Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти перший (бакалаврський)

Веб-застосунок «AI Hub» для роботи зі штучним інтелектом

(тема)

Виконав:

здобувач 4 року навчання,

групи КІУКІ-21-6

Дмитро ХАБАТІЛІН
(власне ім'я, прізвище)

Спеціальність 123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'ютерна інженерія
(повна назва освітньої програми)

Керівник: доц. Наталія БОЛОГОВА
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти перший (бакалаврський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерна інженерія
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Хабатіліну Дмитру Олеговичу
(прізвище, ім’я, по батькові)

1. Тема роботи Веб-застосунок «AI Hub» для роботи зі штучним інтелектом

затверджена наказом по університету від “ 26 ” травня 2025 р. № 424 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 17 червня 2025 р.

3. Вхідні дані до роботи 1) тип застосунку: веб-застосунок; 2) мова програмування: TypeScript; 3) База даних та серверна частина: PocketBase;4) документація Next.js; 5) документація Next.US; 6) документація PocketBase.

4. Перелік питань, що потрібно опрацювати у роботі
1) аналіз проблеми та огляд існуючих рішень;
2) вибір технології розробки та інструментальних засобів;
3) розробка користувацької частини додатку;
4) розробка серверної частини додатку;
5) висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

Слайд-презентація – 16 слайдів

6. Консультанти розділів роботи (заповнюється 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

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Аналіз проблеми та огляд існуючих рішень	27.05.25-30.05.25	
2	Вибір технології розробки та інструменталь-	31.05.25-03.06.25	
	них засобів		
3	Розробка користувацької частини додатку	04.06.25-07.06.25	
4	Розробка серверної частини додатку	08.06.25-09.06.25	
5	Оформлення матеріалів кваліфікаційної	10.06.25-11.06.25	
	роботи		
6	Подання кваліфікаційної роботи керівникові та	12.06.25-13.06.25	
	її попередній захист		
7	Подання кваліфікаційної роботи на рецензу-	14.06.25-16.06.25	
	вання		

Дата видачі завдання " 26 " травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Наталія БОЛОГОВА
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 75 с., 15 рис., 2 дод., 16 джерел.

WEB-ДОДАТОК, ШТУЧНИЙ ІНТЕЛЕКТ, ІНТЕРФЕЙС КОРИСТУВАЧА, LLM, ГЕНЕРАЦІЯ ЗОБРАЖЕНЬ, NEXT.JS, POCKETBASE, OPENROUTER, OPENAI, DALL-E, ЧАТ-БОТ

Метою кваліфікаційної роботи є створення web-застосунку «AI Hub» для інтеграції мовних моделей (LLM) та моделей генерації зображень у єдиному інтерфейсі, що забезпечує зручну взаємодію з інструментами штучного інтелекту.

Метою вирішення задачі є використання сучасного стеку технологій, зокрема фреймворку Next.js, JavaScript, API OpenRouter та OpenAI, а також системи PocketBase для бекенд-функціональності.

У ході виконання кваліфікаційної роботи було розроблено повнофункціональний web-додаток, який дозволяє користувачам взаємодіяти з кількома мовними моделями та інструментами генерації зображень в єдиному середовищі.

Кодова база побудована на основі модульної архітектури з використанням серверного рендерингу, динамічної маршрутизації, REST API та WebSocket для обміну даними в реальному часі.

Програму реалізовано мовою JavaScript з використанням Next.js у середовищі Visual Studio Code. Додаток адаптований для роботи як на десктопних, так і на мобільних пристроях.

ABSTRACT

Bachelor’s thesis: 75 pages, 15 figures, 2 appendices, 16 sources.

WEB APPLICATION, ARTIFICIAL INTELLIGENCE, USER INTERFACE, LLM, IMAGE GENERATION, NEXT.JS, POCKETBASE, OPENROUTER, OPENAI, DALL-E, CHATBOT

The purpose of the qualification work is to create an AI Hub web application for integrating language models (LLM) and image generation models in a single interface that provides convenient interaction with artificial intelligence tools.

The goal of solving the problem is to use a modern technology stack, including the Next.js framework, JavaScript, OpenRouter and OpenAI APIs, as well as the PocketBase system for backend functionality.

In the course of the qualification work, a fully functional web application was developed that allows users to interact with several language models and image generation tools in a single environment.

The code base is based on a modular architecture using server-side rendering, dynamic routing, REST API and WebSocket for real-time data exchange.

The program is implemented in JavaScript using Next.js in the Visual Studio Code environment. The application is adapted to work on both desktop and mobile devices.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ 7
ВСТУП 8
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ АНАЛОГІВ 11
1.1 Стан розвитку інтегрованих ШІ-застосунків 11
1.2 Огляд сучасних платформ для роботи з ШІ 12
1.3 Порівняльна характеристика існуючих рішень 16
1.4 Переваги власного рішення «AI Hub» 17
2 АНАЛІЗ ТА ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗРОБКИ ЗАСТОСУНКУ 20
2.1 Клієнтська логіка 20
2.2 Серверна логіка 22
2.3 Зовнішні API 26
3 ОПИС ЗАСТОСУНКУ 29
4 РЕАЛІЗАЦІЯ ЗАСТОСУНКУ 39
4.1 Архітектура додатку та структура сторінок 39
4.2 Реалізація спілкування з мовними моделями 41
4.3 Реалізація генерації зображень через API та відображення в чаті 44
4.4 Реалізація збереження чатів у базу даних Pocketbase. 45
4.5 Робота з обліковими записами користувачів 48
ВИСНОВКИ 51
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 52
ДОДАТОК А Графічний матеріал кваліфікаційної роботи 54
ДОДАТОК Б Вихідний код 63
Б.1 Сторінка Chat 63
Б.2 Провайдери API 72
Б.3 Вебхук перевірки авторизації 73
Б.4 Сторінка Auth 73

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ШІ – штучний інтелект (англ. Artificial Intelligence)

LLM – велика мовна модель (англ. Large Language Model)

UI – користувацький інтерфейс (англ. User Interface)

UX – користувацький досвід (англ. User Experience)

API – інтерфейс прикладного програмування (англ. Application Programming Interface)

ВСТУП

Стрімкий розвиток інформаційних технологій та комп’ютерної інженерії значною мірою вплинув на всі сфери людської діяльності, зокрема на освіту, науку, бізнес, охорону здоров’я, мистецтво й цифрові медіа. Одним із найважливіших напрямів цього розвитку стало впровадження технологій штучного інтелекту (ШІ), які вже сьогодні здатні виконувати складні когнітивні завдання, що раніше вважалися винятково людською прерогативою. До таких завдань належать розпізнавання мови, генерація текстів, переклад, створення програмного коду, візуального контенту, аналітика, класифікація даних тощо.

Особливо швидко розвиваються мовні моделі великого масштабу (LLM, large language models) та моделі генерації зображень, які стали основою для багатьох сучасних застосунків. Вони дозволяють ефективно взаємодіяти з інформацією, автоматизувати процеси, створювати новий контент, генерувати креативні рішення та надавати персоналізовані відповіді на запити користувачів.

Утім, попри широкий функціонал окремих сервісів, на практиці користувачі стикаються з розпорошенням інструментів ШІ по різних платформах. Наприклад, для генерації тексту зазвичай використовується один сервіс або API, для створення зображень інший, а для аналітики чи обробки даних третій. Така фрагментованість знижує ефективність використання ШІ, створює зайві часові та когнітивні витрати, а також ускладнює можливість інтеграції результатів з різних систем.

У зв’язку з цим виникла потреба у єдиному універсальному середовищі, яке дозволяє працювати з різними типами ШІ-моделей в одному інтерфейсі, швидко перемикатися між ними, порівнювати їх результати, комбінувати генерацію тексту та візуального контенту та адаптувати роботу під потреби конкретного користувача.

Таким середовищем є web-застосунок «AI Hub» – багатофункціональний інструмент для роботи з штучним інтелектом, який дозволяє користувачеві отримати доступ до кількох мовних моделей (наприклад, GPT-3.5, GPT-4, Claude, Gemini тощо) та до моделей генерації зображень (таких як DALL·E 3, Midjourney, Stable Diffusion, Flux тощо) в єдиному чат-інтерфейсі. Важливо, що користувач може самостійно обирати модель, з якою працює, а також порівнювати результати генерації, створюючи тим самим персоналізований і ефективний досвід.

Розробка «AI Hub» ґрунтується на використанні сучасного технологічного стеку, що включає фреймворк Next.js, клієнтську бібліотеку NextUI для створення адаптивного інтерфейсу, а також PocketBase – легкий сервер з вбудованою базою даних та механізмами автентифікації. Для підключення до моделей ШІ застосовуються зовнішні API від OpenRouter, OpenAI та Fal.ai, що забезпечують взаємодію з моделями через єдиний шлюз.

Використання єдиного інтерфейсу для мовних і візуальних моделей суттєво покращує досвід користувача: зменшується час на перемикання між платформами, забезпечується кращий контроль над результатами генерації, з’являється можливість безпосереднього порівняння якості роботи різних моделей. Крім того, такий підхід спрощує знайомство з ШІ для новачків, оскільки усуває потребу у складному налаштуванні або знаннях API, а також дозволяє працювати з потужними моделями у доступному форматі. Збереження історії сесій, адаптивний дизайн та інтуїтивно зрозумілий інтерфейс роблять роботу з додатком зручною як на комп’ютері, так і на мобільних пристроях.

AI Hub дає змогу легко генерувати тексти, створювати зображення на основі текстових підказок, комбінувати результати в одному чаті, зберігати історію, а також розгортати застосунок у хмарному середовищі або на локальному сервері.

Додаток має простий і зрозумілий інтерфейс, що дозволяє використовувати його як досвідченим розробникам, так і пересічним

користувачам без спеціальної підготовки. Адаптивність інтерфейсу забезпечує повноцінну роботу як на стаціонарних пристроях, так і на мобільних.

Таким чином, актуальність теми кваліфікаційної роботи визначається сучасними запитами цифрового суспільства, розвитком ШІ-моделей, потребами в інтегрованих інструментах та запитом на гнучкі, ефективні та інтуїтивні платформи для генерації контенту.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ АНАЛОГІВ

1.1 Стан розвитку інтегрованих ШІ-застосунків

Упродовж останніх років штучний інтелект перетворився з теоретичної концепції на практичний інструмент, що активно впроваджується в цифрові продукти й сервіси. Особливу увагу привертають мовні моделі великого масштабу, які навчені на об’ємних корпусах текстів і здатні відповідати на запити природною мовою, виконувати переклад, здійснювати узагальнення, аналіз, а також створювати оригінальний контент [1]. Поряд із цим активно розвиваються моделі генерації зображень, які формують візуальні об’єкти на основі текстових описів та застосовуються в дизайнерських, рекламних і творчих задачах.

Попри високий рівень технічної зрілості таких моделей, користувачі часто змушені звертатися до окремих сервісів для виконання різнотипних завдань. Наприклад, текстова генерація здійснюється в одному застосунку, тоді як створення зображень потребує переходу в інший інструмент. Це ускладнює взаємодію, призводить до фрагментації робочого середовища та потребує додаткових дій з боку користувача для об'єднання результатів.

У сучасному інформаційному середовищі зростає попит на універсальні рішення, що дозволяють взаємодіяти з різними моделями в межах одного інтерфейсу. Такий підхід значно оптимізує робочі процеси, скорочує витрати часу, підвищує ефективність генерації та спрощує порівняння результатів між різними системами штучного інтелекту. Окрім цього, створюються умови для реалізації персоналізованого підходу, коли користувач самостійно обирає модель, яка найбільше відповідає поточній задачі.

Ідея об’єднання функціоналу кількох моделей у межах одного програмного рішення активно розвивається, набуваючи популярності як

серед розробників, так і серед кінцевих користувачів. Інтегровані застосунки, що поєднують мовні моделі та системи генерації зображень, поступово формують новий стандарт у проєктуванні інтерфейсів для взаємодії з ШІ. Такий підхід не лише розширює функціональність програмних продуктів, а й робить їх більш доступними, зручними та гнучкими в експлуатації.

1.2 Огляд сучасних платформ для роботи з ШІ

Штучний інтелект став основою численних цифрових сервісів, які надають доступ до мовних моделей, генераторів зображень та інструментів обробки даних. Існує велика кількість платформ, що надають користувачам можливість взаємодіяти з потужними LLM-моделями та генеративними системами, однак більшість із них зосереджуються на обмеженому функціоналі [2]. У цьому підрозділі розглянуто найбільш поширені рішення, що демонструють сучасний стан ринку ШІ-застосунків.

Одним із найпопулярніших інструментів є ChatGPT від компанії OpenAI. Ця платформа забезпечує доступ до мовних моделей GPT-3.5 та GPT-4, які демонструють високу якість генерації тексту, здатність до діалогу, вирішення логічних задач, програмування та обробки інформації. ChatGPT має інтуїтивно зрозумілий інтерфейс та підтримує інтеграцію через API. Водночас функціональність сервісу обмежується переважно роботою з текстом, генерація зображень реалізована лише як окрема функція для платних користувачів, і не передбачає широкої кастомізації параметрів.

