таблиць:

– Languages – таблиця, яка відповідає за зберігання інформації про під мови SQL, складається з унікального ідентифікатора мови та назви мови;

– Databases – таблиця, яка відповідає за зберігання інформації про базу даних, складається з унікального ідентифікатора бази даних, назви та інформації до якої під мови належить;

– Tables – таблиця, яка відповідає за зберігання інформації про таблиці в певній базі даних, складається з унікального ідентифікатора таблиці, назви та інформації до якої бази даних належить ця таблиця;

– Attributes – таблиця, яка відповідає за зберігання інформації про атрибути певної таблиці, складається з унікального ідентифікатора атрибуту, назви, типу даних атрибуту, кількості можливих символів, значення по замовченню для певного атрибуту та інформації до якої таблиці належить певний атрибут.

Схема бази даних для Subject Area Infos Service зображена на рисунку 2.4.

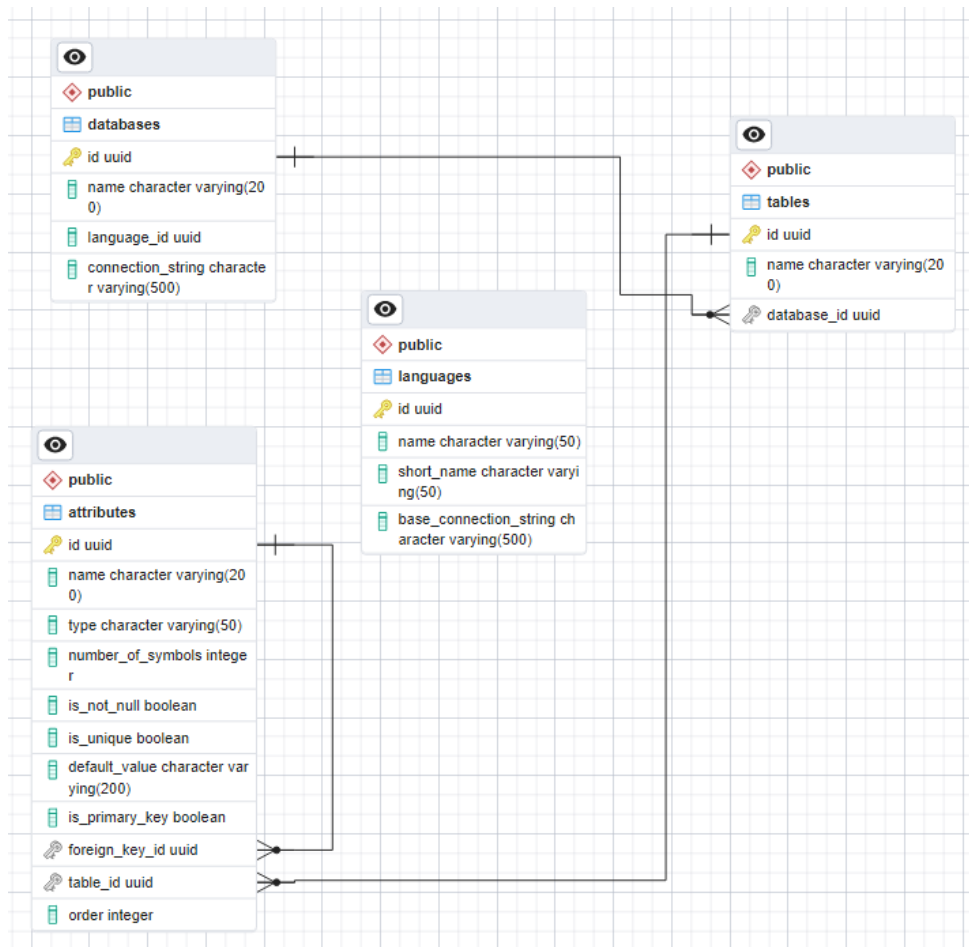


Рисунок 2.4 – Схема бази даних для Subject Area Infos Service

## 2.4 Створення GUI

Усі користувачі мають спільну головну панель з кнопками авторизації та реєстрації, а також навігаційну панель, в якій різні типи користувачів бачать різні пункти меню.

Зокрема, користувачі з ролями «Адміністратор» або «Викладач» мають доступ до 7 пунктів меню на навігаційній панелі (рис. 2.5): «Tests», «Themes», «Questions», «Subjects», «Users», «Groups» та «Schedule Tests».

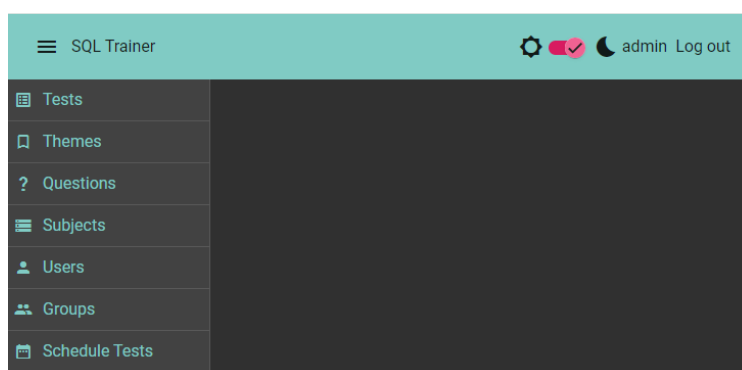
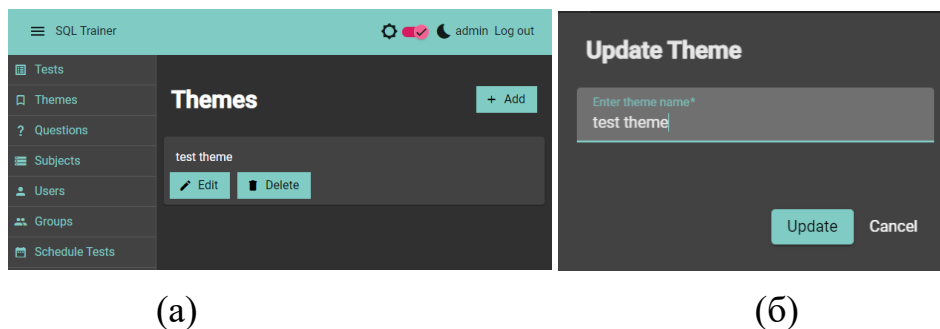


Рисунок 2.5 – Навігаційна панель Викладача/Адміністратора

У розділі «Themes» користувачі можуть переглянути існуючі теми, доступні в системі (рис. 2.6 (а)). Вони також можуть створювати, редагувати та видаляти теми. Однак у розділі створення/редагування (рис. 2.6 (б)) користувачеві доступне лише поле для введення назви теми.



(а)

(б)

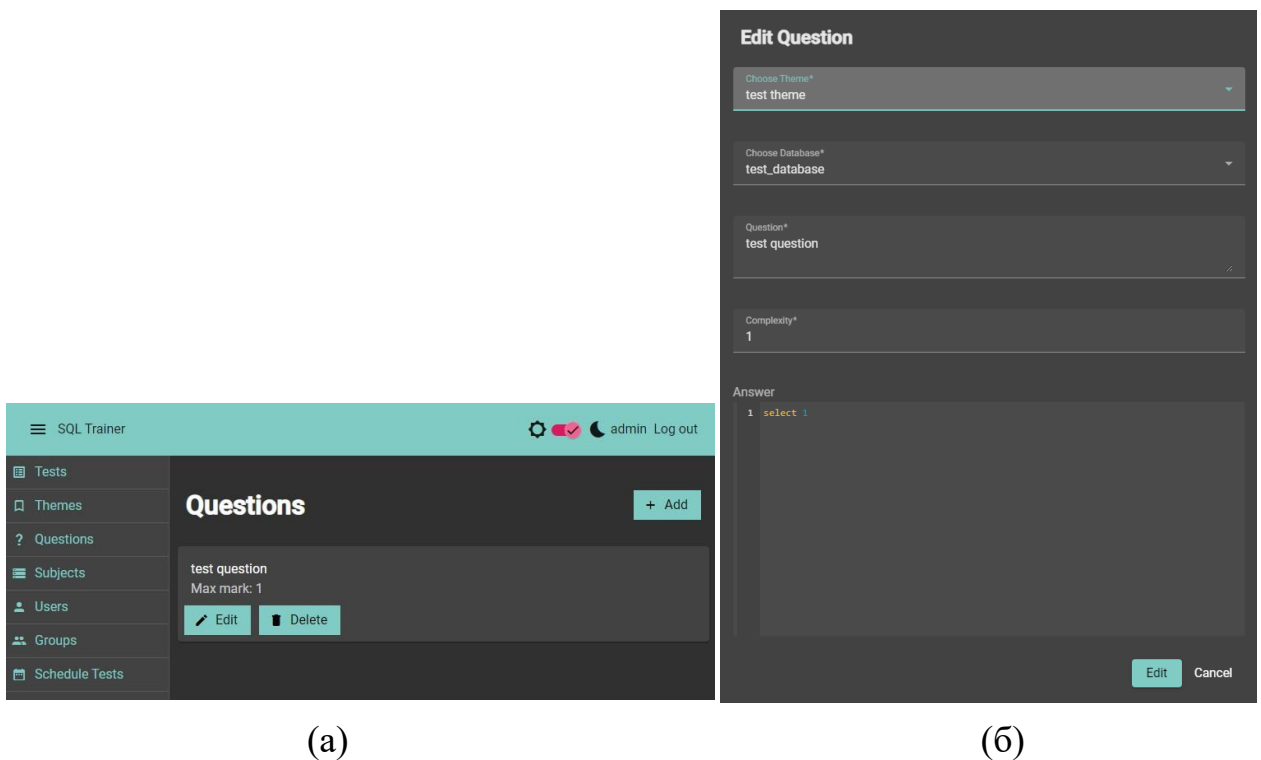
Рисунок 2.6 – Розділ «Themes»:

(а) вигляд меню «Themes»;

(б) вигляд вікна для створення або редагування теми

У розділі «Questions» користувач може переглянути всі доступні питання в базі даних програми (рис. 2.7 (а)). Тут він має можливість додавати нові питання, видаляти існуючі, а також редагувати їх зміст і складність.

У вікні створення/редагування питання (рис. 2.7 (б)) користувачеві необхідно вибрати тему зі списку, вибрати базу даних, до якої буде зроблено запит, ввести текст питання і його складність, а також вказати правильну відповідь.



(a)

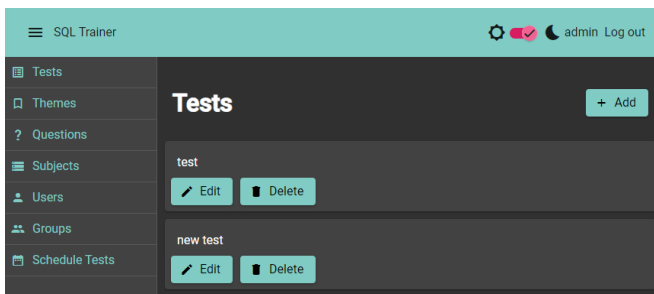
(б)

Рисунок 2.7 – Розділ «Questions»:

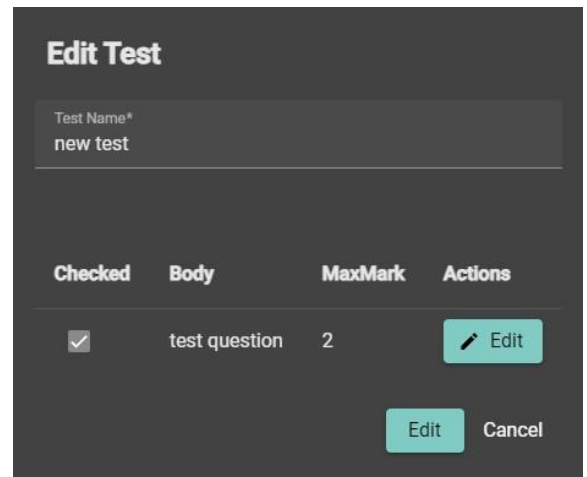
(а) вигляд меню «Questions»;

(б) вигляд вікна для створення/редагування питання

У розділі «Tests» користувач може побачити усі наявні в системі тести (рис. 2.8 (а)). Система надає користувачеві можливість створювати, редагувати та видаляти тести. У вікні створення або редагування тесту (рис. 2.8 (б)) користувачеві необхідно вказати назву тесту та обрати запитання з наявних у системі, які будуть використовуватись у цьому тесті.



(a)



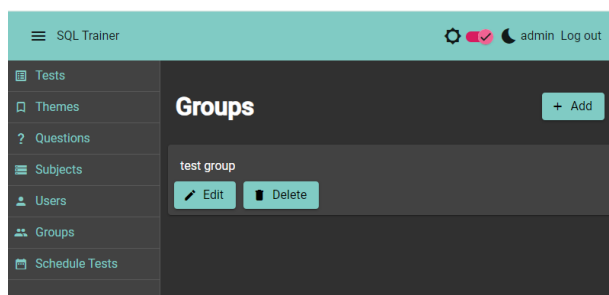
(б)

Рисунок 2.8 – Розділ «Tests»:

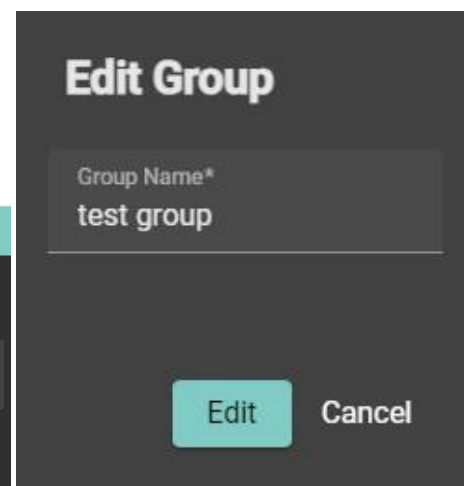
(a) вигляд меню «Tests»;

(б) вікно створення/редагування тесту

У меню «Groups» користувач може бачити усі наявні групи для студентів (рис. 2.9 (a)). Користувач також може створювати, редагувати та видаляти групи. Під час створення або редагування групи (рис. 2.9 (б)) користувачеві необхідно лише вказати або оновити назву групи.



