ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І. І. МЕЧНИКОВА

(повне найменування закладу вищої освіти)

Факультет математики, фізики та інформаційних технологій

(повне найменування факультету)

Кафедра інформаційних технологій

(повна назва кафедри)

Кваліфікаційна робота

на здобуття ступеня вищої освіти «Бакалавр»

«Розробка додатку для впровадження онлайн освіти»

(тема кваліфікаційної роботи українською мовою)

«Development of an Application for Online Education Implementing»

(тема кваліфікаційної роботи англійською мовою)

Виконав: здобувач денної форми навчання

спеціальності 122 Комп'ютерні науки

(код, назва спеціальності)

Освітня програма Комп'ютерні науки

(назва)

Цуркан Артем Ігорович

(прізвище, ім'я, по-батькові здобувача)

Керівник доктор філософії комп. наук, доцент Бучинська І.В.

(науковий ступінь, вчене звання, прізвище, ініціали) (підпис)

Рецензент Клепатська В.В.

(науковий ступінь, вчене звання, прізвище, ініціали)

Рекомендовано до захисту:
Протокол засідання кафедри
Інформаційних технологій .
№ 1 від 09 червня 2024 р.

Завідувачка кафедри

(підпис) КАЗАКОВА Надія .

(прізвище,ім'я)

Захищено на засіданні ЕК № 13 .
протокол № 29 від 21 червня 2024 р.

Оцінка відмінно / A / 95 .

(за національною шкалою/шкалою ECTS/ бали)

Голова ЕК

(підпис) КОПИЧЕНКО Іван .

(прізвище,ім'я)

Одеса 2024

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

API – Application Programming Interface;

IDE – Integrated Development Environment;

ПЗ – Програмне Забезпечення;

ШІ – Штучний Інтелект;

SerpAPI – Search Engine Results Page Application Programming Interface;

HTML – Hyper Text Markup Language;

CSS – Cascading Style Sheets;

AJAX – Asynchronous Java Script and XML;

ML – Machine Learning;

LLMs – Large Language Models;

RL – Reinforcement Learning;

VHDL – Very High Speed Integrated Circuit Hardware Description Language;

GAN – Generative Adversarial Network;

VAEs – Variational Autoencoder;

GPT – Generative Pre-trained Transformer;

UML – Unified Modelling Language;

OAuth – Open Authorization;

PaaS – Platform as a Service;

CLI – Command-Line Interface.

ВСТУП

У сучасному світі інформаційних технологій розвиток онлайн-освіти є одним з ключових напрямків, що визначає якість та доступність навчання для широкого кола людей. В епоху цифрових технологій та глобалізації, потреба у зручних та ефективних засобах навчання зростає, що обумовлює необхідність створення інноваційних платформ для впровадження онлайн-освіти.

Дана кваліфікаційна робота присвячена розробці веб-додатку для впровадження онлайн-освіти, що використовує сучасні технології та інструменти, такі як Langchain, Wolfram Alpha, веб-пошук, локальні PDF-документи та OpenAI. Метою проекту є створення багатофункціональної платформи, яка дозволяє користувачам ефективно обробляти та аналізувати різноманітні джерела інформації, що сприятиме підвищенню якості навчання та досліджень.

Інтеграція з Wolfram Alpha забезпечує користувачам можливість отримувати миттєві відповіді на складні математичні запитання, використовуючи передові алгоритми обчислення та аналізу даних. Додавання локальних PDF-документів та веб-пошуку дозволяє користувачам здійснювати пошук та роботу з власними документами в рамках єдиної платформи. Використання технологій OpenAI відкриває нові горизонти для аналізу та інтерпретації текстових даних, що робить процес навчання більш інтерактивним та ефективним.

Проект включає дослідження предметної області, вибір програмних засобів для реалізації додатку, проектування архітектури додатку та його реалізацію і тестування. Реалізація даного проекту сприятиме покращенню процесу навчання, забезпечуючи студентам доступ до інноваційних інструментів та ресурсів, що дозволить їм краще засвоювати матеріал та застосовувати знання в реальних ситуаціях.

Предметом кваліфікаційної роботи є розробка додатку для впровадження онлайн-освіти. Це включає дослідження предметної області освітніх технологій, вибір програмних засобів для реалізації платформи, проектування архітектури додатку та його реалізацію і тестування.

Об'єктом дослідження є платформа для онлайн-освіти, яка використовує сучасні технології та інструменти, такі як OpenAI, Wolfram Alpha, веб-пошук, локальні PDF-документи та Langchain. Метою проекту є створення багатофункціональної платформи, яка дозволяє користувачам ефективно обробляти та аналізувати різноманітні джерела інформації, що сприятиме підвищенню якості навчання та досліджень

Таким чином, дана кваліфікаційна робота спрямована на створення ефективного інструменту для підтримки процесу навчання, що відповідає сучасним вимогам і тенденціям у сфері інформаційних технологій та освіти.

Для досягнення поставленої мети, наступні завдання ставляться перед проєктом:

- дослідити предметну область;
- вибрати програмні засоби для реалізації додатку;
- спроєктувати архітектуру додатку;
- реалізувати і протестувати додаток;

Кваліфікаційна робота бакалавра складається з 82 сторінок, 5 розділів, 20 рисунків, 3 таблиць, 10 посилань на літературу та додатку.

1 РОЗГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ ТА ФОРМУЛЮВАННЯ ЗАВДАННЯ

1.1 Дослідження предметної області

У галузі освітніх технологій з кожним роком з'являється все більше платформ, що використовують штучний інтелект для покращення процесу навчання. Серед них, Duolingo використовує AI для адаптації мовних курсів до індивідуальних потреб користувачів, що допомагає забезпечити більшу гнучкість у вивченні мов, однак часто не вдається глибоко занурити учнів у складні аспекти мови та культури. Socratic від Google та Chegg залучають AI для вирішення академічних задач, пропонуючи швидкі та ефективні рішення, але такий підхід може сприяти поверхневому засвоєнню матеріалів без розвитку критичного мислення та глибокого розуміння.

Курси від Khan Academy та Coursera покривають широкий спектр академічних дисциплін, проте вони часто стандартизовані та не завжди дозволяють адаптувати навчальний процес під індивідуальні потреби студентів. Це може обмежувати можливість студентів досліджувати теми, що виходять за рамки встановленого курсу. З іншого боку, ChatGPT від OpenAI демонструє вражаючі можливості у створенні відповідей та веденні діалогів, що може служити чудовим інструментом для навчання через бесіду, проте його використання часто не супроводжується глибоким науковим аналізом.

На відміну від цих платформ, платформа Освіта+ створена з метою інтеграції передових технологій штучного інтелекту та аналітичних інструментів. Поєднання AI від OpenAI з аналітичними можливостями Wolfram Alpha дозволяє не тільки відповідати на запитання, а й здійснювати глибокий аналіз даних, що надає студентам персоналізовані та обґрунтовані відповіді. Такий підхід не лише підвищує ефективність навчання, але й допомагає учням краще засвоювати та застосовувати навчальний матеріал в реальних ситуаціях. У зв'язку з цим, інтеграція різноманітних інформаційних

джерел та технологій штучного інтелекту є дуже актуальним на сьогодняшній день.

Інтеграція з Wolfram Alpha є потужною обчислювальною платформою, яка використовує алгоритми для надання відповідей на складні запитання. Ця система здатна аналізувати великі обсяги даних, здійснювати розрахунки та надавати корисну інформацію у вигляді графіків, таблиць та текстових пояснень. Інтеграція Wolfram Alpha в наш вебсайт дозволяє користувачам отримувати швидкі та точні відповіді на різноманітні запити, що значно підвищує їхню продуктивність.

Робота з локальними PDF-документами: Локальні PDF-документи містять велику кількість цінної інформації, яка часто залишається недоступною для автоматизованих систем через специфічний формат файлів. Розробка алгоритмів для аналізу та витягнення даних з PDF-документів є важливим завданням, яке дозволяє користувачам швидко знаходити необхідну інформацію та використовувати її у своїй роботі. Платформа забезпечує зручний інтерфейс для пошуку та аналізу PDF-документів, що значно спрощує роботу з цим типом файлів.

Інтеграція з Google Search є популярним інструментом для створення та зберігання документів у хмарі. Інтеграція з Google Docs дозволяє користувачам працювати з документами безпосередньо на нашій платформі, зберігаючи всі зміни та надаючи доступ до даних з будь-якого пристрою. Це забезпечує зручність та гнучкість у роботі з документами, а також дозволяє ефективно організовувати та обробляти інформацію.

Технології штучного інтелекту, зокрема OpenAI, відкривають нові можливості для автоматизації аналізу та інтерпретації текстових даних. Використання OpenAI дозволяє створювати інтелектуальні системи, які можуть здійснювати глибокий аналіз текстів, витягати ключові ідеї та генерувати відповіді на основі великого обсягу інформації. Це значно підвищує ефективність роботи користувачів та дозволяє отримувати корисні інсайти з різних джерел даних.

1.2 Формулювання завдання

Як вже було зазначено, метою цього проекту є створення інтегрованого веб-сайту, який поєднує можливості Wolfram Alpha, локальної роботи з PDF-документами, пошукової системи Google, а також OpenAI та Langchain для надання користувачам універсального інструменту для отримання інформації та аналізу даних. Це включає розробку Frontend-частини додатку, що складається з інтерфейсу користувача, розробленого за допомогою HTML, CSS та jQuery, а також Backend-частини, що забезпечує серверну частину та обробляє запити до різних API, з використанням Python-фреймворку Flask і веб-хостингу на Heroku. Ключовими завданнями є налаштування інтеграції OpenAI з трьома ключовими інструментами – Wolfram Alpha, локальними PDF файлами та SerpAPI – за допомогою Langcahin. Інтеграція з Wolfram Alpha дозволяє здійснювати запити та отримувати обчислювальні результати, наукові дані та іншу подібну інформацію. Можливість роботи з локальними PDF-файлами дозволяє користувачам легко отримувати необхідні дані з власних документів. Інтеграція з SerpAPI забезпечує можливість здійснення пошукових запитів для отримання актуальної інформації з Інтернету, доповнюючи дані з інших джерел.

