ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Генерація текстів для користувача на основі методів глибокого навчання та хмарних технологій»

на здобуття освітнього ступеня бакалавра
зі спеціальності 122 Комп'ютерні науки
(код, найменування спеціальності)
освітньо-професійної програми Штучний інтелект
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Дар'я АСТАПОВА
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконала:
здобувачка вищої освіти Дар'я АСТАПОВА
група ШІД-41

Керівник: Олександр ЗВЕНІГОРОДСЬКИЙ
науковий ступінь, вчене звання к.т.н., доцент

Рецензент:
науковий ступінь, вчене звання _____
Ім'я, ПРІЗВИЩЕ

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Штучного інтелекту

Ступінь вищої освіти Бакалавр

Спеціальність Комп'ютерні науки

Освітньо-професійна програма Штучний інтелект

ЗАТВЕРДЖУЮ
Завідувач кафедрою Штучного інтелекту

_____ Ольга ЗІНЧЕНКО
«_____» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Астаповій Дар'ї Вячеславівні
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Генерація текстів для користувача на основі методів глибокого навчання та хмарних технологій

керівник кваліфікаційної роботи Олександр ЗВЕНІГОРОДСЬКИЙ к.т.н., доцент,
(Ім'я, ПРІЗВИЩЕ науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2025 р. № 56

2. Строк подання кваліфікаційної роботи «23» травня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, офіційна документація OpenAI API, вимоги до хмарних сервісів

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
Аналіз сучасних мовних моделей GPT (GPT-2, GPT-3.5, GPT-4) та їх особливостей
Дослідження принципів хмарних обчислень і типів хмарних сервісів

Вибір архітектури для реалізації генерації тексту на основі LLM
Інтеграція OpenAI API та формування запитів до моделі
Розробка веб-додатку та користувацького інтерфейсу за допомогою Streamlit
Тестування, розгортання в хмарі та оцінка зручності використання системи

5. Перелік графічного матеріалу: *презентація*
   1. Порівняльна таблиця платформ генерації тексту
   2. Блок-схема логіки обробки запиту
   3. Візуалізація інтерфейсу та роботи веб-додатку

6. Дата видачі завдання «25» лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Дослідження науково-технічної літератури з теми генерації текстів і великих мовних моделей	25.02-15.03.25	Виконала
2	Аналіз архітектури мовних моделей GPT та методів глибокого навчання	16.03-29.03.25	Виконала
3	Вивчення принципів хмарних обчислень та вимог до хмарного розгортання	30.03-15.04.25	Виконала
4	Ознайомлення з OpenAI API, методикою побудови промптів та стилізації тексту	16.04-27.04.25	Виконала
5	Реалізація логіки генерації тексту на основі обраного стилю	27.04-03.05.25	Виконала
6	Розробка інтерфейсу користувача у середовищі Streamlit	04.05-10.05.25	Виконала
7	Хмарне розгортання застосунку за допомогою Streamlit Community Cloud	11.05-13.05.25	Виконала
8	Тестування функціональності сервісу з точки зору користувача	14.05-16.05.25	Виконала
9	Оформлення роботи: вступ, висновки, реферат	17.05-20.05.25	Виконала
10	Створення демонстраційних матеріалів	21.05-23.05.25	Виконала

Здобувачка вищої освіти _____ Дар’я ВЯЧЕСЛАВІВНА
(підпис) *(Ім’я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Олександр ЗВЕНІГОРОДСЬКИЙ
(підпис) *(Ім’я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 68 стор., 34 рис., 39 джерел.

Мета роботи – створення зручного веб-додатку для генерації текстів, що адаптується під потреби користувача (стиль та тема) на основі GPT-3.5-Turbo та розгортання у хмарі.

Об'єкт дослідження – процес генерації текстів за допомогою мовних моделей.

Предмет дослідження – методи генерації текстів.

Короткий зміст роботи: У роботі проведено дослідження можливостей генерації текстів за допомогою трансформерних мовних моделей. Встановлено ефективність їх застосування для створення текстів заданої тематики та стилю. Обґрунтовано доцільність використання хмарних технологій для забезпечення масштабованості та доступності.

Проведено аналіз сучасних підходів до побудови мовних моделей, зокрема методів генерації тексту, стилістичної адаптації та інтеграції з API, методи побудови запитів, що впливають на якість згенерованого контенту.

В ході розробки реалізовано веб-додаток з інтерфейсом на базі фреймворку Streamlit, який дозволяє користувачеві генерувати текст за обраним стилем, використовуючи модель GPT-3.5 Turbo через OpenAI API для обробки запитів. Налаштовано хмарне розгортання з використання Streamlit Community Cloud для забезпечення зручного доступу користувачеві до функціоналу системи.

КЛЮЧОВІ СЛОВА: ГЕНЕРАЦІЯ ТЕКСТУ, ГЛИБОКЕ НАВЧАННЯ, ХМАРНІ ТЕХНОЛОГІЇ, МОВНІ МОДЕЛІ, STREAMLIT, GPT-3.5 TURBO, ВЕБ-ДОДАТОК, PROMPT, OPENAI API

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі кількість текстової інформації стрімко зростає: від наукових публікацій і ділової документації до постів у соціальних мережах і маркетингових текстах. Створення якісного текстового контенту вимагає не лише часу, а й відповідності до стилістичних вимог, що може бути складним для непідготовленого користувача. У цьому контексті все більш актуальними стають інструменти, які здатні генерувати текст відповідно до заданої теми, стилю та мови. Використання мовних моделей на базі глибинного навчання, зокрема GPT, відкриває нові можливості для персоналізованого текстотворення.

Мета роботи. Метою є розробка веб-додатку для генерації текстів різних стилів на основі мовної моделі GPT-3.5 Turbo, який дозволяє користувачеві швидко отримувати граматично правильні, лексично точні та стилістично відповідні тексти. Особливий акцент зроблено на зручності користування інтерфейсом та стабільній роботі сервісу в хмарному середовищі.

Аналіз останніх досліджень. Останні роки спостерігається активне впровадження мовних моделей у різні сфери — від освіти до медіа та бізнесу. В роботах інших дослідникі демонструється високий потенціал моделей GPT у завданнях генерації, редагування та адаптації текстів. Проте більшість готових рішень або складні у використанні, або не враховують мовні особливості, наприклад, українську мову . Саме тому важливим є створення застосунку, який дозволяє отримати стилістично адаптовані тексти українською мовою, із можливістю обирати тему, стиль і тон викладу.

Об’єкт дослідження. Об’єктом виступає процес генерації текстів із використанням мовних моделей глибокого навчання, а предметом — способи реалізації інформаційних систем, що забезпечують зручне створення текстів на основі вибраних стилей користувача.

Предмет дослідження. Предметом дослідження є побудова інформаційної системи генерації тексту з урахуванням обраного стилю,

тематики та мовних вимог користувача, а також використання бібліотеки Streamlit для реалізації веб-інтерфейсу та API OpenAI для підключення мовної моделі.

Ця тема актуальна в умовах зростаючого попиту на створення контенту. У результаті дослідження було створено веб-додаток, який забезпечує простий доступ до генерації якісного тексту з урахуванням стилістичних побажань користувача. Отриманий інструмент здатен полегшити роботу контент-мейкерів, студентів, викладачів та всіх, хто регулярно працює з текстами, зробивши процес генерації змістовним, зручним та індивідуалізованим.

1 ТЕОРЕТИКО-МЕТОДИЧНІ ОСНОВИ ГЕНЕРАЦІЇ ТЕКСТІВ ЗАСОБАМИ ГЛИБОКОГО НАВЧАННЯ

Генерація текстів, або генерація природної мови, є важливою складовою сучасної NLP. Її головне завдання полягає у створенні зв'язного та зрозумілого тексту на основі різноманітних вхідних даних — від текстових запитів до зображень, таблиць чи баз знань. Упродовж останніх десятиліть ця технологія набула широкого застосування в освіті, журналістиці, бізнесі, а також у сфері клієнтського обслуговування.[29]

NLP — напрям штучного інтелекту, який дає змогу комп'ютерним системам аналізувати, інтерпретувати та генерувати людську мову. Ця галузь поєднує елементи обчислювальної лінгвістики, статистичного моделювання та методів машинного і глибокого навчання.[36]

Завдяки відкритим інтерфейсам, такі як: OpenAI API, розробники отримали доступ до сучасних мовних моделей, що здатні створювати тексти на основі коротких текстових запитів. Такі моделі можуть генерувати різноманітні типи тексту — від пояснень і відповідей до програмного коду та структурованих даних. Сучасний прогрес у сфері генерації текстів безпосередньо пов'язаний із розвитком трансформерних архітектур та зростанням можливостей хмарних обчислень, що робить цю технологію доступною для широкого кола користувачів.[5] У наступних розділах ми ознайомимося більш детально із сучасним станом досліджень у сфері генерації текстів, оглянемо популярні мовні моделі, методичні підходи до реалізації генерації текстів та обговоримо основи хмарних обчислень у побудові мовних сервісів.

1.1 Сучасний стан досліджень у сфері генерації текстів

Генеративна модель тексту — це різновид моделей штучного інтелекту, що здатні створювати новий зв'язний текст на основі знань, отриманих у процесі навчання. Така модель є складовою напряму обробки природної мови, який поєднує інструменти лінгвістики та методи штучного інтелекту.[4]

Класифікацію основних завдань NLP наведено (рис.1.1):

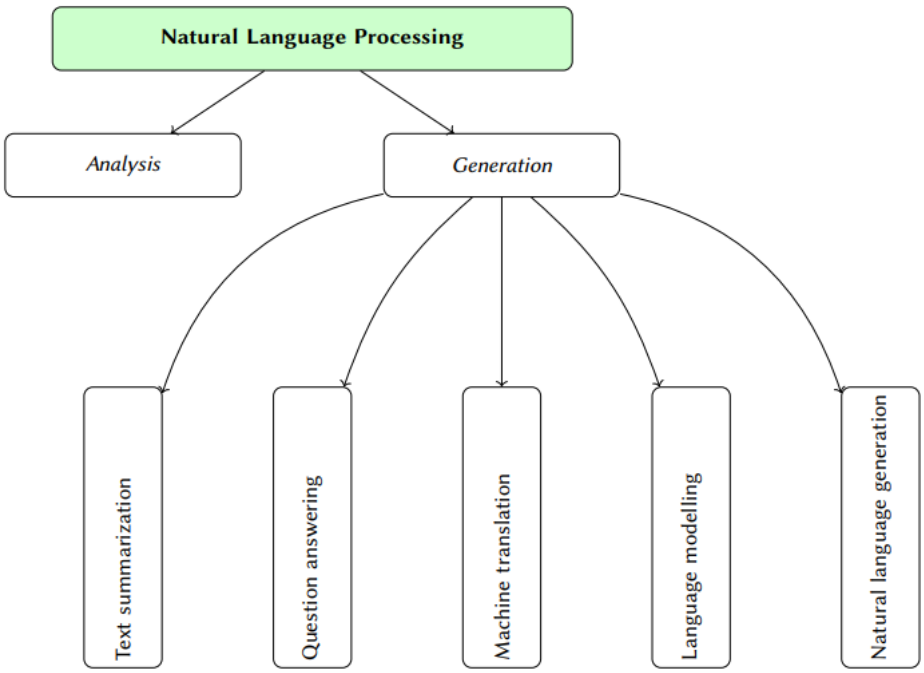

Рис. 1.1 Таксономія популярних завдань обробки природної мови (NLP) для генерації тексту.

Завдяки аналізу великих обсягів тексту такі системи можуть відтворювати логічно пов’язані речення, які відповідають обраній стилістиці та тематиці.[12]

Поява доступу до великих мовних моделей у 2022–2023 роках призвела до значного зростання інтересу до них (рис. 1.2). Це, у свою чергу, актуалізувало необхідність систематизації основних напрямів їх застосування. З цією метою було запропоновано класифікацію (рис. 1.3), яка охоплює п’ять ключових аспектів генерації тексту:[31]

1. За архітектурою нейронної мережі:

- Традиційні архітектури:
  - RNN — рекурентна нейронна мережа для обробки послідовних даних
  - LSTM — покращений варіант RNN для роботи з великими обсягами даних
  - GRU — спрощена версія LSTM із меншою кількістю параметрів

    - CNN — згорткова нейронна мережа, здатна виявляти просторові патерни
- Інноваційні архітектури:
    - Attention-based — моделі з механізмом уваги для вибору релевантних елементів
    - Transformer — архітектура, що реалізує механізм уваги без рекурентності
    - BERT — модель від Google з двонапрямленим механізмом кодування

2. За метриками оцінювання якості:
- Людино орієнтовані:
    - Domain-Expert — оцінювання експертами галузі
- Автоматизовані:
    - BLEU, ROUGE, Cosine Similarity, Content Selection, Diversity Score
3. За сферою застосування: AMR, Language Generation, Speech-to-Text, Script Generation, Machine Translation, Text Summarization, Image Captioning, Shopping Guide, Weather Forecast
4. За мовами генерації:
- Мови з великими ресурсами: англійська, китайська
- Мови з обмеженими ресурсами: бенгальська, корейська, іспанська, гінді, словацька, македонська
5. За типом навчального датасету:
- Тип анотації: розмічені або нерозмічені дані
- Структура: речення, абзац, запитання/відповідь, документ

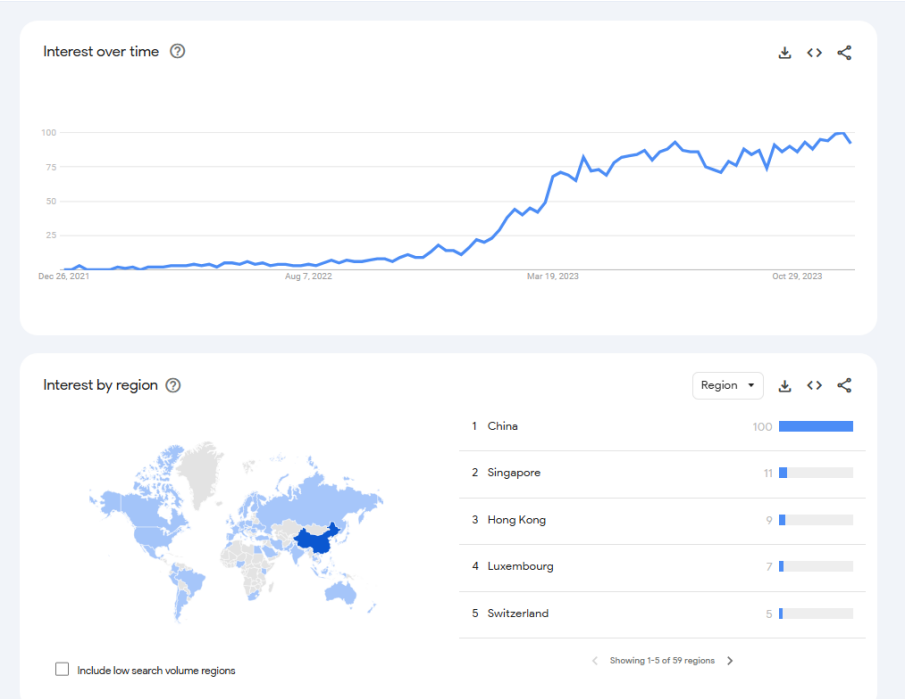

Рис 1.2 Зростання інтересу мовних моделей у користувачів на основі зібраної інформації в GoogleTrends

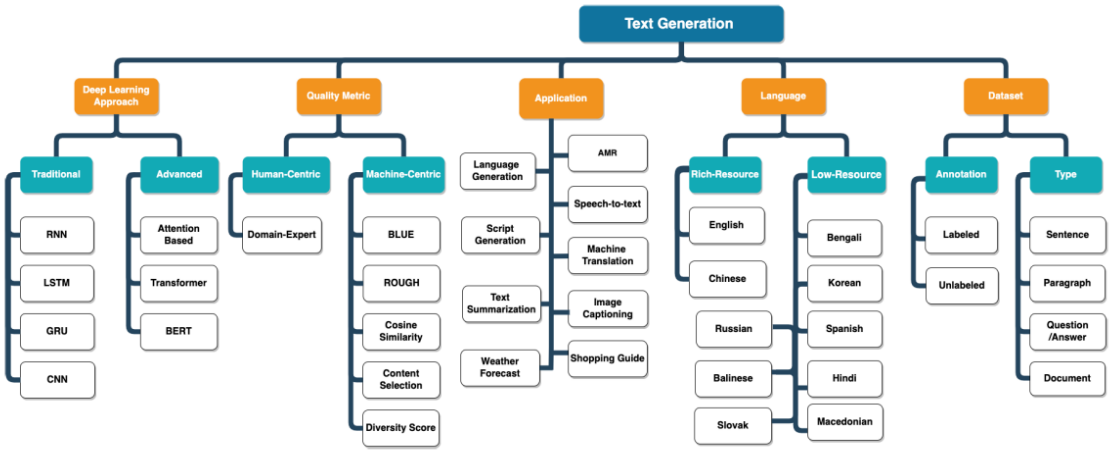

Рис 1.3 Таксономія генерації тексту

Застосування технологій генерації текстів уже охоплює різні сфери: створення чат-ботів, креативний контент, автоматичне складання інструкцій, генерація коду та багато іншого. У дослідницькому середовищі увага зосереджена на вдосконаленні контекстного розуміння, зниженні упередженості моделей, оптимізації ресурсів та підвищенні якості відповідей.