Неможливість самостійно обрати модель для обробки запиту, обмеження у довжині контексту, відсутність прямого управління prompt-інженерією та недостатня прозорість у відображенні механізмів генерації створюють бар’єри для просунутого або технічного використання платформи спілкування зі ШІ ChatGPT (рисунок 1.1).

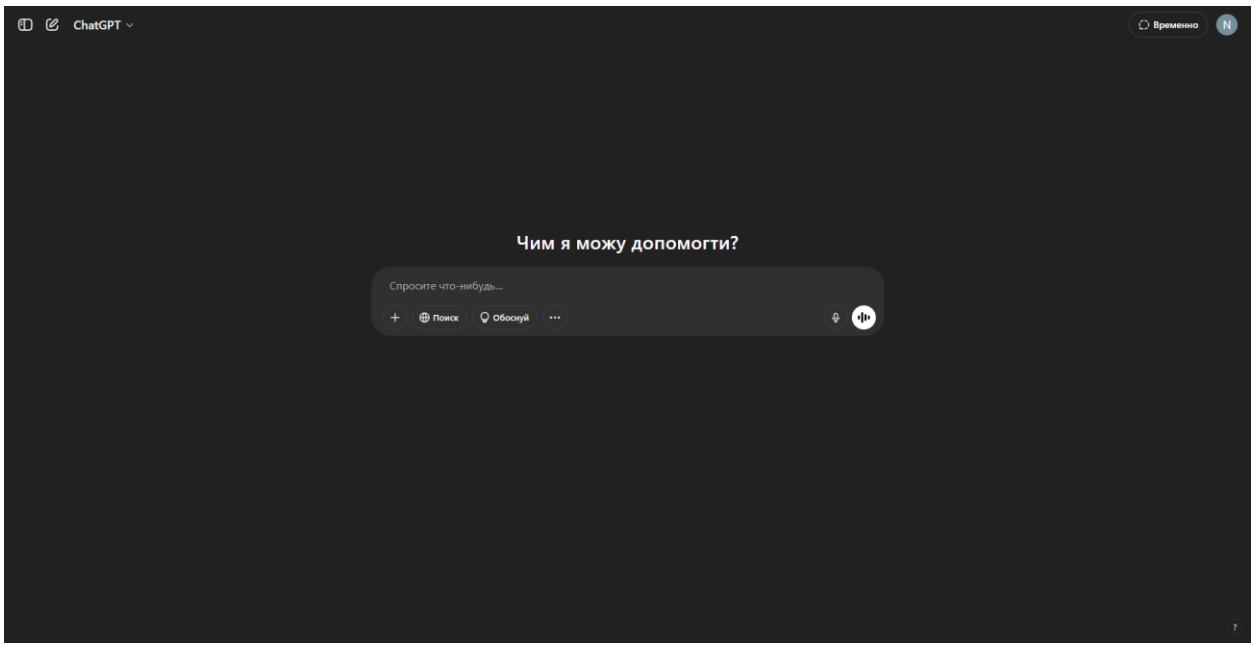

Рисунок 1.1 – Інтерфейс чату з ШІ ChatGPT

Таким чином, попри широкі можливості для звичайного користувача, ChatGPT у своїй базовій формі є скоріше універсальним «асистентом», ніж повноцінною платформою для дослідження та порівняння роботи моделей, що знижує його придатність у контексті завдань, де важливо мати контроль над інструментом і його налаштуваннями.

Інша популярна платформа – DeepAI, яка пропонує базовий функціонал як для мовної, так і для візуальної генерації. Сервіс підтримує кілька моделей, однак має обмежену якість виводу, вузький набір параметрів і не дозволяє користувачеві самостійно обирати модель для конкретного запиту. Перевагою DeepAI є простота використання, проте ця платформа підходить радше для ознайомлення з технологіями ШІ, аніж для професійного використання.

У галузі генерації зображень особливу увагу привертає Midjourney. Це система, яка функціонує через інтеграцію з месенджером Discord і дозволяє створювати високоякісні, стилізовані візуальні роботи. Система відзначається складною архітектурою генерації, підтримкою великої кількості стилів і здатністю формувати оригінальні композиції. Проте сервіс не підтримує текстову генерацію, а взаємодія з ним відбувається в непрямий

спосіб, що може бути незручним для частини користувачів. Крім того, для повноцінного використання необхідне оформлення платної підписки.

Окремо варто згадати Poe – інструмент, розроблений компанією Quora. Ця платформа орієнтована на доступ до різноманітних LLM-моделей, зокрема GPT-4, Claude, Llama та інших, в єдиному інтерфейсі. Poe забезпечує швидке перемикання між моделями та дозволяє зберігати історію чатів. Однак функціональність обмежена текстовими задачами: платформа не підтримує генерацію зображень і не дозволяє реалізовувати комплексну інтеграцію із зовнішніми сервісами (рисунок 1.2).

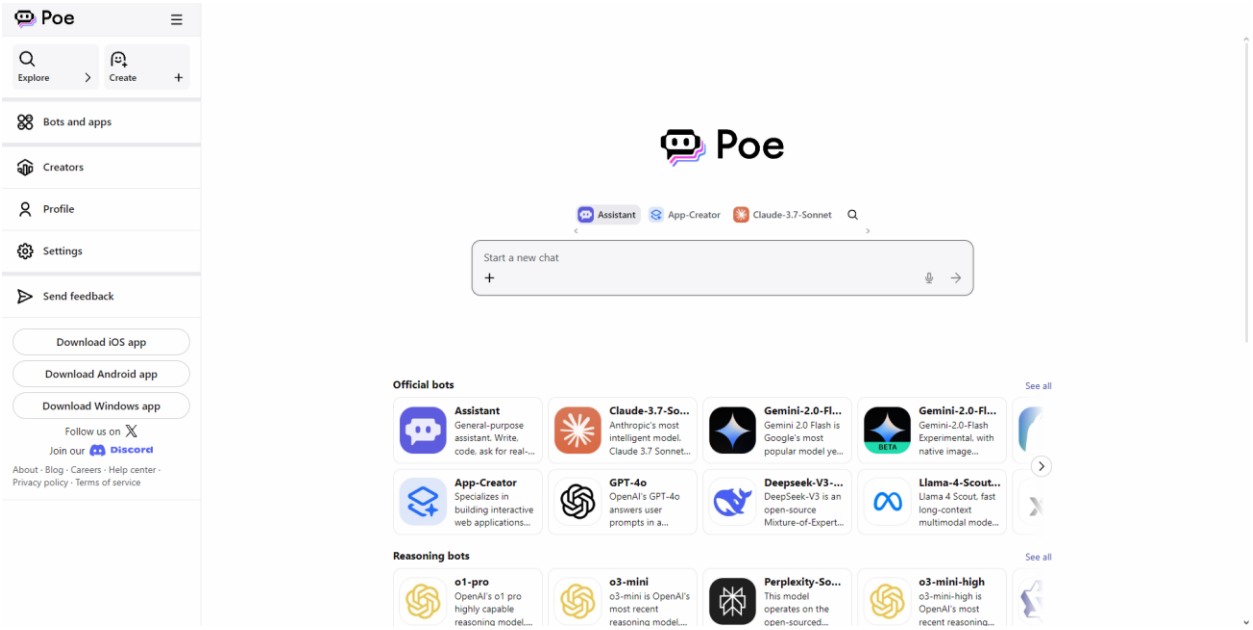

Рисунок 1.2 – Інтерфейс чату з ШІ Poe

Інструмент DALL·E від OpenAI, який спеціалізується на генерації зображень, також активно використовується в креативних сферах. Його головна перевага полягає у здатності інтерпретувати детальні текстові запити та формувати візуальний контент високої якості. Водночас модель не призначена для обробки тексту або коду, тому її використання можливе лише у вузькому сегменті задач (рисунок 1.3).

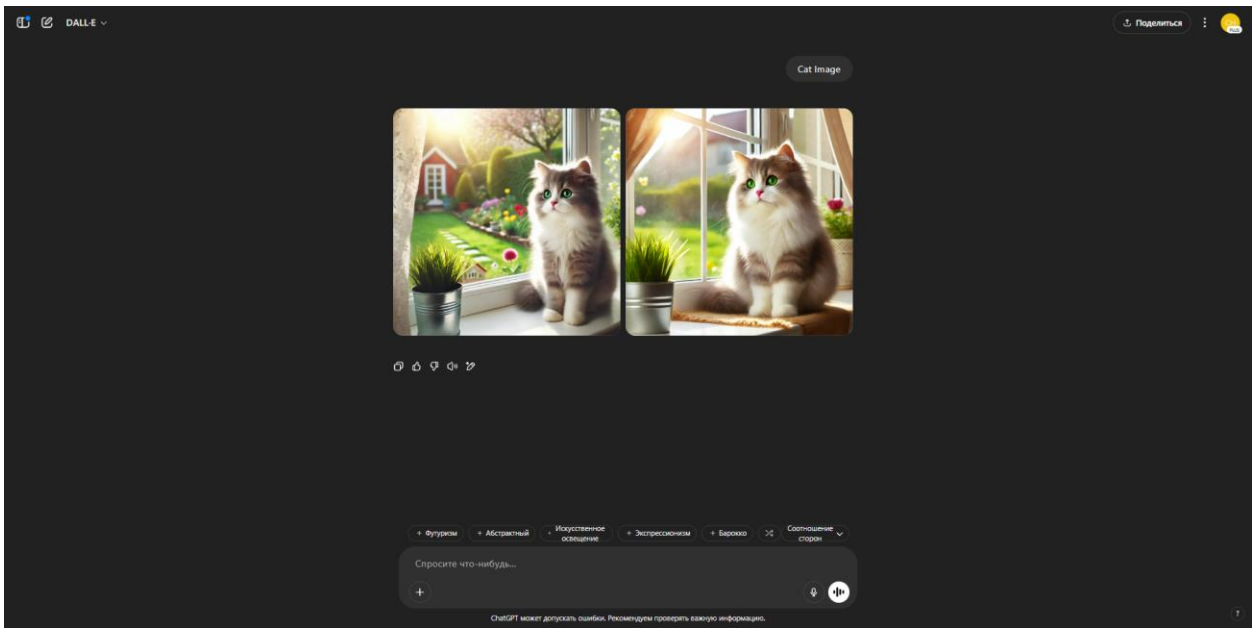

Рисунок 1.3 – Інтерфейс чату з ШІ Dall-E-3

DALL·E має низку обмежень, які суттєво знижують зручність її використання. Модель не має власного окремого інтерфейсу або вебзастосунку – її можна використовувати лише через інтеграцію в інші продукти OpenAI, зокрема ChatGPT, або за допомогою прямого доступу до API.

У вебінтерфейсі ChatGPT ця функція доступна виключно для користувачів із платною підпискою, а кастомізація параметрів генерації є мінімальною.

Використання моделі через API передбачає необхідність самостійної інтеграції, наявності стабільного інтернет-з’єднання, знання документації та обмеження за тарифами. Це створює певний бар’єр для пересічного користувача, який бажає швидко отримати візуальний результат без технічної підготовки.

Таким чином, більшість сучасних рішень мають або обмежений функціонал, або вузьку спеціалізацію. Вони не дозволяють користувачеві вільно комбінувати мовні й візуальні інструменти, що обмежує ефективність взаємодії та створює фрагментований досвід. Це формує передумови для актуальності платформ нового покоління, які поєднують різні типи моделей у

межах єдиного середовища.

1.3 Порівняльна характеристика існуючих рішень

Аналіз сучасних платформ для роботи з мовними моделями та генераторами зображень дозволяє зробити висновок, що кожна з них має свої сильні сторони, проте водночас супроводжується певними обмеженнями, які впливають на зручність та ефективність використання.

Платформи, орієнтовані виключно на текстову взаємодію, такі як ChatGPT або Poe, забезпечують високу якість генерації контенту, гнучке ведення діалогу, здатність до виконання складних аналітичних або програмістських задач. Вони демонструють стабільну роботу, мають добре продуманий інтерфейс і підтримують широкий спектр мов. Проте в них відсутня підтримка мультимодальності, що означає неможливість створення візуального контенту без використання зовнішніх рішень.

Інструменти для генерації зображень, як-от DALL·E чи Midjourney, вирізняються високою якістю візуального результату. Вони здатні інтерпретувати текстові запити та створювати зображення у заданих стилях і форматах. Проте такі сервіси зазвичай не підтримують текстову генерацію або не дозволяють гнучко поєднувати текстові й візуальні дані в одному сценарії. Крім того, доступ до більшості функцій часто вимагає оформлення платної підписки, що може бути бар’єром для нових або непрофесійних користувачів.

Загальною проблемою багатьох платформ є також відсутність можливості інтегрувати кілька моделей одночасно. У разі потреби порівняти результати генерації одного й того ж запиту різними моделями користувачеві доводиться перемикатися між окремими застосунками, копіювати текст, відстежувати відповіді вручну та здійснювати аналіз поза межами системи. Це призводить до втрати цілісності робочого процесу, ускладнює дослідження та значно знижує швидкість взаємодії.