1.3 Визначення вимог до розробки системи

Для успішної реалізації платформи, функціональні та нефункціональні вимоги були поставлені перед додатком (див. табл. 1). Функціональні вимоги визначають конкретні дії, які система повинна виконувати, щоб задовольнити потреби користувачів і досягти поставлених цілей. Вони описують, що система повинна робити, і включають можливості, які повинні бути реалізовані. Нефункціональні вимоги, з іншого боку, визначають критерії, які характеризують роботу системи в цілому.

Таблиця 1 – Функціональні та нефункціональні вимоги

Функціональні вимоги	Нефункціональні вимоги
Зручний користувацький веб-інтерфейс	Комп'ютер, ноутбук, смартфон, планшет або інший подібний пристрій з веб-браузером
Можливість здійснення запитів напряму до OpenAI	Підключення до Інтернету
Можливість здійснення наукових запитів, на відповіді для яких використовується OpenAI у поєднанні з Wolfram Alpha	-
Можливість завантаження PDF-документів та здійснювати запити за їх змістом	-
Можливість використовувати можливості OpenAI у поєднанні з актуальною інформацію (завдяки SerpAPI)	-

Визначивши функціональні та нефункціональні вимоги, можемо перейти до розгляду, порівнянню та вибору програмних засобів, які можна обрати для реалізації платформи.

2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ ПЛАТФОРМИ

2.1 Розгляд основних редакторів коду та IDE

Важливим інструментом розробки будь-якого додатку є редактор коду або IDE. Сьогодні існує широкий вибір таких інструментів. Табл. 2 містить основні особливості, переваги та недоліки популярних редакторів коду / IDE.

Таблиця 2 – Стислий порівняльний аналіз основних редакторів коду / IDE

Редактор/IDE	Особливості	Переваги	Недоліки
VSCode	Легкий, плагіни для різних мов	Гнучкий, велика спільнота	Може споживати багато ресурсів з великою кількістю плагінів
JetBrains PyCharm	Спеціалізований IDE для Python	Потужний, інтеграція з різними бібліотеками Python	Важкий, може бути дорогим для комерційного використання
Atom	Відкритий текстовий редактор	Гнучкість з плагінами, легкість використання	Повільніша швидкість завантаження, розробка призупинена
Sublime Text	Швидкий текстовий редактор	Швидкий, налаштування через JSON	Платний для комерційного використання
Notepad++	Безкоштовний джерело відкритого коду редактор	Легкий, швидкий, підтримка багатьох мов	Обмежені функції порівняно з повноцінними IDE
JetBrains Fleet	Новий легкий IDE від JetBrains	Сучасний дизайн, висока налаштуваність	Ще в ранньому доступі, може мати помилки

2.1 Порівняльний аналіз технологій розробки

Як було зазначено у [1], для frontend-частини додатку треба володіти мовою розмітки HTML, мовою каскадних стилів CSS, веб-скриптінг мовою JavaScript або одним з її фреймворків (jQuery, Angular, React тощо). Якщо HTML та CSS є ключовими елементами будь-якого сучасного веб-додатку і не потребують обгрунтування, то щодо вибору веб-скриптінг мови/ фреймворку та бекенд технології – існує великій вибір і у кожного програмного засобу є свої переваги. У таблиці табл. 3 наведений порівняльний аналіз різноманітних технологій розробки у стислій формі.

Таблиця 3 – Стислий порівняльний аналіз програмних засобів для розробки веб-додатків

Технологія	Особливості	Переваги	Недоліки
jQuery	Легка бібліотека JavaScript для маніпуляції DOM	Простота використання, широке прийняття	Менш сучасний, менш продуктивний за новіші фреймворки
React	Бібліотека JavaScript для створення інтерфейсів	Широка екосистема, висока продуктивність	Вимагає розуміння сучасних JavaScript і build-tools
Vue	Прогресивний фреймворк JavaScript	Легко інтегрується, легкий вивченні	Менше ресурсів у порівнянні з React
Flask	Мікрофреймворк Python для веб-додатків	Легкий, гнучкий	Менш функціональний без додаткових розширень
Node.js	Серверна платформа на основі JavaScript	Спільна мова для клієнта і сервера	Так званий "callback hell", менш ефективний для CPU-інтенсивних операцій

Далі наведений більш розгорнутий опис вищевказаних технологій.

2.1.1 Характеристика jQuery

jQuery – це легка бібліотека JavaScript, яка спрощує маніпуляцію DOM та обробку подій. Її перевага полягає в простоті використання та широкому прийнятті серед розробників, що робить її ідеальним вибором для швидкого створення інтерактивних веб-додатків. Проте, порівняно з сучасними фреймворками, jQuery менш продуктивний та надає менше можливостей.

2.1.2 Опис React

React – це бібліотека JavaScript, призначена для створення інтерфейсів користувача. Вона має широку екосистему, включаючи інструменти для побудови складних додатків, таких як Redux для управління станом. Висока продуктивність React дозволяє створювати швидкі і масштабовані додатки, хоча для її ефективного використання потрібно розуміння сучасних інструментів JavaScript і build-tools.

2.1.3 Опис Vue

Vue – це прогресивний фреймворк JavaScript, який легко інтегрується та легкий у вивченні. Він ідеально підходить для розробників, які тільки починають працювати з JavaScript-фреймворками. Хоча Vue не має такої великої екосистеми, як React, він пропонує достатню функціональність для створення сучасних веб-додатків з мінімальними ресурсами.

2.1.4 Характеристика Flask

Flask – це мікрофреймворк для розробки веб-додатків на Python. Він легкий та гнучкий, дозволяючи швидко розпочати розробку і створювати веб-додатки з мінімальними витратами ресурсів. Основний недолік Flask полягає в меншій функціональності без додаткових розширень, проте це компенсується його простотою та легкістю налаштування.

2.1.5 Аналіз Node.js

Node.js – це серверна платформа на основі JavaScript, яка дозволяє використовувати одну і ту саму мову програмування як на клієнтській, так і на серверній стороні. Це спрощує розробку і дозволяє розробникам використовувати одну мову для всього стеку технологій. Однак, Node.js відомий своїм "callback hell" (ситуація, коли велика кількість вкладених функцій зворотного виклику робить код важким для читання і підтримки) та менш ефективний для CPU-інтенсивних операцій, що може стати перешкодою для деяких проектів.

2.2 Обґрунтування вибору програмних засобів для реалізації

У цьому підрозділі наведені критерії якими був обумовлений саме мій вибір.

Серед фронтенд-технологій дуже популярними є React і Vue. Вони пропонують більш сучасні підходи до побудови інтерактивних інтерфейсів користувача і можуть забезпечити кращу продуктивність і більш сучасну архітектуру порівняно з jQuery. Мій особистий вибір jQuery обумовлений моїм попереднім досвідом роботи з ним.

Для backend-частини вибір технологій теж дуже великій. Мною було обрано Flask – фреймворк мови Python та технологію AJAX, що дозволяє виконувати асинхронні запити до сервера без перезавантаження сторінки.

Щодо веб-хостингу, існує величезна кількість пропозицій. Було обрано Heroku. Це мій особстий суб'єктивний вибір, який базується на попередньому досвіді та простоті налаштування.

Стосовно інших бекенд-технологій (таких як, наприклад, NodeJS) – вони були прийняті до уваги, і невід'ємно мають свої переваги. Наприклад, той самий Node.js дозволяє використовувати JavaScript як на клієнтській, так і на серверній стороні, що може спростити розробку, особливо для тих, хто вже знайомий з JavaScript. Однак Flask може бути привабливим для тих, хто

віддає перевагу Python за його читабельність і лаконічність, що стало однією з головних причин, якими був обумовлений мій вибір.

В якості програмного середовища було обрано VSCode – легкий редактор коду з великим вибором різноманітних плагінів, які підтримують розробку на вищезазначених технологіях. Така гнучкість робить VS Code чудовим інструментом для розробки ПЗ з використанням різних мов програмування і фреймворків. Чому не, наприклад, Notepad++ або надновітній JetBrains Fleet? Останній знаходиться у стадії Public Preview і має досить велику кількість "багів", а щодо Notepad++, то на мій погляд він поступляється VSCode зручністю і функціональністю завдяки величезній кількості доступних плагінів, а також візуальному компоненту – система виділення коду і комфортність для ока на мій погляд набагато кращі у VSCode.

3 ДЕТАЛЬНИЙ РОЗГЛЯД ТЕХНОЛОГІЇ ШТУЧНОГО ІНТЕЛЕКТУ

Штучний інтелект (ШІ) – це галузь інформатики, яка зосереджується на створенні систем, здатних виконувати завдання, що зазвичай вимагають людського інтелекту. До таких завдань належать навчання, міркування, вирішення проблем, розуміння природної мови, сприйняття та навіть прояви соціального інтелекту. Системи ШІ можна поділити на вузький ШІ, який призначений для конкретних завдань, та загальний ШІ, який має на меті виконувати будь-які інтелектуальні завдання, які може виконувати людина.

3.1 Переваги та області використання

Штучний інтелект (ШІ) стає невід'ємною частиною багатьох галузей, підвищуючи ефективність і якість послуг та продуктів. У сфері охорони здоров'я ШІ надає передові діагностичні інструменти і допомагає створювати персоналізовані плани лікування. Наприклад, системи ШІ аналізують медичні зображення, виявляючи аномалії, які можуть свідчити про захворювання, такі як рак. Також ШІ допомагає прогнозувати спалахи хвороб, аналізуючи великі обсяги даних про здоров'я населення, і сприяє швидшому розробленню нових ліків.