Серед ефективних підходів найбільшу популярність здобули трансформерні архітектури, такі як GPT, BERT, що забезпечують точність і гнучкість у генерації тексту. Також спостерігається активний перехід від

застарілих рішень до новітніх моделей, які ґрунтуються на глибокому навчанні й механізмах уваги.[31]

У 2022 та 2023 роках (рис 1.4) помітним був домінантний вплив новаторських методів генерації тексту, в той час як консервативні та змішані зустрічалися набагато рідше. У 2024 році статистика публікацій, де застосовані інноваційні та комбіновані стратегії, демонструє майже ідентичні значення, проте ці дані не є остаточними, бо охоплюється лише частина цього року. Спостерігається тенденція до збільшення кількості досліджень, що використовують новаторські підходи, серед яких особливе місце займають моделі на основі архітектури трансформер та механізмів уваги.

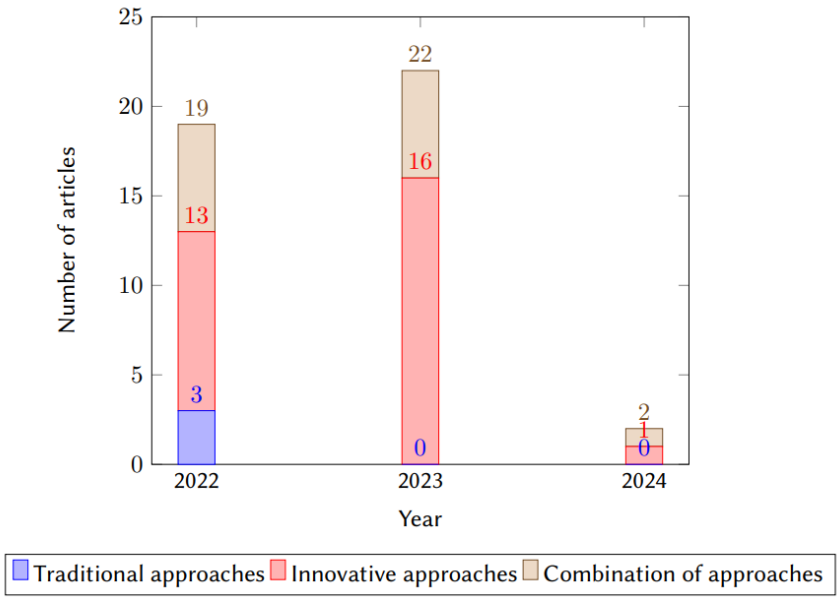

Рис 1.4 Розподіл статей за роками відповідно до категорій підходів до генерації тексту

Підходи до генерації текстів у рамках глибокого навчання можна поділити на три групи:

1. Модель вектор – послідовність — на вхід подається вектор фіксованої довжини, а на виході формується послідовність слів. Такий тип використовують, наприклад, для генерації підписів до зображень.[10]
2. Модель послідовність – вектор — вхідними є дані довільної довжини, результатом є один вектор. Найчастіше застосовується у задачах класифікації.

3. Модель послідовність – послідовність — обидва компоненти, такі як вхід і вихід, мають змінну довжину. Це найпоширеніша структура, яка використовується, зокрема, у машинному перекладі.[7]

1.2 Огляд мовних моделей

Мовні моделі сьогодні активно використовуються у багатьох сферах, де потрібно працювати з текстом. Вони допомагають автоматично перекладати тексти, формувати відповіді в чат-ботах, створювати нові повідомлення або перевіряти орфографію. Основна ідея такої моделі — передбачити наступне слово в реченні, спираючись на вже написаний текст.

Мовна модель — це програма, яка працює на основі машинного навчання. Вона аналізує послідовність слів і визначає, з якою ймовірністю те чи інше слово буде наступним. Наприклад, у реченні «There are two midterms» система може показати, що саме така фраза є правильною і поширеною, а не «Their are two midterms».[16]

Такі моделі мають багато практичних функцій:

- вони можуть автоматично створювати текст
- виправляти помилки у текстах
- розпізнавати мови
- допомагати людям із порушеннями мовлення або письма

Серед мовних моделей особливо виділяються великі мовні моделі (LLM), які здатні працювати з великими обсягами тексту. Більшість із них побудовані на основі трансформерної архітектури — сучасної технології, яка значно покращує здатність системи розуміти контекст та зміст висловлювань. Цей підхід був розроблений фахівцями Google у 2017 році й базується на механізмі самоуваги, що дозволяє моделі зосереджуватись на важливих частинах речення.

Структура трансформера складається з двох основних компонентів — енкодера та декодера, які відповідають за обробку вхідного та вихідного тексту відповідно. Кожен з цих компонентів побудований із послідовних блоків, що

містять механізм багатоголової самоуваги (MSA), двошарову нормалізацію (LN) та повнозв’язну нейронну мережу.[5]

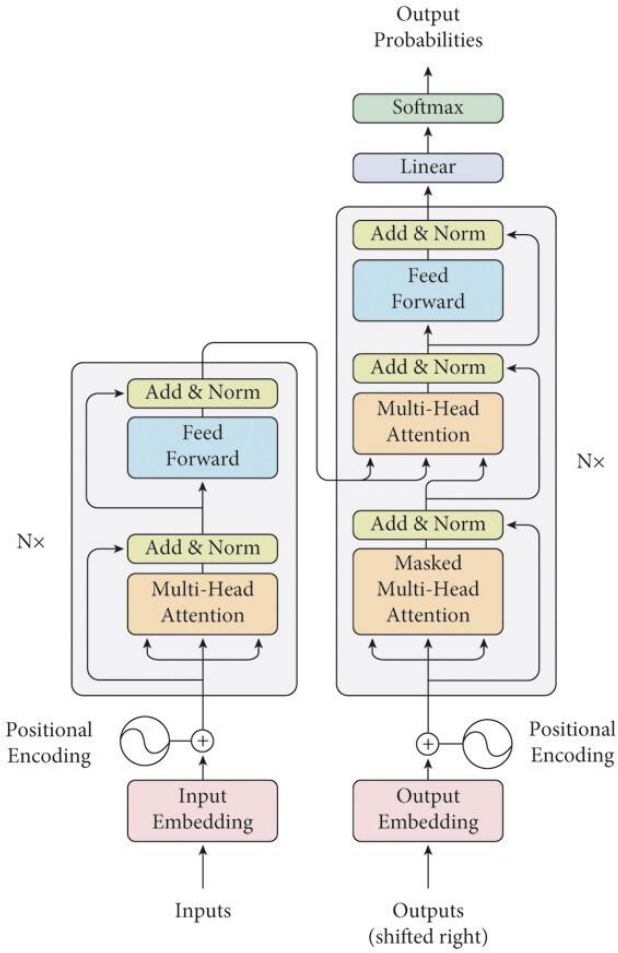

Рис. 1.5 Архітектура моделі трансформаторів

Одним із найвідоміших представників великих мовних моделей є серія GPT (Generative Pre-trained Transformer)**.** Ці моделі спочатку навчаються на великій кількості текстів, а потім здатні генерувати відповіді, створювати статті, перекладати тощо. Перша модель GPT(рис. 1.6) з’явилася у 2018 році, і надалі серія продовжила розвиватися — GPT-2, GPT-3, GPT-4.[18, 25]

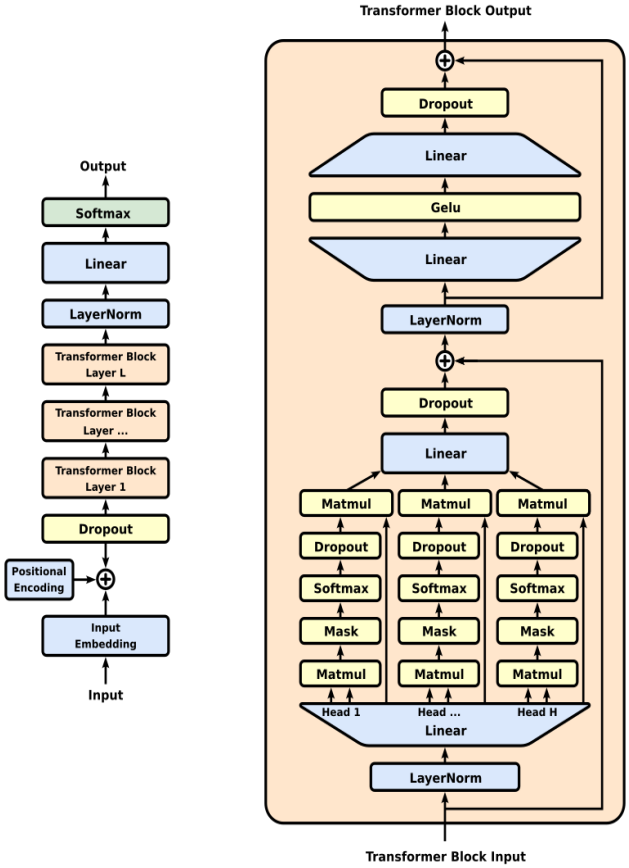

Рис 1.6 Оригінальна модель GPT

Завдяки трансформерній архітектурі GPT-моделі вміють передбачати текст не лише на рівні граматики, а й з урахуванням змісту, стилю та логіки. Проте, оскільки вони навчаються на вже існуючих текстах, тому можуть повторювати помилки та упередження, що містилися в цих даних.[22]

У наступних розділах ми детально проаналізуємо еволюцію та ключові риси трьох ключових моделей серії GPT: GPT-2, GPT-3 та GPT-4. Ці моделі не лише демонструють різні етапи розвитку технології, але й ілюструють важливі концептуальні зміни в підходах до навчання та використання мовних моделей.

1.2.1 Модель GPT-2

Модель GPT-2 є другою версією генеративних попередньо-навчених трансформерів, розроблених компанією OpenAI. Вона була навчена на великому корпусі даних, який охоплював понад 8 мільйонів веб-сторінок, що забезпечило значну варіативність мовного контенту та тематики вхідних даних.[19]

Архітектура GPT-2 складається з низки ключових елементів, які забезпечують її функціональність. Серед головних параметрів моделі варто виділити: словниковий запас розміром 50257 токенів, максимальну довжину вхідної послідовності 1024 токени, розмірність вбудовування — 768, а також 12 головок уваги та 12 шарів трансформера (рис. 1.7). Модель також підтримувала пакетне навчання розміром до 512 прикладів за раз.[38]

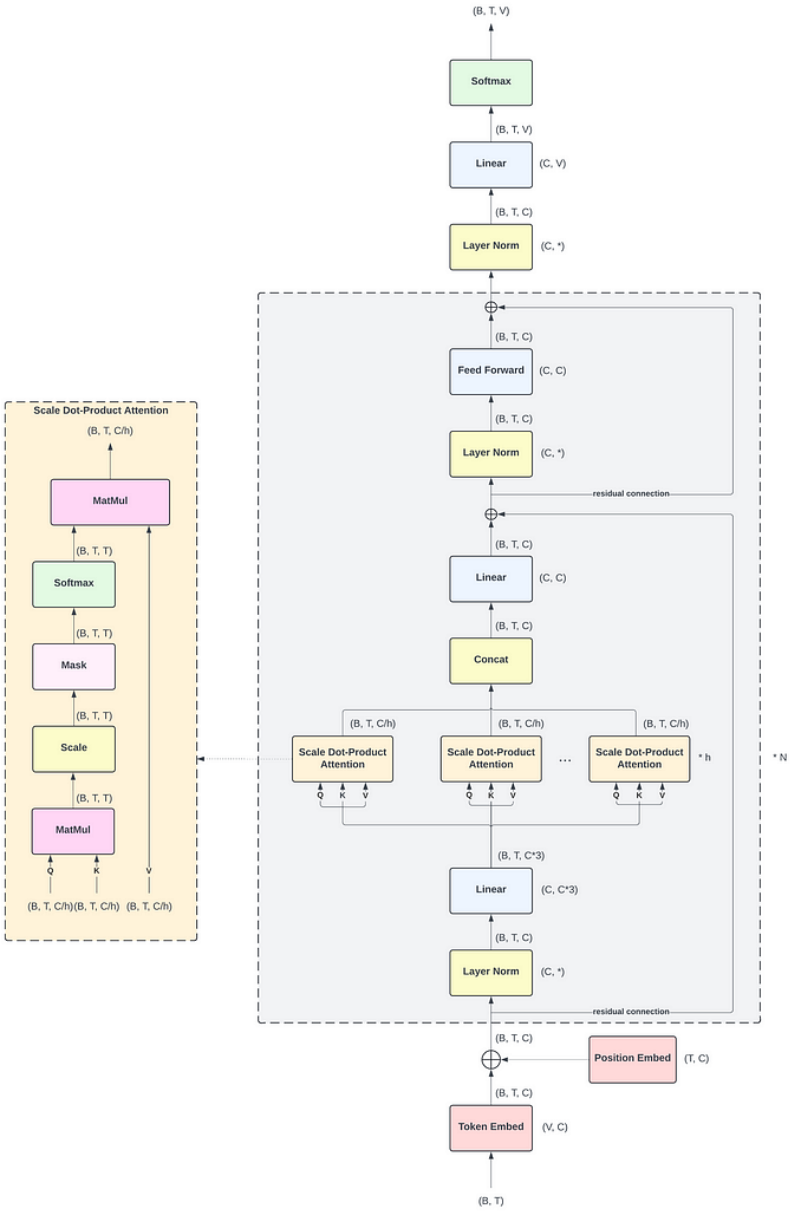

Рис 1.7 Архітектура моделі GPT-2

У процесі генерації мовна модель GPT-2 використовує послідовні блоки механізму уваги та трансформерні шари для обробки вхідного тексту(рис. 1.8). Хоча згенеровані тексти зазвичай граматично правильні, їм може бракувати

логічної зв'язності на рівні кількох абзаців. Проте модель виявила високу гнучкість, адаптуючись до широкого спектру завдань без необхідності додаткового навчання на кожному з них.[35]

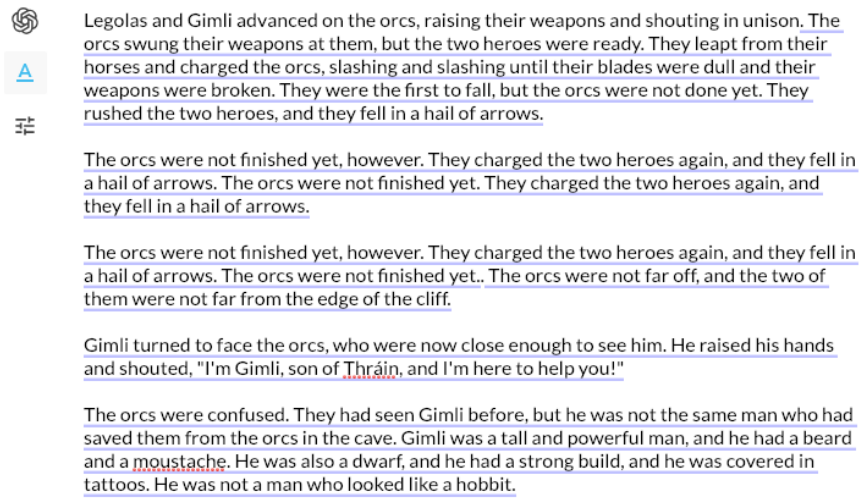

Рис. 1.8 Навички моделі GPT-2 у генерації тексту

Завдяки великому обсягу текстових даних, на яких модель була попередньо навчена, а також використанню методів глибокого навчання, GPT-2 продемонструвала здатність виконувати не лише базове генерування тексту. Вона також може відповідати на запитання, підсумовувати вміст і, у деяких випадках, здійснювати переклад між мовами без додаткового налаштування, покладаючись лише на прогнозування наступного слова в реченні.[35, 28]

Серед прикладів універсальності GPT-2 — здатність до відповіді на запитання, узагальнення змісту та машинного перекладу. Варто зазначити, що навіть без спеціального навчання на іншомовних даних модель змогла здійснювати переклад, наприклад, із французької мови на англійську(рис. 1.9). Це відбувалося, незважаючи на те, що частка французькомовного корпусу у навчальному наборі була надзвичайно малою — лише 10 МБ із загального обсягу у 40 000 МБ.[19]

English reference This re-release, titled The Next Day Extra, was presented in the form of three disks: the original album, unpublished studio sessions and remixes, plus a DVD containing the four clips that have already been unveiled.	**GPT-2 French translation** Les nouvelles re-releases, tout en premier disc, nécessaire de l'album, un studio session et remixes, plus une DVD de l'écran de quelques clips qui ont été déjà échappés.
French reference Un homme expliquait que le fonctionnement de la hernia fonctionnelle qu'il avait reconnaît avant de faire, le fonctionnement de la hernia fonctionnelle que j'ai réussi, j'ai réussi.	**GPT-2 English translation** This release, originally released on three discs: the original album, sessions studio inédites and remixes, a DVD containing the four clips that were released on the previous day.
English reference One man explained that the free hernia surgery he'd received will allow him to work again.	**GPT-2 French translation** Un homme expliquait que le fonctionnement de la hernia fonctionnelle qu'il avait reconnaît avant de faire, le fonctionnement de la hernia fonctionnelle que j'ai réussi, j'ai réussi.
French reference Un homme a expliqué que l'opération gratuite qu'il avait subie pour soigner une hernie lui permettrait de travailler à nouveau.	**GPT-2 English translation** A man told me that the operation gratuity he had been promised would not allow him to travel.
English reference Speaking to an open government conference in London via video link, Mr Kerry said: "There is no question that the President and I and others in government have actually learned of some things that had been happening on an automatic pilot because the ability has been there, going back to World War Two and to the very difficult years of the Cold War, and then, of course, 9/11."	**GPT-2 French translation** Le premier conference de l'édition internationale en ligne sur la vide, le ministre de l'Édition internationale a été vide, lui dit : "Il n'y a qu'une question que le président et moi et tous les autres personnes d'une administration qui a été évident par la fois, par la part de la guerre, par la part de la guer
French reference S'exprimant lors d'une conférence intergouvernementale à Londres par liaison vidéo, M. Kerry a déclaré: "Il est indéniable que le Président, moi-même et d'autres membres du gouvernement avons pris connaissance de certaines choses en mode pilote automatique parce que nous en avions la possibilité, dès la Seconde guerre mondiale et jusqu'aux années difficiles de la Guerre froide, puis bien sûr le 11 septembre."	**GPT-2 English translation** In a conférence between the United States and London, Secretary of State John Kerry said: "It is indeniable that the President, myself and others of the government have been aware of certain certain choices that have been made in the past in order to be able to do certain things in a more automated way."