(a)



(б)

Рисунок 2.9 – Розділ «Groups»:

(a) вигляд меню «Groups»;

(б) вікно створення/редагування групи

У розділі «Users» (рис. 2.10 (а)) користувач із роллю «Адміністратор» може бачити усіх користувачів, користувач із роллю «Викладач» бачить тільки користувачів із роллю «Студент». Користувачі з ролями «Адміністратор» та «Викладач» мають можливість додавати, редагувати та видаляти інших користувачів.

У вікні для створення/редагування користувача (рис. 2.10 (б)) необхідно ввести ім'я користувача, електронну адресу та пароль. Під час створення користувача із роллю «Студент» необхідно обрати групу до якої цей студент належить та завантажити фотографію, де чітко видно обличчя користувача, для проходження майбутньої верифікації перед початком тесту.

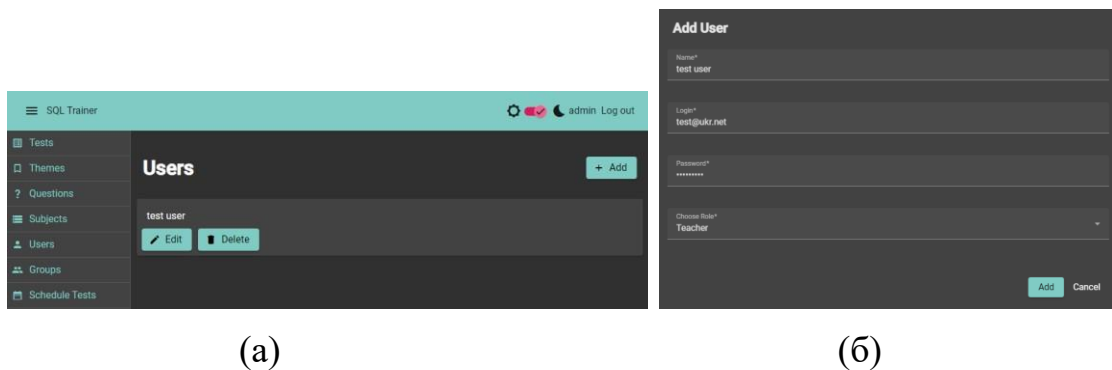


Рисунок 2.10 – Розділ «Users»:

(а) вигляд меню «Users»;

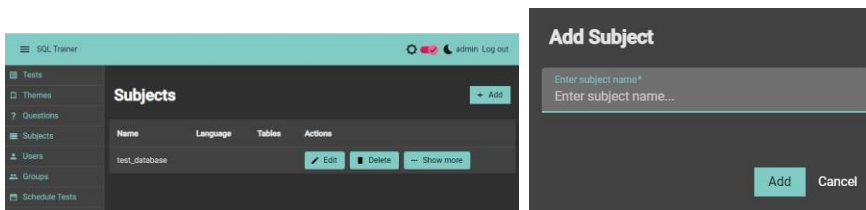
(б) вікно для створення/редагування користувача

У розділі «Subjects» (рис. 2.11 (а)) користувач може додавати, редагувати (рис. 2.11 (б)) або видаляти бази даних, які будуть використовуватись під час проходження тесту.

Також користувач може додавати, редагувати та видаляти таблиці у існуючій базі даних (рис. 2.11 (в)). Додавання або редагування таблиці (рис. 2.11 (г)) потребує вводу назви таблиці та додавання хоча б одного атрибуту до цієї таблиці. Атрибуту необхідно ввести назву, тип та порядок цього атрибуту у полі «Order», ввід даних у поля primary key, not null, unique, default або foreign key не є обов'язковим.

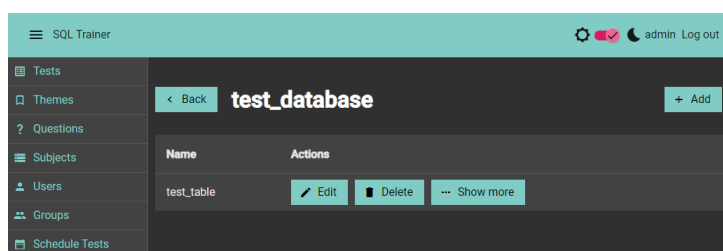
Також користувач має можливість маніпулювати даними певної таблиці (додавати, редагувати або видаляти).

Перегляд даних у таблиці показано на рисунку 2.11 (г), а їх додавання на рисунку 2.11 (д).

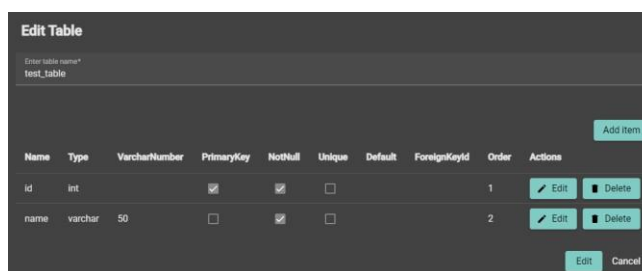


(a)

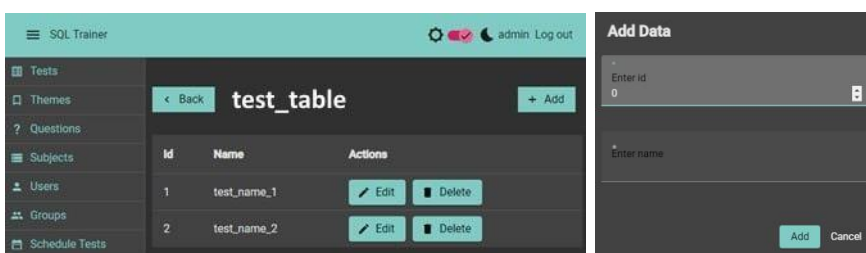
(б)



(в)



(г)



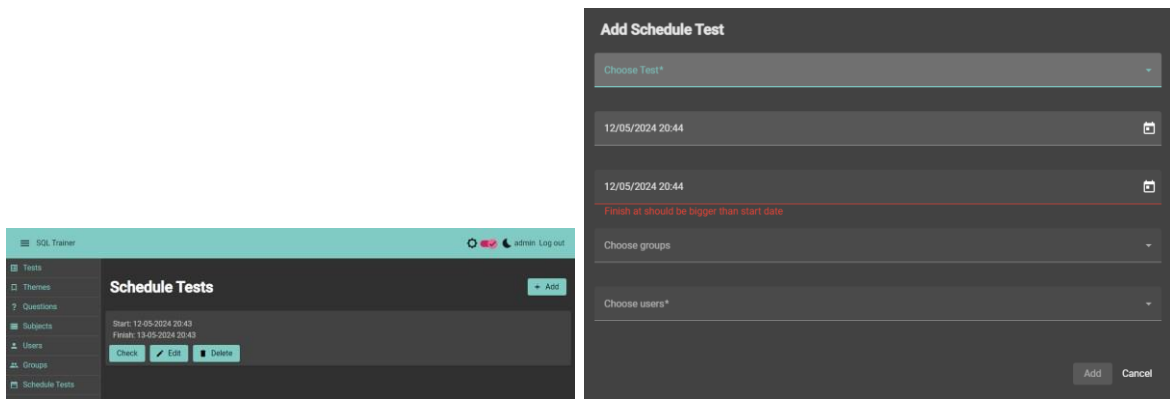
(д)

(д)

Рисунок 2.11 – Розділ «Subjects»:

- (а) вигляд меню баз даних; (б) вікно додавання бази даних;
- (в) вигляд меню таблиць; (г) вікно додавання/редагування таблиці;
- (д) вигляд меню даних таблиці; (д) вікно додавання даних у таблицю

У розділі «Schedule Tests» (рис. 2.12 (а)) користувач бачить розклад тестів які йому назначено. При створенні або редагуванні тесту за розкладом (рис. 2.12 (б)) необхідно обрати тест, обрати початковий та кінцевий час тестуа обрати користувачів, які будуть його проходити.



(а)

(б)

Рисунок 2.12 – Вигляд розділу «Schedule Tests»:

(а) вигляд меню «Schedule Tests»;

(б) вікно для додавання/редагування тесту за розкладом

Також користувач із роллю «Викладач» або «Адміністратор» можуть переглядати відповіді всіх студентів. При перегляді відповідей користувач бачить питання, правильну відповідь на це питання, відповідь студента, максимальну оцінку за правильну відповідь на це питання, оцінку та коментарі, що поставила система, та поле для вводу оцінки від викладача.

Користувач із роллю «Студент» має доступ лише до розділу «Schedule Tests». На цьому розділі він бачить лише ті тести за розкладом, які призначені йому. Користувач може розпочати проходження тесту лише у час початку тесту. Під час проходження тесту користувач бачить меню проходження тесту (рис. 2.13). Це меню складається з декількох частин: заголовок, структура бази даних, поле відповіді на запитання та панель навігації між питаннями. У заголовку відображаються кнопка для виходу із тесту, назва тесту та таймер із залишеним часом. Структура бази даних представляє собою дерево, де перший рівень – назва бази даних, другий рівень – таблиці, а третій

рівень – атрибуту. Панель навігації між питаннями містить список усіх питань та дві кнопки: «Check all» та «Submit». Кнопка «Check all» робить запит на серверну частину в якому передає всі відповіді студента, сервер у свою чергу виконує кожен скрипт, написаний студентом та перевіряє його правильність, та повертає відповідний результат. Натискання кнопки «Submit» призведе до збереження відповідей на запитання та виходу студента із меню проходження тесту.

Після проходження та завершення часу тесту користувач має можливість переглянути результати та побачити коментарі та оцінки за відповіді тесту.

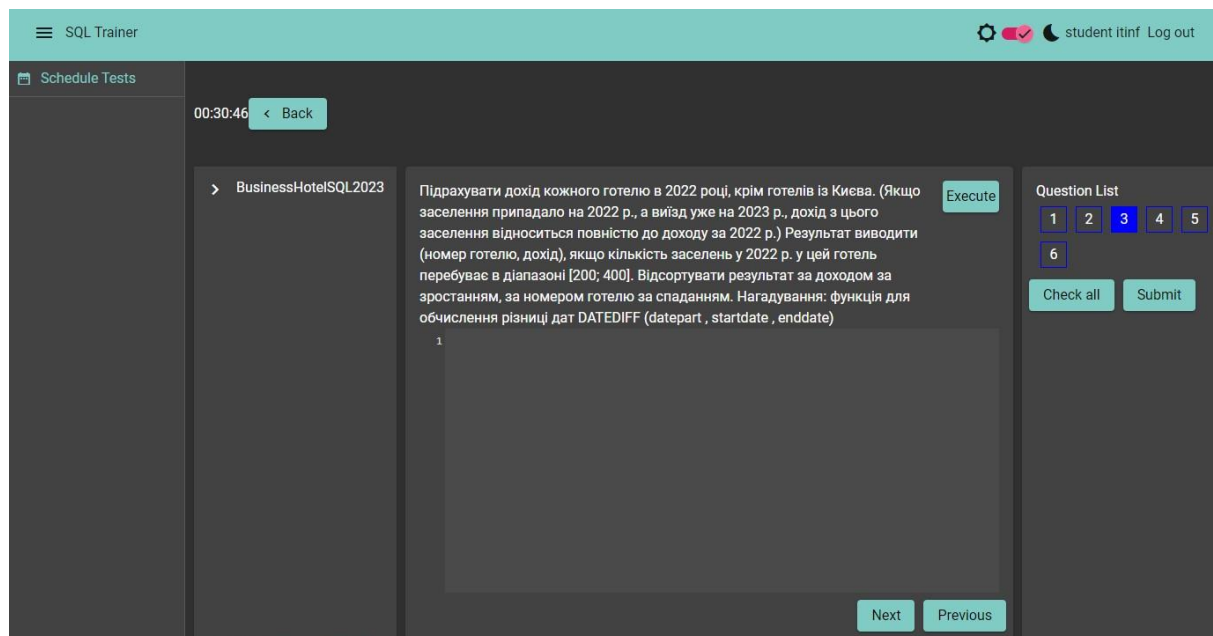


Рисунок 2.13 – Меню проходження тесту

## 2.5 Розробка модулю для оцінювання відповідей студентів

Модуль тестування є центральним у додатку, оскільки він містить основну частину логіки домену. Він відповідає за створення тестів і питань до них, автоматичну перевірку відповідей студентів та інші важливі функції. Крім того, цей модуль має обмежені зв'язки з іншими модулями, такими як

ідентифікатор користувача, який проходить тест, та ідентифікатор бази даних, з якою пов'язані питання в тесті. Однак всередині модуля не зберігається ніякої додаткової інформації про користувачів або бази даних, лише їхні ідентифікатори.

Модуль тестування має у собі найважливішу логіку: алгоритм автоматичного оцінювання SQL-запитів.

При розробці алгоритму автоматичного оцінювання було розглянуто два алгоритми оцінювання, а саме:

- алгоритм оцінювання на основі евристичних правил;
- використання моделей GPT для оцінювання відповідей.

### 2.5.1 Алгоритм оцінювання на основі евристичних правил

Алгоритм оцінювання на основі евристичних правил перевіряє правильність виконання SQL-запитів та не має підтримки для пошуку спеціальних команд, що можуть бути зазначені у питанні.

Сам алгоритм складається з декількох кроків:

Крок 1. Отримання результатів двох сценаріїв: сценарію відповіді учня та сценарію правильної відповіді.