У фінансовому секторі ШІ використовується для виявлення шахрайства, алгоритмічної торгівлі та надання персоналізованих фінансових порад. Моделі ШІ аналізують транзакції в режимі реального часу, виявляючи аномалії, що свідчать про можливу шахрайську діяльність. Також ШІ може аналізувати ринкові дані та виконувати торгівельні стратегії, отримуючи вигоду з незначних змін на ринку, і надавати рекомендації щодо інвестицій, враховуючи фінансовий стан клієнтів.

Освітні інструменти на основі ШІ пропонують персоналізовані навчальні досвіди та автоматизують адміністративні завдання. Системи ШІ адаптують навчальні матеріали відповідно до індивідуальних потреб студентів, а також допомагають викладачам автоматизувати оцінювання

робіт і надання зворотного зв'язку. Мовні моделі ШІ створюють інтерактивні навчальні платформи і контент, роблячи навчання більш доступним та ефективним.

У сфері обслуговування клієнтів чат-боти та віртуальні асистенти на основі ШІ підвищують якість обслуговування, надаючи миттєві відповіді на запити клієнтів. Чат-боти відповідають на поширені питання, надаючи швидку та точну інформацію, а віртуальні асистенти допомагають клієнтам виконувати різні завдання, роблячи процес взаємодії з компанією більш зручним та ефективним.

У галузі розваг ШІ сприяє створенню контенту, системам рекомендацій та інтерактивним досвідам. Генеративні моделі ШІ створюють сценарії для фільмів та телесеріалів, музичні композиції та візуальне мистецтво, сприяючи розвитку креативності. Системи рекомендацій аналізують уподобання користувачів і надають персоналізовані рекомендації, а інтерактивні відеоігри на основі ШІ пропонують нові форми розваг, адаптуючись до дій гравців.

3.2 Труднощі та етичні міркування

Попри численні переваги, ШІ також представляє виклики та етичні питання, які необхідно враховувати для його відповідального впровадження.

Одним з ключових викликів є упередженість та справедливість. Моделі ШІ можуть успадковувати упередження з даних, на яких вони навчаються, що може призвести до несправедливих результатів. Наприклад, якщо дані, на яких навчалася модель, містять расові або гендерні упередження, модель може відтворювати ці упередження у своїх прогнозах або рекомендаціях. Це може мати серйозні наслідки, особливо у таких критичних галузях, як правосуддя або охорона здоров'я. Забезпечення справедливості та усунення упереджень є критично важливими для етичного впровадження ШІ.

Конфіденційність та безпека є ще однією важливою проблемою, оскільки використання ШІ для обробки великих обсягів даних викликає

занепокоєння щодо конфіденційності та безпеки даних. Наприклад, системи на основі ШІ можуть збирати та аналізувати великі обсяги персональних даних, що підвищує ризик витоків інформації або зловживань. Захист чутливої інформації та забезпечення цілісності даних є необхідними для довіри користувачів до технологій ШІ.

Прозорість та відповідальність також є важливими аспектами у впровадженні ШІ. Складність моделей ШІ може ускладнити розуміння їх процесів прийняття рішень, що створює ризик "чорних скриньок". Це означає, що користувачі та навіть розробники можуть не повністю розуміти, як модель дійшла до певного висновку або рекомендації. Сприяння прозорості та притягнення систем ШІ до відповідальності за їхні дії є надзвичайно важливими для забезпечення їх етичного використання.

Витіснення робочих місць є ще одним значущим викликом, оскільки автоматизація завдань за допомогою ШІ може призвести до втрати робочих місць у певних секторах. Наприклад, автоматизація виробничих процесів може зменшити потребу у ручній праці, що може призвести до безробіття серед працівників цих секторів. Вирішення соціально-економічного впливу ШІ та надання можливостей для перенавчання та підвищення кваліфікації є важливими для забезпечення справедливого переходу до більш автоматизованого суспільства.

Таким чином, попри численні переваги, які ШІ приносить у різні галузі, важливо враховувати та вирішувати виклики та етичні питання, щоб забезпечити його відповідальне та корисне впровадження. Врахування цих аспектів допоможе створити технології, які будуть сприяти загальному благополуччю та підвищувати якість життя для всіх.

3.3 Підгалузі штучного інтелекту

Штучний Інтелект (ШІ) в цілому вважається підгалуззю комп'ютерних наук і має багато власних підгалузей, які зображені на рис. 1.

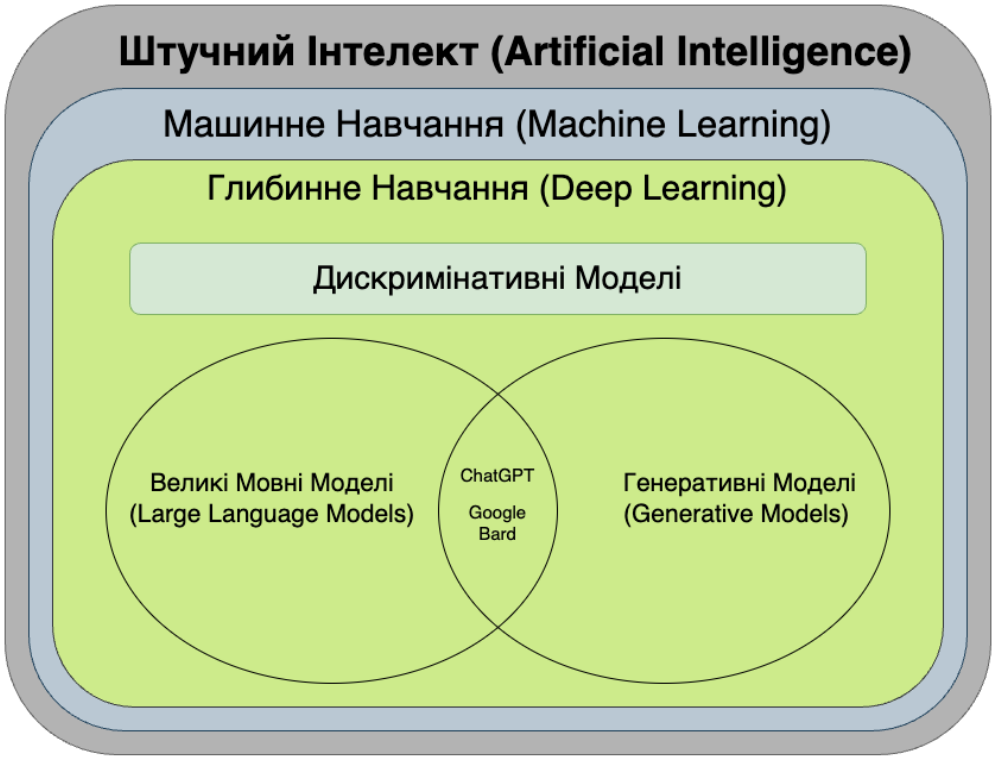

Рисунок 1 – Підгалузі штучного інтелекту

Першою головною підгалуззю ШІ є Машинне навчання (Machine Learning або ML). В свою чергу, підгалуззю Машинного навчання є Глибинне навчання (Deep Learning). Моделі Глибинного навчання поділяються на Дискримінативні Моделі (Discriminative Models), Генеративні Моделі (Generative Models) та Великі Мовні Моделі (Large Language Models). На пересіченні Генеративних Моделей та Великих Мовних Моделей є технологія яка живить додатки які так нам знайомі – ChatGPT та Google Bard.

3.4 Детальний опис машинного навчання

Машинне навчанне сфокусоване на тому, щоб за допомогою тренувальних даних навчити спеціальну комп'ютерну програму – модель – виконувати певні дії. Після тренування моделі таким способом, вона зможе виконувати подібні дії на основі нових даних (це називається висновування (inference)). Наприклад, якщо модель була натренована робити комерційні прогнози на основі даних про продажі кросівок Nike, то після такого

тренування модель зможе прогнозувати продажі кросівок Adidas, якщо моделі будуть надані дані від Adidas.

Виділяють чотири основні методи машинного навчання – контрольоване навчання (supervised learning), неконтрольоване навчання (unsupervised learning), навчання з підкріпленням (reinforcement learning) та навчання з перенесенням (transfer learning).

3.4.1 Контрольоване навчання

Контрольоване навчання (supervised learning) – це метод, при якому модель навчається на маркерованих даних. Вхідні дані мають відповідні виходи, і мета полягає у тому, щоб навчити модель передбачати виходи на основі нових вхідних даних.

На рис. 2 зображено графік чайових відносно типу заказу у ресторані. Дані є маркерованими. Сині крапки позначають закази які були забрані самовивозом, а жовті закази, які були доставлені кур'єром.

Рисунок 2 – Графік чайових відносно типу заказу

Використовуючи модель, яка була натренована методом контрольованого навчання можемо спрогнозувати суму чайових для наступних заказів враховуючи тип доставки.

3.4.2 Неконтрольоване навчання

Метод неконтрольованого навчання (unsupervised learning) передбачає роботу з немаркерованими даними. На рис. 3 зображено графік доходу відносно стажу роботи у компанії. При неконтрольованому навчанні ми працюємо з немаркерованими даними і дивимось чи вони органічно самі собою утворюють логічні групи. Як можна побачити на графіку (див. рис. 3), група працівників, яка обведена зеленим кольором мають відносно велике співвідношення доходу відносно стажу роботи, а групи обведена червоним мале.

Рисунок 3 – Графік доходу відносно стажу роботи

У моделі натренованою методом неконтрольованого навчання можна запитати, наприклад, чи знаходиться новий працівник який щойно почав працювати у компанії на прискореному кар'єрному шляху або ні. Якщо такий

працівник попаде у групу позначену зеленим кольором – відповідь буде "Так". Відповідно, якщо такий працівник попаде у групу позначену червоним, відповідь буде "Ні".

Варто зауважити, що при контрольованому навчанні (на відміну від неконтрольованого), після того, як модель робить прогноз – вона порівнює цей прогноз з даними, які були використані для тренування цієї моделі, і якщо є відхилення – вона намагається його мінімізувати.