Рис 1.9 Переклади з англійської на французьку та з французької на англійську, згенеровані за допомогою GPT-2

Мовна модель GPT-2 виявила здатність до генерації правдоподібного тексту, демонструючи при цьому як сильні сторони: універсальність, масштабованість, так і обмеження, такі як: зниження зв'язності на довгих ділянках тексту. Попри це, також підтвердила ефективність підходу попереднього навчання як основи для подальшої генерації текстів за різним призначенням.

1.2.2 Модель GPT-3.5

GPT-3.5 є вдосконаленою версією попередньої мовної моделі, яка була розроблена для підвищення точності відповідей і кращої адаптації до різних текстових завдань. Її попередниця, GPT-3, представлена у 2020 році, мала понад у 100 разів більше параметрів, ніж GPT-2, і вже тоді демонструвала здатність

вирішувати широке коло завдань навіть за обмеженої кількості прикладів у навчанні.[18]

Удосконалена версія GPT-3.5 стала основою для популярного чат-бота ChatGPT(рис. 1.10), який здатен підтримувати діалог на широкий спектр тем.

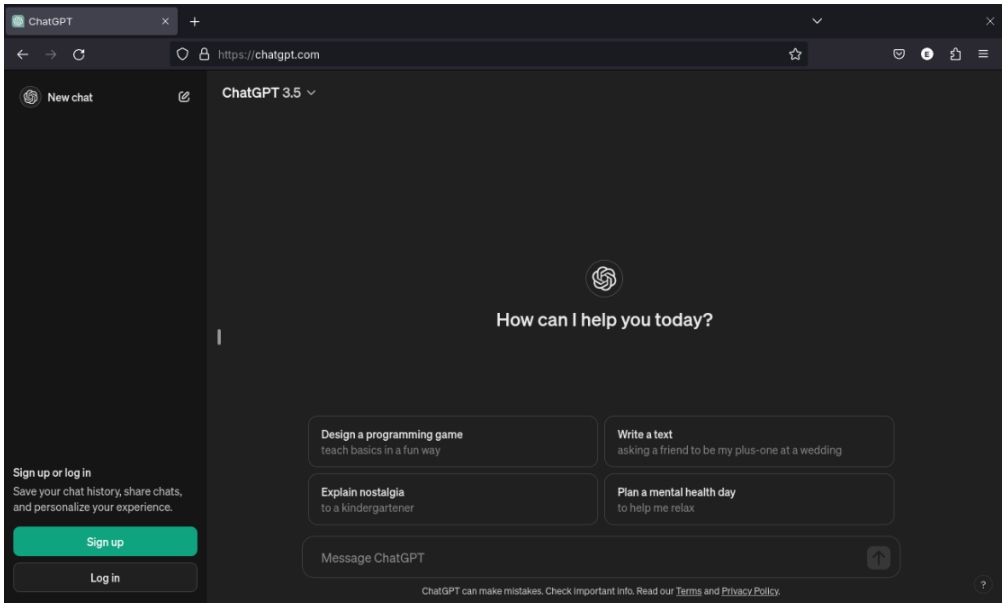

Рис 1.10 Скріншот чат-боту ChatGPT

GPT-3.5 використовує підхід, відомий як RLHF (Reinforcement Learning from Human Feedback) — навчання з підкріпленням на основі зворотнього зв'язку людини. Він дає змогу моделі краще відповідати на запити, враховуючи людські очікування. Попри те, що модель GPT-3.5 має лише трохи менше параметрів, ніж GPT-3 (різниця становить близько 1,3 мільярда), вона демонструє значно кращі результати в окремих завданнях.[21]

Процес RLHF складається з трьох основних етапів:

1. Кероване донавчання :

Людина показує приклади правильних відповідей на задані питання. Ці зразки використовуються для точного налаштування моделі GPT-3, щоби сформувати базову лінію поведінки моделі.

2. Навчання моделі винагород:

Для кожного запиту генерується кілька варіантів, котрі користувачі ранжують відповідно до критерію якості. Ці ранжування є навчальним

матеріалом для спеціальної моделі винагороди, що повинна навчитись передбачати якість відповіді, аналізуючи вхідний текст.

3. Підкріплювальне навчання:

Модель створює відповіді, які підлягають оцінюванню за допомогою моделі винагороди. Ця оцінка застосовується для оновлення стратегії генерації відповіді, використовуючи алгоритм PPO. Це дозволяє поступово покращувати якість згенерованого тексту, відповідно до бажань користувача.

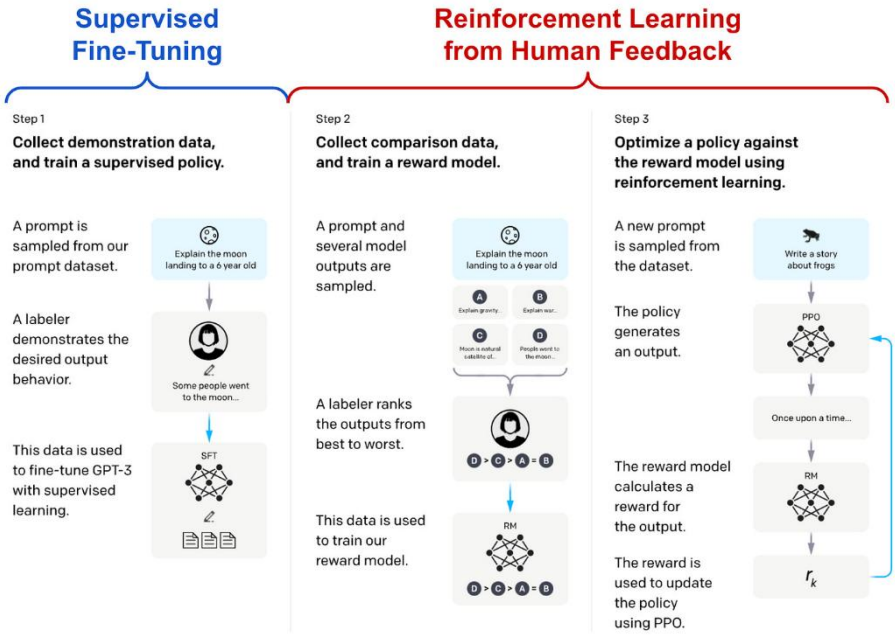

Рис. 1.11 Як працює RLHF для покращення якості генеративних мовних моделей

Однією з переваг GPT-3.5 є її гнучкість і здатність адаптуватися до різноманітних завдань. Наприклад, вона може створювати поезію, як показано у прикладі, де модель без додаткового навчання згенерувала чотири варіанти вірша у стилі Воллеса Стівенса на тему «Тіні на шляху» (рис. 1.12).[18]

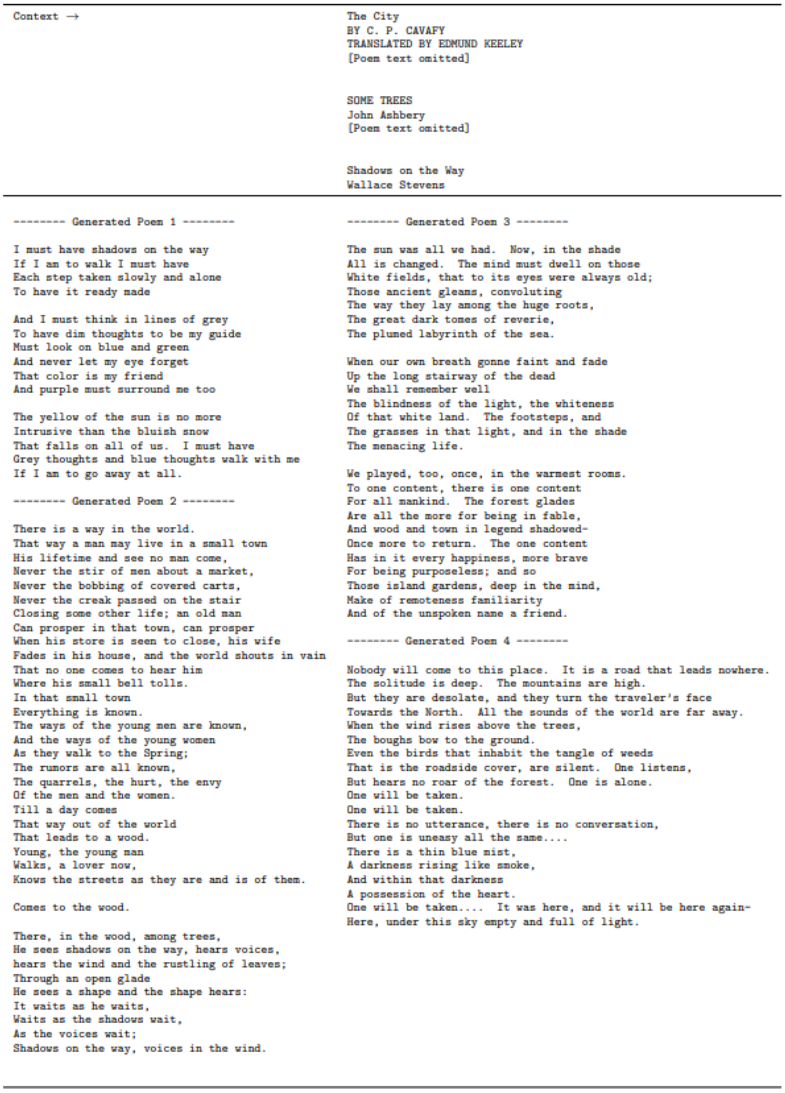

Рис 1.12 Приклад творчого завдання: вірш у стилі Воллеса Стівенса

Модель GPT-3.5 може не лише працювати з текстом, а й виконувати завдання, пов’язані з програмним кодом. Вона добре підходить для створення чат-ботів та віртуальних помічників, які спілкуються з користувачем у форматі діалогу через API.

1.2.3 Модель GPT-4

GPT-4 — це мультимодальна велика мовна модель, створена компанією OpenAI. Вона є четвертим поколінням у серії базових моделей GPT і була розроблена для покращеної обробки як текстової, так і візуальної інформації.[8]

Особливістю GPT-4 є здатність працювати з комбінованим вхідним контентом — користувачі можуть подавати запити, що містять як текст, так і зображення(рис. 1.13). Такий підхід дає змогу виконувати завдання, пов’язані з

аналізом документів, діаграм, скріншотів або інших зображень, де модель формує текстові відповіді, враховуючи візуальний контекст.[26]

Example of GPT-4 visual input:

User — What is funny about this image? Describe it panel by panel.

Source: https://www.reddit.com/r/hmmm/comments/ubab5v/hmmm/

GPT-4 — The image shows a package for a "Lightning Cable" adapter with three panels.

Panel 1: A smartphone with a VGA connector (a large, blue, 15-pin connector typically used for computer monitors) plugged into its charging port.

Panel 2: The package for the "Lightning Cable" adapter with a picture of a VGA connector on it.

Panel 3: A close-up of the VGA connector with a small Lightning connector (used for charging iPhones and other Apple devices) at the end.

The humor in this image comes from the absurdity of plugging a large, outdated VGA connector into a small, modern smartphone charging port.

Table 3. Example prompt demonstrating GPT-4's visual input capability. The prompt consists of a question about an image with multiple panels which GPT-4 is able to answer.

Рис. 1.13 Приклад запиту, що демонструє здатність GPT-4 до обробки візуального вводу

Модель показує високу адаптивність до різних типів запитів, включаючи інтерпретацію зображень, розуміння гумору, логіки й поєднання тексту з візуальними елементами. Проте, як і попередні покоління, GPT-4 не є цілком надійною — можливі помилки в контекстах, де потрібна точність або критично важливе значення має дотримання інструкцій (рис. 1.14).[26]

[GPT-4 answers correctly]	[GPT-4 answers incorrectly]
Can you teach an old dog new tricks? Yes, you can teach an old dog new tricks ← choice You can't teach an old dog new tricks No, you can't teach an old dog new tricks	Son of an actor, this American guitarist and rock singer released many songs and albums and toured with his band. His name is "Elvis" what? Perkins Presley ← choice Elvis Presley His name is Elvis Presley

Рис. 1.14 Приклад правильних і неправильних відповідей GPT-4

Ще одним важливим обмеженням GPT-4 є непрозорість у процесі прийняття рішень. Модель може надавати пояснення своїх відповідей, однак ці пояснення створюються після формування відповіді і не гарантують точності або відповідності дійсним механізмам обробки. У деяких випадках пояснення можуть суперечити попереднім твердженням самої моделі.[32]

На відміну від попередніх моделей, OpenAI не надала публічного доступу до архітектури або параметрів GPT-4. Якщо GPT-2 була повністю відкритою, а GPT-3 — частково документованою, то GPT-4 є повністю закритою моделлю. Це викликало занепокоєння серед дослідників, які вказують на труднощі у вивченні етичних аспектів, упереджень та безпеки моделі. Зокрема, Саша Лучіоні, дослідниця з компанії Hugging Face, назвала GPT-4 «мертвою точкою» для наукової спільноти, оскільки її закритий характер не дозволяє будувати на її основі подальші вдосконалення.[34, 30]

1.3 Основи хмарних обчислень у побудові мовних сервісів

Хмарні обчислення — це модель, яка забезпечує зручний доступ до обчислювальних ресурсів, зокрема серверів, сховищ, мережевих компонентів, додатків та інших послуг(рис. 1.15). Ці ресурси надаються через Інтернет та можуть самостійно керуватися користувачем без необхідності втручання провайдера.[3]

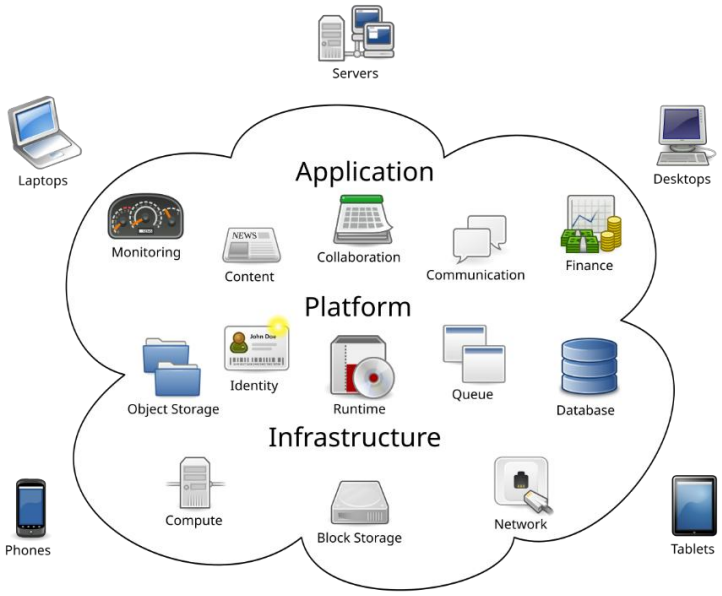

Рис. 1.15 Метафоричне зображення хмарних обчислень

У 2011 році NIST сформулював п’ять ключових характеристик хмарних обчислень. У подальшому, у 2023 році, цей перелік був доповнений і уточнений міжнародною організацією ISO: [14]

- Широкий доступ до мережі — забезпечується можливістю використання ресурсів через стандартні інтерфейси.
- Вимірюваний сервіс — використання послуг контролюється, моніториться та тарифікується.
- Багатокористувацькість — ресурси спільно використовуються, але ізоляція даних між користувачами зберігається.
- Самообслуговування на вимогу — користувач самостійно отримує необхідні ресурси.
- Еластичність — можливість швидкого масштабування ресурсів згідно з потребами.
- Об’єднання ресурсів — спільне використання інфраструктури між користувачами.
- Тип хмарних можливостей — класифікація послуг за рівнем доступу до інфраструктури.

NIST також виділяє три основні моделі хмарних сервісів: IaaS, PaaS та SaaS (рис. 1.16).