Крок 2. Перетворення результатів у масив словників, де ключ – ім'я атрибута, а значення – конкретне значення атрибута для кожного рядка.

Крок 3. Порівняння кількості рядків у результатах запиту. Якщо результати мають різну кількість рядків, кількість балів зменшується вдвічі від максимально можливого.

Крок 4. Порівняння кількості атрибутів у результатах. Якщо кількість атрибутів відрізняється, оцінка зменшується на 10% від оцінки, отриманої на Кроці 3.

Крок 5. Порівняння назви атрибутів у результатах обох сценаріїв. Якщо кількість атрибутів, для яких не знайдено відповідної назви, перевищує

різницю в кількості атрибутів у результатах сценаріїв, оцінка зменшується на 10% від оцінки, отриманої на Кроці 4.

Крок 6. Порівняння даних. Обчислюємо кількість відповідних значень. Обчислюємо максимальну кількість подібних значень: максимальну кількість атрибутів серед двох результатів, помножену на максимальну кількість рядків серед двох результатів. Розділяємо кількість відповідних значень на максимальну кількість подібних значень. Отриманий коефіцієнт множиться на бал, отриманий на Кроці 5, і округлюється до двох знаків після коми.

### 2.5.2 Використання моделей GPT для оцінювання відповідей

Використання моделей GPT для оцінювання відповідей означає застосування мовних моделей, таких як GPT (Generative Pre-trained Transformer), для аналізу та оцінювання текстових відповідей. Цей підхід полягає у використанні штучного інтелекту для автоматичної обробки та розуміння тексту, який надходить від учнів у відповідь на запитання. GPT модель від OpenAI показала свою користь для вирішення багатьох задач.

Оцінка відповідей за допомогою моделей GPT може включати наступні кроки:

- перетворення тексту в числове представлення – текст відповіді перетворюється на числову форму, яку можна обробити за допомогою GPT-моделі;

- генерація контекстно-релевантного тексту – велика мовна модель GPT генерує новий текст, який відображає розуміння і відповідь на запитання, використовуючи контекст тексту запитання і тексту відповіді;

- оцінювання згенерованої відповіді – оцінка базується на різних показниках, таких як правильність відповіді, логічність, повнота, адекватність тощо;

- порівняння з еталоном – згенерована відповідь порівнюється з еталоном або правильною відповіддю, щоб визначити рівень відповідності;
- виведення результату – на основі оцінки модель повертає результат, який може бути використаний для виставлення оцінки студенту.

Для спілкування з моделлю GPT існують такі ролі:

- System – промпт, який допомагає моделі пояснити як треба виконати завдання;
- User – запит користувача;
- Assistant – відповідь моделі.

### 2.5.3 Розробка фінального алгоритму оцінювання відповідей

На основі прикладів двох варіантів методів автоматичного оцінювання можна зазначити, що метод, побудований на евристичних правилах, не є надійним. Це через те, що такий алгоритм не проводить перевірку на те, чи виконує реально студентський скрипт необхідні завдання. Можлива ситуація, коли результати від еталонного скрипту та скрипту студента співпадають, але це може бути випадковістю, і при інших умовах співпадіння може не відбутися, що свідчить про те, що скрипт студента не виконує поставлене завдання. Тому для того, щоб переконатися у оцінці, яку було отримано за допомогою алгоритму на основі евристичних правил, було прийнято рішення підключити велику мовну модель GPT, котра буде паралельно з алгоритмом на основі евристичних правил оцінювати відповідь студента та надавати йому оцінку з коментарем. Після завершення роботи обох алгоритмів, результати їх робіт перевіряються на наявність викидень (коли оцінка від мовної моделі відрізняється від оцінки алгоритму на основі евристичних правил на 20%). Якщо викидень присутній, то система помічає оцінку, що при оцінюванні цієї відповіді є невпевненість, про яку буде повідомлено викладачу, для подальшої перевірки цієї оцінки.

Для того, щоб підключити мовну модель GPT необхідно отримати ключ доступу на сайті OpenAI, і при перевірці відповідей студента генерувати промпт (рис. 2.14), завдяки якому система отримає оцінку в балах, яка потім буде порівняна з оцінкою від алгоритму на основі евристичних правил та коментар-аргументацію (рис 2.15).

```

Ти виконуєш оцінювання відповідей студентів з мови SQL.
Тобі дається завдання, максимальна оцінка за це завдання та
відповідь студента.
Проаналізуй завдання(дивись уважно на вимоги, що зазначено у
завданні в дужках) та перевір чи виконує запит студента поточне
завдання та виводить вірну кількість атрибутів.
Якщо студент не виконує якусь вимогу з завдання(не використовує
потрібний оператор, повертає не вірну кількість атрибутів, не
виконує сортування тощо) знімай 10% від результуючої оцінки.
Якщо відповідь студента не відповідає умові завдання став 0.

Завдання: Підрахувати дохід кожного готелю в 2022 році, крім
готелів із Києва.
(Якщо заселення припадало на 2022 р., а виїзд уже на 2023 р.,
дохід з цього заселення відноситься повністю до доходу за 2022 р.)
Результат виводити (номер готелю, дохід), якщо кількість заселень
у 2022 р. у цей готель перебуває в діапазоні [200; 400].
Відсортувати результат за доходом за зростанням, за номером
готелю за спаданням. Нагадування: функція для обчислення
різниці дат DATEDIFF (datepart , startdate , enddate)

Максимальна оцінка за завдання: 3 бали/балів

Відповідь студента: SELECT Booking.hotelNo, SUM(Price *
DATEDIFF(day, dateB, dateE)) AS TotalSum
FROM Booking INNER JOIN Room ON Booking.hotelNo =
Room.hotelNo AND Booking.roomNo = Room.roomNo
WHERE Booking.hotelNo NOT IN (SELECT hotelNo FROM Hotel
WHERE hotelCity = 'Київ')
AND YEAR(dateB) = 2022
AND Booking.hotelNo IN (
SELECT hotelNo
FROM Booking
WHERE YEAR(dateB) = 2022
GROUP BY hotelNo
HAVING COUNT(*) BETWEEN 200 AND 400
)
GROUP BY Booking.hotelNo
ORDER BY TotalSum ASC, Booking.hotelNo DESC;
Свою відповідь надай у форматі JSON, який буде виглядати так
{
"Mark": твоя оцінка відповіді у форматі double,
"Comment": твій коментар на рахунок відповіді студента
}

```

Рисунок 2.14 – Приклад промпту

"Mark": 2.7,

"Comment": "Відповідь студента загалом правильна, але є кілька незначних помилок. Запит правильно підраховує дохід кожного готелю в 2022 році, виключаючи готелі з Києва. Проте, є одна проблема: у WHERE умові `YEAR(dateB) = 2022` не враховує випадки, коли заселення могло початися в 2021 році, а виїзд припав на 2022 рік. Також варто зазначити, що `DATEDIFF(day, dateB, dateE)` можливо не коректно обчислює дні для всіх можливих сценаріїв. Правильніше було б враховувати тільки дати, які починаються в 2022 році, як вказано в умові.

Рисунок 2.15 – Відповідь моделі на запит з рисунку 2.14

### **3 РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ТЕСТУВАННЯ ЗНАНЬ МОВИ SQL З ВИКОРИСТАННЯМ GPT МОДЕЛІ ДЛЯ АВТОМАТИЧНОГО ОЦІНЮВАННЯ ВІДПОВІДЕЙ**

#### **3.1 Вибір та налаштування програмного середовища**

У постійно мінливому ландшафті веброзробки поєднання надійних бекенд-фреймворків з динамічними фронтенд-технологіями стало основою для створення потужних і масштабованих додатків [19]. Серед динамічних дуетів у цій сфері поєднання .NET та Angular виділяється як переконливий вибір для розробників [20, 21], які прагнуть отримати безперервний, ефективний та багатофункціональний досвід розробки додатків. Від спрощених робочих процесів розробки до покращеного користувацького досвіду, .NET та Angular приносять безліч переваг. Однак, як і в будь-якому технологічному процесі, проблеми та компроміси неминучі. Розуміючи плюси і мінуси цих фреймворків, розробники можуть приймати обґрунтовані рішення, гарантуючи, що обраний ними підхід бездоганно узгоджується з конкретними вимогами і цілями їхніх проєктів.

Тож розглянемо їх плюси та мінуси у поєднанні. Спершу представимо таку технологію для створення серверних застосунків як .NET, яка має наступні переваги:

- .NET підтримує декілька мов, таких як C#, F# та VB.NET, що дозволяє розробникам обирати мову, яка найкраще відповідає їхнім вподобанням та вимогам проєкту;
- екосистема .NET пропонує повний набір інструментів, бібліотек і фреймворків, які спрощують розробку, підвищуючи ефективність і узгодженість;
- з впровадженням .NET Core розробники можуть створювати крос-платформні додатки, забезпечуючи гнучкість розгортання на різних операційних системах;

– .NET відомий своєю масштабованістю та продуктивністю, що робить її чудовим вибором для створення додатків корпоративного рівня, які вимагають високої ефективності та швидкості реагування.

.NET займає лідируючі позиції у своїй категорії. Але він також має деякі недоліки, на які варто звернути увагу:

– проблема з витіком пам'яті через алокацію в замиканнях для стрілочних функцій;

– на жаль, оскільки .NET знаходиться під управлінням Microsoft, будь-які зміни або обмеження, які компанія може накласти, неминуче вплинуть на проєкти, виконані на цій технології. Це означає, що розробники матимуть менше контролю.

В другу чергу розглянемо фронтенд фреймворк Angular, до переваг якого можна віднести:

– двостороння прив'язка даних в Angular спрощує синхронізацію даних між моделлю та поданням, зменшуючи кількість шаблонного коду та підвищуючи продуктивність розробника;

– модульна архітектура Angular дозволяє створювати багаторазові компоненти, що полегшує підтримку та масштабування додатків;

– вбудована інжекція залежностей в Angular полегшує розробку слабо пов'язаних компонентів, що призводить до кращої тестованості та супроводжуваності;

– Angular має велику та активну спільноту, яка забезпечує широку підтримку, навчальні посібники та документацію для розробників, що полегшує усунення проблем при розробці та навчання.

Незважаючи на велику кількість переваг та популярність цього фреймворку у Angular є свої недоліки, які не як не роблять його постійним вибором для розробки фронтенд застосунків:

– широкий набір функцій та складні концепції Angular можуть бути складними для початківців, що вимагає значних витрат часу, щоб зрозуміти весь його потенціал;

– широкі можливості Angular можуть призвести до збільшення розміру пакета, що потенційно може вплинути на час початкового завантаження додатків, особливо на повільних мережах;

– Angular має швидкий цикл випуску з частими оновленнями, що може створювати проблеми для розробників та організацій з точки зору відстеження останніх змін.

.NET, і Angular пропонують потужні інструменти та можливості, вони можуть ефективно доповнювати один одного в розробці сучасних веб-застосунків, де Angular використовується для створення клієнтської частини, .NET – для серверної частини.

Таким, чином саме .NET та Angular було обрано для створення клієнт-серверного застосунку для тестування знань мови SQL.

Для налаштування програмного застосунку краще розгортати кожний сервіс на окремій машині і налаштовувати зв'язки з іншими сервісами за допомогою локальних мереж. Кожен з сервісів, окрім Verification Service, потребує встановлення .NET 8 та Postgres. Для налаштування Verification Service необхідно встановити інтерпретатор Python 3, Flask, DeepFace. Для розгортання клієнтської частини необхідно встановити Node.js та Angular CLI 15.

### 3.2 Ілюстрація роботи застосунка

Розглянемо роботу застосунку на основі створенні груп студентів, створенні тем та завдань, створенні тестів, проходження та перегляд результатів тесту.

### 3.2.1 Створення груп студентів

Для створення тесту та майбутнього його проходження необхідно створити групу для студентів та створити користувачів із роллю студент та закріпити цього користувача за створеною групою.

Створену групу під назвою «test group» зображено на рисунку 3.1.

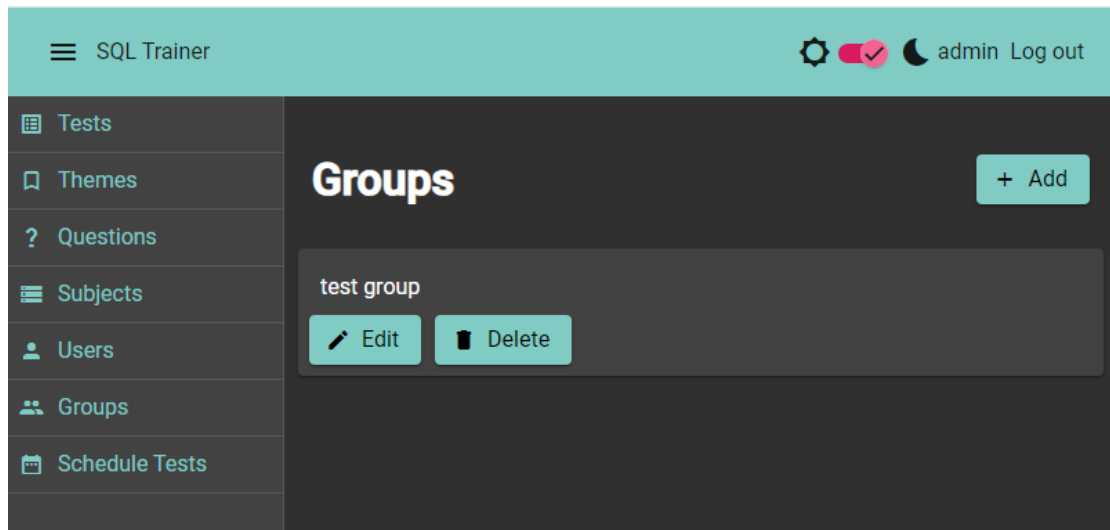


Рисунок 3.1 – Створена група «test group»

Створеного користувача з ім'ям «test student» та роллю Студент зображено на рисунку 3.2.