3.4.3 Опис навчання з підкріпленням

Навчання з підкріплення (reinforcement learning або RL) є однією з гілок машинного навчання, яка концентрується на навчанні моделей взаємодіяти з оточуючим середовищем для досягнення певної мети. У процесі навчання модель отримує зворотний зв'язок у вигляді нагород або штрафів залежно від своїх дій. Метою моделі є максимізація сумарної винагороди за певний період часу, що досягається шляхом вдосконалення своєї стратегії дій.

Основна ідея RL полягає в тому, що модель повинна навчитися приймати рішення в умовах невизначеності. Це досягається через метод проб і помилок, де модель намагається різні дії та аналізує результати цих дій. З часом модель удосконалюється, оскільки накопичує знання про те, які дії приводять до кращих результатів.

Одним з важливих аспектів RL є баланс між дослідженням (exploration) і використанням (exploitation). Дослідження включає в себе пробування нових дій, які модель ще не випробувала, щоб знайти потенційно кращі стратегії. Використання означає застосування вже відомих дій, які приносять високі нагороди. Успішне навчання з підкріплення вимагає збалансованого підходу до цих двох аспектів.

RL знаходить своє застосування в різних галузях. У робототехніці воно використовується для навчання роботів виконувати складні завдання, такі як навігація в незнайомих середовищах або маніпуляція об'єктами. В ігровій

індустрії RL допомагає створювати штучних гравців, які можуть змагатися з людськими гравцями або навіть перевершувати їх. В автономних системах, таких як безпілотні автомобілі, RL використовується для розробки алгоритмів, що дозволяють транспортним засобам безпечно пересуватися в умовах дорожнього руху.

Таким чином, навчання з підкріплення є потужним інструментом у сучасному світі штучного інтелекту, що дозволяє створювати адаптивні та інтелектуальні системи, здатні приймати рішення в умовах невизначеності та змінних обставин.

3.4.4 Аналіз навчання з перенесенням

Навчання з перенесенням (transfer learning) є важливим напрямком у сфері машинного навчання, що дозволяє моделі, навченій на одній задачі, використовувати набуті знання для вирішення іншої, пов'язаної задачі. Це підходить для ситуацій, коли даних для нової задачі недостатньо для навчання моделі з нуля. Основна ідея полягає в тому, щоб скористатися попередньо навченими моделями, які вже мають певні знання, і адаптувати їх до нових умов. Це дозволяє значно скоротити час і ресурси, необхідні для навчання нової моделі, і підвищити її ефективність. Навчання з перенесенням активно використовується в різних галузях, включаючи розпізнавання зображень, обробку природної мови та медичну діагностику, де воно допомагає досягти високих результатів навіть за наявності обмежених даних. Завдяки цьому підходу моделі можуть бути більш гнучкими та універсальними, оскільки вони здатні адаптуватися до різних задач, використовуючи вже набуті знання.

3.5 Характеристика глибинного навчання

Глибинне навчання – це підгалузь машинного навчання, при якому використовуються штучні нейронні мережі (Artificial Neural Networks) –

математичні, програмні та апаратні моделі, побудовані за принципом роботи біологічних нейронних мереж клітин живого організму.

3.5.1 Аналіз штучної нейронної мережі

Нейронні мережі є складною системою, яка включає математичні моделі, програмне забезпечення та апаратне забезпечення. На математичному рівні, кожен нейрон є об'єктом, який обчислює зважену суму вхідних сигналів і передає результат через нелінійну функцію активації. Це імітує функціонування біологічних нейронів, які передають електрохімічні сигнали через синапси. Біологічні нейрони взаємодіють через складні мережі, що дозволяє їм виконувати складні обчислювальні завдання, такі як розпізнавання образів та прийняття рішень.

3.5.1.1 Опис математичного рівня

На рис. 4 показана структура типової нейронної мережі. Кола на зображенні представляють нейрони, які є основними будівельними блоками нейронної мережі. Вони організовані у три шари – шар входу, прихований шар і шар виходу. Зображення іллюструє те, як інформація проходить через різні шари, де кожен нейрон обробляє вхідні дані та передає їх далі.

Чому нейрони позначені колом? Коло є простим і зрозумілим символом, який легко впізнати. Воно допомагає швидко ідентифікувати нейрон як окрему одиницю в мережі. Нейрон виконує складні обчислення, включаючи зважування входів, додавання зміщення і застосування функції активації. Коло дозволяє абстрагувати ці деталі і представити нейрон як простий об'єкт, який приймає входи і видає вихід.

Повертаючись до структури нейронної мережі, розглянемо окремо кожний шар. У шарі входу (input layer) нейрони отримують початкові дані ззовні. Вони приймають вхідні сигнали і передають їх на прихований шар.

У прихованому шарі (hidden layer) нейрони обробляють дані, що надійшли з шару входу. Вони виконують різні обчислення і передають

результати на шар виходу. Кількість прихованих шарів і нейронів у кожному з них може варіюватися залежно від складності та типу нейронної мережі.

У шарі виходу (output layer) нейрони генерують кінцевий результат обробки даних, який видається назовні. Вони отримують оброблені дані з прихованого шару і виробляють вихідні сигнали, які можуть бути кінцевим результатом, наприклад, класифікацією зображення або передбаченням значення.

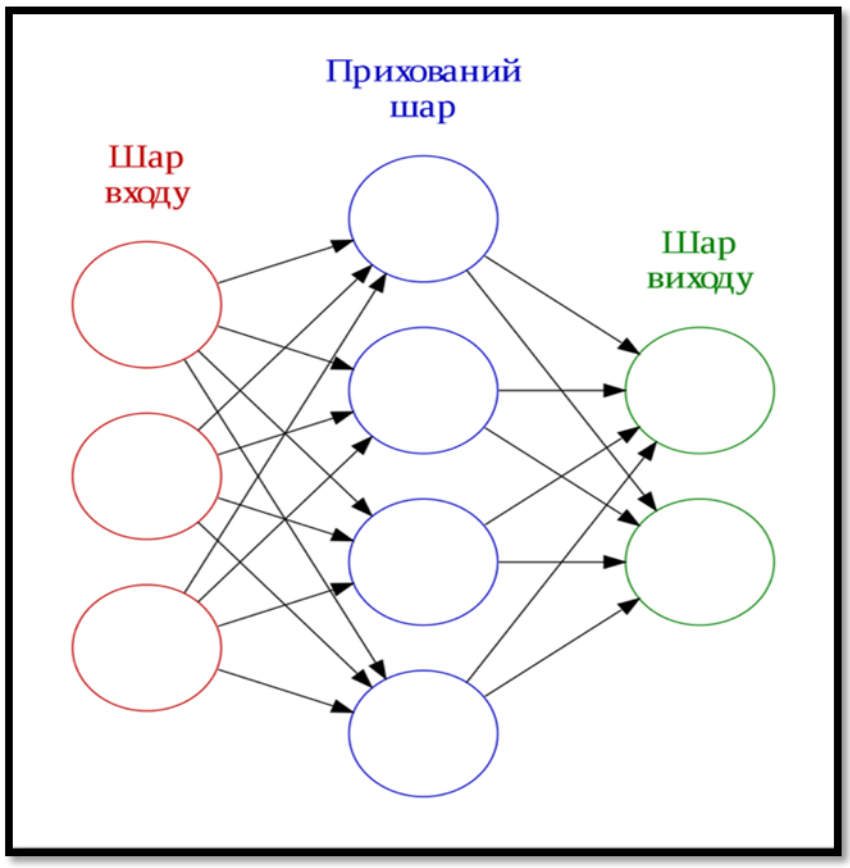

Рисунок 4 – Штучна Нейронна Мережа

Стрілки між колами представляють зв'язки між нейронами, через які передаються зважені сигнали. Кожна стрілка відповідає певній вазі, яка визначає важливість сигналу, що передається.

3.5.1.2 Програмний рівень

У програмному забезпеченні нейронні мережі можуть бути реалізовані за допомогою високорівневих мов програмування, таких як Python, а саме за

допомогою бібліотек TensorFlow, PyTorch та Keras. Ці бібліотеки надають інструменти для визначення архітектури мережі, функцій втрат та оптимізаторів, що дозволяє ефективно створювати, навчати та оцінювати моделі. На нижчому рівні, нейронні мережі можуть бути реалізовані на мові C, що забезпечує кращу продуктивність та контроль над апаратним забезпеченням, особливо у вбудованих системах. Ще нижчий рівень – це реалізація на асемблері, де програміст працює безпосередньо з інструкціями процесора, що забезпечує максимальну продуктивність, але вимагає глибоких знань архітектури процесора.

3.5.1.3 Апаратний рівень

Для апаратної реалізації використовуються мови опису апаратури, такі як VHDL або Verilog, які дозволяють описувати цифрові схеми на рівні регістрів і логічних вентилів. Це дозволяє створювати спеціалізовані чіпи для виконання операцій нейронних мереж.

На найнижчому рівні апаратне забезпечення включає окремі електричні компоненти, такі як транзистори, резистори та конденсатори. Ці компоненти утворюють логічні вентилі та інші базові елементи цифрових схем, які в свою чергу необхідні для апаратного втілення штучних нейронних мереж.

3.5.2 Нейронні мережі і напівконтрольоване навчання

Застосування глибоких нейронних мереж у машинному навчанні є надзвичайно потужним інструментом, оскільки вони здатні навчатися складним паттернам і особливостям у великих обсягах даних. Однак, не завжди існує достатня кількість маркерованих даних для повноцінного тренування таких моделей. Тут на допомогу приходить напівконтрольоване навчання, яке дозволяє ефективно використовувати як маркеровані, так і немаркеровані дані. Глибокі нейронні мережі особливо корисні в цьому контексті, оскільки вони можуть автоматично витягувати корисні

особливості з немаркованих даних і використовувати їх для покращення загальної продуктивності моделі.