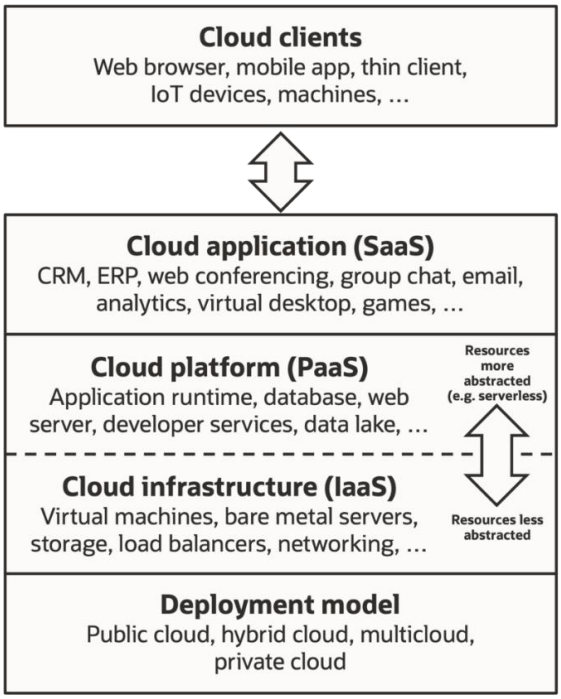

Рис. 1.16 Моделі хмарних послуг, розташовані як шари в стеку

У 2023 році ISO додала нові моделі: NaaS, CaaS, CompaaS, DSaaS: [14]

- SaaS — користувач отримує готове програмне забезпечення через веб або API без потреби керувати інфраструктурою.
- PaaS — надається середовище для створення, тестування та розгортання додатків.
- IaaS — базові обчислювальні ресурси, включно з мережами, сховищами й віртуальними машинами.
- NaaS — управління мережевими з'єднаннями без контролю над фізичною інфраструктурою.
- CaaS — сервіси комунікацій у реальному часі (чат, відео тощо).
- CompaaS — обчислювальні потужності для запуску ресурсомістких програм.
- DSaaS — сервіси зберігання, резервного копіювання, аналітики великих даних.

Моделі розгортання хмари визначають рівень контролю та взаємодії між споживачами й постачальниками:[14]

- Приватна хмара — модель, у якій ресурси використовує лише один споживач і контролює їх самостійно або через третю сторону. Може бути на території CSC або поза нею. Забезпечує високий контроль над безпекою, даними та відповідністю вимогам. Підтримує багатокористувацьке середовище з логічною ізоляцією. Забезпечує еластичність, але потребує резервних потужностей, вартість яких несе CSC.
- Публічна хмара — модель, у якій хмарні ресурси доступні багатьом споживачам через публічну мережу. Керується постачальником і може належати комерційній, державній чи академічній організації. CSC не має контролю над іншими користувачами чи розміщенням даних. Ресурси зазвичай спільні між кількома CSC.
- Гібридна хмара — це модель, яка поєднує приватну та публічну хмари. Вони залишаються окремими, але з'єднані технологіями для переносу даних та застосунків. Може бути керована самою організацією або третьою стороною, розміщена локально чи зовні. Дозволяє використовувати послуги кількох CSP одночасно.
- Хмара спільноти — це модель, де хмарні послуги використовуються лише групою CSC з спільними цілями чи вимогами. Ресурси контролюються одним або кількома учасниками. Може бути розміщена локально або зовні, управлятись учасниками чи третьою стороною. Поєднує вигоди публічної хмари з підвищеним контролем.

Під час використання хмарних сервісів користувачу стають доступні переваги хмарних технологій:[33]

- Високу відмовостійкість — забезпечення доступності даних навіть при збої обладнання
- Економію витрат на інфраструктуру — користувач не турбується про апаратне забезпечення
- Безпеку даних — використання сучасного шифрування
- Високу швидкість обробки — за рахунок оптимізованого обладнання

- Глобальний доступ і мобільність — можливість роботи з будь-якого пристрою
- Простота інтеграції та управління — автоматизація, API, централізоване адміністрування

Попри численні переваги хмарних технологій, варто також враховувати потенційні обмеження та ризики їх використання:

- Ризики втрати контролю над конфігурацією та даними
- Небезпека витоку інформації — зломи, уразливості в API
- Складність міграції між хмарними платформами
- Фінансові ризики — труднощі в прогнозуванні витрат
- Залежність від провайдера — технічна та юридична прив’язаність
- Невизначеність SLA — не всі збої покриваються умовами угоди
- Технічні уразливості та правові ризики зберігання даних

Великі мовні моделі, прикладом яких є модель GPT-4, потребують значних обчислювальних потужностей як для процесу навчання, так і для отримання результатів. Саме хмарні технології забезпечують необхідні обчислювальні ресурси, необхідну для ефективної роботи сучасних мовних моделей.

Розгортання великих мовних моделей потребує ретельного підбору технічних рішень, що враховують обчислювальні та організаційні особливості конкретного застосування:

1. Хмарні платформи:
   - Google Cloud Platform — спеціалізується на машинному навчанні з використанням TPU, що оптимізує навчання LLM
   - Amazon Web Services — лідер ринку з широким спектром послуг, зокрема SageMaker для автоматичного налаштування моделей і масштабування
   - Microsoft Azure — інтеграція з корпоративними сервісами, що забезпечує повноцінну інфраструктуру для розгортання LLM у великих масштабах

- Streamlit — легкий фреймворк для Python, призначений для створення простих і ефективних веб-інтерфейсів до LLM

2. Параметри обчислень:
    - CPU: Економічно ефективний для менших моделей, але недостатній для великих LLM
    - GPU: Ідеальний для масштабних LLM-застосунків завдяки паралельній обробці
    - TPU: Призначений для завдань AI на GCP, пропонує високу продуктивність, але потребує оптимізації
3. Масштабованість:
    - Горизонтальне масштабування: Додавання машин до пулу ресурсів, що добре підходить для LLM через паралелізацію та автоматичне масштабування
    - Вертикальне масштабування: Збільшення ресурсів на одній машині для специфічних завдань, але з обмеженнями за ємністю
    - Комбінація обох: Для більшості розгортань LLM оптимальним є поєднання горизонтального та вертикального масштабування

Оцінка потреб у ресурсах для розгортання моделей у хмарі включає в себе врахування кількох ключових факторів, що дозволяють вибрати оптимальну інфраструктуру та уникнути перевитрат чи недостатнього забезпечення:[24]

- Характеристики моделі: Розмір і тип моделі значно впливають на потреби в ресурсах. Моделі глибокого навчання, зокрема трансформатори, такі як GPT, потребують значно більше обчислювальних потужностей та пам'яті через велику кількість параметрів і шарів. Важливі також розмір пакета та точність обчислень
- Вимоги до обчислень: Час виведення та кількість запитів на секунду визначають, чи потрібно використовувати GPU або TPU для прискорення. Якщо прогнозування в реальному часі не потрібне, достатньо може бути стандартних процесорів

- Вимоги до пам'яті: Моделі з великою кількістю параметрів потребують більше пам'яті для логічного висновку. Також важливий розмір вхідних даних та пакетна обробка
- Вимоги до зберігання: Для великих моделей і кількох версій необхідне значне сховище для вагових коефіцієнтів, даних і журналів. Важливе також врахування швидкості вводу/виводу даних

Ключові проблеми розгортання LLM у хмарі включають:[24]

- Високі обчислювальні вимоги: Для програм реального часу потрібні GPU/TPU для забезпечення низької затримки.
- Місткість сховища: Великий обсяг пам'яті для зберігання контрольних точок, параметрів моделей та результатів
- Обмеження пам'яті: Великі пакети вхідних даних можуть перевищити обсяги пам'яті
- Масштабованість і еластичність: Стратегічне масштабування для обробки коливань навантаження
- Управління витратами: Оптимізація витрат через використання точкових екземплярів і управління ресурсами
- Безпека та відповідність вимогам: Шифрування даних і дотримання нормативних вимог
- Споживання енергії: Зниження впливу на навколишнє середовище через оптимізацію споживання енергії

Отже, хмарні технології відіграють ключову роль у реалізації великих мовних моделей, забезпечуючи необхідні ресурси для їхнього навчання, розгортання та масштабування. Завдяки доступності, гнучкості та швидкості, хмарні платформи дають змогу ефективно впроваджувати мовні сервіси в різних сферах — від автоматизованого обслуговування користувачів до обробки фінансових або медичних даних. З урахуванням постійного розвитку великих мовних моделей, роль хмарних обчислень лише зростатиме, відкриваючи нові можливості для інтеграції штучного інтелекту у прикладні інформаційні системи.

У 2023 році 82 % ринку IaaS займали п’ять провідних провайдерів. Лідером залишалась Amazon (39 %, $54,6 млрд), за нею Microsoft (23 %), далі Google (8,2 %) і Alibaba (7,9 %)(рис. 1.17).[11]

Company	2023 Revenue	2023 Market Share (%)	2022 Revenue	2022 Market Share (%)	2022-2023 Growth (%)
Amazon	54,648	39.0	48,123	39.9	13.6
Microsoft	32,197	23.0	25,889	21.5	24.4
Google	11,454	8.2	9,072	7.5	26.3
Alibaba Group	11,119	7.9	9,222	7.7	20.6
Huawei	5,980	4.3	5,248	4.4	13.9
Others	24,601	17.6	22,943	19.0	7.2
Total	**139,999**	**100**	**120,497**	**100**	**16.2**

Рис 1.17 Світова частка ринку послуг публічної хмари IaaS, 2022–2023 роки (у мільйонах доларів США)[11]

Станом на 2025 рік найбільшими гіперскейлерами залишаються Amazon Web Services, Microsoft Azure та Google Cloud.

1.4 Методичні підходи до реалізації генерації текстів

Одним із ключових компонентів сучасних систем генерації текстів є інтерфейс прикладного програмування, відомого як — API. Він забезпечує взаємодію між окремими програмними модулями, дозволяючи швидко підключати зовнішні сервіси без необхідності створення функціоналу з нуля.[13]

Для використання API розробник отримує спеціальний ключ доступу й надсилає HTTP-запити до відповідних кінцевих точок. У відповідь надходять дані, зазвичай у форматі JSON, які обробляються програмою. Наприклад, при роботі з OpenAI API надсилається текстовий запит, а у відповідь отримується згенерований результат.

Одним із найпоширеніших рішень у сфері обробки природної мови є платформа Hugging Face, що пропонує бібліотеку transformers, сумісну з PyTorch, TensorFlow і JAX(рис. 1.18). Вона включає реалізації популярних

моделей, які можна застосовувати для роботи з текстом, зображенням або аудіо.[39]

```python
from transformers import pipeline

# Load a pre-trained model for sentiment analysis
classifier = pipeline("sentiment-analysis")

# Analyze sentiment of a text
result = classifier("I love using Hugging Face models!")
print(result)
```

Рис. 1.18 Код для аналізу тональності тексту за допомогою попередньо навчених моделей бібліотеки transformers від Hugging Face

Під час роботи з великими мовними моделями важливо враховувати вимоги до обчислювальних ресурсів. Якщо для базових задач достатньо процесорів CPU, то при масштабніших обчисленнях, пов'язаних із глибинним навчанням, доцільно використовувати графічні процесори GPU або навіть спеціалізовані обчислювальні модулі.

При впровадженні мовних сервісів розробники обирають між хмарною (Cloud AI) або локальною (On-Premise AI) інфраструктурою. Хмарні рішення забезпечують масштабованість та гнучкість, але можуть нести ризики щодо конфіденційності. Локальні системи дозволяють повністю контролювати дані, проте потребують більших фінансових вкладень на початковому етапі.[6]

Для роботи з великими мовними моделями критично важливим є конструювання запитів (prompt engineering). Це процес точного формулювання вхідних даних, що дозволяє отримати найбільш релевантні та коректні відповіді. Правильна побудова запиту враховує стиль, тон, інструкції та контекст.[27]

Одним із сучасних підходів до підвищення точності генерації є RAG (Retrieval-Augmented Generation). Його суть полягає в тому, що перед

генерацією модель отримує доступ до зовнішніх джерел — баз даних або документів і формує відповіді на основі актуальної інформації.[37]

Завдяки впровадженню підходу RAG (Retrieval-Augmented Generation) великі мовні моделі, зокрема ChatGPT, демонструють покращення якості відповідей. На відміну від традиційних LLM, які використовують лише наперед задані дані, RAG дозволяє здійснювати пошук актуальної інформації з баз даних, завантажених документів або веб-ресурсів перед генерацією відповіді. Наприклад, ChatGPT може виконувати веб-пошук, пов'язаний із запитом користувача, і надавати відповідь із посиланням на перевірене джерело. Попри це, навіть із використанням додаткових джерел, моделі можуть створювати неточну або вигадану інформацію. Такі помилки в науковій літературі називають «галюцинаціями», хоча цей термін часто критикують за надання штучному інтелекту людських рис. Як альтернативу пропонують поняття «конфабуляція», яке в психології означає несвідоме заповнення прогалин у пам'яті без наміру дезінформувати.[9]

Подальшим розвитком RAG є підхід GraphRAG(рис. 1.19), який поєднує моделі з графами знань для кращого розуміння складних концепцій та логічного зв'язку між даними. Дослідження Microsoft показало ефективність GraphRAG у аналізі новинних статей за 2022-2024 рік.[20]

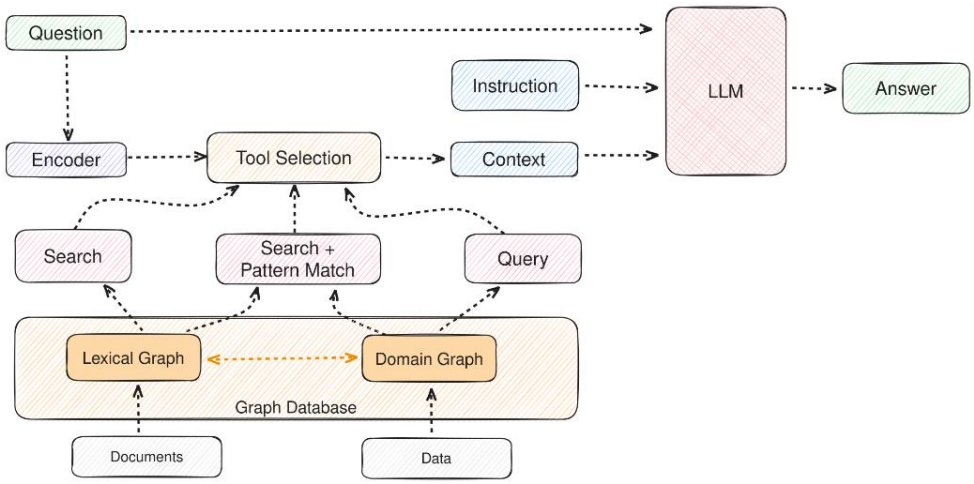

Рис. 1.19 Схема GraphRAG — генерація на основі графа знань

Ще одним важливим параметром генерації є температура — гіперпараметр, що регулює випадковість вибору слів. При низьких значеннях ($t < 1$) модель

надає передбачувані відповіді. При t > 1 — відповіді стають менш точними, але більш креативними.[15]

Загальні режими температури:

- Низька (t < 1.0) — точність та передбачуваність
- Середня (t = 1.0) — баланс між точністю і варіативністю
- Висока (t > 1.0) — більше креативності та непередбачуваності

Top-p та top-k — це методи обмеження вибору токенів під час генерації тексту:

- Top-p враховує токени, сукупна ймовірність яких перевищує заданий поріг p, що дозволяє контролювати ступінь випадковості.
- Top-k враховує k найімовірніших токенів.

Токенізація — ще один базовий етап генерації. Вона перетворює текст у числове представлення, яке модель може обробити. Для цього використовуються алгоритми BPE або WordPiece. Також застосовуються спеціальні токени [MASK], [UNK] для позначення маскованих або невідомих слів.[17]

Для реалізації генерації тексту використовуються різноманітні мови програмування, бібліотеки, хмарні платформи і таке інше, серед найпопулярніших та зручних інструментів для реалізації генерації тексту використовують:

- Python: основна мова для роботи з алгоритмами генерації тексту завдяки наявності корисних бібліотек.
- OpenAI API: зручний фреймворк для створення API, що дозволяє швидко інтегрувати мовні моделі для генерації текстів у особистий майбутній сервіс.
- Render: хмарна платформа для простого розгортання веб-додатків з автоматичним масштабуванням, підтримує Python та інтеграцію з GitHub.
- Streamlit: фреймворк для швидкого створення інтерактивних сервісів на Python, що дозволяє легко візуалізувати моделі генерації тексту та взаємодіяти з користувачами.

2 АНАЛІЗ ТА ДОСЛІДЖЕННЯ ПІДХОДІВ ДО ПОБУДОВИ СИСТЕМ ГЕНЕРАЦІЇ ТЕКСТІВ

2.1 Аналіз функціонування існуючих сервісів генерації тексту

В епоху стрімкого розвитку AI, створення текстів засобами LMM, набула широкого розповсюдження як серед фахівців, так і в загальному користуванні, зокрема серед молоді. У цьому розділі представлено огляд найбільш популярних платформ для генерування тексту, котрі демонструють значний прорив у використанні мовних моделей.

Розглянемо популярний чат-бот GPT, який був розроблений компанією OpenAI та представлений широкій публіці 30 листопада 2022 року. Платформа ChatGPT стала надзвичайно популярною за дуже короткий проміжок часу, набагато перевищуючи популярність будь-яких інших чат-ботів зі штучним інтелектом, випущених для громади.

Наразі у рамках чат-боту ChatGPT використовує свої основні дві мовні моделі: GPT-3.5, яка використовується як безкоштовна версія моделі, та GPT-4-turbo, яка доступна через платну підписку. Також є інші моделі, доступні в ChatGPT Pro або через API: GPT-4.5, GPT-4.1, o3, o4-mini. Наприклад, GPT-4.1 краще справляється з програмуванням, але доступна лише через API.