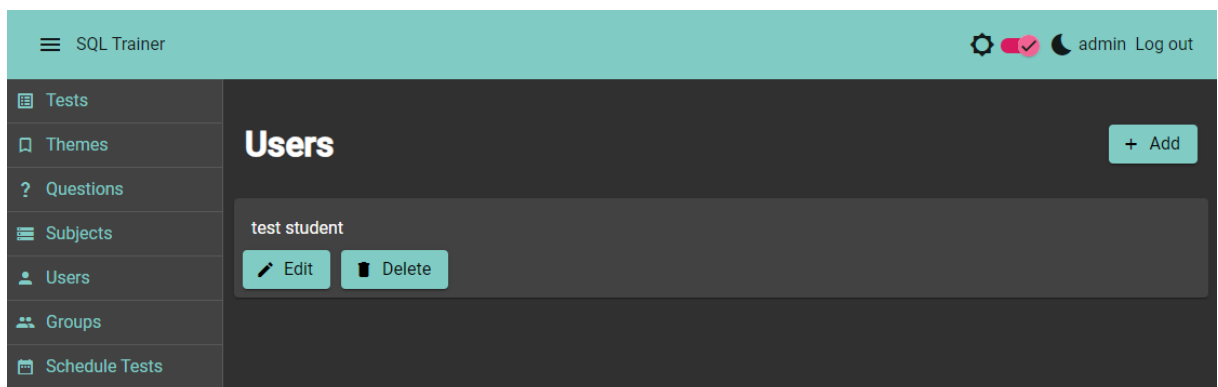


Рисунок 3.2 – Створений користувач з ім'ям «test student» та роллю Студент

### 3.2.2 Створення тем, завдань та тестів

Для створення тесту за розкладом необхідно створити питання для цього тесту та прив'язати питання до відповідної теми.

Створена тема під назвою «test theme» зображена на рисунку 3.3.

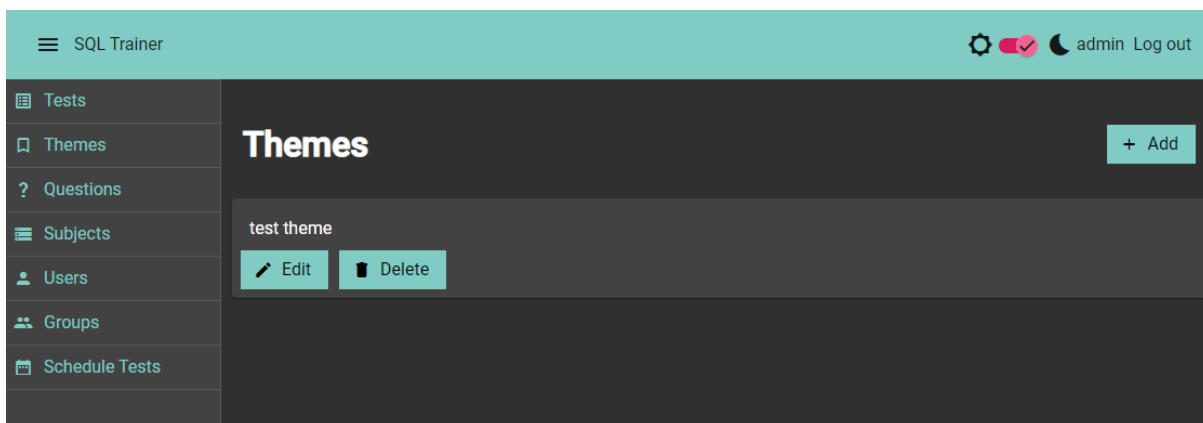


Рисунок 3.3 – Створена тема під назвою «test theme»

Створимо запитання та закріпимо їх за цієї темою. Для створення запитання необхідно створити базу даних, таблиці у цій базі даних та заповнити ці таблиці тестовими даними.

Створена база даних з назвою «test\_database» зображена на рисунку 3.4.

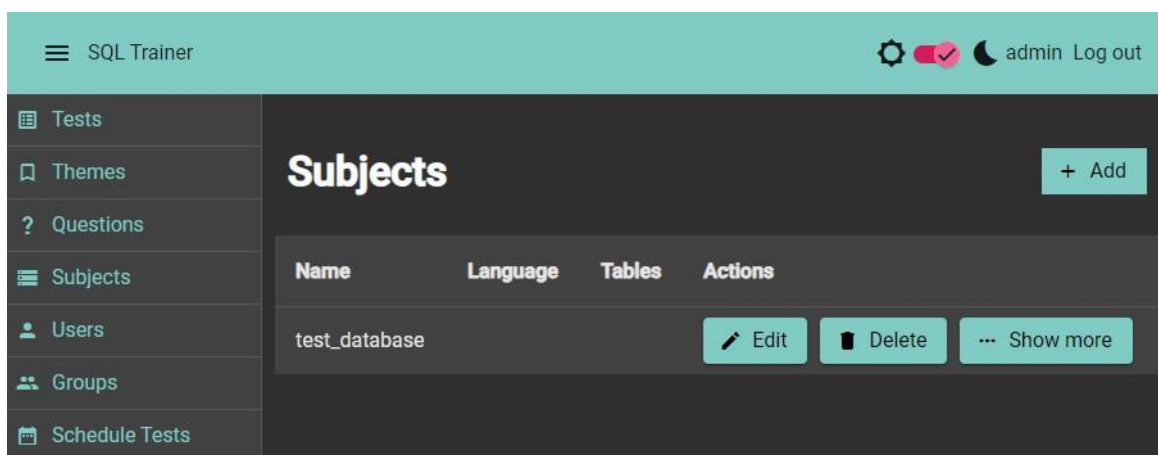


Рисунок 3.4 – Створена база даних з назвою «test\_database»

Створені таблиці для цієї бази даних зображені на рисунку 3.5.

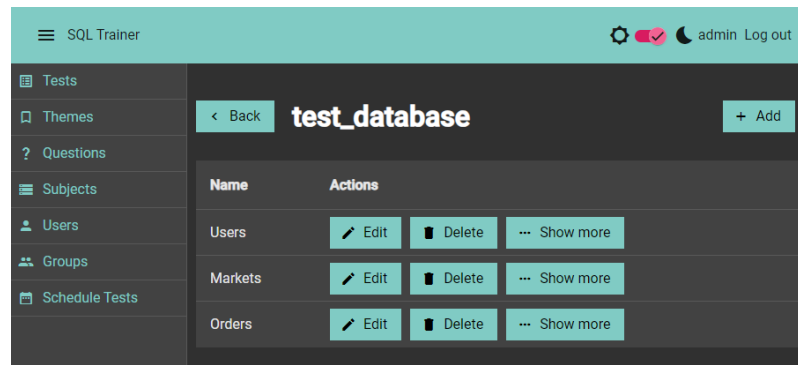


Рисунок 3.5 – Створені таблиці для бази даних

Таблиця «Users» (рис. 3.6) зберігає у собі інформацію про користувачів бази даних. Складається за полів Id (ідентифікатор), Name (ім'я користувача).



Рисунок 3.6 – Наповнення таблиці «Users»

Таблиця «Markets» (рис. 3.7) зберігає у собі інформацію про магазини бази даних. Складається з полів Id (ідентифікатор), Name (назва магазину).

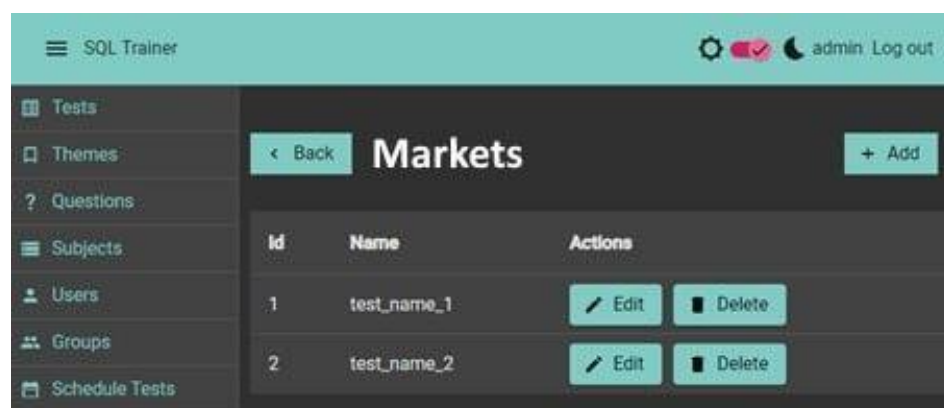


Рисунок 3.7 – Наповнення таблиці «Markets»

Таблиця «Orders» (рис. 3.8) зберігає у собі інформацію про замовлення бази даних. Складається з полів Id (ідентифікатор), UserId (ідентифікатор користувача, який зробив замовлення), OrderDate (дата створення замовлення).

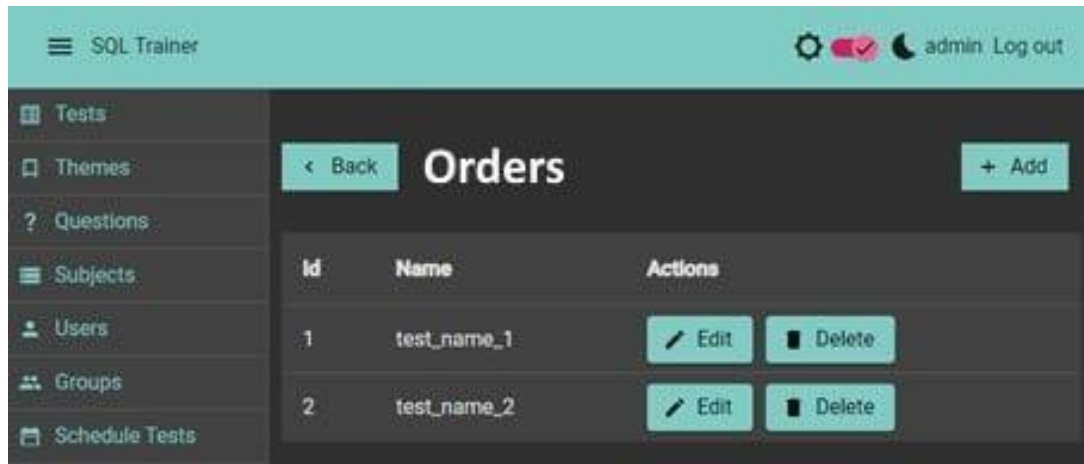


Рисунок 3.8 – Наповнення таблиці «Orders»

Після створення бази даних, таблиць та заповнення таблиць даними, перейдемо до створення запитань. Створені запитання зображені на рисунку 3.9.

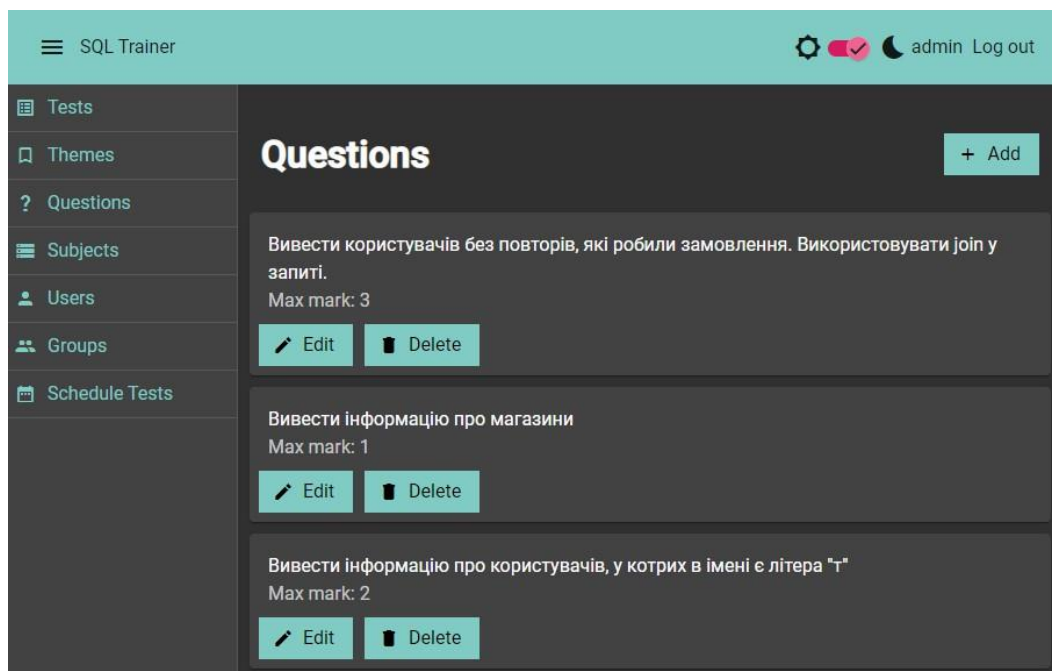


Рисунок 3.9 – Створені запитання

Після створення запитань можна створити тест за розкладом та назначити його початковий час та кінцевий і назначити цей тест групам студентів. Створений тест за розкладом зображено на рисунку 3.10.