Без використання нейронних мереж напівконтрольоване навчання також можливе, проте воно покладається на інші алгоритми, такі як методи підтримувальних векторів, дерева рішень або графові методи. Ці методи можуть бути менш ефективними у витягуванні складних паттернів і можуть вимагати більше ручного налаштування та попередньої обробки даних для досягнення подібного рівня точності та узагальнюваності моделі.

Розглянемо напівконтрольоване навчання більш детально. На рис. 5 зображено основний принцип роботи напівконтрольованого навчання на прикладі виявлення шахрайства.

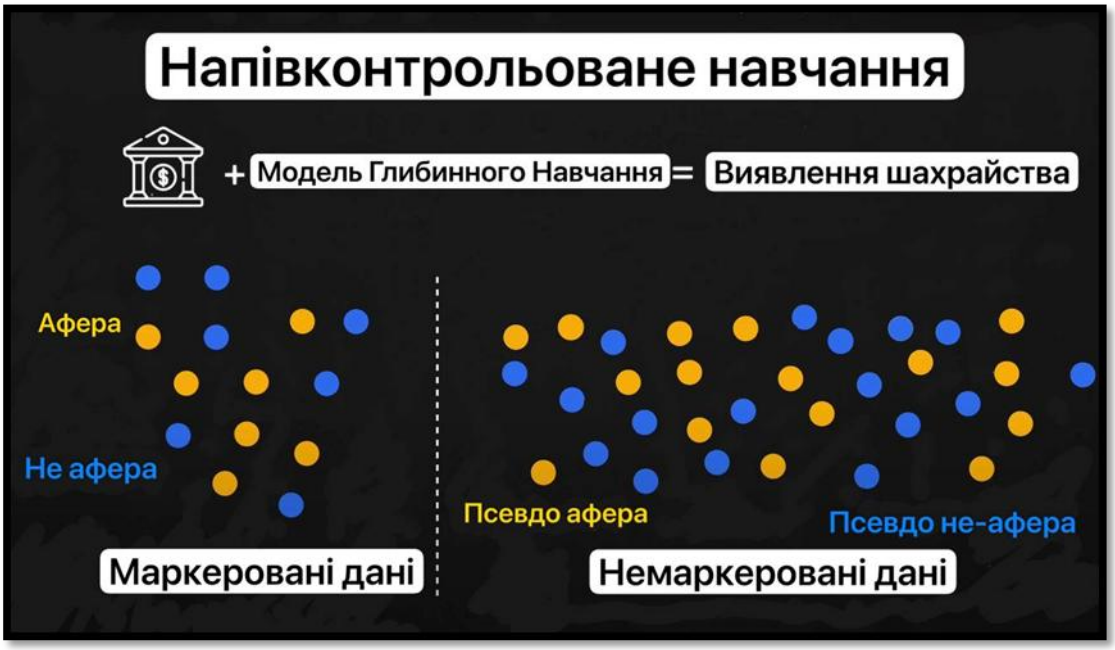

Рисунок 5 – Напівконтрольоване навчання

Береться великий обсяг даних і витрачаються зусилля на те, щоб промаркувати, наприклад, 5% відсотків всіх даних, бо на маркування всіх даних не вистачить часу або ресурсів. Використовуючи напівконтрольоване навчання, модель глибинного навчання використає ці 5% маркерованих даних, щоб робити передачення чи містять аферу ті 95% операцій, що залишилися.

Розглянувши принципи роботи моделей глибинного навчання, треба зазначити що ці моделі поділяються на дискримінативні, генеративні та великі мовні моделі, які будть розглянуті нижче у відповідних підзрозділах.

3.5.3 Дискримінативні моделі

Дискримінативні моделі використовуються для класифікації та розпізнавання об'єктів. Наприклад, якщо у нас є набір фотографій, де зображені коти та собаки, дискримінативна модель допомагає визначити, на яких з них зображені коти, а на яких – собаки.

Ці моделі фокусуються на тому, щоб знайти відмінності між класами даних, у нашому випадку між котами та собаками, не намагаючись змоделювати всі можливі варіанти зображень. Завдяки цьому дискримінативні моделі часто простіші у використанні та налаштуванні. Вони забезпечують високу точність класифікації, оскільки оптимізовані для того, щоб правильно розпізнавати клас об'єкта.

Наприклад, у [3] детально розглядаються переваги дискримінативних моделей у порівнянні з генеративними моделями. Крім того, у [4] також підкреслюється ефективність використання дискримінативних моделей у задачах класифікації.

Дискримінативні моделі широко застосовуються в задачах розпізнавання образів (таких як визначення, чи є на зображенні кіт або собака) і є ефективними інструментами для задач класифікації. Вони допомагають точно розпізнавати об'єкти, забезпечуючи високу ефективність та простоту у використанні.

3.5.4 Опис генеративних моделей

Генеративні моделі використовуються для створення нового тексту, зображень, аудіо та навіть відео. Генеративні моделі імітують людську творчість. Генеративні моделі намагаються змоделювати розподіл усіх даних, а не лише визначити відмінності між класами. Наприклад, якщо у нас є набір

фотографій, де зображені коти та собаки, генеративна модель вивчає всі можливі варіанти зображень котів та собак, щоб потім вміти створювати нові зображення, що виглядають як реальні коти та собаки.

Ці моделі намагаються зрозуміти, як виглядають дані в кожному класі, і створюють нові дані на основі цього розуміння. Завдяки цьому генеративні моделі можуть бути використані не лише для класифікації, а й для створення нових даних, таких як зображення, текст або музика. Це робить їх дуже потужним інструментом у багатьох творчих та наукових задачах.

Однією з найбільш відомих генеративних моделей є генеративно-змагальні мережі (GANs), які були представлені у [5]. Ця робота стала основою для багатьох досліджень у сфері генеративного машинного навчання. Іншим важливим ресурсом є стаття [6], яка описує варіаційні автоенкодери (VAEs), ще один тип генеративних моделей.

Генеративні моделі широко використовуються у різних галузях. Наприклад, вони можуть створювати нові реалістичні зображення на основі навчання на великій кількості існуючих зображень. В обробці природної мови, моделі, такі як GPT (Generative Pre-trained Transformer), можуть створювати текст, що виглядає як написаний людиною. У наукових дослідженнях генеративні моделі допомагають у симуляції даних для експериментів, що особливо корисно у біології та фізиці.

3.5.5 Характреистика великих мовних моделей

Великі мовні моделі (Large Language Models або LLMs) – це тип ШІ-моделей, створених для розуміння та генерації людської мови. Ці моделі навчаються на великих обсягах текстових даних, що дозволяє їм вивчати складнощі мови, включаючи граматику, контекст та нюанси. LLMs побудовані з використанням глибинних навчальних технік, зокрема нейронних мереж, і мають мільярди параметрів, які сприяють їх здатності до розуміння мови.

Однією з ключових особливостей LLMs є їх здатність виконувати широкий спектр завдань, пов'язаних з мовою. До таких завдань належать переклад мови, резюмування текстів, відповіді на запитання та навіть креативне письмо. LLMs можуть генерувати зв'язний та контекстуально релевантний текст, що робить їх корисними у різних застосуваннях в різних галузях.

Великі мовні моделі (LLMs) спеціально розроблені для обробки та генерації природної мови. Вони навчаються на величезних обсягах текстових даних, що дозволяє їм розуміти контекст, структуру та зміст мови. Наприклад, якщо у нас є набір текстів, великі мовні моделі можуть передбачати наступне слово або навіть генерувати цілі абзаци, які виглядають як написані людиною.

Одним з найвизначніших досягнень у цій сфері є розробка моделі GPT (Generative Pre-trained Transformer) компанією OpenAI. В статті [7] було представлено архітектуру трансформерів, яка стала основою для багатьох сучасних великих мовних моделей. Модель GPT-3, описана у статті [8], показала здатність генерувати високоякісні тексти та виконувати широкий спектр мовних завдань.

Великі мовні моделі використовуються в різних галузях. Наприклад, вони можуть автоматично створювати контент для веб-сайтів, відповідати на запитання клієнтів у чат-ботах, перекладати тексти з однієї мови на іншу, а також виконувати багато інших завдань, що пов'язані з обробкою природної мови. У наукових дослідженнях LLMs допомагають аналізувати великі обсяги текстових даних, знаходити нові знання та робити передбачення.

3.5.6 Взаємозв'язок LLMs та GenAI

Великі мовні моделі (LLMs) та генеративний штучний інтелект (GenAI) є потужними інструментами в сучасному машинному навчанні, які тісно пов'язані між собою. LLMs, такі як GPT (Generative Pre-trained Transformer)

та Google Bard, є прикладами моделей, що знаходяться на перетині цих двох технологій.

LLMs навчаються на величезних обсягах текстових даних, що дозволяє їм розуміти контекст, структуру та зміст мови. Це робить їх надзвичайно ефективними для генерації текстів, перекладу, відповідей на запитання та інших завдань, пов'язаних з обробкою природної мови. Однією з найвизначніших робіт у цій сфері є стаття [7], де було представлено архітектуру трансформерів, що стала основою для багатьох сучасних LLMs.

Генеративний штучний інтелект, з іншого боку, спеціалізується на створенні нових даних, таких як зображення, текст або музика, на основі вивчених патернів. Одним з найпопулярніших прикладів GenAI є генеративно-змагальні мережі (GANs), описані у статті [5].

Перетин LLMs та GenAI найбільш яскраво демонструється на прикладі таких моделей, як GPT-3 та Google Bard. GPT-3, описана у статті [8], є великою мовною моделлю, що може генерувати високоякісні тексти на основі контексту, заданого користувачем. Вона здатна виконувати широкий спектр завдань, включаючи написання статей, переклад текстів, створення віршів та багато іншого. Google Bard, аналогічно, використовує можливості LLMs для генерації текстів і відповіді на запитання, інтегруючи це в широкий спектр додатків Google.

Однією з важливих особливостей GPT-3 є підтримка few-shot learning, яка дозволяє моделі виконувати нові завдання з мінімальною кількістю прикладів. Це означає, що модель може навчитися розв'язувати нову задачу, отримавши всього кілька прикладів, що значно розширює її можливості та ефективність у реальних застосуваннях.