Недоліком є обмежені можливості персоналізації та кастомізації. У більшості випадків користувач позбавлений гнучкого налаштування параметрів моделі, вибору формату виводу, адаптації інтерфейсу під власні потреби або об'єднання результатів у спільне середовище.

Також варто зазначити, що більшість платформ не підтримують одночасну роботу з кількома типами моделей – наприклад, текстовою та графічною – у межах одного інтерфейсу. Відсутність єдиного середовища, де можна генерувати текст і візуальний контент, знижує продуктивність і ускладнює створення цілісного цифрового продукту або документа. Для реалізації повноцінних мультимодальних задач користувачеві часто доводиться комбінувати кілька платформ вручну.

Ще одним суттєвим обмеженням є неможливість одночасної взаємодії з кількома мовними моделями в межах однієї сесії. У більшості платформ користувач може працювати лише з однією обраною LLM-моделлю за раз, без можливості перемикання або паралельного порівняння відповідей від інших моделей у тому ж чаті. Це унеможливлює безпосереднє порівняння стилю, точності або логіки генерації між різними ШІ-системами й потребує вручну дублювати запити в інших інтерфейсах.

Таким чином, сучасні сервіси здебільшого надають фрагментовані рішення, зосереджені на вузьких задачах, що обмежує їхнє використання у комплексних сценаріях. Усе це свідчить про наявність потреби у створенні платформи, яка поєднує в собі гнучкість, багатомодельність, мультимодальність та простоту у використанні, з можливістю персоналізованої взаємодії з різними типами штучного інтелекту.

1.4 Переваги власного рішення «AI Hub»

На тлі аналізу існуючих платформ стає очевидною необхідність створення єдиного, гнучкого й доступного інструменту для взаємодії з різними мовними та візуальними моделями штучного інтелекту. Саме таку

концепцію реалізовано у розробленому web-застосунку «AI Hub», який об’єднує в одному інтерфейсі роботу з LLM-моделями, генераторами зображень і механізмами збереження історії сесій користувача.

Однією з ключових переваг «AI Hub» є можливість вибору конкретної мовної моделі для генерації відповіді без необхідності перемикання між сервісами або вкладками. Це дозволяє користувачеві самостійно вирішувати, яка модель найбільше підходить для поточного запиту, з урахуванням таких критеріїв, як стиль відповіді, точність, креативність або швидкість. Такий підхід значно підвищує ефективність роботи та забезпечує гнучкість у використанні системи.

Ще однією важливою перевагою є підтримка моделей генерації зображень безпосередньо в межах чату. Користувач має змогу надсилати текстовий опис і миттєво отримувати відповідну візуалізацію, яка відображається поряд із відповіддю текстової моделі. Це створює інтегрований мультимодальний досвід, який відсутній у більшості існуючих платформ.

На відміну від багатьох конкурентів, де користувач має обмежений контроль над сеансом взаємодії, в «AI Hub» реалізовано механізм збереження історії запитів. Це дозволяє не лише повертатися до попередніх діалогів, а й здійснювати контекстну роботу, аналізувати варіанти відповідей від різних моделей і використовувати збережені сесії для повторної генерації або навчання.

Використання сучасного технологічного стеку – зокрема фреймворку Next.js для клієнтської частини, PocketBase для керування даними й авторизації, а також API OpenRouter і Fal.ai для підключення моделей – забезпечує високу продуктивність, масштабованість і простоту підтримки додатку. Завдяки модульній структурі кодова база легко розширюється, що дозволяє інтегрувати нові моделі або функціональні компоненти без суттєвого перепроектування архітектури.

Інтерфейс додатку побудовано з урахуванням принципів адаптивного

дизайну. Це означає, що «AI Hub» однаково зручно використовувати як на десктопі, так і на мобільних пристроях. Усі елементи керування інтуїтивно зрозумілі, що забезпечує комфорт навіть для нових користувачів, які не мають попереднього досвіду роботи зі штучним інтелектом.

Платформа також надає технічну основу для подальшого впровадження функціоналу колективної роботи — наприклад, розширення чату на кількох користувачів, обмін сесіями, коментування відповідей або порівняння результатів у командному середовищі. Завдяки модульному підходу до побудови архітектури такі можливості можуть бути легко додані без потреби у глибокій перебудові системи.

Ще одним важливим аспектом є відкритість до кастомізації: структура застосунку дозволяє адаптувати платформу під конкретні потреби організацій, інтегрувати власні мовні моделі або додати фільтрацію, сортування, обмеження доступу за ролями. Це робить AI Hub не лише готовим продуктом, а й основою для створення корпоративних або освітніх платформ на базі вже реалізованого ядра.

Таким чином, запропоноване рішення не тільки усуває основні недоліки наявних платформ, а й формує новий підхід до взаємодії з ШІ – централізований, персоналізований та мультимодальний. Це робить «AI Hub» перспективним інструментом для широкого кола користувачів – від звичайних користувачів і студентів до дослідників, розробників і працівників інших різноманітних сфер діяльності.

2 АНАЛІЗ ТА ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ РОЗРОБКИ ЗАСТОСУНКУ

2.1 Клієнтська логіка

Клієнтська частина web-додатку «AI Hub» відіграє ключову роль у взаємодії користувача з мовними моделями та генераторами зображень, саме тому вибір інструментів для її реалізації має стратегічне значення. Основу фронтенд-розробки становить Next.js. Це сучасний React-фреймворк, що поєднує переваги серверного рендерингу, статичної генерації, клієнтської інтерактивності та розширеної маршрутизації. Його застосування дозволяє забезпечити швидке завантаження сторінок, високу продуктивність, гнучкість у роботі з контентом і покращену індексацію у пошукових системах, що є важливим для web-додатків зі складною структурою [3].

Архітектура клієнтської частини базується на концепції App Router, яка дозволяє чітко структурувати маршрути, розмежовуючи логіку відображення, запити до API та компоненти інтерфейсу. Такий підхід сприяє модульності, що особливо актуально для масштабованих систем, де кожен розділ додатку – наприклад, чат, історія, генерація зображень – може розвиватися незалежно.

У якості основної бібліотеки інтерфейсів використовується React, який дозволяє реалізовувати динамічні, компонентно-орієнтовані UI-рішення. React забезпечує ефективне оновлення DOM через механізм Virtual DOM, що критично важливо при частих змінах стану інтерфейсу – зокрема, під час генерації повідомлень, оновлення чатів чи завантаження зображень. Реактивна модель рендерингу дозволяє будувати масштабовані додатки з високим рівнем інтерактивності.

Для стилізації у проєкті застосовується бібліотека NextUI, яка надає набір готових компонентів для швидкої розробки сучасного та візуально

привабливого інтерфейсу. Її використання дозволяє зосередитись на логіці застосунку, оскільки всі компоненти – кнопки, селектори, модальні вікна, вкладки тощо – вже оптимізовані для адаптивного відображення та мають вбудовану підтримку тем оформлення. Особливо корисною є можливість легко перемикатися між світлою та темною темами, що покращує UX у різних умовах використання.

NextUI чудово інтегрується з TailwindCSS, даючи можливість поєднувати гнучкість утилітарної стилізації з готовими шаблонними рішеннями. Бібліотека також підтримує анімації, управління фокусом, кастомізацію інтерфейсних елементів та доступність (accessibility), що дозволяє створювати інтерфейс із високим рівнем взаємодії без написання додаткового коду з нуля.

Код клієнтської частини реалізовано на TypeScript – надмножині JavaScript з підтримкою статичної типізації. Завдяки цьому забезпечується більша передбачуваність поведінки додатку, підвищується безпека типів і зменшується кількість потенційних помилок на етапі виконання. Це особливо актуально для взаємодії з зовнішніми API, де точність формату даних має критичне значення.

Формування користувацького інтерфейсу реалізовано через набір власних компонентів, які повторно використовуються у різних частинах додатку. Такий підхід відповідає принципам DRY (Don't Repeat Yourself) та сприяє підтримці чистоти коду. Компоненти структуровано у відповідні директорії, що полегшує навігацію в проєкті та пришвидшує розробку.

Ще однією перевагою обраного стеку є підтримка асинхронної взаємодії з сервером. Кожен запит до мовної або візуальної моделі реалізується через асинхронні функції з обробкою станів: завантаження, очікування, завершення або помилка. Це дозволяє забезпечити стабільну роботу навіть у випадках великої кількості запитів або довготривалих відповідей (рисунок 2.1).

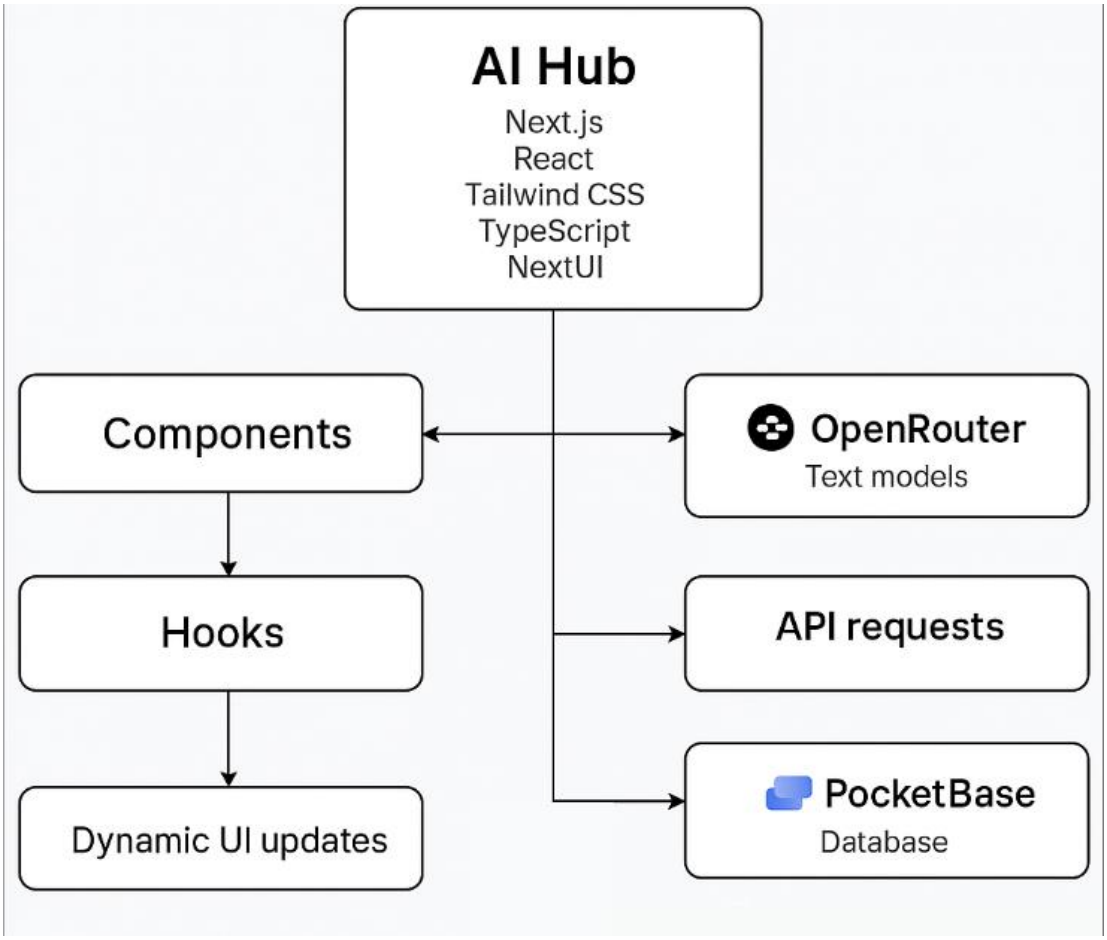

Рисунок 2.1 – Структура застосунку

Завдяки використанню Next.js, React, NextUI та TypeScript клієнтська частина додатку «AI Hub» не лише відповідає сучасним вимогам до продуктивності, а й забезпечує зручність у розробці, модифікації та масштабуванні. Обрані технології дозволяють реалізувати адаптивний, кросплатформений інтерфейс з високим рівнем інтерактивності та стабільності, що є критично важливим для продукту, орієнтованого на щоденну роботу з інструментами штучного інтелекту.

2.2 Серверна логіка

Серверна частина web-застосунку «AI Hub» виконує допоміжну, але важливу функцію. Вона відповідає за збереження основних даних користувачів та історії їхньої взаємодії зі штучним інтелектом [4]. Для

реалізації цієї логіки було обрано рішення на базі PocketBase – сучасного легковагового бекенду, який поєднує в собі функції бази даних, API-сервера, механізмів аутентифікації та WebSocket-підтримки. Такий підхід дозволяє розгорнути повноцінну серверну інфраструктуру без залучення зовнішніх сервісів або побудови окремого backend-додатку, що значно скорочує час розробки та спрощує підтримку проєкту (рисунок 2.2).