Для аналізу роботи моделі було надано промт для генерування тексту обсягом 100 слів у інформативно-формальному стилі(рис.2.1).

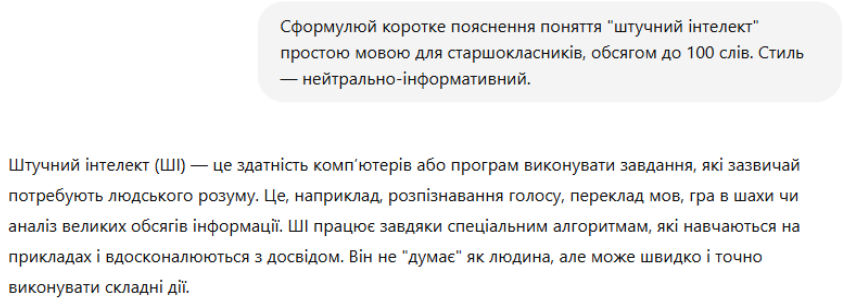

Рис 2.1 Відповідь ChatGPT на запит користувача

Технічні можливості:

- Підтримує різноманітні стилі генерації текстів
- Підтримка декількох моделей.

- Розширений контекст, що дозволяє вести довгі діалоги.

Особливості платформи:

- Можливість задавати багато промптів для налаштування відповіді.
- Створення користувацьких моделей через Custom GPT.
- Підтримує понад 25 мов, включаючи українську.
- Зручний інтерфейс як для чатування, так і для розробників.

Доступ до API:

- API доступний через платформу OpenAI.
- Можливість вказувати у коді температуру, максимальну кількість токенів для інтеграції в сторонні сервіси.

Наступна платформа, яку ми розглянемо – Claude, яка була розроблена компанією Anthropic, як стартап, колишніми розробниками OpenAI.

Claude використовує особисту мовну модель – Claude 3.7 Sonet, яка доступна для користувачів, що мають безкоштовну підписку. Однак, як і ChatGPT, також платформа надає можливість до інших мовних моделей за платну підписку – Claude 3.5 Haiku та Claude 3 Opus.

Технічні можливості:

- Використання декількох моделей.
- Підтримка обробки великого контексту, більшій за ChatGPT.
- Присутнє фільтрування від негативних відповідей та зберігає етичність.

Особливості:

- Безпечна генерація тексту та зниження ймовірності помилок у відповіддях.
- Надає логічні відповідь на питання.
- Застосування в аналітиці та створенні контенту із чітко визначеними цілями.

Доступ до API:

- API надається через платформу розробників, однак доступ може бути обмежений в залежності від деяких факторів.
- Передбачено налаштування параметрів моделі для конкретних завдань.

Останнього, кого ми розглянемо у цьому розділі – Gemini, розроблений компанією Google 21 березня 2023 року у відповідь на зростання популярності ChanGPT.

Чат-бот має також декілька мовних моделей для користування: 2.0 Flash – повністю безкоштовна для користування, справляється зі звичайними задачами та 2.5 Flash, яка доступна як зовсім нова модель для пробного користування.

Технічні можливості:

- Можливість обробки контексту сягає до 1 мільйона токенів.
- Підтримка багатомодальних запитів - текст, зображення, PDF.
- Інтеграція з системою Google, такі як: Google Workspace, Android.

Особливості:

- Орієнтований на інтеграцію із сервісами Google для забезпечення роботи між продуктами.
- Великі можливості для аналізу та генерації тексту завдяки розширеному контексту.
- Акцент на велику обробку даних та високу швидкість обробки.

Доступ до API:

- Доступний через платформу Vertex AI в рамках Google Cloud.
- Інтерфейс API забезпечує можливості інтеграції з іншими продуктами Google і сторонніми сервісами.

Проаналізувавши функціональність цих трьох платформ із можливістю генерувати текст, я виділила важливі пункти та створила таблицю(рис.2.2):

Платформа	Модель	Обсяг контексту	Мовна підтримка	API-доступ	Особливості	Кастомізація	Типові стилі тексту
ChatGPT	Безкоштовно: GPT-3.5, платна: GPT-4-turbo	В залежності від моделі, у GPT-3.5 обсяг 200 тис. токенів	25+ мов	Через офіційну платформу OpenAI	Можливість задавати багато промптів для налаштування відповідей, створення користувацьких моделей, зручний інтерфейс для загального користувача та розробників	Висока (через Custom GPT)	Будь-які стилі, залежно від промпту
Claude	Безкоштовно: Claude 3.7 Sonet, платна: Claude 3.5 Haiku та Claude 3 Opus.	200 тис. токенів	Багатомовна	API надається через платформу розробників, однак доступ може бути обмежений в залежності від деяких факторів	Зниження ймовірність помилок у відповідях, надання логічних відповідей,добрий у застосуванні для аналітики та створення контенту	Обмежена	Формальний, аналітичний
Gemini	2.0 Flash основна модель, 2.5 Flash пробна	до 1 млн. токенів	Багатомовна	Платформа Vertex AI	Великі можливості аналізу, співпрацює з Google-сервісами, велика обробка данних та висока швидкість роботи	Інтеграція через Google Cloud	Універсальний, адаптивний

Рис 2.2 Порівняння трьох платформ

2.2 Дослідження можливостей OpenAI API для генерації тексту

Одним з найвпливовіших інструментів сучасної текстової генерації є великі мовні моделі , які компанія OpenAI надає для використання через API. У даному розділі розглядаються основні можливості OpenAI API для генерації тексту, доступні моделі, методи інтеграції та параметри налаштування, що дозволяють оптимізувати генерацію для різних застосувань.

Розглянемо для прикладу дуже популярні моделі, які часто використовуються розробниками для створення всласного сервісу:

Gpt-3.5 Turbo недорога модель, яка ефективно та швидко генерує текст, основне застосування – чат-боти, підтримках простих задач.

Gpt-4 одна із нових моделей компанії, більш точна та контекстуально глибока, основне застосування – аналітика, генерація складного тексту.

Gpt-4 Turbo вдосконалена версія минулої, оптимізована, дешевша у використанні, основне застосування – великі запити, добрий у кодуванні та у створенні персональних GPT-агентів.

Інтеграція з API здійснюється через HTTP-запити з використанням унікального API-ключа. Розробники можуть взаємодіяти з API як через офіційні бібліотеки для популярних мов програмування, так і через прямі REST-запити. Було наведено базовий приклад взаємодії з API(рис.2.3) Процес автентифікації передбачає отримання API-ключа після реєстрації в системі OpenAI.

```python
import openai

client = openai.OpenAI(api_key="sk-...")

response = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "Ви є асистентом, що надає наукові відповіді."},
        {"role": "user", "content": "Поясніть принцип роботи трансформерів у NLP."}
    ],
    temperature=0.7,
    max_tokens=500
)

generated_text = response.choices[0].message.content
```

Рис. 2.3 Приклад заємодії з API OpenAI

Інтерфейс OpenAI API надає ряд параметрів, що дозволяють контролювати процес генерації тексту:

Temperature:

- 0.0-0.3: Максимально детермінована генерація, підходить для фактичних відповідей
- 0.4-0.7: Збалансована генерація для більшості застосувань
- 0.8-2.0: Підвищена креативність та різноманітність відповідей

Max_tokens — максимальна довжина згенерованої відповіді:

- Визначає довжину згенерованого тексту.
- Впливає на вартість використання API.

Наведемо рекомендовані значення параметру temperature для якісної генерації тексту(рис 2.4):

Тип завдання	Temperature	Характеристика відповідей
Фактичні відповіді	0.0-0.2	Максимально точні та консистентні
Наукові тексти	0.1-0.3	Логічні, послідовні
Освітні матеріали	0.3-0.5	Збалансовані, інформативні, структуровані
Діалогові системи	0.5-0.8	Природні, конверсаційні, з варіативністю
Творчі тексти	0.7-1.0	Різноманітні, оригінальні, висока креативність

Рис. 2.4 Рекомендовані значення, згідно отриманої інформації

У більших випадках якість генерації тексту залежить від формулювання запитів. На основі аналізу взаємодії з API можна виділити основні фактори, що впливають на якість результатів:

1. Чіткість: конкретні та однозначні запити дають більш релевантні відповіді
2. Контекстуалізація: використання системних інструкцій, які задають роль або стиль відповіді.
3. Структурування : поділ складного завдання на етапи або логічні блоки.

При порівнянні моделей встановлено, що GPT-3.5 Turbo забезпечує задовільну якість для більшості стандартних запитів, тоді як GPT-4 демонструє вищу точність та здатність дотримуватися складних інструкцій.

2.3 Аналіз форматів запитів та результатів у різних сценаріях

Застосування мовних моделей генерації тексту передбачає здатність адаптувати результати під бажання користувача: коригування стилю, тональності чи структури тексту. Така варіативність досягається передусім завдяки коректному формулюванню запитів, які називаються промптами, без потреби втручання в технічні параметри моделі. У цьому розділі зробимо аналіз, як змінюється характер згенерованого тексту в залежності від стилістики запиту, а також наведено приклади текстів у різних стилях.

Для аналізу в цьому розділі обрано чотири поширені стилі, які часто зустрічаються при запиті генерування тексту:

- Діловий стиль
- Науковий стиль
- Пояснювальний
- Неформальний стиль

У розділі «2.1 Аналіз функціонування існуючих сервісів генерації тексту» ми розглядали три популярні платформи – ChatGPT, Claude та Gemini. Якщо не використовувати API, де можна впливати на параметри temperature чи max_tokens, моделі автоматично обирають ці параметри, тому зміна стилю йде за рахунок правильно зформульованного запиту. Для данного аналізу нам буде достатньо взяти один чат-бот, я використаю самий розповсюджений: ChatGPT.

Я надала всім однакове запитання: «Що таке штучний інтелект?», - але з використанням певних промптів в залежності від стилю.

1. Діловий стиль:

Промпт: «Сформулюй визначення поняття «штучний інтелект» у діловому стилі, використовуючи формальні конструкції»

Відповідь(рис. 2.5):

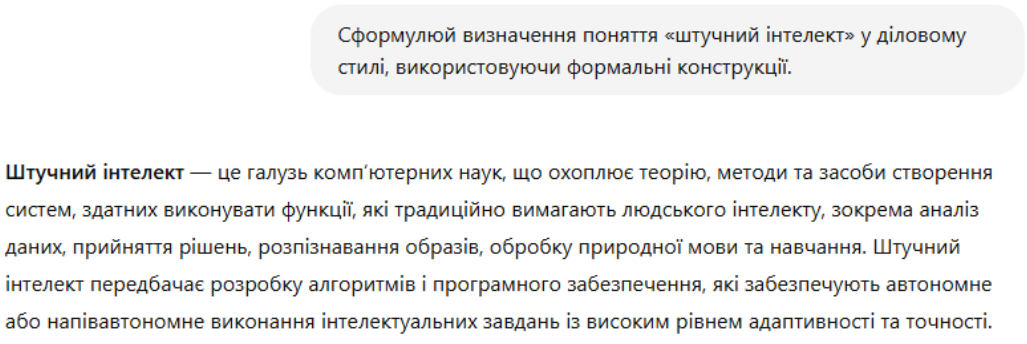

Рис. 2.5 Відповідь у діловому стилі

У відповіді дотримано офіційного стилю, структура чітка, використано терміни «галузь комп'ютерних наук», «автономне виконання завдань», що відповідає діловому стилю. Такий текст можна використовувати для звітів, презентацій та технічної документації.

2. Науковий стиль:

Промпт: «Поясни, що таке штучний інтелект, у науковому стилі, з використанням відповідної термінології»

Відповідь(рис. 2.6):

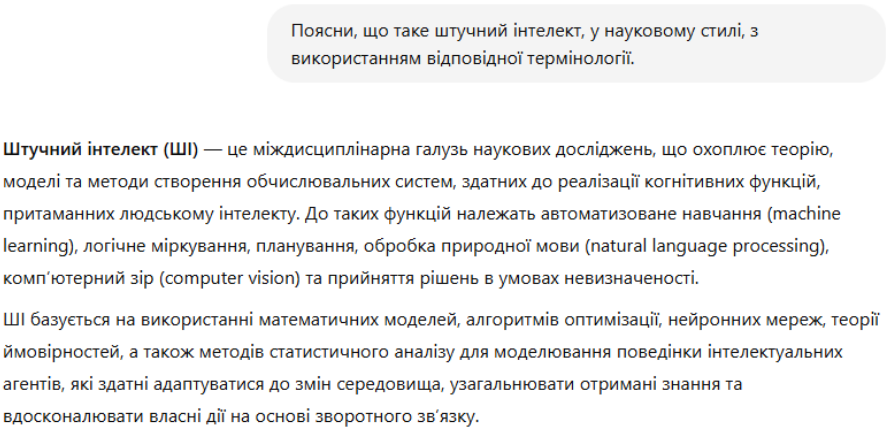

Рис. 2.6 Відповідь у науковому стилі

У відповіді текст наповнений термінами, англомовними словами «machine learning», «computer vision», які розкривають значення перекладенних слів, сама структура відповідає науковим публікаціям.

3. Пояснювальний стиль:

Промпт: «Поясни, що таке штучний інтелект, простими словами, як для учня старших класів.»

Відповідь(рис 2.7):

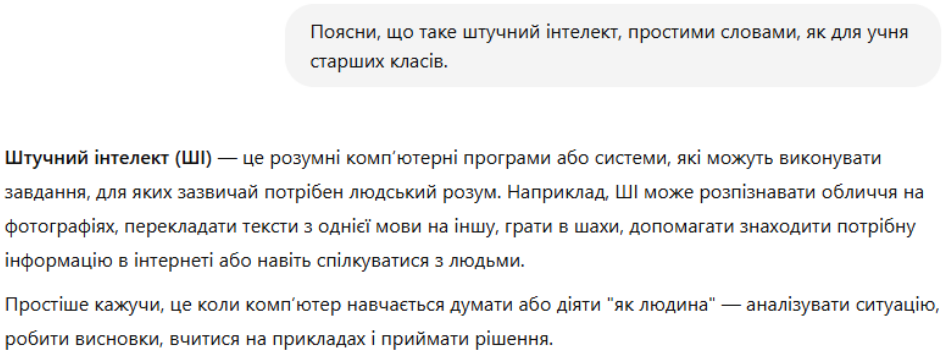

Рис. 2.7 Відповідь у пояснювальному стилі

Текст орієнтований на широке коло користувачів, має приклади з повсякденного життя, присутня проста лексика. Підходить школярам, які тільки вивчають поняття штучного інтелекту і бажають використати інформацію у рефераті.

4. Неформальний стиль:

Промпт: «Поясни, що таке штучний інтелект, у неформальному стилі, якби ти розповідав це другові.»

Відповідь(рис. 2.8):

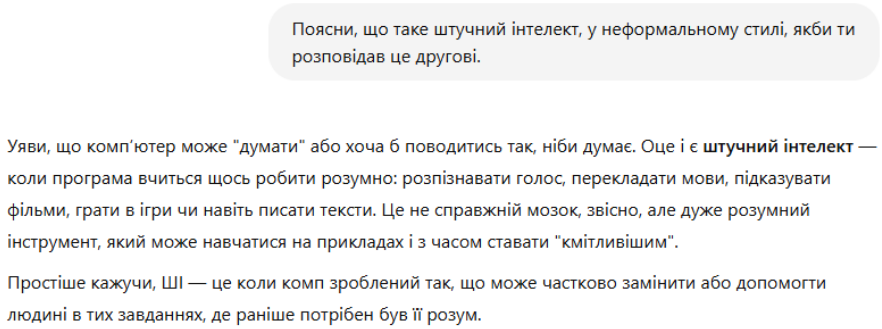

Рис 2.8 Відповідь у неформальному стилі

Текст відтворює живе мовлення, містить розмовні вислови, риторичні прийоми та яскраво виражену емоційність. Такий формат підходить у розмові з другом.

Отже, внаслідок проведеного аналізу було зроблено висновки, що один і той же запит здатний бути заданним різними стилями, залежно від того, як сформульовано промт та задані параметри генерації.

2.4. Оцінка якості згенерованих текстів

Для аналізу якості результатів, згенерованих різними стилями, використовуються метрики. Вони дозволяють оцінити, наскільки близький текст до еталонного варіанта. В цьому дослідженні застосовано три поширені метрики: BLEU, ROUGE та METEOR. Для прикладу візьмемо відповідь ChatGPT, яку він надав у науковому стилі.

2.4.1 Метрика BLEU

Метрика BLEU використовується для порівняння кандидата на переклад з одним або кількома еталонними перекладами. Результат знаходиться в діапазоні від 0 до 1, де бал ближчий до 1 вказує на якісні переклади.[23]

Еталон – текст, який взятий з певних джерел: «Штучний інтелект (ШІ) — це галузь інформатики, яка займається розробкою інтелектуальних машин, здатних виконувати завдання, які зазвичай потребують людського інтелекту»

Кандидат – текст, який попередньо був згенерований нейромережою: «Штучний інтелект (ШІ) — це міждисциплінарна галузь наукових досліджень, що охоплює теорію, моделі та методи створення обчислювальних систем, здатних до реалізації когнітивних функцій, притаманних людському інтелекту»

Нижче представлено лістинг 2.1 з використанням метрики BLEU, який оцінює вищезазначений текст.