Ці моделі знаходяться на перетині LLMs та GenAI, оскільки вони не тільки розуміють та обробляють природну мову, але й створюють нові текстові дані, що виглядають як написані людиною. Це робить їх надзвичайно корисними в багатьох галузях, таких як автоматизація контенту, обслуговування клієнтів, наукові дослідження та багато інших.

4 ПРОЄКТУВАННЯ ПЛАТФОРМИ

Цей розділ присвячений проєктуванню платформи. У підрозділі 4.1 наведено загальну архітектуру платформи, а наступні підрозділи фокусуються на UML-діаграмах (варіантів використання (прецедентів) системи, діяльності (активностей), послідовностей).

4.1 Загальна архітектура платформи

Загальна архітектура додатку проіллюстрована нижче (див. рис. 6).

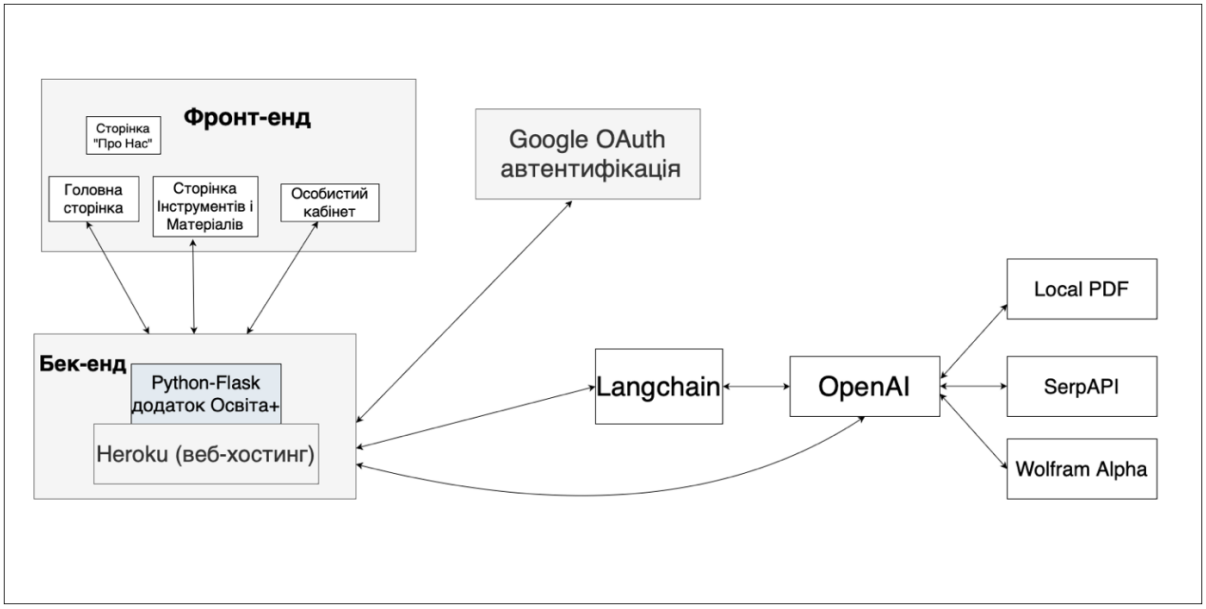

Рисунок 6 – Загальна архітектура додатку

Ця архітектура забезпечує інтеграцію різних інструментів та сервісів, що дозволяє користувачам легко орієнтуватися та зручно використовувати всі функціональні можливості платформи.Розділи сайту, такі як головна сторінка, сторінка з інструментами та матеріалами, а також особистий кабінет користувача, створені для забезпечення зручності використання та доступу до різноманітних функцій. Інтеграція з Google OAuth дозволяє забезпечити безпечний вхід користувачів до системи, використовуючи їхні облікові записи Google. Розміщення додатку на платформі Heroku забезпечує легко-налаштовуємий веб-хостинг з можливістю масштабізації за потреби у

майбутньому. Інтерфейс програми побудований таким чином, щоб користувачі могли легко орієнтуватися та використовувати всі функціональні можливості платформи.

4.2 Побудова діаграми варіантів використання (прецедентів) системи

Діаграма варіантів використання (прецедентів) системи є важливим елементом в процесі розробки програмного забезпечення. Вона дозволяє візуалізувати взаємодію користувачів (акторів) із системою через різні сценарії використання, які називаються прецедентами. Кожен прецедент відображає конкретний спосіб використання системи, описуючи, як користувачі взаємодіють з функціональністю програмного продукту.

Створення діаграми варіантів використання допомагає розробнику краще зрозуміти вимоги до системи, забезпечує чітке уявлення про функціональні можливості і дозволяє виявити потенційні проблеми на ранніх етапах розробки.

Діаграма варіантів використання складається з акторів, які представляють зовнішні системи або користувачів, та прецедентів, які описують взаємодію акторів із системою. Зв’язки між акторами та прецедентами відображають, які функції системи є доступними для яких користувачів. Ця діаграма може містити додаткові елементи, такі як узагальнення, включення та розширення, що допомагають деталізувати та структурувати варіанти використання.

Діаграма варіантів використання (прецедентів) системи для платформи Освіта+ наведена на рис. 7.

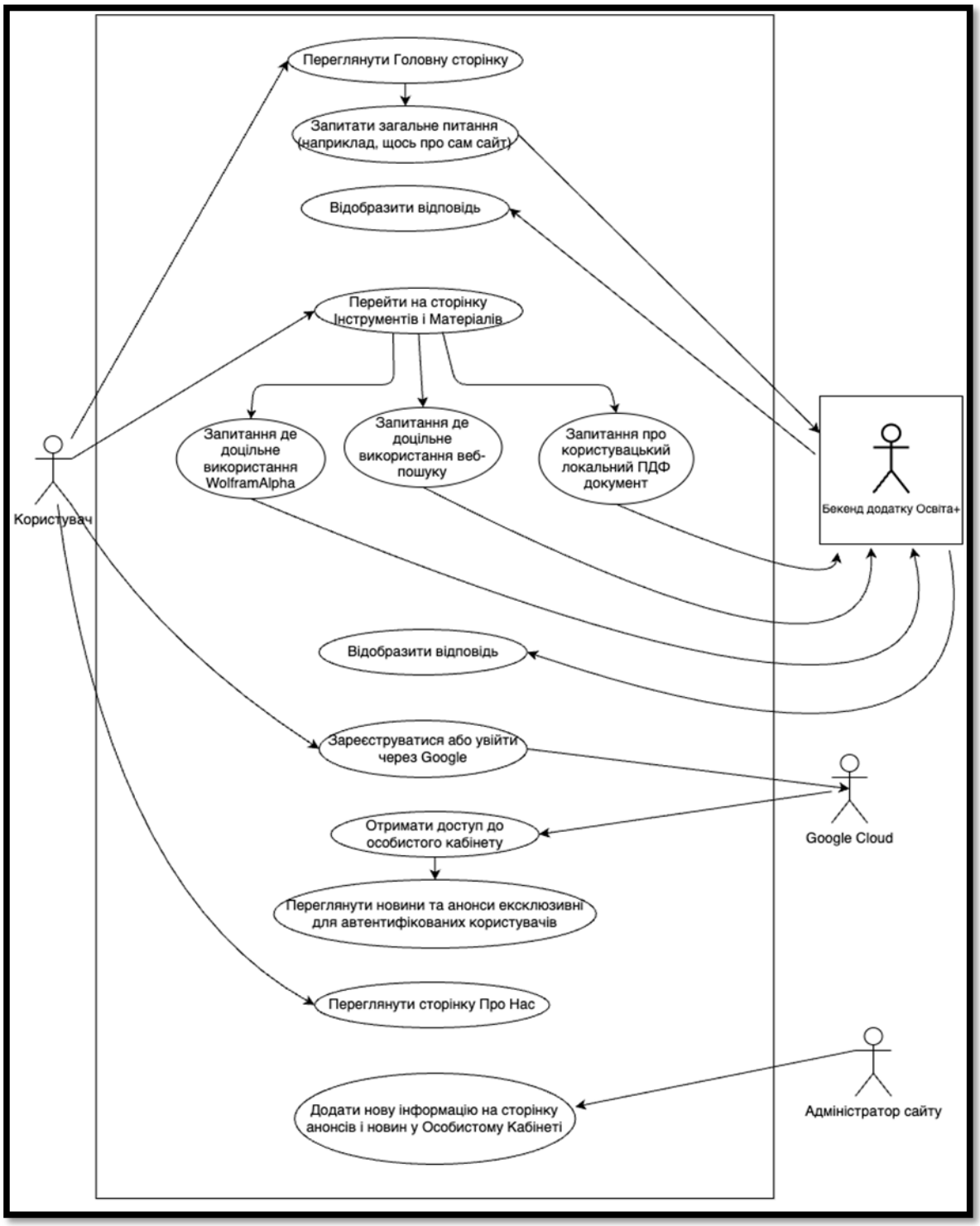

Рисунок 7 – Діаграма варіантів використання (прецедентів) системи

Загалом, діаграма варіантів використання є потужним інструментом для моделювання поведінки системи та забезпечення її відповідності вимогам замовника. Вона надає можливість побудувати чітку та зрозумілу документацію, що сприяє успішній реалізації проекту.

Діаграми варіантів використання також грають важливу роль у плануванні тестування програмного забезпечення. Вони допомагають визначити основні сценарії, які необхідно перевірити, щоб забезпечити правильну роботу системи. На основі прецедентів можна розробити тестові випадки, що охоплюють усі можливі взаємодії користувачів із системою, включаючи як стандартні, так і виняткові ситуації. Це дозволяє виявити потенційні недоліки та помилки ще на стадії моделювання, що значно знижує ризик виникнення критичних проблем під час експлуатації системи.