Важливою перевагою PocketBase є його готовність до використання без додаткових налаштувань. Сервіс дуже швидко розгортається без зайвихдій та не потребує складного конфігурування чи встановлення додаткових компонентів.

```
PS D:\A_XHYPE\A_diploma\Backend> .\pocketbase.exe
PocketBase CLI

Usage:
  pocketbase.exe [command]

Available Commands:
  admin       Manages admin accounts
  migrate     Executes app DB migration scripts
  serve       Starts the web server (default to 127.0.0.1:8090 if no domain is specified)
  update      Automatically updates the current app executable with the latest available version

Flags:
      --automigrate            enable/disable auto migrations (default true)
      --dev                    enable dev mode, aka. printing logs and sql statements to the console
      --dir string             the PocketBase data directory (default "D:\\A_XHYPE\\A_diploma\\Backend\\pb_data")
      --encryptionEnv string   the env variable whose value of 32 characters will be used
                               as encryption key for the app settings (default none)
  -h, --help                   help for pocketbase.exe
      --hooksDir string        the directory with the JS app hooks
      --hooksPool int          the total prewarm goja.Runtime instances for the JS app hooks execution (default 25)
      --hooksWatch             auto restart the app on pb_hooks file change (default true)
      --indexFallback          fallback the request to index.html on missing static path (eg. when pretty urls are used with SPA) (
default true)
      --migrationsDir string   the directory with the user defined migrations
      --publicDir string       the directory to serve static files (default "D:\\A_XHYPE\\A_diploma\\Backend\\pb_public")
      --queryTimeout int       the default SELECT queries timeout in seconds (default 30)
  -v, --version                version for pocketbase.exe

Use "pocketbase.exe [command] --help" for more information about a command.
PS D:\A_XHYPE\A_diploma\Backend> .\pocketbase.exe serve
2025/04/12 13:17:36 Server started at http://127.0.0.1:8090
├─ REST API: http://127.0.0.1:8090/api/
└─ Admin UI: http://127.0.0.1:8090/_/
```

Рисунок 2.2 – Розгортання PocketBase сервісів

Після першого запуску розробник одразу отримує доступ до вбудованої веб-адміністративної панелі (рисунок 2.3), яка відкриває широкі можливості для управління застосунком без написання коду. Через цю панель можна створювати колекції (аналог таблиць у БД), додавати поля, задавати правила валідації, редагувати записи, керувати користувачами, ролями та правами

доступу. Крім того, панель дозволяє візуально переглядати всю інформацію, що зберігається в базі даних, у режимі реального часу. Це суттєво підвищує зручність у процесі розробки, оскільки відпадає потреба в додаткових клієнтах для роботи з БД або написанні запитів вручну.

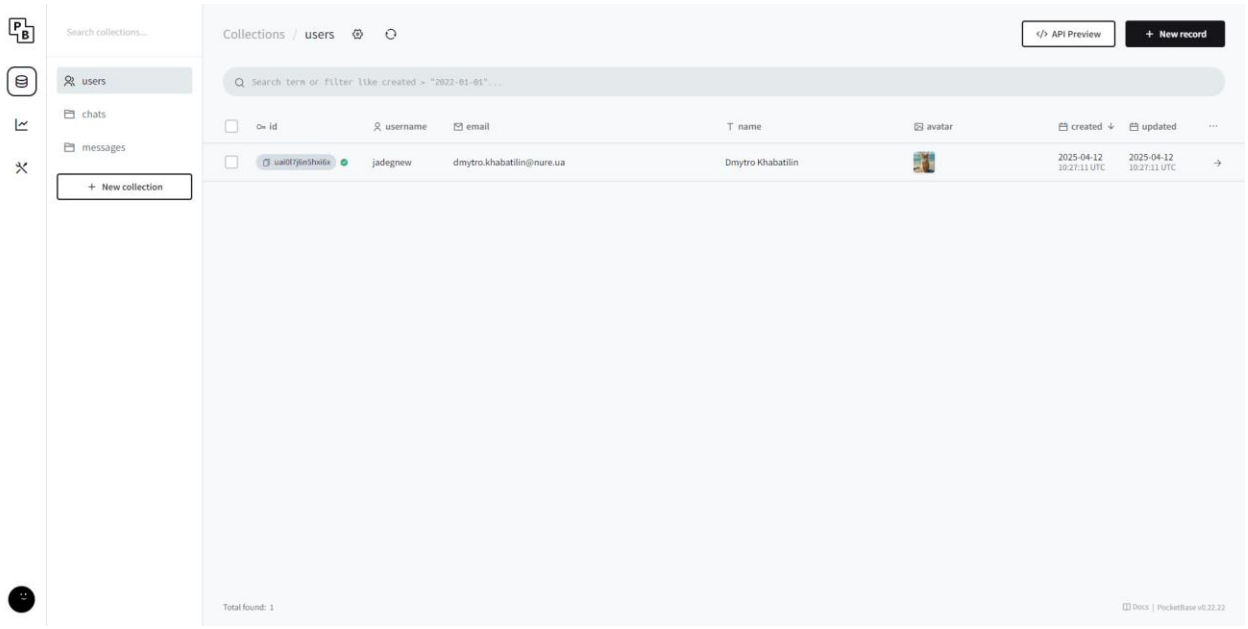

Рисунок 2.3 – Веб-адміністративна панель PocketBase

Логіка, реалізована на стороні PocketBase, охоплює зберігання облікових записів користувачів, які реєструються та авторизуються через стандартні API-ендпоінти з валідацією пароля, генерацією токенів доступу та можливістю оновлення сесій. Платформа також підтримує механізми підтвердження електронної пошти, скидання пароля та обмеження доступу на основі токенів, що дозволяє легко масштабувати функціонал у майбутньому [5]. Створення користувача, його вхід у систему та перевірка прав на перегляд даних реалізуються на рівні REST-запитів до колекції, яку система автоматично формує як users.

Окрім цього, PocketBase використовується як централізоване сховище для чатів і повідомлень. Кожен чат, створений користувачем у клієнтському інтерфейсі, фіксується як окремий запис у відповідній колекції, при цьому всі надіслані або отримані повідомлення зберігаються у зв’язаній колекції, із

зазначенням часу, автора, типу повідомлення (текст або зображення), а також моделі, яка брала участь у генерації відповіді. Така структура дозволяє забезпечити логічну цілісність історій, створює передумови для реалізації функцій перегляду попередніх сесій, аналізу якості відповідей різних моделей або навіть візуалізації взаємодії у вигляді діаграм у майбутньому.

Усі дані обробляються через REST API безпосередньо з клієнтської частини додатку. Це дозволяє уникнути потреби у проміжному сервері або проксі, а також робить взаємодію з базою максимально прозорою. Кожен користувач після авторизації отримує токен доступу, який додається до заголовків запитів і дозволяє виконувати захищені операції, як створення чатів, читання історії або видалення сесій.

У проєкті реалізована реєстрація та авторизація користувачів. Кожен користувач має унікальний профіль, що дозволяє зберігати й відновлювати історію взаємодії з ШІ-моделями у межах особистого кабінету.

Чати зберігаються у вигляді окремих сутностей, прив’язаних до конкретного користувача. Повідомлення, надіслані у процесі спілкування з моделями, додаються до відповідного чату разом із метаданими – датою, назвою використаної моделі, типом (текст чи зображення) та вмістом. Завдяки структурованій моделі даних реалізовано повноцінне відновлення історії, підтримку контексту і можливість роботи з попередніми сесіями.

Перевагою такого підходу є його лаконічність і достатність – система не обтяжена зайвою логікою, мікросервісною структурою чи складною маршрутизацією. Усе зосереджено на мінімально необхідному функціоналі: підтримці користувача та збереженні його історії. Разом із тим архітектура залишається відкритою до розширення – у разі потреби в майбутньому можна буде додати нові поля, налаштувати ролеву модель доступу або навіть винести окремі частини логіки на кастомні сервери.

Таким чином, серверна логіка «AI Hub» побудована за принципами простоти, ефективності та керованості. PocketBase надає всі необхідні можливості для повноцінної реалізації необхідного функціоналу без

перевантаження проєкту зайвими технологічними залежностями.

2.3 Зовнішні API

Однією з ключових переваг застосунку «AI Hub» є його здатність взаємодіяти з різними моделями штучного інтелекту через зовнішні API [6]. Такий підхід дозволяє забезпечити широкі можливості генерації контенту, гнучкість у виборі моделей та високу масштабованість системи. У реалізації проєкту використовуються два незалежні напрями інтеграції – перший охоплює мовні моделі, другий – моделі генерації зображень (рисунок 2.4).

Для роботи з мовними моделями обрано платформу OpenRouter, яка виступає як універсальний маршрутизатор запитів до широкого спектра моделей від різних постачальників. Через єдину REST API OpenRouter забезпечує доступ до моделей GPT-4o, GPT-Omni, Claude, Gemini, LLaMA, Mistral, Cohere Command R та багатьох інших. У застосунку реалізовано можливість динамічного перемикання між моделями – користувач самостійно обирає, з якою моделлю бажає працювати, а система автоматично формує відповідний запит з урахуванням назви моделі та поточного повідомлення. Це дозволяє не лише отримувати різні стилі відповіді, а й порівнювати якість та поведінку моделей у межах одного інтерфейсу, не змінюючи логіку чату.

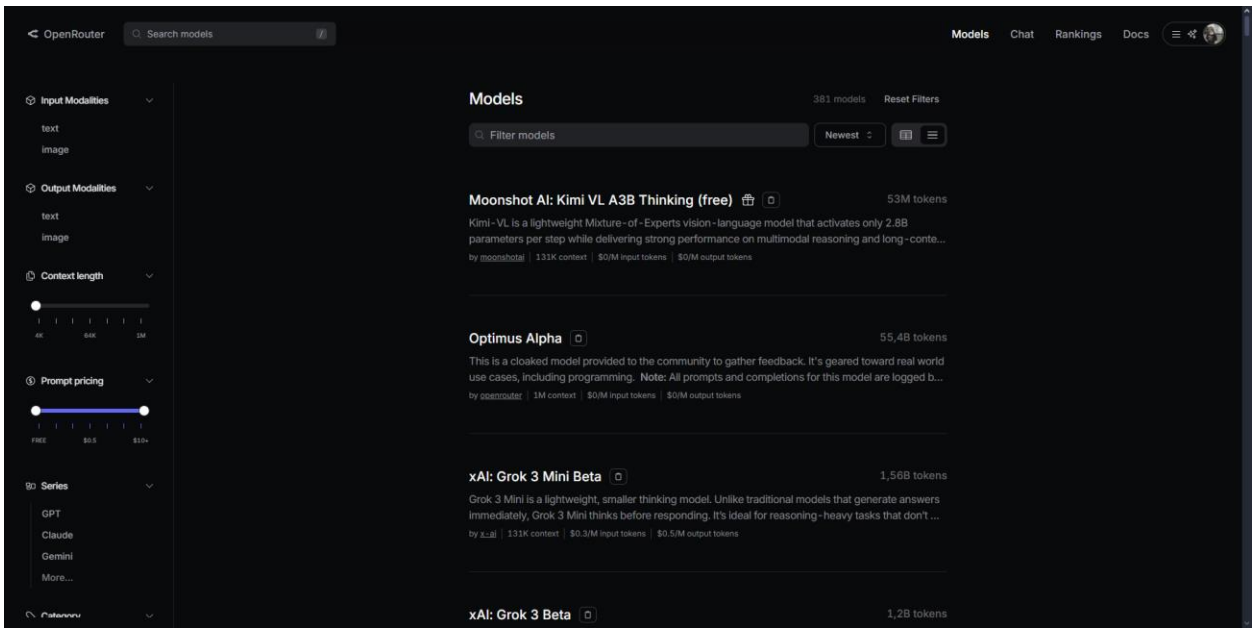

Рисунок 2.4 – Перелік моделей, що підтримуються в OpenRouter

Важливою особливістю OpenRouter є підтримка стрімінгу – відповіді надходять частинами в реальному часі, і це дозволяє відображати результат генерації поступово, як у живому діалозі. Така функція реалізована через обробку потокових відповідей (Server-Sent Events або text/event-stream), що надає користувачеві більш природний досвід спілкування з ШІ та зменшує час очікування на повну відповідь [7].

На відміну від мовних моделей, генерація зображень у застосунку реалізується через безпосереднє підключення до API DALL·E 3 від OpenAI. У цьому випадку інтеграція відбувається напряму, без використання OpenRouter. Власне API забезпечує більший контроль над параметрами запиту, стабільність у роботі та повну сумісність із внутрішніми механізмами обробки результатів. Користувач вводить текстовий опис, після чого клієнтська частина надсилає запит до ендпоінту OpenAI, вказуючи prompt та параметри розміру зображення. У відповідь застосунок отримує URL згенерованого зображення, який інтегрується в потік повідомлень, і користувач бачить зображення безпосередньо в межах поточної сесії.