Лістинг 2.1 Використання метрики BLEU

```
from nltk.translate.bleu_score import sentence_bleu, SmoothingFunction
reference = [["Штучний", "інтелект", "це", "галузь", "інформатики", "яка", "займається", "розробкою", "інтелектуальних", "машин"]]
candidate = ["Штучний", "інтелект", "це", "міждисциплінарна", "галузь", "наукових", "досліджень", "що", "охоплює", "теорію"]
score = sentence_bleu(reference, candidate, smoothing_function=SmoothingFunction().method1)
print(f"BLEU score: {score:.2f}")
```

Метрика оцінила якість на 0.11, що означає низьку лексичну схожість. Такий результат пояснюється тим, що ChatGPT використовував інше формулювання та синоніми при збереженні загального змісту. Демонструється обмеження BLEU при оцінці вільної генерації, де роль відіграє сенс, а не відповідність слів.

2.4.2 Метрика ROUGE

Метрика ROUGE вимірює збіг між фразами з оригінального тексту та згенерованого. Метрики мають значення від 0 до 1, до того ж вищі бали вказують на високу схожість між автоматично створеною анотацією та посиланням. ROUGE-1 порівнює окремі слова, тим часом як ROUGE-L враховує довжину спільної послідовності слів.

Нижче представлено лістинг 2.2 з використанням метрики ROUGE, який оцінює той же самий текст.

Лістинг 2.1 Використання метрики ROUGE

```
from rouge_score import rouge_scorer
reference = (
    "Штучний інтелект (ШІ) — це галузь інформатики, яка займається розробкою інтелектуальних машин,"
    "здатних виконувати завдання, які "
    "зазвичай потребують людського інтелекту."
)
candidate = (
    "Штучний інтелект (ШІ) — це міждисциплінарна галузь наукових досліджень, "
    "що охоплює теорію, моделі та методи створення обчислювальних систем, "
    "здатних до реалізації когнітивних функцій, притаманних людському інтелекту."
)
```

```
scorer = rouge_scorer.RougeScorer(['rouge1', 'rougeL'], use_stemmer=True)
scores = scorer.score(reference, candidate)
print("ROUGE-1:")
print(f" Precision: {scores['rouge1'].precision:.2f}")
print(f" Recall:    {scores['rouge1'].recall:.2f}")
print(f" F1 Score:  {scores['rouge1'].fmeasure:.2f}\n")
print("ROUGE-L:")
print(f" Precision: {scores['rougeL'].precision:.2f}")
print(f" Recall:    {scores['rougeL'].recall:.2f}")
print(f" F1 Score:  {scores['rougeL'].fmeasure:.2f}")
```

Результат, який надала метрика: ROUGE-1: Precision: 0.00 Recall: 0.00 F1 Score: 0.00 ROUGE-L: Precision: 0.00 Recall: 0.00 F1 Score: 0.00. Це означає, що між оригінальним тектом та згенерованним не знайшлося однакового слова чи послідовності слів. Така ситуація поширена при генерації текстів у науковому стилі, оскільки є високий рівень варіативності мовних конструкцій.

2.4.3 Метрика METEOR

Останню метрику, яку ми розглянемо для оцінки нашого згенерованого тексту – METEOR. Ця метрика здійснює оцінювання точності, враховуючи відповідність до синонімів, морфологічні особливості та порядок слів. На відміну від BLEU, метрика демонструє більшу чутливість до граматичної структури, у зв'язку з чим часто використовується в задачах, де необхідно генерувати текст.

Нижче представлено лістинг 2.3 з використанням метрики METEOR, який оцінює той же самий текст.

Лістинг 2.3 Використання метрики METEOR

```
from nltk.translate.meteor_score import meteor_score
reference = """Штучний інтелект (ШІ) — це галузь інформатики, яка займається розробкою інтелектуальних машин,
```

здатних виконувати завдання, які зазвичай потребують людського інтелекту.""".split()

candidate = """Штучний інтелект (ШІ) — це міждисциплінарна галузь наукових досліджень, що охоплює теорію,

моделі та методи створення обчислювальних систем, здатних до реалізації когнітивних функцій, притаманних людському інтелекту.""".split()

score = meteor_score([reference], candidate)

print(f"METEOR Score: {score:.2f}")

Результат, який надала метрика: METEOR Score: 0.36. Це означає, що згенерований текст має низьку схожість з оригіналом за лексикою та грааматикою, але при цьому зберігає загальну суть і зміст.

3 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ГЕНЕРАЦІЇ ТЕКСТІВ

3.1 Вибір архітектури застосунку

Для розробки сервісу генерації текстів була обрана клієнт-серверна архітектура з використанням платформи Streamlit, яка дозволяє створювати прості веб-додатки для користувачів, що можуть інтерактивно взаємодіяти з системою через зручний інтерфейс. Завдяки інтеграції з OpenAI API, сервіс отримує доступ до мовних моделей, які можуть генерувати тексти в різних стилях, відповідно до введеного запиту.

Система розгортається в хмарному середовищі Streamlit Cloud, що забезпечує стабільність, а також можливість масштабування. Весь код зберігається в репозиторії GitHub із можливістю редагування, а додаток автоматично розгортається через Streamlit, що забезпечує безперервну роботу сервісу без необхідності в додаткових серверних налаштуваннях.

Блок-схема (рис. 3.1) демонструє основні етапи роботи сервісу:

1. Користувач вводить запит через веб-інтерфейс Streamlit.
2. Запит передається на сервер, де виконується обробка.
3. Система звертається до OpenAI API для генерації тексту.
4. Згенерований текст повертається користувачу через вікно інтерфейсу.

Рис. 3.1 Блок-схема обробки запиту користувача

3.2 Розробка системи генерації текстів

У цьому розділі ми розглянемо основні етапи розробки системи генерації тексту, яка дозволяє користувачеві генерувати тексти різних стилів. Система реалізована за допомогою бібліотеки Streamlit для побудови інтерфейсу, а також API OpenAI для генерації текстів.

Основні елементи системи:

1. Вибір стилю тексту: користувач обирає один із запропонованих стилів для створення тексту. Стилі включають офіційний, креативний, науковий, академічний та інші. Це дає змогу користувачам отримувати текст у потрібному контексті.
2. Ввід тексту: користувач вводить текст, до якого хоче, щоб була застосована певна стилістика. Текст надсилається для обробки на сервер, де відбувається генерація з урахуванням вибраного стилю.
3. Обробка запиту та генерація тексту: система формує запит до OpenAI API, використовуючи вибраний стиль та вхідний текст користувача. GPT-3.5 Turbo генерує відповідь, яка повертається у вигляді стилістично адаптованого тексту.
4. Виведення результату: згенерований текст відображається у вигляді результату на інтерфейсі, а користувач може скопіювати його або зберегти у файл.

3.3 Інтеграція з OpenAI API

Інтеграція з OpenAI API є важливою частиною створення сервісу генерації текстів, оскільки саме через неї здійснюється зв’язок між користувацьким запитом і мовною моделлю GPT-3.5 Turbo. В цьому проекті обрано бібліотеку openai, яка забезпечує зручну взаємодію з API OpenAI.

На початковому етапі відбувається ініціалізація клієнта, через якого здійснюється взаємодія з API OpenAI. Для забезпечення безпеки підключення використовується механізм збереження ключа доступу у захищеному середовищі за допомогою Streamlit Secrets, що дозволяє уникнути публічного

розголошення конфіденційної інформації у коді. Після введення користувацького запиту система формує prompt, який відповідає обраному стилю генерації тексту. Разом із цим prompt моделі передаються інструкції, в яких вказано, що вона має виконувати роль професійного мовного редактора. Такий підхід дозволяє отримати текст, адаптований відповідно до заданого стилю, з дотриманням правила правопису української мови.

Промпт складається з двох частин:

- Стилістична інструкція: обраний користувачем стиль (офіційний, креативний, науковий, академічний, формальний, мотиваційний, розповідь)
- Змістова частина: тема або вхідний текст, введений користувачем

При зверненні до мовної моделі gpt-3.5-turbo використовується наступний текстовий запит, який складається з двох системних інструкцій і запиту користувача на основі запису з Log OpenAI API, де вказується академічний стиль тексту:

Системне повідомлення 1: «Ти професійний мовний редактор, який перетворює вхідний текст на граматично, лексично та стилістично бездоганний український текст у заданому стилі. Уникай русизмів, кальок і нетипових конструкцій.»

Системне повідомлення 2: «Усі речення повинні бути логічно зв’язані; не забувай використовувати відмінки та пунктуацію згідно з правилами українського правопису.»

Запит користувача: «Створи текст в академічному стилі: формулюй думки логічно, об’єктивно, з використанням складнопідрядних речень і термінів, властивих студентським і дипломним роботам; уникай емоційної лексики.»

Такий підхід дозволяє чіткому розділенню ролей: перші два повідомлення задають загальні правила обробки тексту, а третє — конкретизує стиль і тему.

Усі компоненти запиту надсилаються до методу chat.completions.create, де незмінні повідомлення system та змінні повідомлення user поєднуються з

параметрами max_tokens і temperature. Після генерації отриманий текст зберігається у st.session_state і відображається у спеціальному полі веб-додатку.

Ця частина логіки не охоплює побудову інтерфейсу чи стилістичну обробку тексту у веб-додатку, а відповідає виключно за зв’язок між локальною системою і зовнішньою моделлю штучного інтелекту, що й зображено на лістингу 3.1

Лістинг 3.1 Інтеграція OpenAI

```
from openai import OpenAI
client = OpenAI(api_key=st.secrets["openai_api_key"])
response = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {"role": "system", "content": "Ти професійний мовний редактор, який перетворює вхідний текст на граматично, лексично та стилістично бездоганний український текст у заданому стилі. Уникай русизмів, кальок і нетипових конструкцій."},
        {"role": "system", "content": "Усі речення мають бути логічно зв’язані, не забувай використовувати відмінки у словах та пунктуацію в реченнях згідно правопису української мови."},
        {"role": "user", "content": full_prompt}
    ],
    max_tokens=1000,
    temperature=0.8
)
# Отриманий результат зберігається у session_state
st.session_state.generated_text = response.choices[0].message.content
```

Ефективна взаємодія з мовною моделлю здійснюється через промпти — текстові інструкції, які об’єднують вибраний користувачем стиль із його темою або вхідним фрагментом. У реалізованій системі промпти формуються динамічно за такою шаблонною структурою:

3.4 Розробка функцій системи

Розробка функцій сервісу генерації текстів передбачала створення повноцінного інтерактивного інтерфейсу, який забезпечує зручну взаємодію користувача з мовною моделлю GPT-3.5 Turbo. Основний функціонал було реалізовано за допомогою бібліотеки Streamlit, що дає змогу ефективно побудувати веб-додаток з мінімальними витратами на ресурси.

Після завантаження сторінки користувач бачить привітальне повідомлення та коротку інструкцію щодо користування сервісом. У інтерфейсі реалізовано основні компоненти, такі як: випадаючий список стилів тексту, текстове поле для введення запиту, кнопки для генерації та очищення вмісту, а також блок для відображення результату, де можна зберегти текст. З метою збереження стану змінних під час взаємодії з інтерфейсом застосовується st.session_state, що дозволяє уникнути втрати даних при повторному завантаженні.

Функція генерації ініціюється натисканням кнопки «Згенерувати текст», після чого формується prompt, який містить системні інструкції та стилістичне уточнення залежно від вибору користувача. Згенерований текст виводиться у текстовому полі й доступний для копіювання або завантаження у вигляді окремого .txt-файлу. Для зручності також реалізовано кнопку очищення, яка скидає як вхідні, так і вихідні дані.

Загальна структура коду є модульною, а всі елементи логічно розділені, що забезпечує простоту подальшого масштабування або адаптації під інші задачі. Нижче наведено повний лістинг 3.2 реалізованого коду:

Лістинг 3.2 Повний код main.py

```
import streamlit as st
from openai import OpenAI
client = OpenAI(api_key=st.secrets["openai_api_key"])
st.title("Стильовий генератор тексту")
st.write("""
Вітаю!
Я - генератор, який допоможе згенерувати вам текст за обраним стилем!
```

```
_____________________________

1. Оберіть стиль тексту, який вам потрібний
2. Напишіть запит у спеціальне поле
3. Натисніть "Згенерувати"
4. Готово!

_____________________________

Ви отримаєте граматично правильний та стильово адаптований результат українською мовою.
Автор: Астапова Дар'я
""")
if "user_text" not in st.session_state:
    st.session_state.user_text = ""
if "generated_text" not in st.session_state:
    st.session_state.generated_text = ""
style = st.selectbox(
    "Оберіть стиль тексту:",
    ["Офіційний", "Креативний", "Науковий", "Академічний", "Формальний", "Мотиваційний", "Розповідь"]
)
prompt_styles = {
    "Офіційний": "Створи текст у формальному, офіційному чи офіційно-діловому стилі:",
    "Креативний": "Створи текст у форматі для використання рекламних банерів, сторітелінгу, презентацій, неймінгу, підписів для соцмереж і закадрових текстів вірусних роликів. Це можуть бути також маленькі слогани:",
    "Науковий": "Створи текст у науковому стилі: формальний, об’єктивний, з точними формулюваннями, нейтральною лексикою та безособовими конструкціями:",
```

```
    "Академічний": "Створи текст в академічному стилі: формулюй думки логічно, об’єктивно, з використанням складнопідрядних речень і термінів, властивих студентським і дипломним роботам. Уникай емоційної лексики.",
    "Формальний": "Створи текст простою, формальною мовою. Якщо просять пояснити щось, відповідай формально мовою також:",
    "Мотиваційний": "Напиши цей текст як мотиваційне звернення:",
    "Розповідь": "Напиши цей текст у форматі вигаданої історії, це може бути казка чи фанфікшн:"
}
# Поле вводу керується через session_state
st.session_state.user_text = st.text_area(
    "Введіть свій текст:",
    value=st.session_state.user_text,
    height=200,
    key="text_area"
)
if st.button("Згенерувати текст") and st.session_state.user_text.strip():
    full_prompt = f"{prompt_styles[style]} {st.session_state.user_text.strip()}"
    with st.spinner("Генерується..."):
        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "system", "content": "Ти професійний мовний редактор, який перетворює вхідний текст на граматично, лексично та стилістично бездоганний український текст у заданому стилі. Уникай русизмів, кальок і нетипових конструкцій."},
                {"role": "system", "content": "Усі речення мають бути логічно зв’язані, не забувай використовувати відмінки у словах та пунктуацію в реченнях згідно правопису української мови."},
                {"role": "user", "content": full_prompt}
```

```
            ],
            max_tokens=1000,
            temperature=0.8
        )
        st.session_state.generated_text = response.choices[0].message.content
if st.button("Очистити"):
    st.session_state.user_text = ""
    st.session_state.generated_text = ""
    st.experimental_rerun()
        if st.session_state.generated_text:
            st.subheader("Згенерований текст:")
            st.text_area("Результат", value=st.session_state.generated_text, height=300,
key="result_area")
            st.download_button("Зберегти текст",
        st.session_state.generated_text, file_name="zghenerovanyi_tekst.txt")
        else:
                st.warning("Будь ласка, введіть текст.")
```

3.5 Вибір хмарної платформи та особливості перенесення системи

У процесі створення сервісу генерації текстів постало завдання вибору хмарної платформи, яка б відповідала критеріям простоти інтеграції з Python, підтримувала роботу з бібліотекою Streamlit, а також забезпечувала зручне розгортання без складної серверної структури. Між наявними рішеннями було обрано Streamlit Cloud. Він забезпечує просте, швидке та безкоштовне розгортання проєктів через GitHub-репозиторій, що набагато спрощує процес розгортання для невеликих застосунків із відкритим кодом.

Порівняно з альтернативними хостингами, такими як Render, Vercel чи Heroku, Streamlit має кілька переваг:

- відсутність потреби в додатковій конфігурації веб-сервісу
- автоматичне визначення залежностей на основі requirements.txt

- просте управління Streamlit Secret через веб-інтерфейс
- швидке перезавантаження додатку після внесення змін у код репозиторію

Таким чином, підхід, який було обрано дозволяє уникнути складної серверної структури, зберігаючи при цьому надійність, доступність і масштабованість сервісу. Streamlit Cloud добре підходить для розгортання демонстраційних проєктів, які активно взаємодіють із зовнішніми API.

3.6 Розгортання та налаштування сервісу у хмарі

Після вибору хмарної платформи Streamlit Cloud для розгортання сервісу генерації текстів наступним етапом стало налаштування середовища та підготовка до перенесення локального за стосунку в доступний формат. Такий підхід дозволяє максимально спростити розгортання, не вдаючись до складної серверної інфраструктури чи ручного налаштування залежностей.

Процес розгортання починається з формування структури проєкту, що має відповідати вимогам платформи. Одним із ключових елементів є файл requirements.txt, у якому перелічено всі необхідні бібліотеки для коректної роботи застосунку. Цей файл використовується системою Streamlit для автоматичного встановлення залежностей у хмарному середовищі. До найважливіших бібліотек, які зазначено у файлі, належать Openai — для взаємодії з API мовної моделі, та Streamlit — для створення інтерфейсу користувача.

Після створення конфігураційних файлів весь проєкт завантажується у публічний репозиторій на GitHub, що є обов’язковою умовою для розгортання через Streamlit Cloud. У веб-інтерфейсі платформи виконується прив’язка до репозиторію, після чого запускається автоматичний процес встановлення залежностей та запуску застосунку. При цьому відпадає потреба в додатковій конфігурації серверів чи налаштуванні середовища вручну — усе відбувається у напівавтоматичному режимі.