4.3 Побудова діаграми діяльності (активностей)

Для моделювання процесу роботи з додатком Освіта+ використовується діаграма діяльності. На діаграмі діяльності відображається логіка або послідовність переходу від однієї діяльності до іншої, при цьому увага фіксується на результаті діяльності. Сам результат може призвести до зміни стану системи або повернення деякого значення.

На рис. 8 зображена діаграма діяльності, яка відображає основний процес роботи з додатком, включаючи входження користувача, використання інструментів та доступ до персонального кабінету.

Діаграма діяльності також допомагає виявити можливі вузькі місця та оптимізувати процеси. Вона надає чітке уявлення про те, які дії виконуються послідовно, а які можуть виконуватися паралельно, забезпечуючи більш ефективне використання ресурсів.

Крім того, діаграма діяльності може бути корисною для документування бізнес-процесів та навчання нових користувачів і розробників, оскільки вона демонструє візуальне представлення роботи додатка.

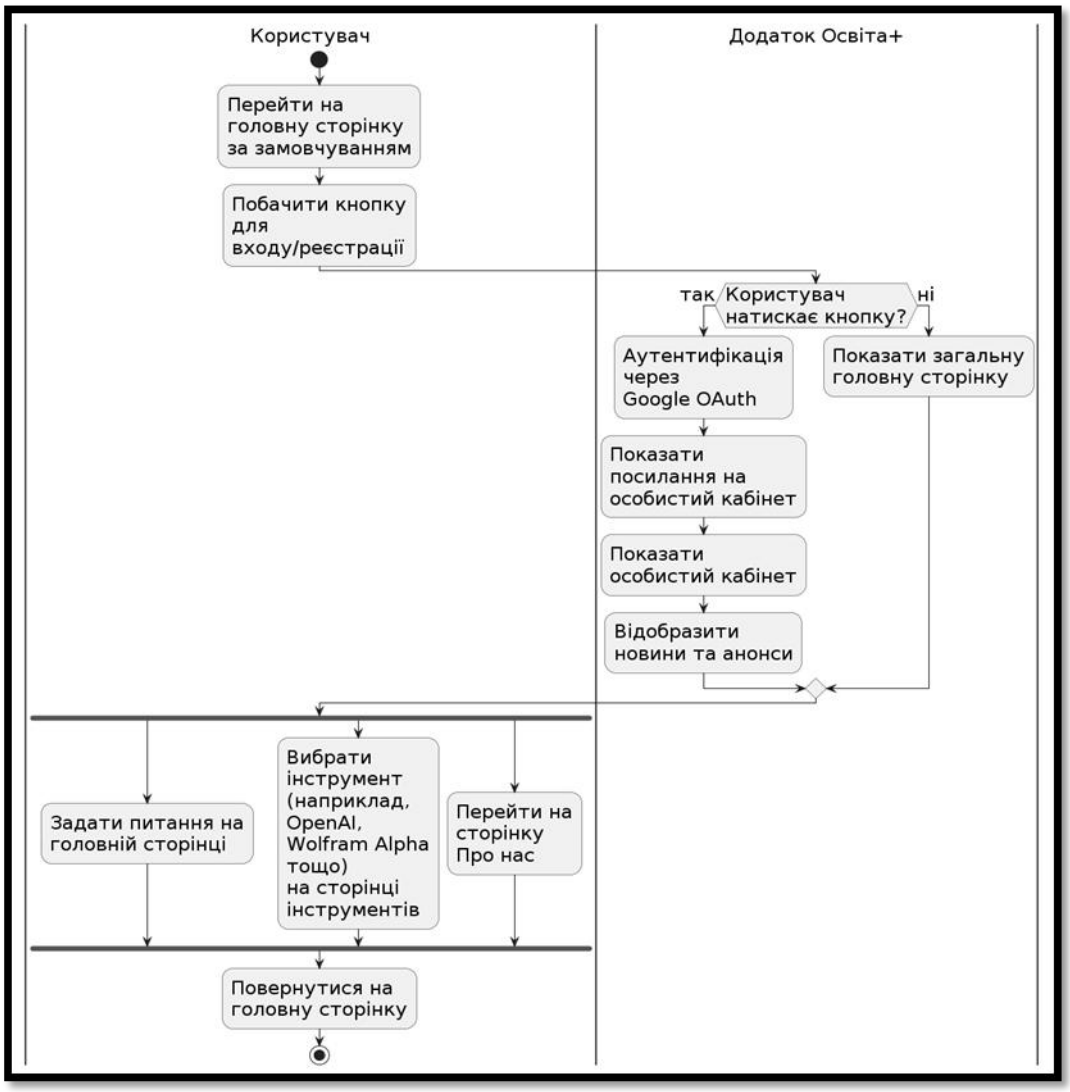

Рисунок 8 – Діаграма діяльності (активностей)

4.4 Побудова діаграми послідовностей

На рис. 9 показана діаграма послідовностей для додатку Освіта+. Такі діаграми використовуються для уточнення логіки взаємодії між різними компонентами системи і більш детального опису сценаріїв використання. Головна послідовність сценарію для додатку Освіта+ складається з кількох ключових взаємодій, які відображені на діаграмі: користувач переходить на головну сторінку, натискає кнопку для входу/реєстрації, проходить аутентифікацію через Google OAuth, і після успішної аутентифікації отримує доступ до персонального кабінету, де відображаються новини та анонси.

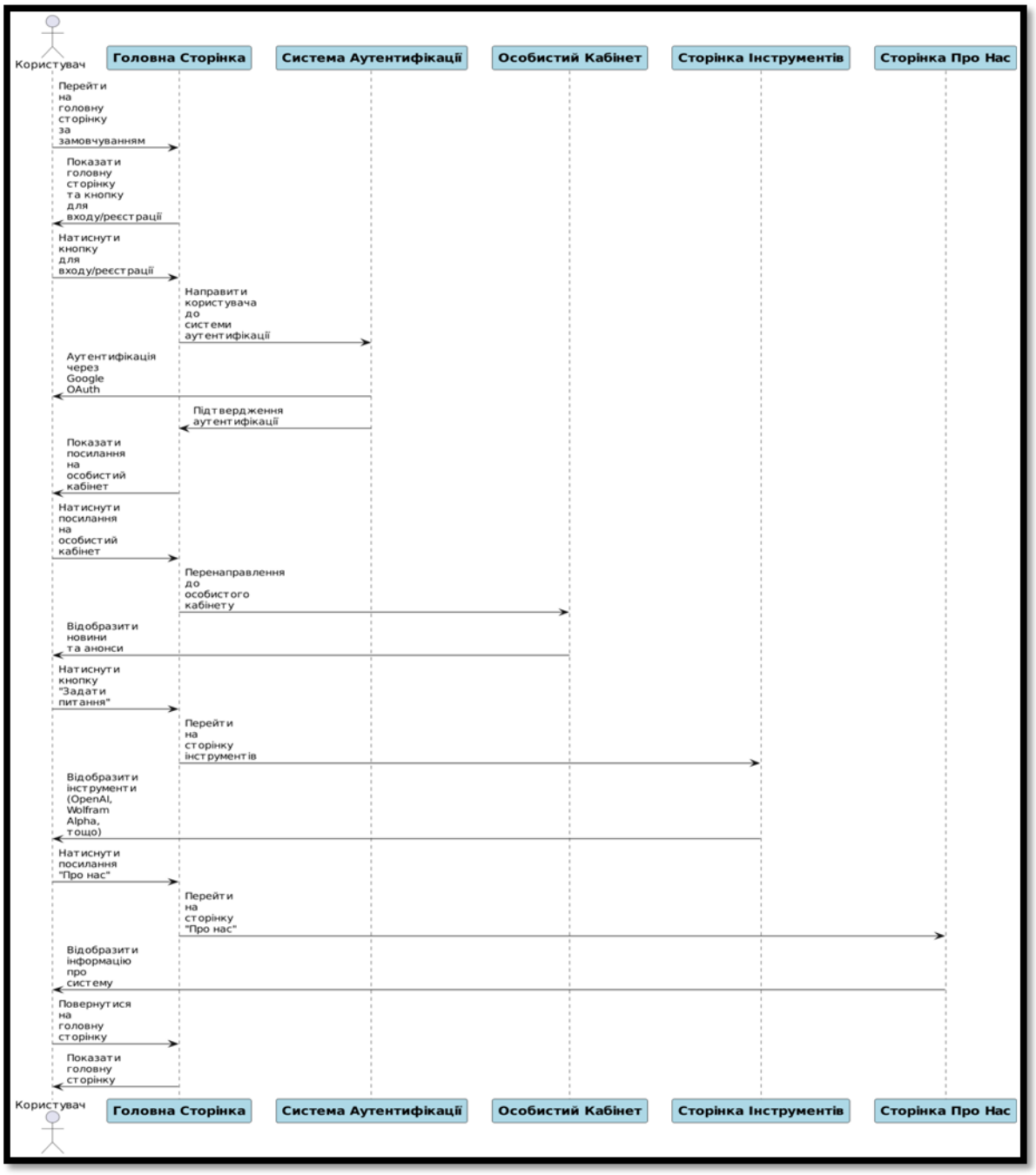

Рисунок 9 – Діаграма послідовностей

Крім того, користувач має можливість використовувати інструменти, такі як OpenAI і Wolfram Alpha, або переглядати інформацію про систему на сторінці "Про нас". Ці взаємодії детально відображені на діаграмі послідовностей для забезпечення чіткого розуміння процесу роботи з додатком.

5 РЕАЛІЗАЦІЯ ПЛАТФОРМИ

Повний програмний код додатку наведено у додатку А.

5.1 Отримання API-ключів

Backend-частина додатку потребує отриманя API-ключів від OpenAI, WolframAlpha та Serp API. Розглянемо процес отримання ключів та основні кроки інтеграції.

5.1.1 Отримання API-ключа OpenAI

Для отримання API-ключа OpenAI потрібно перейти на офіційний сайт OpenAI (https://www.openai.com/) та зареєструватися або увійти у свій обліковий запис (рис. 10).