Такий підхід дозволяє ізолювати канали взаємодії: текстова генерація обробляється через єдиний шлюз (OpenRouter), що спрощує логіку вибору

моделей, тоді як генерація зображень залишається контрольованим окремим модулем із чітко визначеною схемою. Це дає змогу легше масштабувати обидва напрями окремо, не створюючи надмірної взаємозалежності між ними. У майбутньому заплановано додавання підтримки інших моделей зображень – таких як Midjourney або Stable Diffusion – шляхом винесення генерації в окремі API-модулі із загальною структурою запиту, уніфікованою для різних провайдерів.

Архітектура взаємодії з зовнішніми API побудована з урахуванням можливості обробки як синхронних, так і асинхронних відповідей. Усі запити реалізовані як асинхронні функції, що обробляють статус генерації, повідомляють про прогрес користувачу та підтримують можливість відображення помилок при недоступності моделей або перевищенні лімітів.

Таким чином, застосунок «AI Hub» реалізує гнучку, масштабовану і розділену інтеграцію з зовнішніми системами штучного інтелекту: мовні моделі підключені через універсальний шлюз OpenRouter, а зображення генеруються безпосередньо через API DALL·E 3 від OpenAI. Це забезпечує максимальну стабільність, розширюваність та функціональність системи в межах єдиного інтерфейсу.

Додатковою перевагою такого підходу є можливість централізованого контролю над усіма зовнішніми інтеграціями. Завдяки тому, що всі запити до мовних моделей обробляються через OpenRouter, спрощується керування API-ключами, логування активності та моніторинг запитів. Це особливо важливо у випадках масштабування системи, введення обмежень на кількість генерацій або поділу доступу між різними користувачами.

Інтеграція зовнішніх API також враховує питання відмовостійкості: передбачено обробку помилок на стороні користувача із виведенням інформативних повідомлень про збій, тайм-аут, недоступність моделі або перевищення ліміту. Така логіка дозволяє уникнути збоїв у роботі інтерфейсу та забезпечує надійність при реальній експлуатації системи.

3 ОПИС ЗАСТОСУНКУ

Web-застосунок «AI Hub» – це платформа для взаємодії з різними штучними інтелектами в одному вікні. Основна ідея полягає у створенні централізованого середовища, яке дозволяє вести повноцінні діалоги з мовними моделями (LLM), генерувати зображення, зберігати історію спілкування та працювати в мультимодальному режимі без необхідності перемикатися між окремими сервісами або вкладками.

Інтерфейс додатку є мінімалістичним, сучасним і повністю сфокусованим на комфорті користувача [8]. Після запуску користувача зустрічає вітальний екран (рисунок 3.1), який слугує точкою входу до функціоналу чату. У верхній частині інтерфейсу розміщено головне меню навігації з двома основними пунктами: Dashboard та Chat, а також елементи керування профілем, перемиканням теми оформлення та зовнішніми соціальними посиланнями.

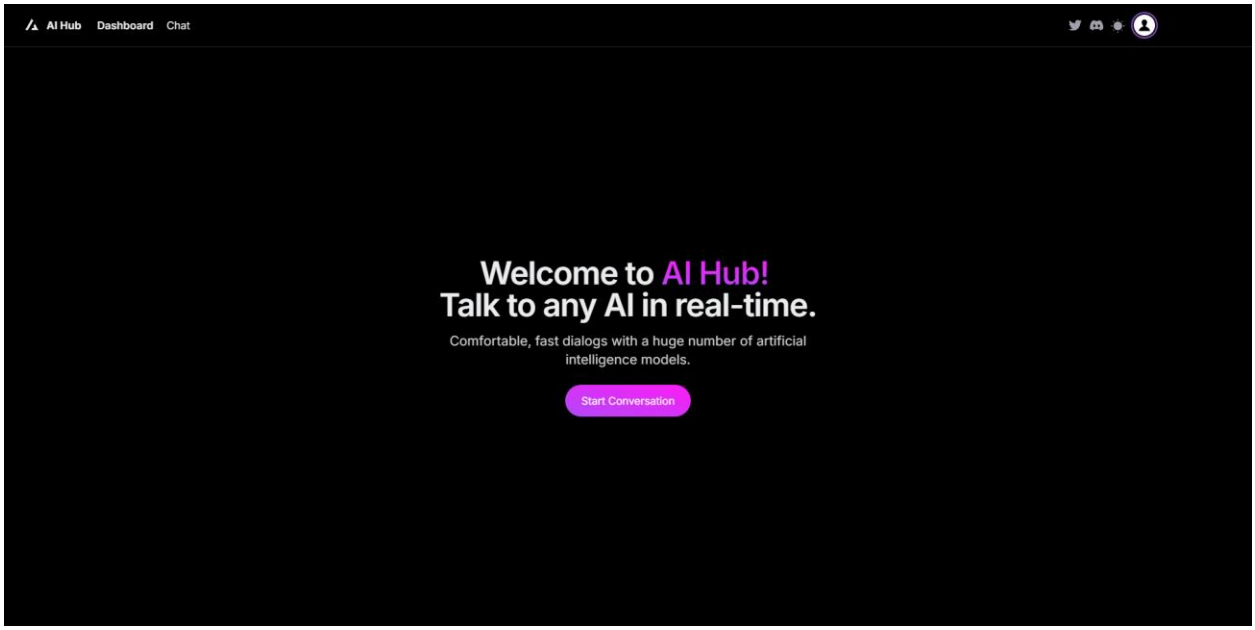

Рисунок 3.1 – Головна сторінка застосунку з вітальним екраном

Взаємодія з платформою починається після натискання кнопки Start

Conversation на головному екрані або Chat в меню навігації, що переводить користувача до основного вікна чату. Інтерфейс чату побудовано таким чином, щоб забезпечити повну концентрацію на процесі спілкування. Центр екрана займає область повідомлень, нижче – текстове поле введення, а у верхній частині – елемент вибору моделі, з якою ведеться діалог.

Кожен новий користувач має змогу одразу почати роботу з системою без необхідності попереднього налаштування або додаткових інтеграцій. Всі необхідні механізми – обробка запитів, відповідей, генерація зображень та відображення історії – реалізовані у вигляді єдиного реактивного потоку подій. Це створює відчуття «живого» чату, де зміна моделей, отримання тексту або зображення відбувається без перезавантажень, очікувань чи втрати контексту.

Кожен елемент інтерфейсу адаптовано до темної та світлої теми оформлення, що перемикаються в один клік. Дизайн створено з урахуванням вимог доступності, адаптивності та інтуїтивної навігації. Таким чином, уже на стартовому етапі користувач отримує відчуття цілісної, зручної й продуктивної екосистеми, яка адаптована до сучасного стилю взаємодії з інструментами штучного інтелекту.

Однією з ключових особливостей застосунку «AI Hub» є можливість одночасної взаємодії з різними мовними моделями в межах одного чату. Такий підхід докорінно відрізняється від більшості сучасних платформ, які прив’язують користувача до єдиної моделі або змушують відкривати окремі сесії для кожного інструменту. У «AI Hub» реалізовано гнучкий механізм перемикання між моделями в реальному часі, без втрати історії повідомлень або перезавантаження інтерфейсу.

У верхній частині чатового вікна розміщено випадаючий список моделей, доступних для взаємодії. Користувач має змогу обрати одну з представлених моделей – GPT-4o, Claude 3.5 Haiku, Gemini 2 Flash, Qwen 2, Grok 2 та інші – і моментально продовжити діалог, надсилаючи запити вже новообраній системі (рисунок 3.2). Усі відповіді зберігаються в загальному

потоці повідомлень, що дозволяє безпосередньо порівнювати якість, стиль та логіку різних моделей у межах єдиного інтерфейсу.

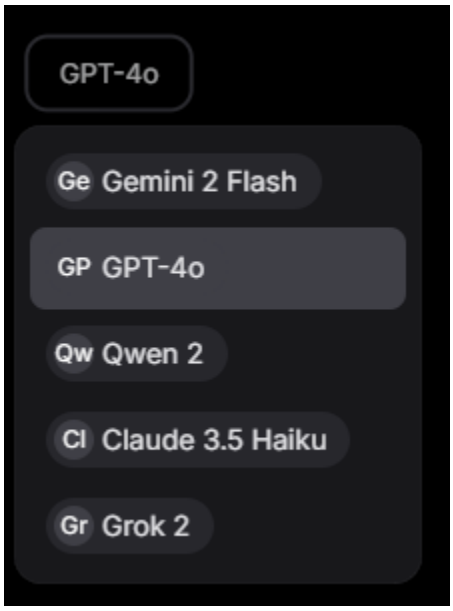

Рисунок 3.2 – Випадаючий список вибору моделі в чаті

Завдяки цьому користувач може використовувати сильні сторони кожної конкретної моделі у відповідний момент – наприклад, швидкість генерації у Gemini, логічну послідовність GPT-4o або стислість Claude. Така функціональність особливо корисна в аналітичних задачах, під час навчання або при тестуванні генеративних можливостей різних систем.

Крім перемикання між моделями, система підтримує повноцінну генерацію текстових відповідей у режимі реального часу. Користувач може надіслати будь-який запит у текстове поле вводу – від простих побутових запитань до складних технічних або аналітичних задач. Наприклад, на рисунку 3.3 показано приклад запиту «Напиши мені 5 фактів про котів», після якого модель повертає розгорнуту відповідь, структуровану у вигляді маркованого списку, з поясненнями та прикладами (рисунок 3.3).

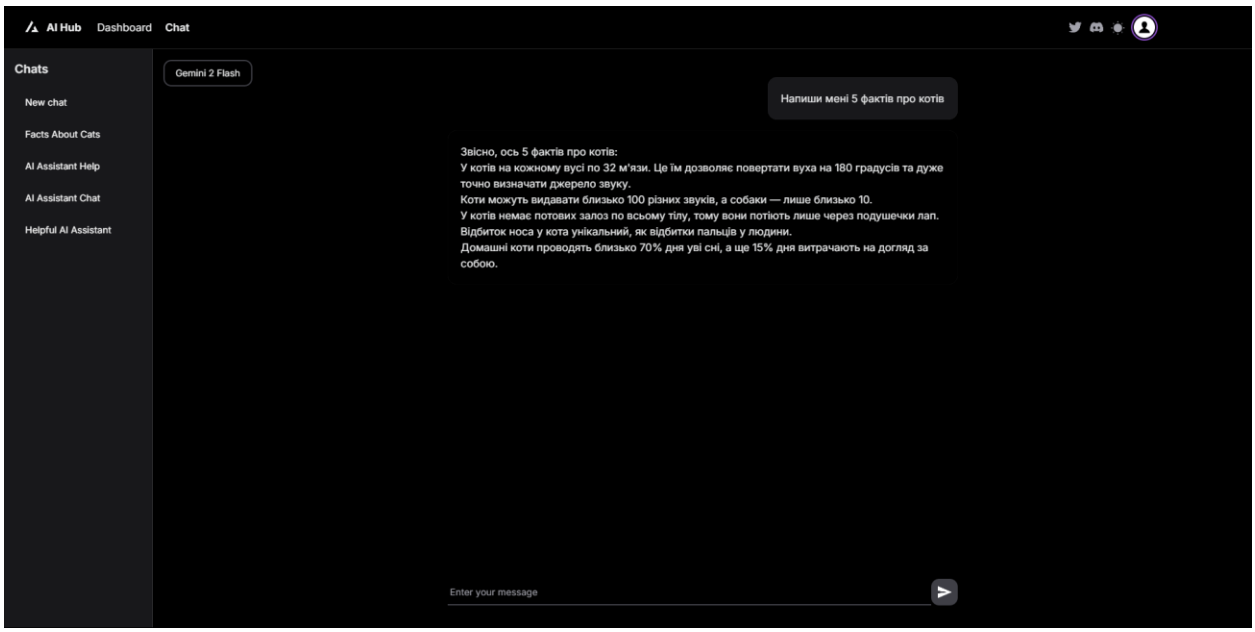

Рисунок 3.3 – Взаємодія з моделлю: текстовий запит та відповідь у чаті

Система підтримує як короткі, так і довгі відповіді. Якщо модель має стрімінгову підтримку, текст поступово з’являється на екрані, імітуючи «живе» спілкування. Це створює природне відчуття діалогу, підвищуючи загальну інтерактивність платформи. Кожна відповідь супроводжується міткою моделі, що надала її, – це дозволяє відстежити, як саме різні AI інтерпретують одне й те саме питання.

Формат відповіді може включати маркування, виділення, кодування, таблиці та навіть багатомовні фрагменти. Це дозволяє використовувати платформу не лише для розмовного спілкування, а й для серйозної роботи з текстовим контентом: написання статей, пояснень, навчальних матеріалів або генерації коду.

Інтеграція OpenRouter як бекенду для текстових моделей забезпечує надійний і гнучкий доступ до широкого переліку моделей із єдиною системою автентифікації та форматування запитів. Це дозволяє в перспективі розширювати список доступних моделей без внесення змін у логіку взаємодії або інтерфейс.