У підготовленому репозиторії, який розгортається через Streamlit Cloud, містяться лише два ключові файли: main.py — основний файл програми, що

містить усю логіку взаємодії з користувачем та OpenAI API, і requirements.txt — файл, у якому вказано необхідні версії бібліотек для запуску застосунку. Така структура дозволяє забезпечити швидке й безпомилкове розгортання в хмарному середовищі.

На рис. 3.2 подано загальний вигляд структури репозиторію, підготовленого до розгортання на Streamlit Cloud.

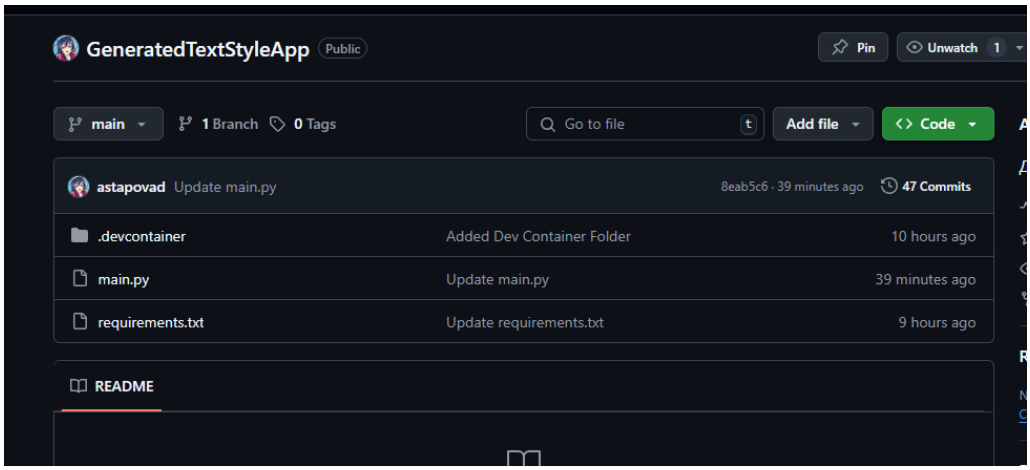

Рис. 3.2 Репозиторій у GitHub

Після завантаження основних файлів до GitHub, відбувається прив’язка репозиторію до Streamlit Cloud. Для цього користувач переходить на офіційний сайт, авторизується через обліковий запис GitHub та надає доступ до репозиторії, де у результаті з’являється можливість обрати потрібний зі списку, правильну гілку та основний файл з кодом, при бажанні змінити App URL та натиснути кнопку «Deploy» для запуску процесу розгортання, не забуваючи ввести секретний код, який нам надав OpenAI API.(рис. 3.3)

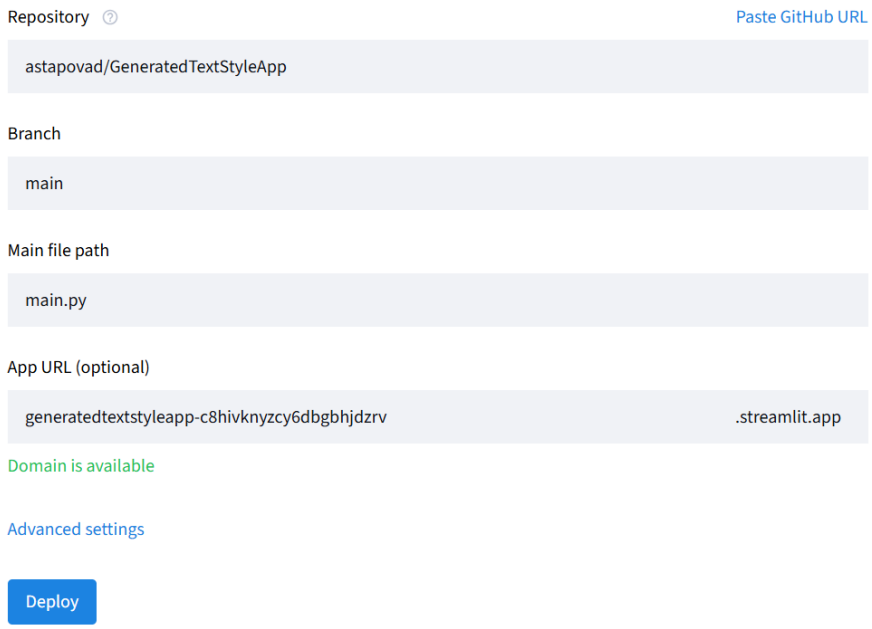

Рис 3.3 Налаштування перед розгортанням в Streamlit Cloud

Інтерфейс платформи автоматично розпізнає основний код та файл вимог requirements.txt, після чого починає збірку середовища виконання. У разі наявності змін у коді, при кожному оновленні репозиторію, Streamlit робить автоматичне оновлення. Це дозволяє швидко тестувати оновлення та миттєво виводити їх у користувацьке середовище без необхідності ручного втручання.

Після завершення збірки додаток автоматично розгортається, і користувач отримує доступ до готового веб-інтерфейсу, який дозволяє формувати текстові запити до моделі. Таким чином, результатом усіх етапів є стабільно працюючий сервіс генерації текстів, доступний за унікальним URL-посиланням, що забезпечує інтерактивну взаємодію між користувачем та мовною моделлю без необхідності локального запуску коду або складного налаштування структури.

3.7 Тестування веб-додатку

Реалізований веб-додаток забезпечує зручний і зрозумілий інтерфейс для генерації текстів на основі запиту користувача. При вході на сторінку сервісу користувач потрапляє в зручний інтерфейс, що містить назву застосунку, короткий опис і панель керування генерацією тексту. (рис.3.4)

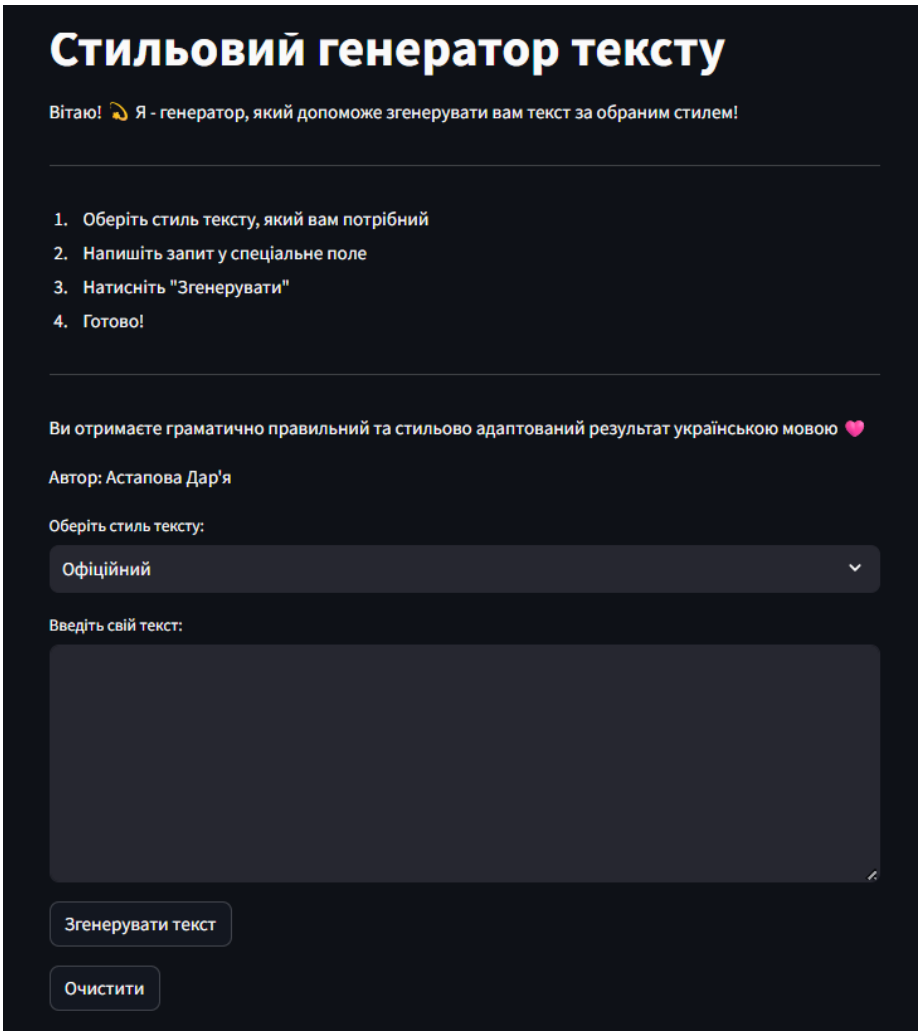

Рис. 3.4 Інтерфейс веб-додатку «Стильовий генератор тексту»

У верхній частині сторінки розташовано випадаючий список, де користувач може обрати стиль тексту: офіційний, креативний, науковий, академічний, формальний, мотиваційний або розповідь(рис.3.5). Після вибору стилю в текстове поле необхідно ввести запит або тему, на яку потрібен згенерований текст.

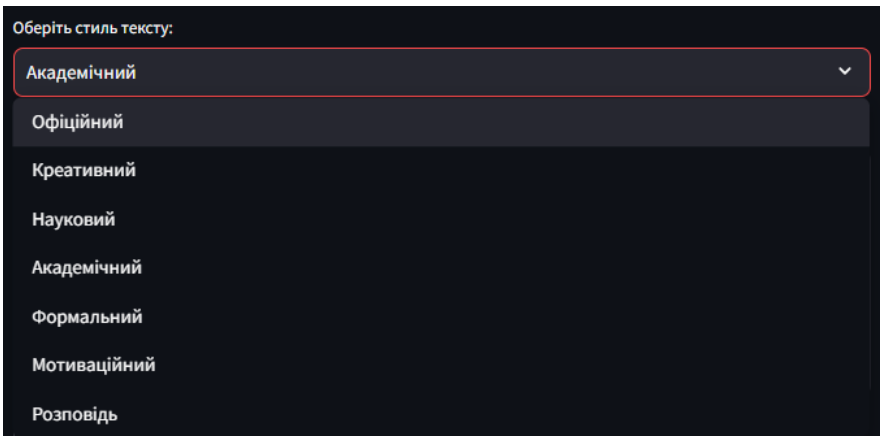

Рис. 3.5 Список стилів тексту, які можна обрати у веб-додатку

Кнопка «Згенерувати» запускає процес генерації. Весь процес займає кілька секунд, після чого нижче відображається результат. Згенерований текст подається одразу у відформатованому вигляді, що зручно для копіювання чи подальшого використання(рис.3.6). У випадку порожнього запиту виводиться повідомлення з проханням ввести текст(рис.3.7), що забезпечує мінімальну перевірку введення.

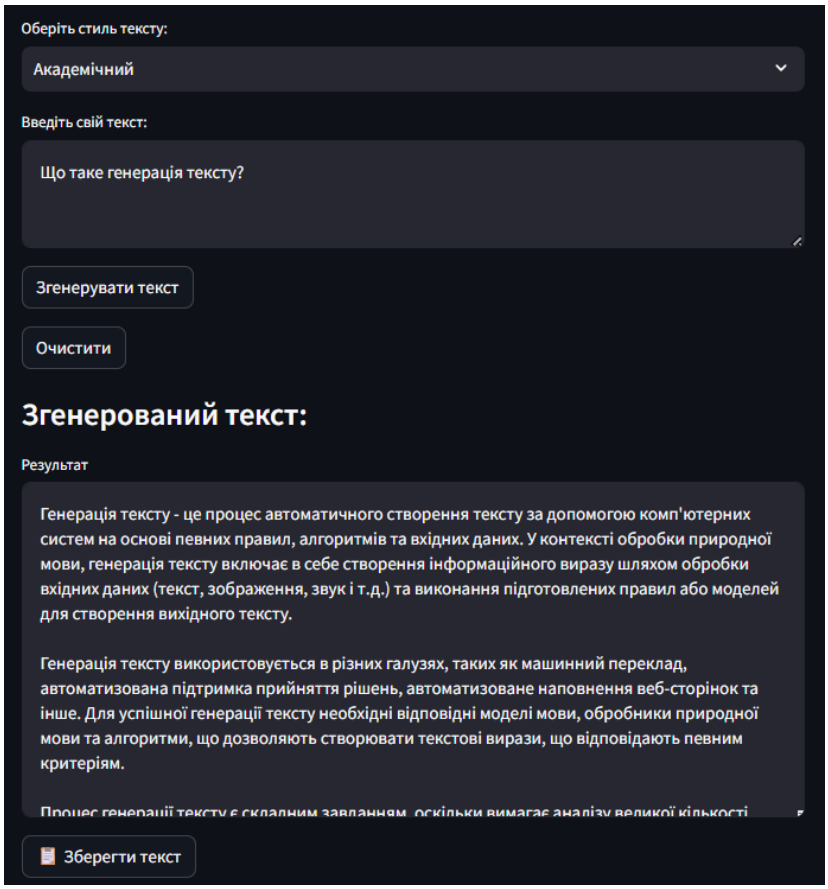

Рис 3.6 Результат згенерованного тексту, де вказано академічний стиль на запит «Що таке генерація тексту?»

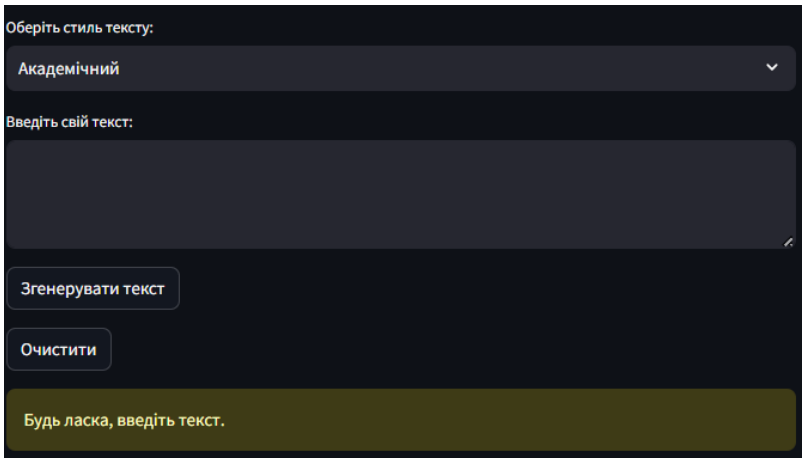

Рис. 3.7 Виведення помилки при порожньому запиті

Інтерфейс побудований так, щоб ним зручно користуватися як на комп’ютерах, так і на телефонах — усі елементи правильно відображаються незалежно від розміру екрана, а взаємодія з веб-додатком зрозуміла навіть без технічної підготовки.

ВИСНОВКИ

1. Мовні моделі на основі глибокого навчання демонструють великий потенціал у сфері штучного інтелекту. Вони дозволяють ефективно виконувати завдання генерації текстів, що знайшло підтвердження в огляді моделей GPT, таких як: GPT-2, GPT-3.5 та GPT-4. Розгляд архітектур цих моделей показав переваги трансформерної архітектури, що забезпечує гнучкість, масштабованість і можливість роботи з контекстом. Попри високу якість генерації, моделі мають обмеження, пов’язані з "галюцинаціями", упередженнями та потребою в потужних обчислювальних ресурсах.
2. Сучасні хмарні обчислення відіграють ключову роль у розгортанні мовних сервісів. У дипломній роботі було розглянуто різні моделі хмарної структури — IaaS, PaaS, SaaS, їх розгортання (приватна, публічна, гібридна хмара), а також переваги і ризики, пов’язані з їх використанням. Зокрема, підкреслено важливість таких платформ, як Google Cloud, AWS, Azure, а також використання Streamlit для створення зручного інтерфейсу генерації тексту. Використання хмари дозволило досягти гнучкості, відмовостійкості та масштабованості у проєкті.
3. У розділі про реалізацію було створено веб-додаток із мовою програмування Python та використанням фреймворку Streamlit, що дозволяє користувачу обрати стиль генерації тексту та отримати результат за допомогою API OpenAI. Такий підхід спростив взаємодію користувача з великою мовною моделлю й продемонстрував практичну інтеграцію технологій у хмарному середовищі. Була обрана модель GPT-3.5 Turbo, що забезпечила баланс між якістю відповіді та швидкістю обробки.
4. У процесі реалізації були враховані методичні підходи до роботи з LLM, зокрема prompt engineering, керування параметрами генерації та токенізацією, а також використання сучасних технологій, таких як RAG та GraphRAG. Це дозволило підвищити точність і релевантність відповідей, забезпечити адаптивність системи до запитів користувача й інтегрувати актуальні дані у відповіді моделі.