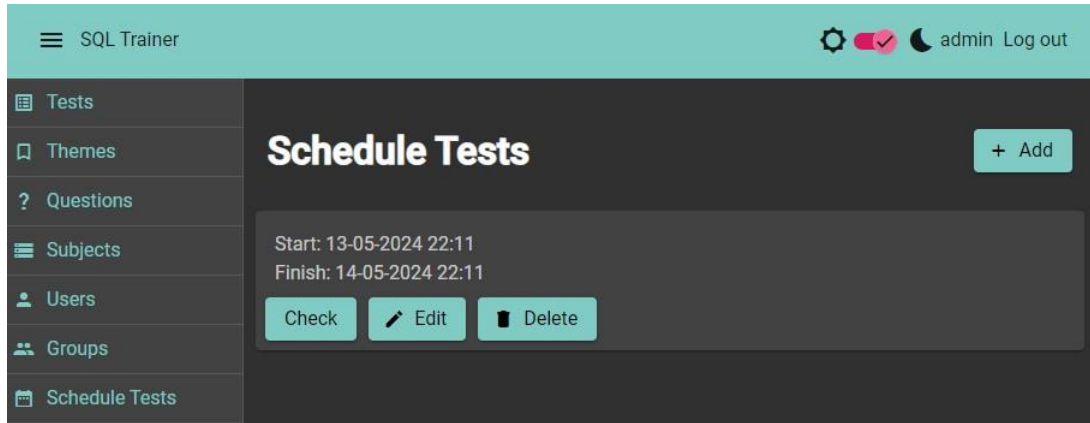


Рисунок 3.10 – Створений тест за розкладом

Тепер студенти, яким назначили цей тест можуть пройти його з 13 травня по 14 травня.

Для проходження цього тесту на найвищу оцінку користувачу необхідно правильно відповісти на наступні запитання:

- вивести інформацію про магазини;
- вивести інформацію про користувачів, у котрих в імені є літера «т»;
- вивести користувачів без повторів, які робили замовлення.

Використовувати join у запиті;

– вивести ім'я користувача, назву магазину та дату замовлення в цьому магазині;

– вивести кількість замовлень кожного користувача;

– вивести кількість замовлень магазину;

– вивести користувачів, котрі робили замовлення у магазині з ідентифікатором «1»;

– вивести магазини у яких робив замовлення користувач з ідентифікатором 2 (використовувати під-запит);

- вивести магазини у котрих не робилися замовлення (використовувати оператор «IN»);
- вивести користувачів, котрі ще не робили замовлення (використовувати оператор «EXISTS»);
- вивести всю інформацію про замовлення (вивести ім'я користувача, назву магазину та дату замовлення).

Правильні відповіді на питання, показані вище представленні лістингах з 3.1 по 3.11.

Лістинг 3.1 Правильна відповідь на питання «Вивести інформацію про магазини»:

```
select *
from Markets
```

Лістинг 3.2 Правильна відповідь на питання «Вивести інформацію про користувачів, у котрих в імені є літера «Т»»:

```
select *
from users
where name like 't%'
```

Лістинг 3.3 Правильна відповідь на питання «Вивести користувачів без повторів, які робили замовлення. Використовувати join у запиті»:

```
select distinct u.*
from users u join orders o on u.id = o.userid
```

Лістинг 3.4 Правильна відповідь на питання «Вивести ім'я користувача, назву магазину та дату замовлення в цьому магазині»:

```
select u.name, m.name, o.orderDate  
from users u join orders o on u.id = o.userId  
join markets m on o.marketId = m.id
```

Лістинг 3.5 Правильна відповідь на питання «Вивести кількість замовлень кожного користувача»:

```
select u.name, count(*)  
from users u join orders o on u.id = o.UserId  
group by u.Name
```

Лістинг 3.6 Правильна відповідь на питання «Вивести кількість замовлень магазину»:

```
select m.name, count(*)  
from Markets m join orders o on m.id = o.MarketId  
group by m.Name
```

Лістинг 3.7 Правильна відповідь на питання «Вивести користувачів, котрі робили замовлення у магазині з ідентифікатором «1»»:

```
select *  
from users  
where id in (select distinct userId  
from orders  
where MarketId = 1)
```

Лістинг 3.8 Правильна відповідь на питання «Вивести магазини у яких робив замовлення користувач з ідентифікатором 2 (використовувати підзапит)»:

```
select *  
from Markets  
where id in (select distinct MarketId  
from orders  
where UserId = 2)
```

Лістинг 3.9 Правильна відповідь на питання «Вивести магазини у котрих не робилися замовлення (використовувати оператор «IN»)»:

```
select *  
from Markets  
where id not in (select distinct MarketId  
from orders)
```

Лістинг 3.10 Правильна відповідь на питання «Вивести користувачів, котрі ще не робили замовлення (використовувати оператор «EXISTS»)»:

```
select *  
from Users  
where NOT EXISTS (select distinct UserId  
from orders)
```

Лістинг 3.11 Правильна відповідь на питання «Вивести всю інформацію про замовлення (вивести ім'я користувача, назву магазину та дату замовлення)»:

```
select u.name, m.name, o.orderDate  
from Users u join Orders o on u.Id = o.UserId  
join Markets m on m.id = o.MarketId
```

### 3.2.3 Проходження тестів та перегляд результатів

Перед проходження тесту за розкладом студенту необхідно пройти верифікацію у застосунку, для того щоб бути впевненим що саме цей студент проходить тест і ніхто інший. Логіка проходження верифікації побудована з використанням новітніх алгоритмів та підходів [22–37]. Проходження верифікації зображено на рисунку 3.11.

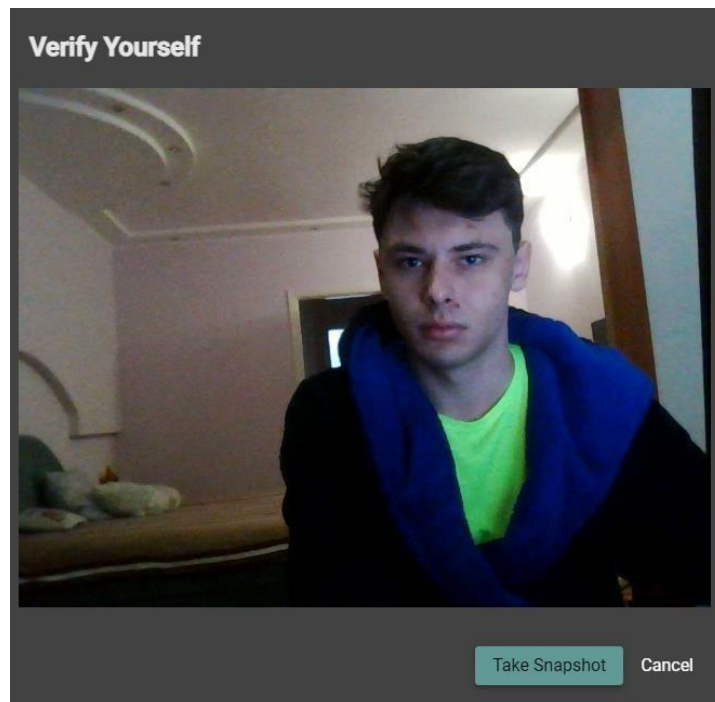


Рисунок 3.11 – проходження верифікації

Під час проходження тесту система періодично кожні 10 секунд проводить верифікацію обличчя студента. Якщо правила верифікації порушено тричі, відповіді студента фіксуються, і тест вважається невдалим, при цьому за кожне питання ставиться оцінка 0.

Під час проходження тесту користувач має навігаційну панель питань, для того щоб переключатись між запитаннями, структуру бази даних, для розуміння структури бази даних, кнопку «Execute» для того, щоб виконати запит, написаний користувачем, та бути впевненим, що запит відпрацьовує

правильно. При правильному відпрацюванні запиту користувач побачить дані, які повернув його запит. Екран проходження тесту зображено на рисунку 3.12.

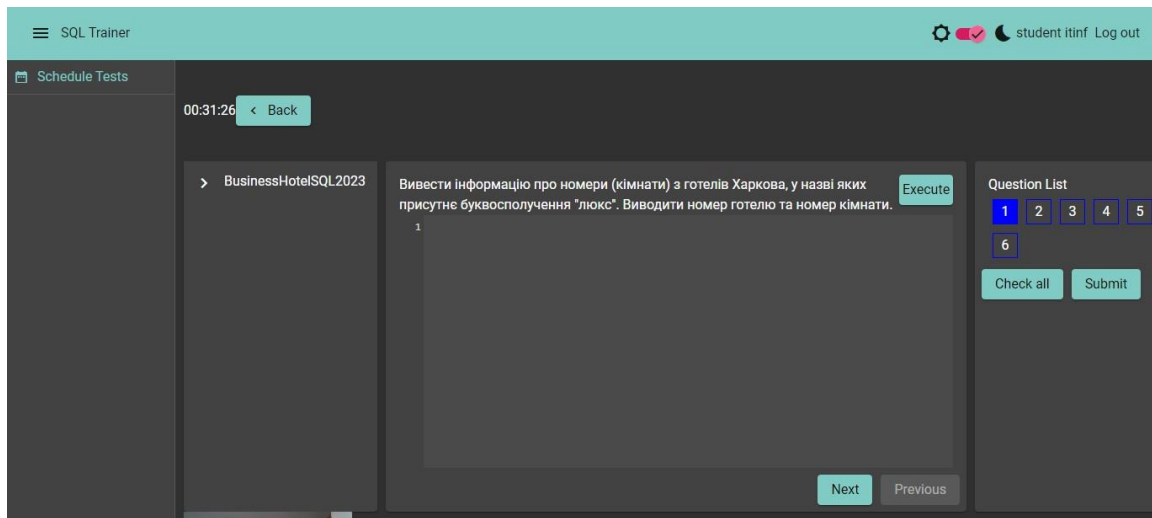


Рисунок 3.12 – Екран проходження тесту

Після проходження тесту користувач буде в змозі переглянути результати пройденого тесту, коли час для тесту закінчився. Користувач буде в змозі подивитися свою відповідь на запитання та оцінку з коментарями від. Екран перегляду результатів тесту зображено на рисунку 3.13.

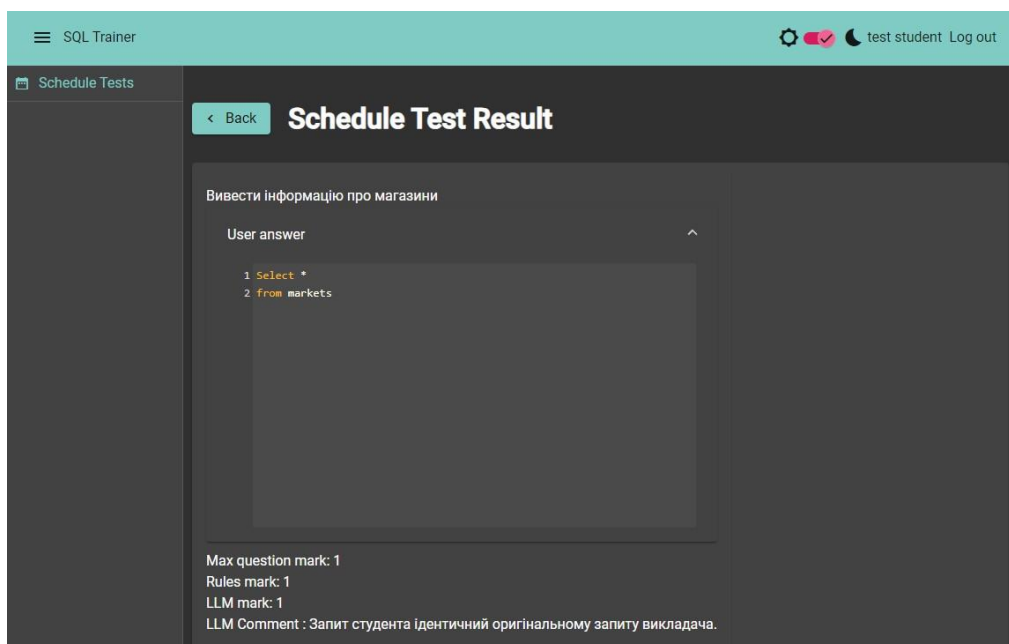


Рисунок 3.13 – Екран перегляду результатів тесту

### 3.3 Аналіз роботи алгоритму автоматичного оцінювання

Для того, щоб проаналізувати роботу алгоритму автоматичного оцінювання за допомогою GPT моделі, необхідно пройти створений тест з декількома неправильними відповідями, для того, щоб зрозуміти як GPT модель може аналізувати запитання та робити певні висновки при порівнянні відповіді студента та правильної відповіді, наведеної викладачем.

Перше питання «Вивести інформацію про магазини». У цьому питанні необхідно написати SQL-запит, котрий виведе всю інформацію про магазини. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.14.

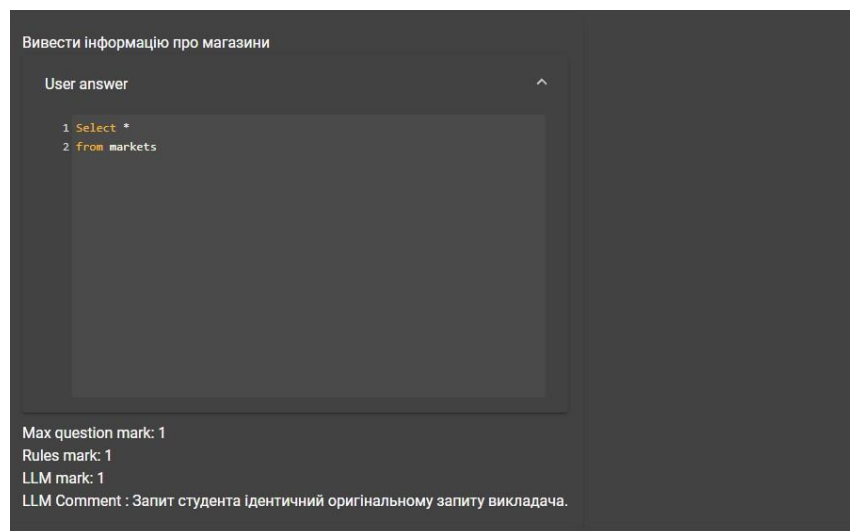


Рисунок 3.14 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на перше питання

Відповідь на питання ідентичне з правильною відповіддю, наданою викладачем, тому обидва алгоритми відпрацювали правильно та надали максимальну оцінку за цю відповідь.