Таким чином, мультимодельна архітектура «AI Hub» створює принципово новий досвід взаємодії з штучним інтелектом: вона дозволяє

вивчати, тестувати та комбінувати можливості різних LLM у межах одного інтерфейсу, не перериваючи сесій та не виходячи за межі чату.

Застосунок «AI Hub» з самого початку проєктувався як інструмент, орієнтований на глобальну аудиторію. Саме тому одним із ключових аспектів його функціональності є підтримка мультимовності – здатність взаємодіяти з користувачами будь-якою мовою світу. Це стало можливим завдяки інтеграції з мовними моделями через OpenRouter, які мають багатомовну підготовку та демонструють високі результати у розумінні запитів і генерації відповідей різними мовами.

Користувач може вільно спілкуватися українською, англійською, німецькою, французькою, китайською, арабською або будь-якою іншою мовою – система автоматично визначає мову запиту та адаптує відповідь відповідно до нього. Немає необхідності попередньо обирати мову інтерфейсу чи вводити спеціальні команди – усе працює на основі природної мови, що значно підвищує зручність використання.

На практиці це означає, що користувач може задати одне й те саме питання різними мовами і отримати коректні, лексично багаті та граматично правильні відповіді. Наприклад, на рисунку 3.4 показано діалог, де запит сформульовано українською, а відповідь моделі збережена в загальному чаті поряд з іншими англомовними результатами. Така взаємодія відкриває широкі можливості для багатомовного навчання, перекладу, локалізації або тестування ШІ-моделей у різних лінгвістичних контекстах (рисунок 3.4).

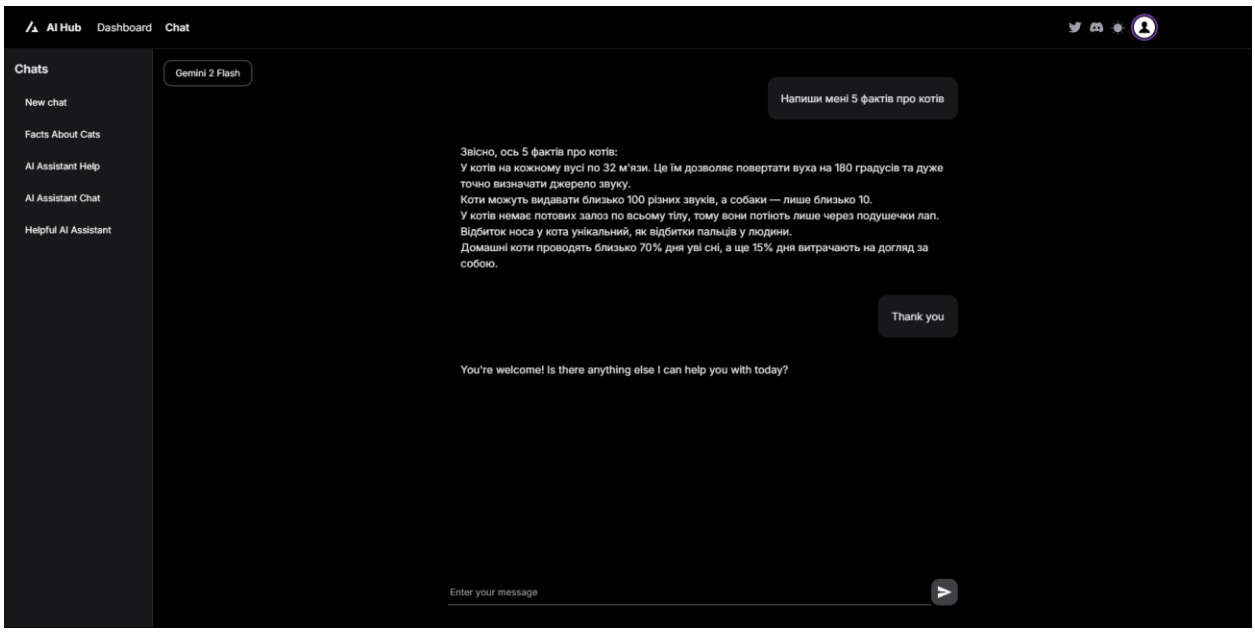

Рисунок 3.4 – Приклад взаємодії з моделлю різними мовами

Однією з функцій, реалізованих у платформі «AI Hub», є можливість генерації зображень безпосередньо в чаті – без потреби перемикання між окремими інструментами чи вибору спеціальної моделі. Це стало можливим завдяки використанню механізму Function Calling, що дозволяє мовним моделям не лише відповідати на запит, а й викликати вбудовану функцію генерації зображення у відповідь на відповідний запит користувача.

Користувачеві достатньо ввести звичайне текстове повідомлення із проханням згенерувати зображення, наприклад: "Згенеруй зображення кота web-розробника". Модель аналізує запит, розпізнає його як такий, що потребує візуальної генерації, і автоматично викликає функцію, яка надсилає запит до API DALL·E 3 від OpenAI. Після обробки запиту користувач отримує зображення безпосередньо в межах того самого чату, як звичайну відповідь моделі (рисунок 3.5).

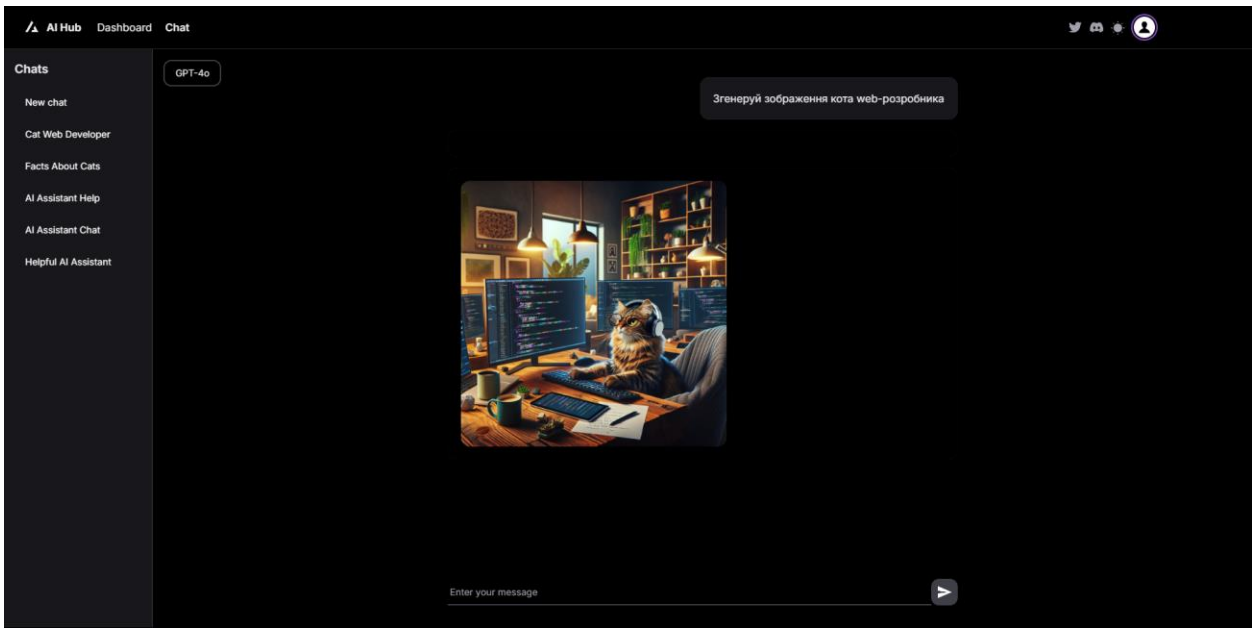

Рисунок 3.5 – Приклад запиту на генерацію зображення та результат у чаті

Цей підхід має кілька переваг. По-перше, користувач не змінює контекст взаємодії – як текстові, так і візуальні запити обробляються єдиною логікою. По-друге, відсутність окремого інтерфейсу для генерації зображень робить роботу швидшою, зрозумілішою та інтегрованішою. Користувач сприймає генерацію зображення як частину звичайного діалогу з мовною моделлю.

Візуалізація відбувається з короткою затримкою, що пояснюється часом обробки на стороні API. При цьому користувач бачить повідомлення, що зображення генерується, а після завершення – фінальний результат із можливістю відкриття у повному розмірі.

Такий рівень інтеграції створює мультимодальний досвід, де текст і зображення можуть комбінуватися у природний, послідовний потік. Це особливо корисно в креативних задачах, презентаціях, візуалізації концептів або освітніх сценаріях, де візуальна складова відіграє не меншу роль, ніж текстова. Інтерфейс платформи «AI Hub» реалізовано з урахуванням підтримки обох режимів візуального оформлення – світлого та темного. Це дозволяє користувачам самостійно обирати вигляд застосунку відповідно до особистих уподобань, умов освітлення або режиму використання пристрою.

Перемикання між темами відбувається миттєво, без перезавантаження сторінки або втрати поточного стану сесії.

Кнопка перемикання розміщена у правій частині верхньої панелі інтерфейсу і доступна з будь-якої сторінки додатку. Система автоматично зберігає вибір користувача – при наступному вході тема відкривається у попередньо встановленому режимі. Це підвищує зручність роботи, особливо у випадках тривалого використання застосунку або роботи в умовах обмеженого освітлення (рисунок 3.6).

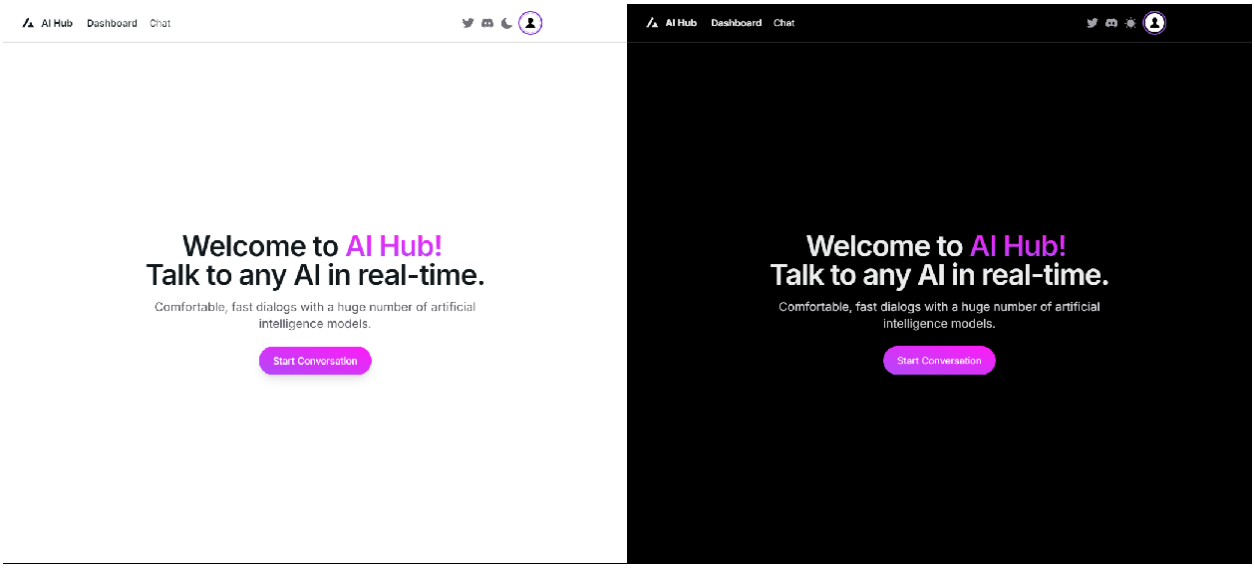

Рисунок 3.6 – Інтерфейс застосунку у світлій та темній темі

Темна тема оформлена у темно-сірій гамі з акцентними елементами у синьому або білому кольорі. Вона особливо зручна для роботи у вечірній або нічний час, знижує навантаження на зір і дозволяє зосередитись на контенті. Світла тема має світлий фон, чіткі межі блоків і добре підходить для презентацій, навчання або денного використання на мобільних пристроях.

Усі візуальні компоненти інтерфейсу – від чатів до списків моделей – адаптовані під обидва режими. Це означає, що незалежно від вибору теми, жоден елемент функціоналу не втрачає своєї контрастності, читабельності або логічної побудови. TailwindCSS у поєднанні з UI-бібліотекою NextUI забезпечує узгоджене відображення елементів у різних темах та на різних

пристроях.

Таким чином, підтримка двох режимів оформлення не лише покращує візуальне сприйняття, а й підкреслює увагу до досвіду користувача – незалежно від того, як, коли та з якого пристрою він працює з застосунком.

Застосунок «AI Hub» створено з урахуванням вимог адаптивного дизайну, що дозволяє забезпечити коректну роботу інтерфейсу як на великих екранах (настільні ПК, ноутбуки), так і на мобільних пристроях. Незалежно від розміру екрана, інтерфейс зберігає свою логічну структуру, зручність навігації та повний функціонал.

У мобільному режимі застосунок автоматично адаптується до вертикального розміщення елементів. Всі основні блоки – чат, список моделей, поле введення повідомлень, кнопки керування – перебудовуються у зручний формат, оптимізований для роботи пальцями та перегляду однією рукою. Меню приховується за іконку-бургер, а елементи інтерфейсу масштабуються відповідно до ширини екрана (рисунок 3.7).