5. Отже, розроблена система генерації тексту на основі мовної моделі GPT успішно реалізує поставлені завдання, забезпечує гнучкий інтерфейс для користувача та демонструє ефективне поєднання глибокого навчання з хмарними технологіями. Надалі система може бути розширена через додавання багатомовної підтримки, покращення стилістичних параметрів генерації, інтеграцію з іншими сервісами та удосконалення системи аналізу якості згенерованих текстів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Астапова Д.В., Звенігородський О.С. Використання нейронних мереж для обробки даних в IOT-системах моніторингу здоров'я / Науково-практична конференція «Використання нейронних мереж для обробки даних в iot-системах моніторингу здоров'я». Збірник тез. – К.: ДУІКТ, 2025 р.
2. Зінченко О.В., Антонов В.В., Астапова Д.В. Штучний інтелект у візуальному пошуку: дослідження можливостей / Науково-практична конференція «Сучасні досягнення компанії Hewlett Packard Enterprise в галузі IT та нові можливості їх вивчення і застосування». Збірник тез. – К.: ДУІКТ, 2023 р.
3. Поняття хмарних обчислень: основні моделі та характеристики. *ON your BUSINESS*. URL: https://onbiz.biz/cloud-computing-models/ (дата звернення: 09.05.2025).
4. A systematic literature review on text generation using deep neural network models / N. Fatima та ін. *IEEE Access*. 2022. URL: https://ieeexplore.ieee.org/document/9771452 (дата звернення: 06.05.2025).
5. Attention is all you need / A. Vaswani та ін. *ArXiv*. 2023. URL: https://arxiv.org/pdf/1706.03762 (дата звернення: 06.05.2025).
6. Bruckner D. Cloud AI vs. on-premises AI: what you need to know. *Tamr*. URL: https://www.tamr.com/blog/cloud-ai-vs-onpremise-ai-what-you-need-to-know (дата звернення: 12.05.2025).
7. Dusek O., Novikova J., Rieser V. Evaluating the state-of-the-art of end-to-end natural language generation: the E2E NLG challenge. *ArXiv*. 2019. URL: https://arxiv.org/pdf/1901.07931 (дата звернення: 06.05.2025).
8. Edwards B. OpenAI's GPT-4 exhibits "human-level performance" on professional benchmarks. *arstechnica*. URL: https://arstechnica.com/information-technology/2023/03/openai-announces-gpt-4-its-next-generation-ai-language-model/ (дата звернення: 08.05.2025).

9. Edwards B. Why ChatGPT and Bing Chat are so good at making things up. *arstechnica*. URL: https://arstechnica.com/information-technology/2023/04/why-ai-chatbots-are-the-ultimate-bs-machines-and-how-people-hope-to-fix-them/ (дата звернення: 08.06.2025).
10. Exploration into deep learning text generation architectures for dense image captioning / M. Toshevska та ін. *15th Conference on Computer Science and Information Systems (FedCSIS 2020)*. 2020. URL: https://www.researchgate.net/publication/344468185_Exploration_into_Deep_Learning_Text_Generation_Architectures_for_Dense_Image_Captioning (дата звернення: 06.05.2025).
11. Gartner says worldwide iaas public cloud services revenue grew 16.2% in 2023. *Gartner*. URL: https://www.gartner.com/en/newsroom/press-releases/2024-07-22-gartner-says-worldwide-iaas-public-cloud-services-revenue-grew-16-point-2-percent-in-2023 (дата звернення: 10.05.2025).
12. Glassman B. Повний посібник для початківців з генеративного ІІІІ. *DreamHost*. URL: https://www.dreamhost.com/blog/uk/posibnik-z-generativnogo-shtuchnogo-intelektu/ (дата звернення: 06.05.2025).
13. Goodwin M. What is an API (application programming interface)?. *IBM*. URL: https://www.ibm.com/think/topics/api (дата звернення: 08.06.2025).
14. ISO/IEC 22123-2:2023. Information technology – cloud computing – part 2: concepts. Вид. офіц. ISO, 2023. 35 с.
15. Is temperature the creativity parameter of large language models? / M. Peeperkorn та ін. *ArXiv*. 2024. URL: https://arxiv.org/pdf/2405.00492 (дата звернення: 14.05.2025).
16. Jurafsky D., Martin J. H. Speech and language processing (3rd edition). Stanford University. URL: https://web.stanford.edu/~jurafsky/slp3/3.pdf (дата звернення: 06.05.2025).
17. Kaushal A., Mahowald K. What do tokens know about their characters and how do they know it?. *ArXiv*. 2022. URL: https://arxiv.org/pdf/2206.02608 (дата звернення: 14.05.2025).

18. Language models are few-shot learners / T. B. Brown та ін. *ArXiv*. 2020. URL: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf (дата звернення: 07.05.2025).
19. Language models are unsupervised multitask learners / A. Radford та ін. 2019. URL: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf (дата звернення: 07.05.2025).
20. Larson J. GraphRAG: a new approach for discovery using complex information. *Microsoft Research Blog*. URL: https://www.microsoft.com/en-us/research/blog/graphrag-unlocking-llm-discovery-on-narrative-private-data/ (дата звернення: 12.05.2025).
21. Mandour A. GPT-3.5 model architecture. *OpenGenus IQ: Learn Algorithms, DL, System Design*. URL: https://iq.opengenus.org/gpt-3-5-model/ (дата звернення: 08.05.2025).
22. Manning C. D. Human language understanding & reasoning. *Daedalus*. 2022. URL: https://doi.org/10.1162/daed_a_01905 (дата звернення: 07.05.2025).
23. Mishra P. Automated metrics for evaluating the quality of text generation. *DigitalOcean*. URL: https://www.digitalocean.com/community/tutorials/automated-metrics-for-evaluating-generated-text (дата звернення: 15.05.2025).
24. Navigating deployment of llms to cloud servers / B. Mohta та ін. *Walturn*. URL: https://www.walturn.com/insights/navigating-deployment-of-llms-to-cloud-servers (дата звернення: 10.05.2025).
25. NeOn-GPT: a large language model-powered pipeline for ontology learning / N. Fathallah та ін. *ESWC 2024*. URL: https://2024.eswc-conferences.org/wp-content/uploads/2024/05/77770034.pdf (дата звернення: 06.05.2025).
26. OpenAI. GPT-4 technical report. *ArXiv*. 2024. URL: https://arxiv.org/pdf/2303.08774 (дата звернення: 08.05.2025).

27. Paraphrase types elicit prompt engineering capabilities / J. P. Wahle та ін. *ArXiv*. 2025. URL: https://arxiv.org/pdf/2406.19898 (дата звернення: 12.05.2025).
28. Piper K. An AI helped us write this article. *Vox*. URL: https://www.vox.com/future-perfect/2019/2/14/18222270/artificial-intelligence-open-ai-natural-language-processing (дата звернення: 06.05.2025).
29. Pre-trained language models for text generation: a survey / J. Li та ін. *ArXiv*. 2022. URL: https://arxiv.org/pdf/2201.05273 (дата звернення: 05.05.2025).
30. Sanderson K. GPT-4 is here: what scientists think. *Nature*. URL: https://www.nature.com/articles/d41586-023-00816-5 (дата звернення: 08.06.2025).
31. Slobodianiuk A. V., Semerikov S. O. Advances in neural text generation: a systematic review. *ArXiv*. URL: https://ceur-ws.org/Vol-3917/paper59.pdf (дата звернення: 06.05.2025).
32. Sparks of artificial general intelligence: early experiments with GPT-4 / S. Bubeck та ін. *ArXiv*. 2023. URL: https://arxiv.org/pdf/2303.12712 (дата звернення: 09.05.2025).
33. Tonya. 4 переваги хмарних технологій: підсилення вашого бізнесу. *UCloud*. URL: https://ucloud.ua/perevagy-hmarnyh-tehnologij/ (дата звернення: 10.05.2025).
34. Vincent J. OpenAI co-founder on company's past approach to openly sharing research: "We were wrong". *The Verge*. URL: https://www.theverge.com/2023/3/15/23640180/openai-gpt-4-launch-closed-research-ilya-sutskever-interview (дата звернення: 09.05.2025).
35. Vincent J. OpenAI's new multitalented AI writes, translates, and slanders. *TheVerge*. URL: https://www.theverge.com/2019/2/14/18224704/ai-machine-learning-language-models-read-write-openai-gpt2 (дата звернення: 06.05.2025).

36. What is NLP (natural language processing)?. *IBM*. URL: https://www.ibm.com/think/topics/natural-language-processing (дата звернення: 05.05.2025).
37. Williams R. Why Google's AI Overviews gets things wrong. *MIT Technology Review*. URL: https://www.technologyreview.com/2024/05/31/1093019/ (дата звернення: 12.05.2025).
38. Wu H. GPT-2 detailed model architecture. *Medium*. URL: https://medium.com/@hsinhungw/gpt-2-detailed-model-architecture-6b1aad33d16b (дата звернення: 07.05.2025).
39. Zahoor Ch S. APIs - openai and hugging face introduction. *LinkedIn*. URL: https://www.linkedin.com/pulse/apis-openai-hugging-face-introduction-sumiyya-zahoor-ch-r29sf?trk=public_post_main-feed-card_feed-article-content (дата звернення: 10.05.2025).

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ШТУЧНОГО ІНТЕЛЕКТУ

Кваліфікаційна робота
на тему: «Генерація текстів для користувача на основі методів глибокого навчання і хмарних технологій»

Виконала: здобувачка вищої освіти гр. ШІД-42
Дар'я АСТАПОВА
Керівник: кандидат технічних наук, доцент
Олександр ЗВЕНІГОРОДСЬКИЙ

2025 рік

Мета роботи: Створення зручного веб-додатку для генерації текстів, що адаптується під потреби користувача (стиль та тема) на основі GPT-3.5-Turbo та розгортання у хмарі.

Об'єкт дослідження: Процес генерації текстів за допомогою мовних моделей.

Предмет дослідження: Методи генерації текстів.

Основні завдання кваліфікаційної роботи:

• Провести огляд сучасних мовних моделей, зокрема GPT-2, GPT-3.5 і GPT-4, та проаналізувати їх.

• Дослідити можливості хмарних технологій для інтеграції мовного сервісу у веб-середовище.

• Реалізувати веб-додаток для генерації текстів з вибором стилю тексту з використанням Streamlit та OpenAI API.

• Розгорнути готову систему у хмарному середовищі з GitHub-репозиторію.

• Провести тестування застосунку через веб-інтерфейс і проаналізувати результати роботи з погляду користувача.

Актуальність теми дослідження полягає в стрімкому розвитку мовних моделей та зростанням потреби генерації текстів у різних сферах людської діяльності. Поєднання глибокого навчання та хмарних технологій відкриває нові можливості для створення зручних та доступних веб-додатків для користувачів без додаткових передплат та реєстрацій.

Проблематика теми дослідження полягає в малодоступності мовних моделей для широкого кола користувачів через складність у використанні та відсутність зрозумілого інтерфейсу. Виникає потреба у створенні зручного інструменту, який дозволить генерувати тексти у заданому стилі без залучення технічних знань.

Використані технології:

- Python
- GPT 3.5
- OpenAI API
- Streamlit
- Streamlit Community Cloud
- GitHub

Платформа	Модель	Обсяг контексту	Мовна підтримка	API-доступ	Особливості	Кастомізація	Типові стилі тексту
ChatGPT	Безкоштовно: GPT-3.5, платна: GPT-4-turbo	В залежності від моделі, у GPT-3.5 обсяг 200 тис. токенів	25+ мов	Через офіційну платформу OpenAI	Можливість задавати багато промптів для налаштування відповідей, створення користувацьких моделей, зручний інтерфейс для загального користувача та розробників	Висока (через Custom GPT)	Будь-які стилі, залежно від промпту
Claude	Безкоштовно: Claude 3.7 Sonet, платна: Claude 3.5 Haiku та Claude 3 Opus.	200 тис. токенів	Багатомовна	API надається через платформу розробників, однак доступ може бути обмежений в залежності від деяких факторів	Зниження ймовірність помилок у відповідях, надання логічних відповідей,добрий у застосуванні для аналітики та створення контенту	Обмежена	Формальний, аналітичний
Gemini	2.0 Flash основна модель, 2.5 Flash пробна	до 1 млн. токенів	Багатомовна	Платформа Vertex AI	Великі можливості аналізу, співпрацює з Google-сервісами, велика обробка данних та висока швидкість роботи	Інтеграція через Google Cloud	Універсальний, адаптивний

Рис. 1 Порівняння трьох платформ для генерації тексту

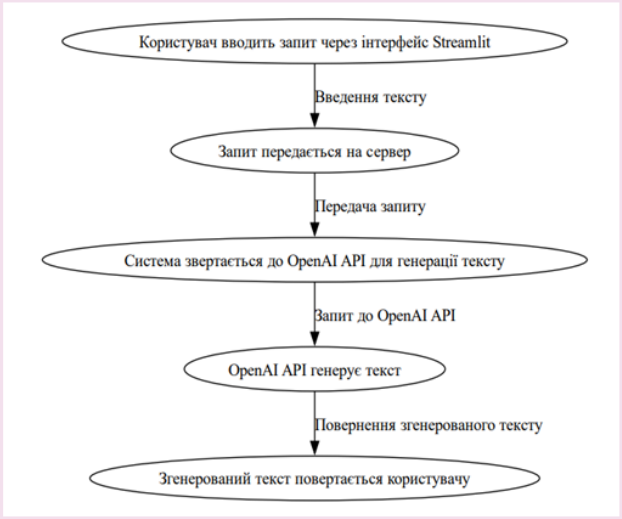

Рис. 2 Блок-схема обробки запиту користувача

Демонстрація проекту

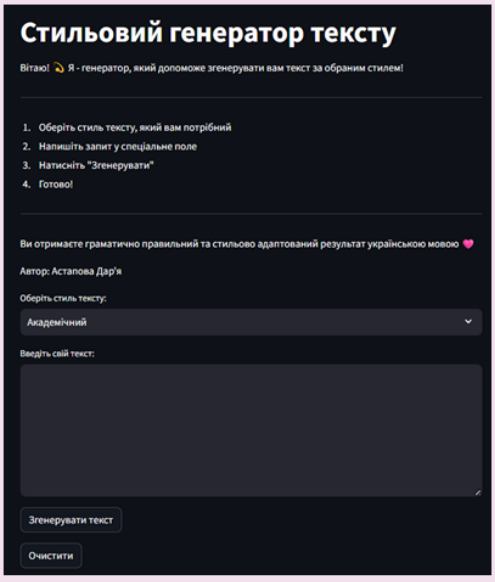

Рис. 4 Сторінка веб-додатку "Стильовий генератор тексту"

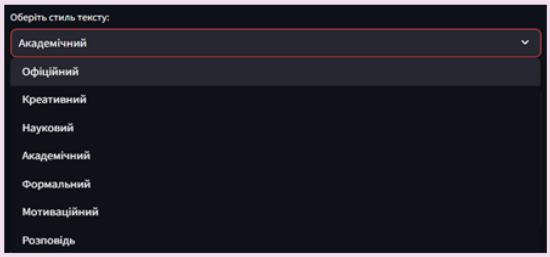

Рис. 5 Список стилів тексту, які можна обрати у веб-додатку

Демонстрація роботи

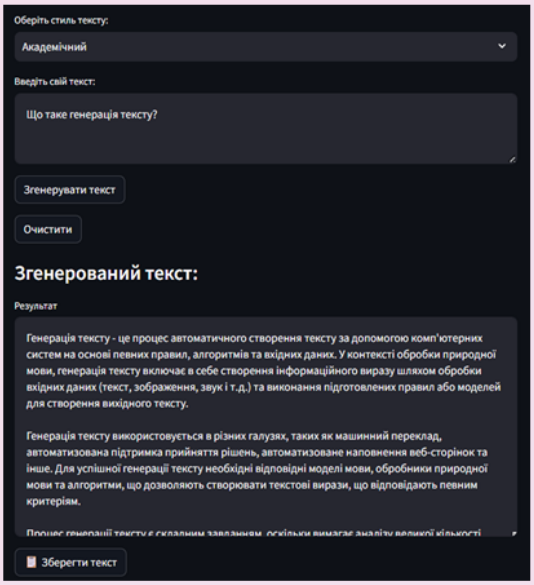

Рис. 6 Результат згенерованного тексту, де вказано академічний стиль на запит «Що таке генерація тексту?»

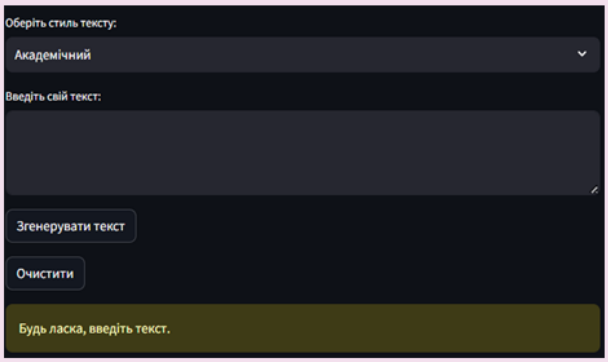

Рис. 7 Виведення помилки при порожньому запиті

ВИСНОВКИ

➢ Реалізовано веб-додаток, що дозволяє генерувати тексти у різних стилях відповідно до запиту користувача. Основою слугує модель GPT-3.5, побудована на трансформерній архітектурі — сучасному методі глибокого навчання, що використовує механізм уваги, попереднє навчання на великому корпусі даних та налаштування за допомогою зворотного зв'язку від людини.

➢ Інтегровано OpenAI API, реалізовано систему промптів, які адаптуються до обраного стилю: офіційний, науковий, креативний, та підвищують якість і доречність згенерованих текстів.

➢ Проведено огляд архітектури сучасних мовних моделей GPT-2, GPT-3.5 та GPT-4, розглянуто їх технічні особливості, етапи навчання та переваги у контексті генерації тексту.

➢ Реалізовано інтерфейс користувача за допомогою фреймворку Streamlit, що забезпечує простоту використання. Систему розгорнуто у хмарному середовищі Streamlit Community Cloud.

➢ Проведено тестування на функціональність роботи з позиції користувача. Встановлено, що система є стабільною, адаптивною та зручною у використанні — особливо в умовах обмеженого технічного досвіду користувачів.