Друге питання «Вивести інформацію про користувачів, у котрих в імені є літера «т»». У цьому питанні нам необхідно вивести лише тих користувачів, у котрих в імені наявна літера «т». Відповідь студента та оцінку від алгоритму

на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.15.

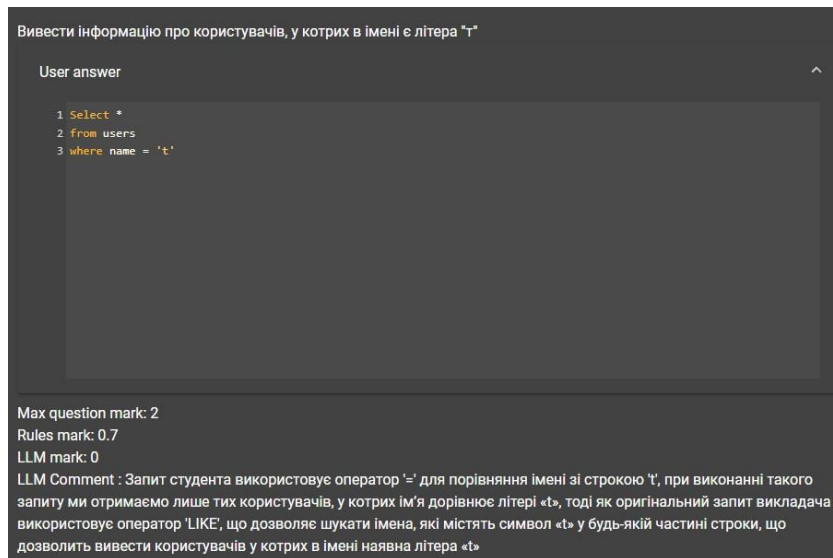
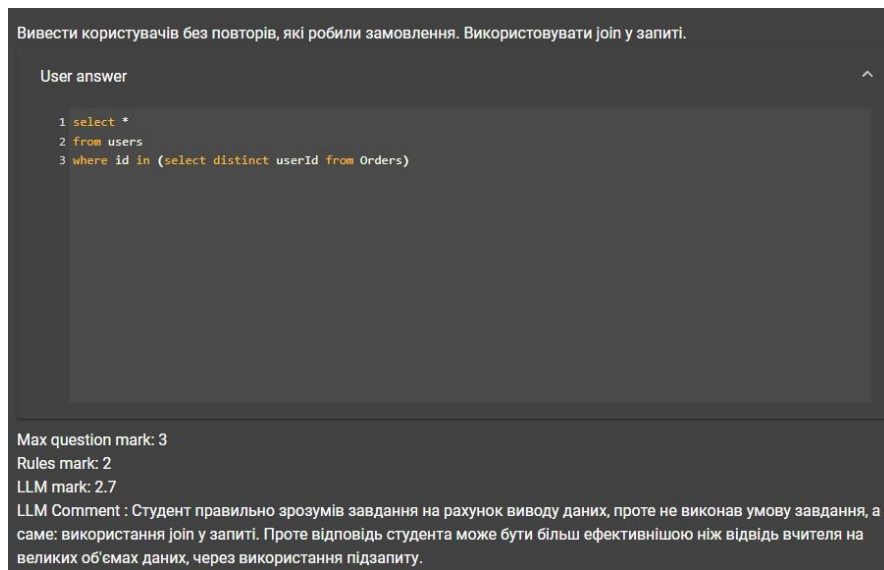


Рисунок 3.15 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на друге питання

Опираючись на відповідь студента, що зазначена вище, можна побачити, що алгоритм на основі евристичних правил поставив оцінку 0,7, хоча відповідь студента є невірною на це запитання. Це могло статись через те, що SQL-запит студента повернув правильну кількість атрибутів, проте не вивів правильну кількість рядків з бази даних, у той момент як оцінка від моделі GPT є 0, через те що відповідь студента є неправильною, про що говорить нам коментар від цієї моделі, а саме: «Запит студента використовує оператор «=» для порівняння імені зі строкою «t», при виконанні такого запиту ми отримаємо лише тих користувачів, у котрих ім'я дорівнює літері «t», тоді як оригінальний запит викладача використовує оператор «LIKE», що дозволяє шукати імена, які містять символ «t» у будь-якій частині строки, що дозволить вивести користувачів у котрих в імені наявна літера «t»». Спираючись на цей коментар ми бачимо, що мовна модель правильно проаналізувала завдання, та

правильно порівняла скрипт студента та еталонний скрипт викладача, та надала правильну оцінку та коментар на рахунок цієї відповіді.

Третє питання «Вивести користувачів без повторів, які робили замовлення. Використовувати join у запиті.». У цьому завданні необхідно вивести інформацію без повторів про користувачів, які робили замовлення. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.16.



```

Вивести користувачів без повторів, які робили замовлення. Використовувати join у запиті.

User answer

1 select *
2 from users
3 where id in (select distinct userId from Orders)

Max question mark: 3
Rules mark: 2
LLM mark: 2.7
LLM Comment : Студент правильно зрозумів завдання на рахунок виводу даних, проте не виконав умову завдання, а саме: використання join у запиті. Проте відповідь студента може бути більш ефективнішою ніж відповідь вчителя на великих об'ємах даних, через використання підзапиту.

```

Рисунок 3.16 – Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями на третє питання

Згідно з відповіді студента можна зрозуміти, що студент правильно зрозумів завдання, проте не дотримався всіх умов при написанні відповіді на це запитання. Відповідь студента дійсно повертає користувачів без повторів, які робили замовлення, та написана без помилок. Проте умовою було використовувати оператор «join» при написанні запиту, чого студент не дотримався. Обидва алгоритми надали студенту оцінку на його відповідь, проте алгоритм з використанням GPT моделі надав більшу оцінку, ніж алгоритм на основі евристичних правил і пояснив чому. Насправді відповідь студента буде більш ефективнішою ніж еталонний запит від викладача, тому

що студент використовує у своїй відповіді під-запит, що спрацює набагато швидше ніж запит з використанням оператора «join», тому що перша ітерація виконання SQL-запиту – це виконання блоку «from» і при використанні оператора «join» таблиці які вказані у блоку «from» будуть об'єднуватись між собою за вказаною умовою, що може призвести до довгого виконання запиту на великих об'ємах даних, коли як відповідь студента виконається набагато швидше, тому що базі не потрібно буде витратити ресурси на об'єднання таблиць між собою, що призведе до швидкого отримання необхідного набору даних. Про це нам і каже мовна модель GPT своїм коментарем на рахунок оцінки студенту на дану відповідь, що говорить про те що модель правильно проаналізувала завдання та має певні знання в оптимізації SQL-запитів, щоб пришвидшити вибірку даних з якого запиту.

Четверте питання «Вивести ім'я користувача, назву магазину та дату замовлення в цьому магазині». У цьому завданні необхідно вивести ім'я користувача, назву магазину та дату замовлення в цьому магазині. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.17.

```

Вивести ім'я користувача, назву магазину та дату замовлення в цьому магазині

User answer

1 select u.name, m.name, o.orderDate
2 from markets m join orders o on m.id = o.userId
3 join users u on o.marketId = u.id

Max question mark: 2
Rules mark: 0.75
LLM mark: 0
LLM Comment : У запиті студента помилково здійснено з'єднання таблиць markets та users по неправильному полю, що призводить до невірного результату. Необхідно змінити умову з'єднання на o.marketId = m.id, щоб запит був коректним.

```

Рисунок 3.17 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на четверте питання

Відповідь студента не є вірною, тому що студент допустився помилок у написанні умов з'єднання таблиць у операторі «join», проте запит студента має правильну кількість атрибутів, за що і отримав оцінку більше 0 від алгоритму оцінювання на основі евристичних правил, проте логіка запиту невірна, про що каже мовна модель GPT та надає правильну оцінку та коментарі на рахунок цього.

П'яте питання «Вивести кількість замовлень кожного користувача». У цьому запитанні необхідно вивести ім'я користувача та кількість його замовлень. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.18.

```

Вивести кількість замовлень кожного користувача

User answer

1 select u.name, count(o.Id)
2   from users u join orders o on u.id = o.UserId
3   group by u.Name

Max question mark: 2
Rules mark: 2
LLM mark: 2
LLM Comment : Обидва запити правильно виконують групування та підрахунок кількості замовлень кожного користувача. У запиті студента використано поле o.Id для підрахунку, що також правильно, але зайве, оскільки можна використовувати будь-яке поле, що мається в наявності.

```

Рисунок 3.18 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на п'яте питання

Обидва алгоритма відпрацювали правильно та надали вірну оцінку на рахунок запиту, незважаючи на те, що еталонний запит викладача трохи відрізняється від відповіді студента.

Шосте питання «Вивести кількість замовлень магазину». У цьому завданні необхідно вивести назву магазину та кількість його замовлень.

Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.19.

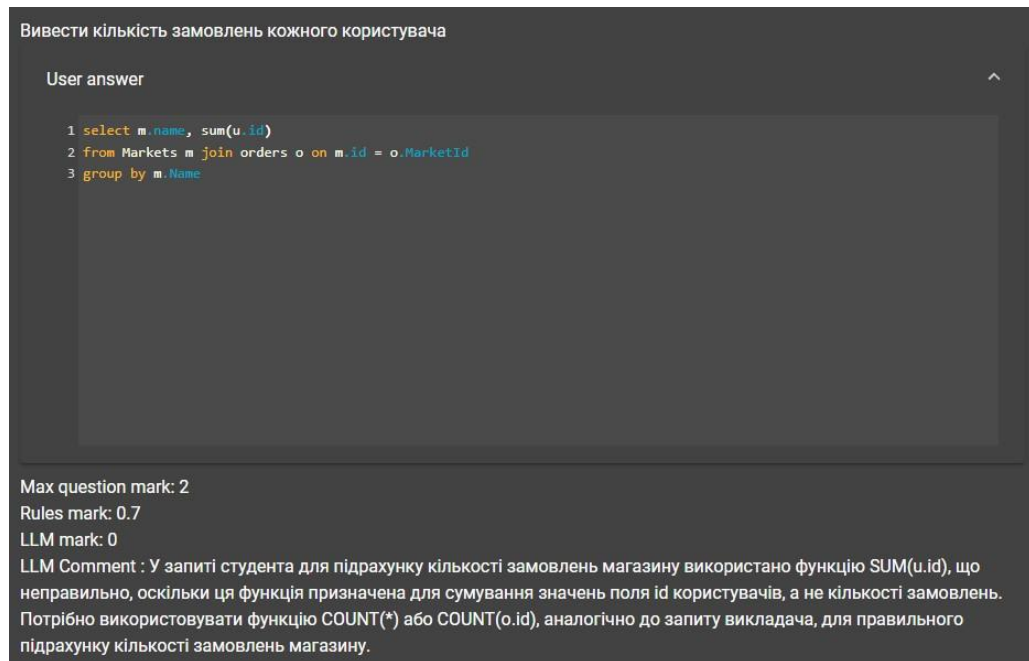


Рисунок 3.19 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на шосте питання

Відповідь студента не є правильною на це запитання, тому що як пояснює мовна модель він використовує неправильну функцію для рахування кількості, що призведе до неправильного вигляду кінцевої вибірки даних. Проте алгоритм на основі евристичних правил поставив студенту оцінку 0,7 за вірну кількість атрибутів, які повертає запит, проте запит не є вірним і оцінка від мовної моделі GPT є вірною та справедливою.

Сьоме питання «Вивести користувачів, котрі робили замовлення у магазині з ідентифікатором «1»». У цьому питанні необхідно вивести користувачів, котрі зробили замовлення у магазині з ідентифікатором «1» у будь-який проміжок часу. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.20.

```

Вивести користувачів, котрі робили замовлення у магазині з ідентифікатором «1»

User answer
1 select u.*
2 from users u join orders o on u.id = o.userId
3 where o.userId = 1

Max question mark: 2
Rules mark: 0.7
LLM mark: 0
LLM Comment : У запиті студента, він порівнює ідентифікатор користувача з ідентифікатором магазину, що є помилково. Таким чином, цей запит не відображає коректний результат, оскільки використовує неправильну умову для фільтрації результатів.

```

Рисунок 3.20 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на сьоме питання

Відповідь студента є невірною, тому що він виконує не правильну умову у запиті за рахунок чого отримає неправильний результат у вибірці даних. Оцінка мовної моделі є справедливою та добре аргументованою на рахунок відповіді студента. Модель побачила помилку та розписала її у коментарях, що дає більше розуміння того, чого саме студент отримав оцінку 0. Коли від алгоритму на основі евристичних правил студент отримав оцінку 0,7, тому що дотримався правильної кількості атрибутів, проте запит студента є невірним та повертає неправильну вибірку даних, за що студент справедливо отримує оцінку 0.

Восьме питання «Вивести магазини у яких робив замовлення користувач з ідентифікатором «2» (використовувати під-запит)». У цьому питанні необхідно використовувати під-запит для пошуку необхідного користувача з ідентифікатором «2» та вивести всі дані про магазин у якому робились замовлення. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.21.