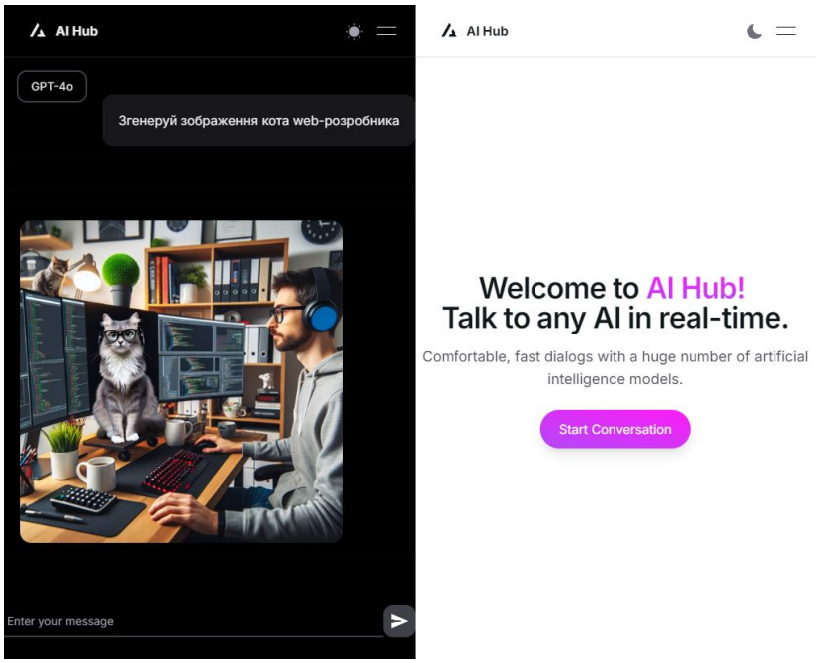

Рисунок 3.7 – Мобільна версія застосунку на смартфоні

Повна функціональність чату, перемикання моделей, генерація зображень, темна та світла тема – усе це доступно і на мобільних пристроях

без обмежень. Інтерфейс протестовано на сучасних браузерах Android та iOS, що дозволяє використовувати застосунок у будь-яких умовах – у дорозі, на перерві, у кафе або під час навчання.

Особливу увагу приділено оптимізації швидкодії на мобільних платформах. Компоненти інтерфейсу не перевантажені анімаціями, графіка оптимізована, а логіка обробки запитів виконується асинхронно, що забезпечує стабільну роботу навіть при повільному інтернет-з'єднанні. TailwindCSS та можливості Next.js дозволяють динамічно адаптувати вміст без додаткових стилів або дублювання розмітки.

Наявність повноцінної мобільної підтримки розширює сферу застосування «AI Hub», роблячи платформу по-справжньому універсальною та зручною для щоденного використання. Користувач має можливість спілкуватися з моделями та створювати зображення у будь-який момент – незалежно від місця перебування чи типу пристрою.

Інтерфейс платформи побудовано з урахуванням зручності та універсальності: користувач може обирати модель, мову, тему оформлення, а також отримувати як текстові, так і візуальні відповіді в межах єдиного інтерфейсу. Завдяки адаптивному дизайну забезпечено повноцінну роботу як на десктопах, так і на мобільних пристроях. Реалізація генерації зображень без перемикання контексту чату створює справжній мультимодальний досвід [9].

Платформа враховує недоліки наявних рішень, усуває фрагментованість взаємодії з ШІ і пропонує централізоване середовище для щоденної, професійної чи навчальної роботи з сучасними моделями штучного інтелекту. Таким чином, «AI Hub» поєднує зручність, гнучкість, технічну досконалість і відкритість до подальшого масштабування.

4 РЕАЛІЗАЦІЯ ЗАСТОСУНКУ

4.1 Архітектура додатку та структура сторінок

Web-застосунок «AI Hub» розроблений із використанням сучасного підходу до побудови архітектури на базі фреймворку Next.js 14 з підтримкою App Router [10]. Основною метою при побудові структури було досягнення гнучкості, масштабованості та логічного розподілу відповідальностей між окремими частинами системи.

У кореневій директорії проєкту основні робочі файли розташовані у папках app, components, hooks та lib. Кожна папка виконує окрему роль у структурі. app – містить сторінки, макети, логіку маршрутизації, components зберігає багаторазові UI-компоненти, hooks – користувацькі React-хуки, lib – утиліти для взаємодії з PocketBase та авторизації (рисунок 4.1).

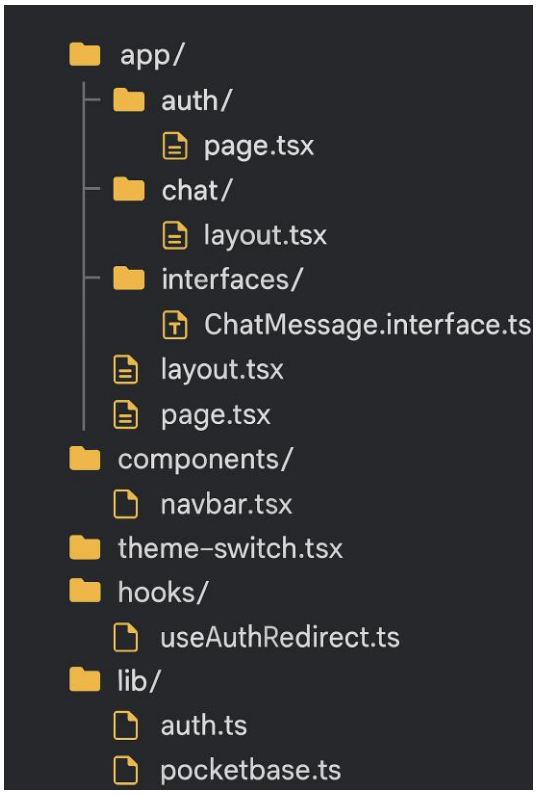

Рисунок 4.1 – Файлова структура додатку «AI Hub»

Використання App Router дозволило організувати маршрути як вкладені папки, де кожен маршрут (chat, auth) має свій власний page.tsx, layout.tsx та допоміжні файли за потребою [11].

Головний layout у файлі app/layout.tsx задає базову структуру HTML-документу та загальні стилі для всього застосунку (лістинг 4.1). Він обгортає дочірні компоненти в провайдери теми та інтерфейсу.

Лістинг 4.1 – Загальний макет для сторінок (файл app/layout.tsx)

```
<Providers themeProps={{ attribute: "class", defaultTheme:
"dark" }}>
  <div className="relative flex flex-col h-screen">
    <Navbar />
    <main className="w-screen h-screen">
      {children}
    </main>
  </div>
</Providers>
```

Це рішення дозволяє централізовано підключити такі сервіси, як NextUI Provider для стилізації елементів і Theme Provider для перемикання теми (світла або темна).

Сторінка авторизації (auth) реалізована у файлі app/auth/page.tsx і містить окрему логіку для реєстрації та входу користувачів. Додаткові вкладені компоненти, такі як інпут форми або кнопки, використовують готові елементи бібліотеки NextUI для прискорення розробки [12].

Сторінка чату (chat) має власний layout (app/chat/layout.tsx), який встановлює висоту вікна чату та розташування повідомлень. Головний компонент app/chat/page.tsx відповідає за відображення списку чатів, діалогового вікна, поля вводу повідомлень, а також перемикача моделей.

У загальній структурі також передбачено глобальний файл app/page.tsx, який є стартовою сторінкою застосунку. Він перенаправляє користувача або на chat, якщо сесія активна, або на auth для входу, залежно від стану автентифікації в PocketBase (лістинг 4.2).

Лістинг 4.2 – Перенаправлення користувача в залежності від статусу сесії

(файл app/page.tsx)

```
export default function Home() {
  const router = useRouter();

  const handleClick = () => {
    if (pb.authStore.isValid) {
      router.push("/chat");
    } else {
      router.push("/auth");
    }
  };
```

Це рішення забезпечує плавний і безпечний перехід між основними сторінками без ризику доступу до приватних маршрутів неавторизованим користувачем.

Навігація по сайту реалізована через компонент Navbar, який містить динамічні посилання на основні розділи, перемикач теми оформлення та випадаюче меню профілю користувача.

Кожен компонент інтерфейсу розроблено так, щоб максимально спростити розширення системи у майбутньому. Завдяки компонентному підходу та правильній структурі директорій нові сторінки, модулі або окремі функціональні блоки можуть бути додані до проєкту без необхідності реорганізації всієї кодової бази.

Таким чином, архітектура «AI Hub» забезпечує простоту навігації, зручність підтримки, швидкість розробки та можливість безболісного масштабування в разі збільшення функціональності платформи.

4.2 Реалізація спілкування з мовними моделями

Однією з основних функціональних складових застосунку «AI Hub» є організація інтерактивного спілкування з мовними моделями штучного інтелекту [13]. Для цього реалізовано повноцінну інтеграцію з платформою OpenRouter, яка надає доступ до великої кількості моделей різних провайдерів, таких як OpenAI, Google DeepMind, Anthropic та інших.

В основі комунікації з моделями лежить підключення до API через OpenAI, налаштований на використання OpenRouter як проміжного маршрутизатора. Підключення ініціалізується за допомогою динамічного імпорту модуля openai-js (лістинг 4.3).

Лістинг 4.3 – Ініціалізація провайдеру OpenAI з маршрутизатором OpenRouter (файл app/chat/page.tsx)

```
async function initializeOpenAI() {
      const OpenAI = (await import("openai")).default;
      openai = new OpenAI({
        dangerouslyAllowBrowser: true,
        baseURL: "https://openrouter.ai/api/v1",
        apiKey:
          ENV.OPENROUTER
        defaultHeaders: {
          "HTTP-Referer": "https://your-site-url.com",
          "X-Title": "AI Hub",
        },
      });
```

Після підключення OpenAI клієнта користувач має змогу вибрати конкретну модель зі списку підтримуваних. Поточна вибрана модель зберігається у локальному стані компонента через хук useState, що дозволяє легко змінювати модель без перезавантаження сторінки чи сесії.

Надсилання повідомлень і отримання відповідей організоване за допомогою асинхронної функції fetchChatResponseStream, яка надсилає історію діалогу разом із новим повідомленням і обробляє відповідь у режимі реального часу. Реалізовано підтримку стрімінгового отримання даних через об'єкт ReadableStream, що дозволяє відображати відповідь частинами без затримки.

Основна логіка обробки наведена у лістингу 4.4.

Лістинг 4.4 – Основна логіка обробки повідомлень (файл app/chat/page.tsx)

```
for await (const chunk of response) {
        const delta = chunk.choices[0]?.delta;

        if (delta?.tool_calls) {
```

```
          callId = delta.tool_calls[0]?.id;
          functionArgs +=
delta.tool_calls[0]?.function?.arguments || "";
        } else if (delta?.content) {
          onToken(delta.content);
        }

        if (chunk.choices[0]?.finish_reason === "tool_calls") {
          const args = JSON.parse(functionArgs);
          const result = await
functionMap.generateImageWithDalle3(args);
          break;
        }
      }
```

Кожне нове повідомлення, яке відправляє користувач, додається до поточного списку повідомлень у чаті. Якщо чат ще не створений (тобто немає активного chatId), система автоматично створює новий запис у базі даних PocketBase, прив'язуючи його до авторизованого користувача.

Особливістю реалізації також є підтримка стрімінгового оброблення function calling. Модель може, за потреби, ініціювати виклик функції для додаткової обробки запиту, наприклад, генерації зображення через DALL·E 3. Таке розширення функціоналу робить платформу по-справжньому мультимодальною.

Всі повідомлення організовуються відповідно до інтерфейсу ChatMessage, який підтримує текстові та візуальні відповіді, включаючи вказівку типу повідомлення (isImage: boolean).

Таким чином, реалізація спілкування з мовними моделями у «AI Hub» забезпечує:

- динамічний вибір моделей без перезавантаження сторінки;
- стрімінгову передачу відповідей у реальному часі;
- створення та збереження історії діалогів у базі даних;
- підтримку function calling для розширення можливостей моделей.

Завдяки використанню OpenRouter та гнучкої архітектури клієнта, платформа дозволяє швидко адаптуватися до нових моделей, змінювати параметри запитів і розширювати функціонал без зміни базових принципів

роботи.

4.3 Реалізація генерації зображень через API та відображення в чаті

Однією з унікальних функцій застосунку «AI Hub» є можливість генерації зображень безпосередньо в чаті на основі текстового запиту користувача. Для цього реалізовано інтеграцію з моделлю DALL-E-3 від OpenAI через пряме API підключення.

Особливістю реалізації є використання механізму function calling, який підтримується мовними моделями через OpenRouter. Якщо користувач у своєму повідомленні вказує запит на створення зображення (наприклад, «Створи зображення кота в стилі ретро»), модель може ініціювати виклик функції генерації зображення, не змінюючи логіку основного чату [16].

Підключення до API DALL-E-3 здійснюється окремо від OpenRouter. Клієнт налаштовується напряму через офіційний API OpenAI (лістинг 4.5).

Лістинг 4.5 – Ініціалізація провайдеру OpenAI для використання DALL-E-3 (файл app/chat/page.tsx)

```
dalle3 = new OpenAI({
        dangerouslyAllowBrowser: true,
        baseURL: "https://api.openai.com/v1",
        apiKey:
          ENV.DALLE3
      });
```

Після обробки текстового запиту від користувача функція fetchChatResponseStream аналізує відповіді моделі. Якщо модель виявляє запит на генерацію зображення через function_call, система викликає спеціальну локальну функцію generateImageWithDalle3, передаючи їй сформульований запит на генерацію зображення (лістинг 4.6).

Лістинг 4.6 – Виклик функції генерації зображення після завершення виклику

(файл app/chat/page.tsx)

```
if (chunk.choices[0]?.finish_reason === "tool_calls") {
    const args = JSON.parse(functionArgs);
    const result = await
functionMap.generateImageWithDalle3(args);
    break;
}
```

Отримане посилання на згенероване зображення вставляється у чат як окреме повідомлення типу assistant, де поле isImage встановлено у true. Це дозволяє на стороні інтерфейсу легко відрізняти зображення від текстових відповідей та відображати їх відповідно.

В інтерфейсі повідомлення з зображенням виводиться за допомогою компонента Image із підтримкою масштабування та відкриттям у модальному вікні для перегляду у великому розмірі. Це підвищує зручність роботи з візуальними результатами генерації без залишення поточної сесії.

Користувач бачить статус процесу за допомогою спінера «The image is generating...», що відображається у чаті під час очікування відповіді від API.

Таким чином, реалізація генерації зображень у «AI Hub» забезпечує:

- Безперервний мультимодальний досвід спілкування з ШІ;
- Інтеграцію текстових і візуальних відповідей у єдиному чаті;
- Автоматичне визначення запитів на генерацію зображень без участі користувача;
- Оптимізовану взаємодію через function_call та асинхронну обробку результатів.

Функціональність генерації зображень значно розширює можливості застосунку і робить його більш універсальним для креативних та дослідницьких задач.

4.4 Реалізація збереження чатів у базу даних Pocketbase.

Однією з важливих функцій застосунку «AI Hub» є збереження історії чатів користувачів для подальшого доступу, перегляду та продовження

роботи. Для реалізації цієї можливості інтегровано базу даних PocketBase, яка виступає як автономний backend-сервер, забезпечуючи керування колекціями користувачів, чатів та повідомлень [14].

Всі запити до PocketBase здійснюються через клієнт, ініціалізований у файлі lib/pocketbase.ts, що підключається до локального сервера (лістинг 4.7).

Лістинг 4.7 – Підключення до PocketBase (файл lib/pocketbase.ts)

```
import PocketBase from 'pocketbase';

const pb = new PocketBase(ENV.POCKETBASE_URL);
export default pb;
```

Чати зберігаються у колекції chats, а кожне повідомлення – у колекції messages. Структура даних побудована так, щоб кожен чат був прив’язаний до конкретного користувача, а повідомлення були прив’язані до чату через зовнішній ключ.

При створенні нового чату, якщо користувач ще не має активного чату в сесії, система автоматично створює запис у базі (лістинг 4.8).

Лістинг 4.8 – Створення нового чату у базі PocketBase при першому повідомленні (файл app/chat/page.tsx)

```
const newChat = await pb.collection("chats").create({
    user: pb.authStore.model?.id,
    title: "New Chat",
});
currentChatId = newChat.id;
setChatId(newChat.id);
setChats((prev) => [newChat, ...prev]);
```

Повідомлення зберігаються у колекції messages при кожному надсиланні або отриманні відповіді від мовної моделі. Кожне повідомлення містить текст, роль відправника (user або assistant), прив'язку до чату та ідентифікацію моделі, яка генерувала відповідь.

Так само, при отриманні відповіді від моделі або генерації зображення, відповідь також фіксується у базі даних. Повідомлення-зображення

відрізняються від звичайних текстових повідомлень наявністю прапорця isImage: true, що дозволяє правильно обробляти їх при відображенні у чаті.

Кожного разу при завантаженні чату за його id виконується запит до PocketBase для отримання повного списку повідомлень.

Історія завантажується у правильному порядку (від найстаріших до найновіших), що дозволяє відобразити повний контекст розмови без втрати даних. Повідомлення типізовані через інтерфейс ChatMessage, що забезпечує уніфіковане оброблення текстових і візуальних даних.

Ще однією важливою деталлю реалізації є механізм автоматичного оновлення заголовку чату. Після першої взаємодії, система надсилає додатковий запит до моделі із проханням згенерувати коротку назву чату (лістинг 4.9).

Лістинг 4.9 – Автоматичне оновлення заголовку чату після першої відповіді (файл app/chat/page.tsx)

```
const result = await openai.chat.completions.create({
    model: selectedModel.id,
    messages: renamePrompt,
});

const newTitle = result.choices[0]?.message?.content
    ?.replace(/["."]/g, "")
    .trim();

if (newTitle) {
    await pb.collection("chats").update(currentChatId, {
    title: newTitle,
});
setChats((prev) =>
    prev.map((chat) =>
    chat.id === currentChatId ? { ...chat, title: newTitle } :
chat
    )
);
```

Таким чином, реалізація збереження чатів і повідомлень через PocketBase дозволяє:

- підвищити зручність користування шляхом збереження повної історії

взаємодії;

- реалізувати продовження розмови у будь-який момент;
- забезпечити логічну структуру зберігання даних: користувач, чати, повідомлення;
- автоматизувати організацію діалогів без додаткового втручання користувача.

Завдяки гнучкості PocketBase та правильній структурі запитів застосунок «AI Hub» отримав можливість надійно працювати зі складними багаточатовими сценаріями, забезпечуючи при цьому високу швидкодію і масштабованість.

4.5 Робота з обліковими записами користувачів

Для забезпечення безпечної взаємодії користувачів із застосунком «AI Hub» було реалізовано повноцінну систему облікових записів на базі можливостей PocketBase. Система включає реєстрацію нових користувачів, вхід у систему, перевірку сесії та вихід із облікового запису [15].

Уся логіка роботи з обліковими записами зосереджена у модулі lib/auth.ts, який містить функції реєстрації, авторизації та виходу користувача (лістинг 4.10).

Лістинг 4.10 – Функції реєстрації, логіну, виходу та отримання даних користувача через PocketBase (файл lib/auth.ts)

```
export async function register(email: string, username: string,
password: string, passwordConfirm: string) {
  return await pb.collection('users').create({ email, username,
password, passwordConfirm });
}

export async function login(email: string, password: string) {
  return await pb.collection('users').authWithPassword(email,
password);
}

export function logout() {
```

```
  pb.authStore.clear();
}

export function getCurrentUser() {
  return pb.authStore.model;
}
```

Процес реєстрації і авторизації користувача винесено у окрему сторінку /auth, реалізовану у файлі app/auth/page.tsx. На сторінці доступні дві вкладки: одна для реєстрації нового користувача, інша – для входу в існуючий обліковий запис.

Використання бібліотеки компонентів NextUI дозволяє реалізувати сучасний і адаптивний інтерфейс форм.

При реєстрації система збирає введені дані (пошта, ім’я користувача, пароль) і надсилає їх до колекції users в PocketBase.

Після успішної реєстрації користувач автоматично залогується і перенаправляється на сторінку чату chat. Так само під час входу користувач вводить свої облікові дані, які перевіряються через authWithPassword, а у разі успіху сесія зберігається у pb.authStore.

Щоб обмежити доступ до захищених сторінок лише для авторизованих користувачів, реалізовано спеціальний хук useAuthRedirect.ts, який перевіряє валідність сесії при кожному завантаженні сторінки (лістинг 4.11).

Лістинг 4.11 – Перевірка авторизації користувача через хук (файл hooks/useAuthRedirect.ts)

```
export default function useAuthRedirect() {
  const router = useRouter();

  useEffect(() => {
    if (!pb.authStore.isValid) {
      router.replace("/auth");
    }
  }, []);
}
```

У разі натискання кнопки виходу з облікового запису користувач може розлогінитися вручну через функцію розавторизації, яка очищає локальне

збереження токену і перенаправляє користувача назад на сторінку авторизації.

Таким чином, система роботи з обліковими записами у «AI Hub» забезпечує:

- простий та зрозумілий процес реєстрації та входу;
- захист закритих сторінок для неавторизованих користувачів;
- збереження сесії авторизації між оновленнями сторінки;
- можливість безпечного виходу з системи.

Реалізація через PocketBase дозволяє швидко масштабувати функціонал облікових записів у майбутньому – наприклад, додати ролі користувачів, двофакторну автентифікацію або систему підписок.

ВИСНОВКИ

У наш час штучний інтелект стрімко розвивається, і все більше користувачів хочуть мати з ним зручну та зрозумілу взаємодію. Чати з мовними моделями вже давно перестали бути новинкою, але більшість платформ досі залишаються обмеженими: одна модель, один тип відповіді, обмежений функціонал. Саме тому створення універсального застосунку, який дозволяє в одному місці спілкуватися з різними AI-моделями, генерувати зображення, перемикатися між режимами, мовами та темами – є не просто корисним, а дійсно актуальним рішенням.

AI Hub – це сучасний web-застосунок, який зібрав усе в одному: зручний інтерфейс, підтримку кількох моделей, генерацію зображень без перемикань, адаптацію під мобільні пристрої, темну і світлу теми, а головне – простоту й швидкість. У користувача немає потреби відкривати кілька вкладок, реєструватися на різних платформах або копіювати відповіді між сервісами. Усе працює прямо «тут і зараз».

Проєкт реалізовано на базі сучасного технологічного стеку (Next.js, React, TailwindCSS, OpenRouter API), що дає високу продуктивність, стабільність і можливість подальшого розвитку. Завдяки модульній архітектурі застосунок легко розширити – наприклад, додати збереження чатів, історію сесій або власну систему реєстрації.

Таким чином, AI Hub – це не просто ще один чат з моделями. Це платформа, яка справді зручна, швидка, сучасна та продумана. Вона дозволяє як відпочити й поспілкуватися з ШІ, так і протестувати кілька моделей, згенерувати зображення або працювати з текстами різними мовами. І саме це робить її гідним прикладом того, як має виглядати застосунок майбутнього для взаємодії з AI.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Mitchell M. Artificial Intelligence A Guide for Thinking Humans : Picador Paper; Reprint edition, 2020. 336 p.

2. Scott M. Fraley AI for Beginners: A Practical Guide to Artificial Intelligence in Daily Life, Real-World Applications, Ethics, and Future Trends : Scott M. F., 2025. 219 p.

3. Godbolt M. Frontend Architecture for Design Systems: A Modern Blueprint for Scalable and Sustainable Websites : O'Reilly Media, 1st edition, 2016. 261 p.

4. Hermans K. Mastering Back-End Development: A Comprehensive Guide to Learn Back-End Development : Cybellium Ltd, 2023. 405 p.

5. Pocketbase Pocketbase Relational Database Usage Documentation : [Online]. Available: https://pocketbase.io/docs/collections/.

6. Clark W. REST APIs Step by Step: A Practical Guide with Examples : William Clark, 2025. 387 p.

7. OpenRouter OpenRouter API Documentation : [Online]. Available: openrouter.ai/docs/.

8. Yablonski J. Laws of UX: Using Psychology to Design Better Products & Services : O'Reilly Media, 2nd edition, 2024. 183 p.

9. Pilot L. The Ultimate Handbook of UI/UX Product Design: How to Thrive in the Age of Artificial Intelligence (Artificial Intelligence for UX Designers) : Pilot L., 2024. 192 p.

10. Emmanuel E. Learn Next.js 14, Typescript and Tailwind CSS by Building an Amazing Blog : Emmanuel E., 2024. 206 p.

11. Next.js Next.js 14 Usage Documentation : [Online]. Available: https://nextjs.org/docs/14/getting-started/.

12. NextUI NextUI Componentd Documentation : [Online]. Available: https://www.heroui.com/docs/guide/introduction/.

13. Goldberg J. Learning TypeScript: Enhance Your Web Development Skills Using Type-Safe JavaScript : O'Reilly Media, 1st edition, 2022. 317 p.

14. Qiang H. Grokking Relational Database Design : Manning, 1st edition, 2025. 280 p.

15. Pocketbase Pocketbase Authentication Usage Documentation : [Online]. Available: https://pocketbase.io/docs/authentication/.

16. Хабатілін Д. О. Розробка веб-додатку «AI HUB» для роботи зі штучним інтелектом / Д. О. Хабатілін // Радіоелектроніка та молодь у XXI столітті : матеріали 29-го Міжнар. молодіж. форуму, 23–25 квітня 2025 р. – Харків : ХНУРЕ, 2025. – Т. 5. – С. 69. – УДК 004.774:004.8.