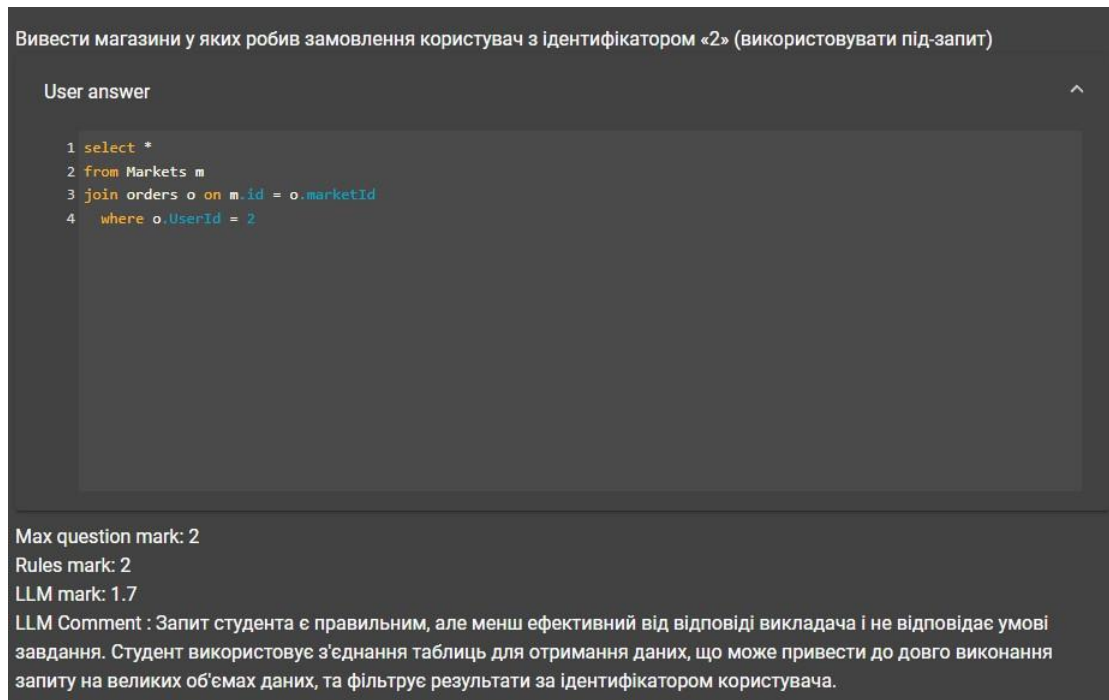


Рисунок 3.21 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на восьме питання

Відповідь студента з точки зору виконання запиту у базі даних є правильною, але не виконує умови завдання у якій сказано, що треба використовувати підзапит, про що зазначає мовна модель і ставить оцінку нижче від максимальної за недотримання умови завдання, у той момент як алгоритм на основі евристичних правил поставив відповіді максимальну оцінку через те, що запит повернув правильну кількість атрибутів таблиці та повернув правильну вибірку даних як результат, проте запит не повністю виконує умови завдання.

Дев'яте питання «Вивести магазини у котрих не робилися замовлення (використовувати оператор «IN»)». У цьому питанні необхідно використовувати оператор «IN», щоб знайти магазини, в яких не робилося замовлення, для правильного виконання завдання. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.22.

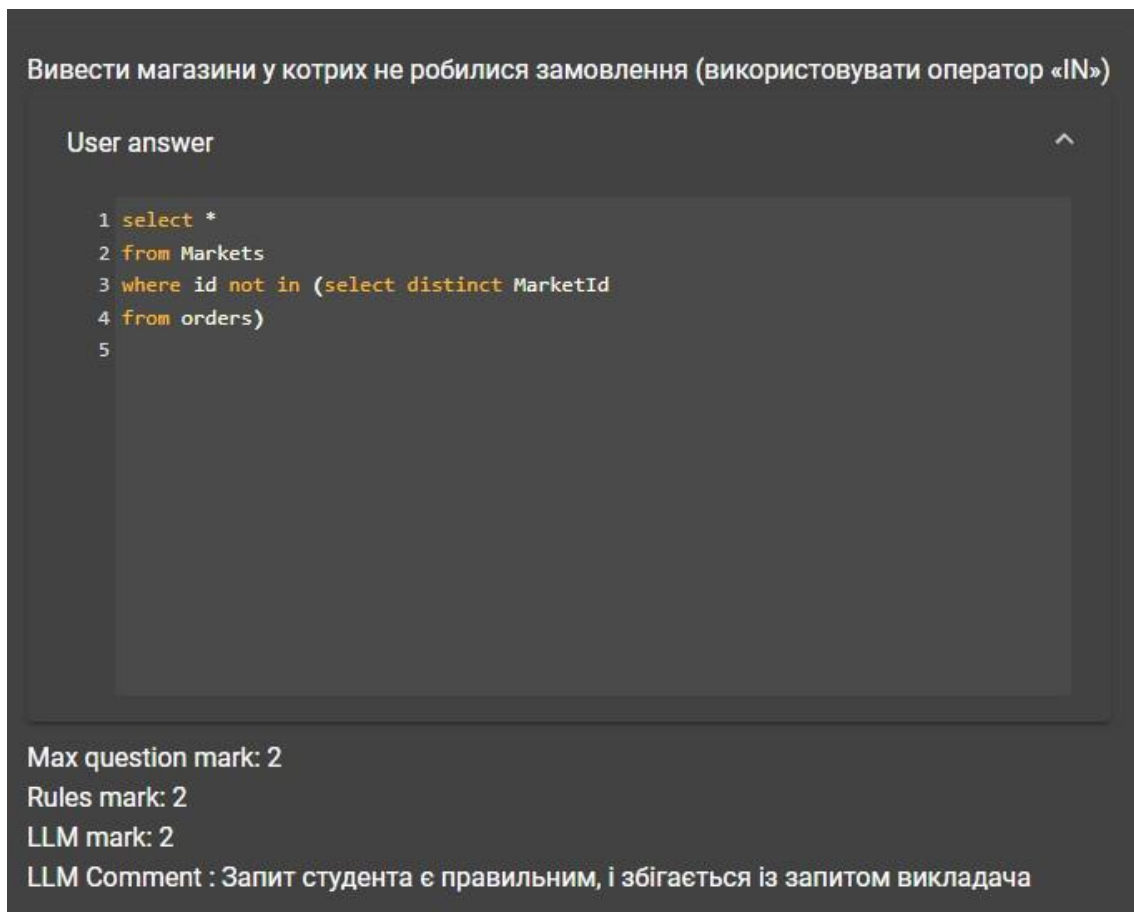


Рисунок 3.22 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на дев'яте питання

Відповідь студента є вірною та студент дотримався всіх вимог завдання. Обидва алгоритми оцінювання поставили правильну оцінку.

Десяте питання «Вивести користувачів, котрі ще не робили замовлення (використовувати оператор «EXISTS»)». У цьому питанні треба використовувати оператор «EXISTS» для того, щоб отримати інформацію про користувачів, котрі ще не робили замовлення. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.23.

Вивести користувачів, котрі ще не робили замовлення (використовувати оператор «EXISTS»)

User answer

```

1 select *
2 from Users
3 where id NOT in (select distinct UserId
4                 from orders)
5

```

Max question mark: 2  
Rules mark: 1.7  
LLM mark: 1.7  
LLM Comment : У запиті студента використовується оператор NOT IN, що порівнює ідентифікатори користувачів зі списком ідентифікаторів користувачів, які зробили замовлення. Запит виконаний правильно та поверне той самий результат, що і запит викладача, проте студент не дотримався всіх умов завдання. Запит викладача, який використовує оператор EXISTS, правильно перевіряє наявність записів у таблиці замовлень для кожного користувача, тому він відповідає умові завдання.

Рисунок 3.23 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на десяте питання

Відповідь студента є частково вірною, тому що запит студента поверне той самий результат, що і запит викладача, проте студент не дотримався всіх умов завдання, за що отримав оцінку 0,7 та справедливі коментарі від мовної моделі GPT. Алгоритм на основі евристичних правил поставив ідентичну оцінку як і мовна модель GPT, тому що запит студента повертає правильну кількість атрибутів і правильну кількість рядків у результаті.

Одинадцяте питання «Вивести всю інформацію про замовлення (вивести ім'я користувача, назву магазину та дату замовлення)». У цьому питанні необхідно написати запит, який зможе вивести ім'я користувача, назву магазину та дату замовлення. Відповідь студента та оцінку від алгоритму на основі евристичних правил та оцінку від мовної моделі GPT з коментарями зображено на рисунку 3.24.

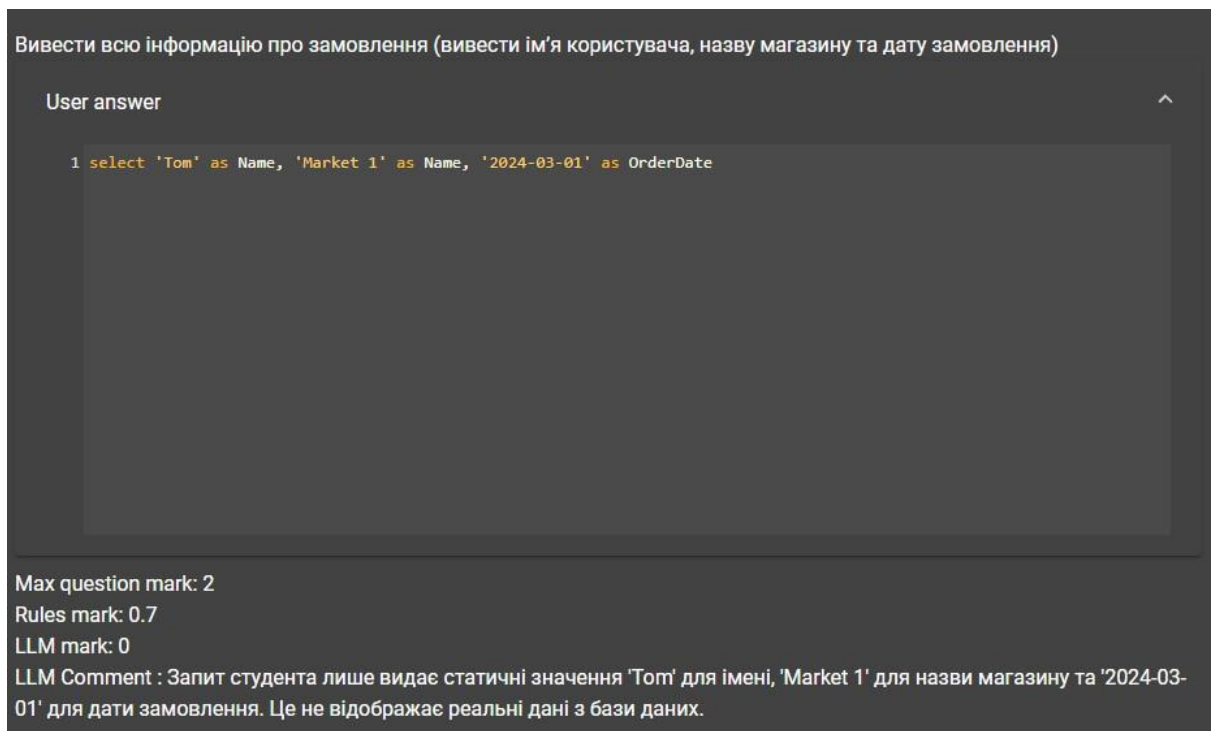


Рисунок 3.24 – Відповідь студента та оцінка від алгоритму на основі евристичних правил та оцінка від мовної моделі GPT з коментарями на одинадцяте питання

Відповідь студента повністю неправильна. Студент зробив імітацію запиту, який повертає один рядок даних та правильну кількість атрибутів. За рахунок правильної кількості атрибутів студент отримав оцінку 0,7 від алгоритму на основі евристичних правил, хоча запит не є вірною відповіддю на запитання та не відображає реальних даних з бази даних, про що свідчить коментар від мовної моделі GPT. Мовна модель правильно проаналізувала відповідь студента та зрозуміла, що відповідь студента видає статичні дані, які не є фактичними даними з бази даних, за що і поставила студенту справедливую оцінку 0.

Для більш глибокого дослідження та майбутнього аналізу роботи алгоритмів було проведено тестування на 100 питань між 11 студентами (всього 1100 відповідей). В таблиці 3.1 наведено результати алгоритмів для перших 11 відповідей.

Таблиця 3.1 – Результати роботи алгоритмів для перших 11 відповідей

Номер питання	Оцінка від алгоритму на основі евристичних правил, $X_{\text{оцін\_евр}}$	Оцінка від мовної моделі GPT, $X_{\text{оцін\_GPT}}$	Оцінки викладача, $X_{\text{еталон}}$	Максимальна оцінка за питання, $X_{\text{max}}$
1	1	1	1	1
2	0,7	0,4	0	2
3	2	2,2	2,8	3
4	0,75	0,4	0	2
5	2	2	2	2
6	0,4	0,15	0	2
7	0,7	0	0	2
8	2	1,4	1,9	2
9	2	2	2	2
10	1,7	1,7	1,8	2
11	0,7	1,7	0	2

Для визначення якості оцінювання відповідей студента розглянутими підходами в роботі запропоновано використати суму нормованих модулів різниці відхилення отриманої оцінки від оцінки вчителя (еталонної оцінки):

$$\rho(X_{\text{оцін}}, X_{\text{етал}}) = \frac{1}{N} \sum_{i=1}^N \frac{|X_{\text{оцін}_i} - X_{\text{еталон}_i}|}{X_{\text{max}_i}}, \quad (3.1)$$

де  $X_{\text{еталон}_i}$  – еталонна оцінка, з якою йде порівняння;

$X_{\text{оцін}_i}$  – оцінка від алгоритму ( $X_{\text{оцін\_евр}}$  – оцінка за евристичними правилами,  $X_{\text{оцін\_GPT}}$  – оцінка, що отримана за допомогою моделі GPT);

$X_{max_i}$  – максимальна оцінка за відповідь на запитання;

$N$  – кількість відповідей.

Величина  $\rho(X_{оцін}, X_{етал})$  буде знаходитися в діапазоні між 0 та 1. Ідеальним оцінюванням буде вважатися ситуація, коли оцінювання збігається з оцінкою вчителя, і  $\rho(X_{оцін}, X_{етал})$  буде дорівнювати нулю.

В результаті підрахунку  $\rho$  для 1100 відповідей студентів отримано такі значення:  $\rho(X_{оцін_{евр}}, X_{етал}) = 0,27$  ,  $\rho(X_{оцін_{GPT}}, X_{етал}) = 0,18$ . Також було підраховано відхилення значень оцінок між запропонованими підходами  $\rho(X_{оцін_{GPT}}, X_{оцін_{евр}}) = 0,21$ .

Також було визначено відсоток викидів  $C_{outlier}(X_{оцін}, X_{етал})$  значень оцінок запропонованими підходами від оцінки вчителя. Під викидами розуміються випадки коли оцінки розрізняються більш ніж на 20%. Підрахунки показали, що підхід отримання оцінки на базі евристичних правил має більше викидів, а саме  $C_{outlier}(X_{оцін_{евр}}, X_{етал}) = 11\%$ , коли як підхід на основі GPT моделі мав менш викидів  $C_{outlier}(X_{оцін_{GPT}}, X_{етал}) = 6\%$ .

Тобто за двома показниками  $\rho$ ,  $C_{outlier}$  підхід на базі GPT моделі мав перевагу перед підходом на основі евристичних правил. Отримані результати демонструють, що GPT модель забезпечує точніше та стабільніше оцінювання, що є важливим для підвищення якості автоматизованої оцінки знань. Успішне застосування цього алгоритму може сприяти покращенню процесу навчання і підвищенню об'єктивності оцінювання. Крім того, зниження відсотка викидів свідчить про більш узгоджені результати, що є ключовим аспектом для забезпечення справедливості та точності у процесі оцінювання

Але цей підхід все ж має відхилення від оцінки викладача та не може гарантувати, що система оцінить відповіді так само як викладач. Тому, щоб підвищити якість оцінювання було запропоновано алгоритм оцінювання, яких спирається на результати одночасно двох підходів (підпункт 2.5.3). Якщо їх результати не відрізняються більш ніж на заданий поріг, то як фінальна оцінка

та коментар береться оцінка від GPT моделі, оскільки вона має кращі значення показників  $\rho$ ,  $C_{outlier}$ . Якщо б поріг перевищений, то відповідь позначається поміткою про низку впевненість.

Також, як показано в підрозділі 1.5, використання GPT моделі через API є платним. Експерименти з 1100 відповідями та моделлю GPT 3.5turbo показали, що оцінювання коштувало 0,47\$.

В середньому на оцінювання 1 відповіді запропонований алгоритм витрачав 2,43 сек., де алгоритм на основі евристичних правил у середньому відпрацював за 0,27 сек., а алгоритм на базі GPT – 2,4 сек. Тобто алгоритм на базі GPT повільніший приблизно у 10 разів.

У майбутньому, запропонований підхід може бути адаптований для ще більшого зменшення відсотка викидень, що призведе до того, що процес автоматичного оцінювання стане подібний до звичайного оцінювання викладачем. Цього було б можливо досягти одночасним використанням декількох GPT моделей, та усередненням їх результатів, але це підвищить вартість та час оцінювання. Такий факт говорить, що питання підвищення якості із зберіганням невеликої вартості та часу потребує подальших досліджень.

## ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений вебзастосунок для моніторингу знань з мови SQL з використанням LLM у рамках запропонованого методу автоматичного оцінювання. Для досягнення цієї мети були розглянуті різні LLM, які могли б бути використанні для автоматичного оцінювання відповідей студентів, та способи їх використання у мікросервісній архітектурі, і серед великої кількості мовних моделей була обрано мовна модель GPT від OpenAI. Такий вибір був обумовлений тим, що мовна модель гарно показала та зарекомендувала себе у різних сферах життя та бізнесу, автоматизуючи різні процеси для пришвидшення роботи та надання відповідей та результатів, які найближче збігаються з людськими.

Виконано усі поставлені задачі, а саме:

- розглянуто сучасний стан цього питання розробки вебзастосунку для моніторингу знань студентів з використанням LLM моделі для методу автоматичного оцінювання, було розглянуто сервіси, які допомагають організувати учбовий процес, сервіси, які надають задачі для вирішення та сервіси, які використовують штучний інтелект;
- розглянуто сучасні застосунки для вивчення мови SQL;
- ознайомлено з різними LLM моделями та підібрано найбільш зручну мовну модель для вирішення цього питання;
- навчилися генерувати запити до LLM моделі для отримання найбільш бажаного оцінювання, яке є подібним до оцінювання викладача, щодо відповіді студента;
- спроектовано архітектуру вебзастосунку;
- спроектовані бази даних для вебзастосунку;
- спроектовано модуль тестування знань з мови SQL для таких типів користувачів: адміністратор, вчитель, студент, який буде забезпечувати автоматичну перевірку відповідей за допомогою обраної мовної моделі;

- спроектувано модуль оцінювання відповідей студента для отримання оцінки та коментарів щодо помилок студента;
- розроблено зручний та простий у використанні інтерфейс користувача;
- реалізувано спроектовані модулі тестування та ідентифікації особи;
- проведено тестування розробленого застосунку й проведено оцінювання роботи кожного з модулів, та застосунку в цілому та визначено переваги та недоліки цього вебзастосунку.

Даний вебзастосунок необхідно використовувати для організації та проведення навчального процесу при викладанні дисциплін з мови SQL та автоматичного оцінювання знань студента. Алгоритм автоматичного оцінювання з використанням мовної моделі допомагає автоматизувати роботу викладача та надає справедливо оцінювання та коментарі на рахунок відповіді студента, які є наближені до реальних оцінок та коментарів від викладача, проте мовна модель не дуже добро показує себе при оцінюванні складних SQL-запитів, в яких використовуються великі під-запити, або велика кількість різних перевірок, що свідчить про можливі викидення від мовних моделі та несправедливе оцінювання. Через це у майбутньому планується провести більше досліджень та розробити каскадну модель на основі великих мовних моделей, для того, щоб одна модель могла виправити іншу у разі несправедливої оцінки, що у майбутньому зможе призвести до можливої повної автоматизації процесу оцінювання відповідей студенті.

Результати роботи апробовано у вигляді 2 тез доповідей під час 28-го Міжнародного молодіжного форуму «РАДІОЕЛЕКТРОНІКА ТА МОЛОДЬ У XXI СТОЛІТТІ» [38] та 15-ої Міжнародної науково-практичної конференції «FREE AND OPEN SOURCE SOFTWARE» [39].

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Coursera | Degrees, Certificates, & Free Online Courses. URL: <https://www.coursera.org/> (дата звернення 13.05.2024).
2. edX – безкоштовна інтернет платформа масових відкритих інтерактивних курсів. URL: <https://www.edx.org/> (дата звернення 13.05.2024).
3. Udeemy – бібліотека курсів. URL: <https://www.udemy.com/> (дата звернення 13.05.2024).
4. Codecademy – інтерактивна онлайн-платформа з навчання 7 мовам програмування. URL: <https://www.codecademy.com/> (дата звернення 13.05.2024).
5. Quizlet – це сервіс, який дозволяє легко запам'ятовувати будь-яку інформацію, яку можна подати у вигляді навчальних карток. URL: <https://quizlet.com/> (дата звернення 13.05.2024).
6. Duolingo – безкоштовний додаток для вивчення іноземних мов. URL: <https://uk.duolingo.com/> (дата звернення 13.05.2024).
7. Cerego – хмарна технологічна платформа адаптивного навчання. URL: <https://www.cerego.com/> (дата звернення 13.05.2024).
8. SMARTeacher – діє під назвою Prodigy Education, пропонує платформу Prodigy Math Game, що має на меті покращити навчання математики. URL: <https://www.prodigygame.com/> (дата звернення 13.05.2024).
9. What is a Relational Database (RDBMS)? URL: <https://www.oracle.com/database/what-is-a-relational-database/> (дата звернення 13.05.2024).
10. 10 popular database management systems (DBMS). URL: <https://www.stackscale.com/blog/popular-database-management-systems/> (дата звернення 13.05.2024).
11. LeetCode – сервіс, для вирішування задач різними мовами програмування. URL: <https://leetcode.com/> (дата звернення 13.05.2024).

12. CodeWars - Achieve mastery through coding practice and developer mentorship. URL: <https://www.codewars.com/> (дата звернення 13.05.2024).
13. What is a large language model (LLM)? URL: <https://www.cloudflare.com/learning/ai/what-is-large-language-model/> (дата звернення 13.05.2024).
14. Cherednichenko, O., Kanishcheva, O., Yakovleva, O., & Arkatov, D. (2020). Collection and Processing of a Medical Corpus in Ukrainian. In COLINS (pp. 272-282).
15. Cherednichenko, O., Vovk, M., Yanholenko, O., & Yakovleva, O. (2020). Towards the technology of employers' requirements collection development. In Integrated Computer Technologies in Mechanical Engineering: Synergetic Engineering (pp. 228-239). Cham: Springer International Publishing.
16. Yakovleva, O., Nebeský, L., & Kirichenko, A. (2023). Using the GPT models for responses based on custom content to develop neural consultant for university applicants. In V International Scientific and Practical Conference. Madrid, Spain (pp. 172-178).
17. Comparison of Models: Quality, Performance & Price Analysis. URL: <https://artificialanalysis.ai/models> (дата звернення 13.05.2024).
18. What web application development tools can you use to boost productivity and teamwork? URL: <https://www.linkedin.com/advice/1/what-web-application-development-tools-can-you> (дата звернення 13.05.2024).
19. Skeet, J. (2019). C# in Depth. Simon and Schuster.
20. Bampakos, A., & Deeleman, P. (2023). Learning Angular: A no-nonsense guide to building web applications with Angular 15. Packt Publishing Ltd.
21. Albahari, J. (2022). C# 10 in a Nutshell. " O'Reilly Media, Inc."
22. Gorokhovatskyi, V., Tvoroshenko, I., Yakovleva, O., Hudáková, M., & Gorokhovatskyi, O. (2024). Application a Committee of Kohonen Neural Networks to Training of Image Classifier Based on Description of Descriptors Set. *IEEE Access*.

23. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., & Al-Dhaifallah, M. (2021). Methods of classification of images on the basis of the values of statistical distributions for the composition of structural description components. *IEEE Access*, 9, 92964-92973.
24. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., & Zeghid, M. (2022). Tools for fast metric data search in structural methods for image classification. *IEEE Access*, 10, 124738-124746.
25. Lyashenko, V., Kobylin, O., & Selevko, O. (2020). Wavelet analysis and contrast modification in the study of cell structures images.
26. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень.
27. Gorokhovatskyi, V., Tvoroshenko, I., & Olena, Y. (2024). Transforming image descriptions as a set of descriptors to construct classification features.
28. Ibrahim Daradkeh, Y., Gorokhovatskyi, V., Tvoroshenko, I., & Al-Dhaifallah, M. (2022). Classification of Images Based on a System of Hierarchical Features. *Computers, Materials & Continua*, 72(1).
29. Lyashenko, V., Kobylin, O., Ryazantsev, O., Ryazantsev, I., Barbaruk, V., & Zhychenko, Y. (2020). General Ideology of Analysis Digital Medical Images in RGB Format.
30. Ibrahim, D. Y., Gorokhovatskyi, V., Tvoroshenko, I., & Zeghid, M. (2022). Cluster representation of the structural description of images for effective classification.
31. Гороховатський, В. О., & Творошенко, І. С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.
32. Гороховатський, В. О., & Творошенко, І. С. (2022). Аналіз багатовимірних даних за описом у формі множини компонент.
33. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., & Al-Dhaifallah, M. (2023). Statistical data analysis models for determining the relevance of structural image descriptions. *IEEE Access*, 11, 126938-126949.

34. Tvoroshenko, I., & Zarivchatskyi, R. (2020). Analysis of existing methods for searching object in the video stream, in Proc. VI Int. Sci. Practic. Conf. «About the problems of science and practice, tasks and ways to solve them», Milan, pp. 500-505.

35. Gorokhovatskyi, O., Peredrii, O., Gorokhovatskyi, V., Vlasenko, N. (2023) Explanation of CNN Image Classifiers with Hiding Parts. In: J. Benois-Pineau, R. Bourqui, D. Petkovic, G. Quenot (eds), Explainable Deep Learning Artificial Intelligence, pp. 125-146, Academic Press, 346 p.

36. Гороховатський В., Творошенко І., Сидоренко Д. (2021) Класифікація зображень із використанням кластерного подання, Міжн. наук. симпозиум «Інтелектуальні рішення-С». Обчислювальний інтелект. Теорія прийняття рішень (Вересень 29, 2021). Київ – Ужгород, С. 44-45.

37. Gadetska, S. V., Gorokhovatskyi, V. O., Stiahlyk, N. I., & Vlasenko, N. V. (2021). Statistical data analysis tools in image classification methods based on the description as a set of binary descriptors of key points. Radio Electronics, Computer Science, Control, (4), pp. 58-68.

38. Науменко В.В., Яковлева О.В. (2024). Розробка методу автоматичного оцінювання відповідей студентів з використанням GPT-моделей від OPENAI для веб-застосунку з тестування знань мови SQL. 28-ий міжнародний молодіжний форум «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ».

39. Naumenko, V., Shelest, V., & Yakovleva, O. (2024). COMBINATION OF .NET TECHNOLOGY AND ANGULAR FRAMEWORK TO DEVELOP APPLICATION FOR TESTING SQL LANGUAGE KNOWLEDGE. Матеріали XV-ої Міжнародної науково-практичної конференції «Free and Open Source Software», Харків, 13-14 лютого 2024 р. Харків: Харківський національний економічний університет імені Семена Кузнеця, 2024. 148 с., 63.