Після входу до облікового запису треба перейти до розділу "API" у профілі користувача та натиснути на "Create API Key" для створення нового ключа.

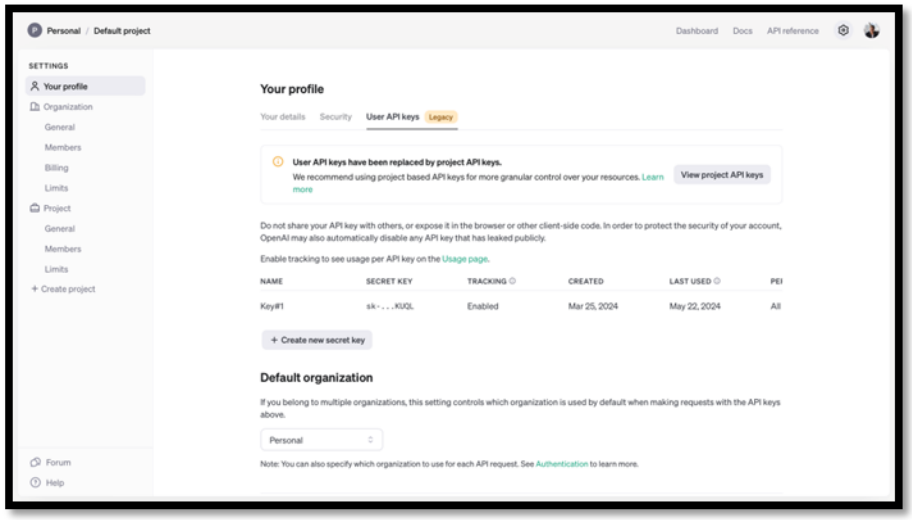

Рисунок 10 – Отриманий OpenAI API ключ

Після цього треба зберегти отриманий API ключ у безпечному місці.

5.1.2 Отримання API-ключа WolframAlpha

Для отримання API-ключа WolframAlpha потрібно перейти на офіційний сайт Wolfram Alpha (https://www.wolframalpha.com/) та зареєструватися або увійти у свій обліковий запис (див. рис. 11). Після входу до облікового запису треба перейти до розділу "Developer Portal" або "API" у профілі користувача та натиснути на "Get API Key" для створення нового ключа.

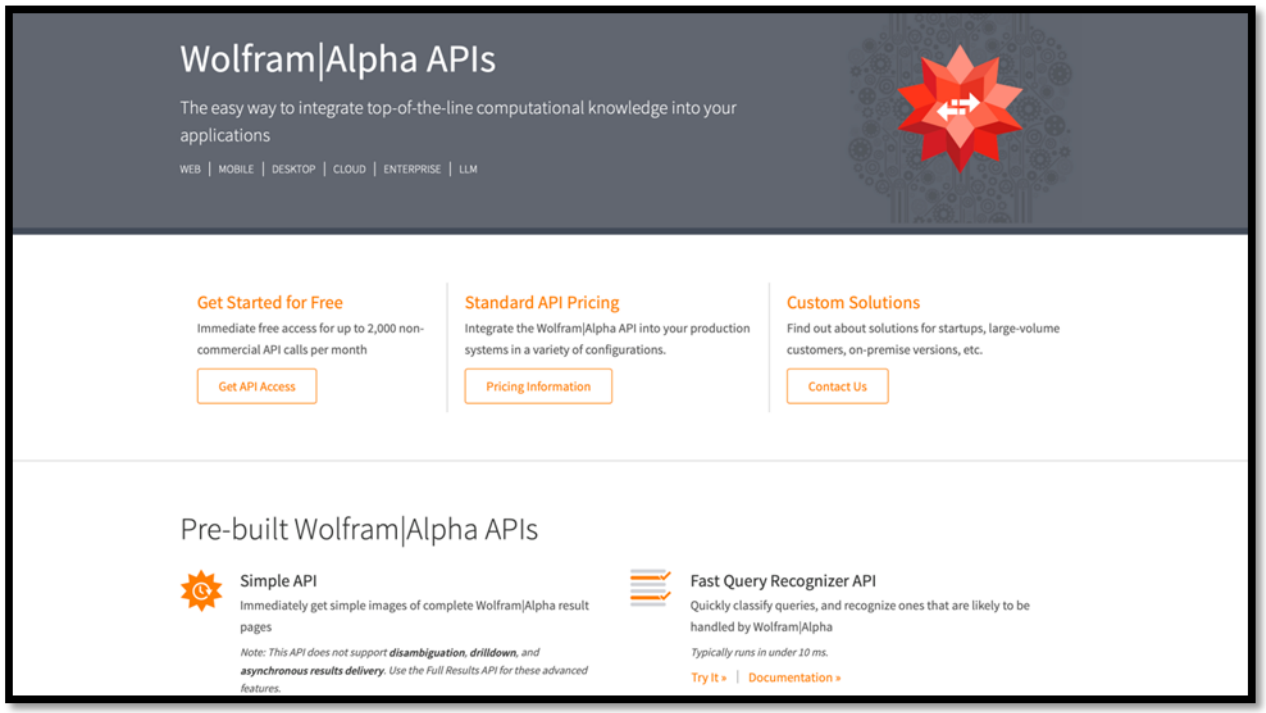

Рисунок 11 – Отримання WolframAlpha API ключа

Після цього зберегти отриманий API ключ у безпечному місці.

5.1.3 Отримання API-ключа SerpAPI

Для отримання API-ключа SerpApi потрібно перейти на офіційний сайт SerpAPI (https://serpapi.com/) та зареєструватися або увійти у свій обліковий запис (рис. 12). Після входу до облікового запису треба перейти до розділу "Dashboard" або "API Keys" у профілі користувача та натиснути на "Create New API Key" для створення нового ключа.

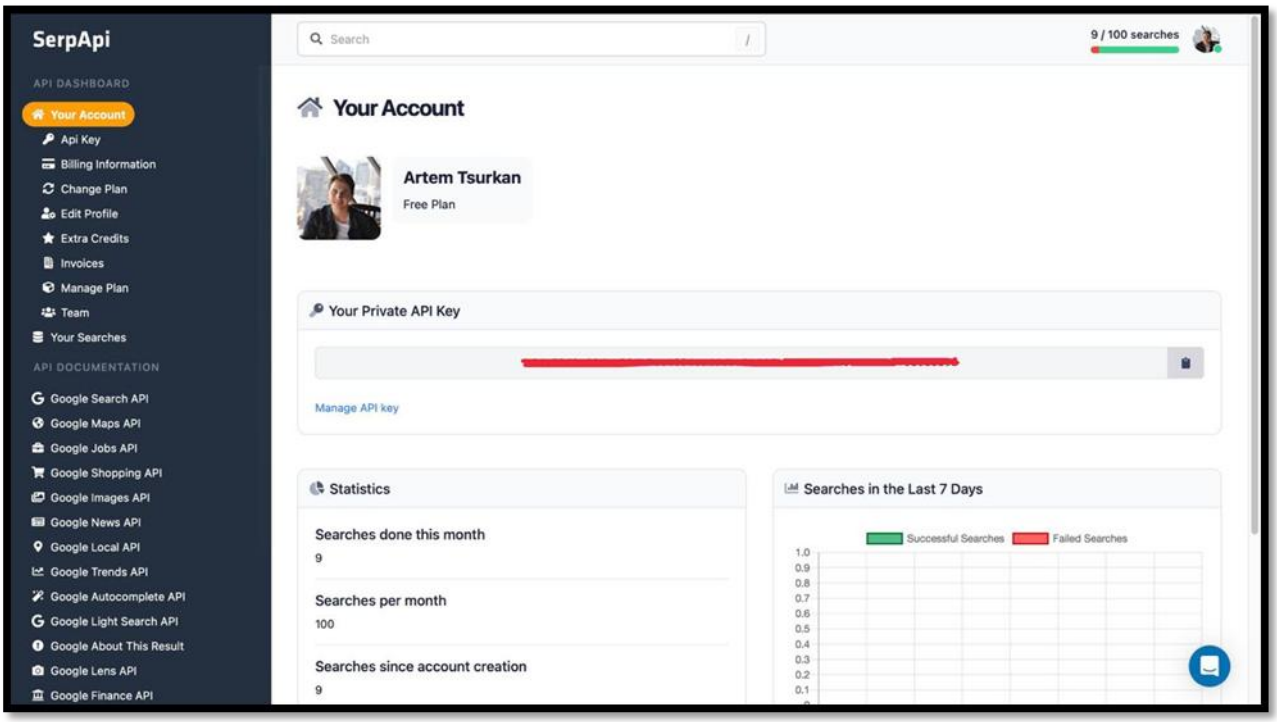

Рисунок 12 – Отримання ключа SerpAPI

Після цього зберегти отриманий API ключ у безпечному місці.

5.2 Реалізація фронтенд та бекенд додатку

Фронтенд додатку реалізовано з використанням HTML, CSS, та JavaScript (зокрема, jQuery). Основні компоненти включають головну сторінку, сторінку інструментів та матеріалів, а також особистий кабінет користувача. Бекенд реалізовано з використанням Flask (Python) для обробки запитів та інтеграції з різними API.

5.2.1 Опис головної сторінки

На домашній сторінці Освіта+ знаходиться вступна секція, мета якої зацікавити користувачів. Тут можна легко ознайомитись із основними можливостями та концепцією сайту, а також отримати швидкий доступ до всіх інших розділів (див. рис. 13). У шапці сайту знаходиться кнопка входу або реєстрації, яка інтегрована через систему Google OAuth, забезпечуючи безпечний та зручний доступ до особистих акаунтів користувачів. Ця кнопка

доступна з будь-якої сторінки сайту. Після того, як користувач авторизувався, кнопка входу або реєстрації змінюється кнопкою виходу, а у шапці сайту з'являється можливість перейти до особистого кабінету, звідки користувач зможе ознайомитись з останніми новинами та анонсами, які будуть періодично там публікуватися (див. рис. 14).

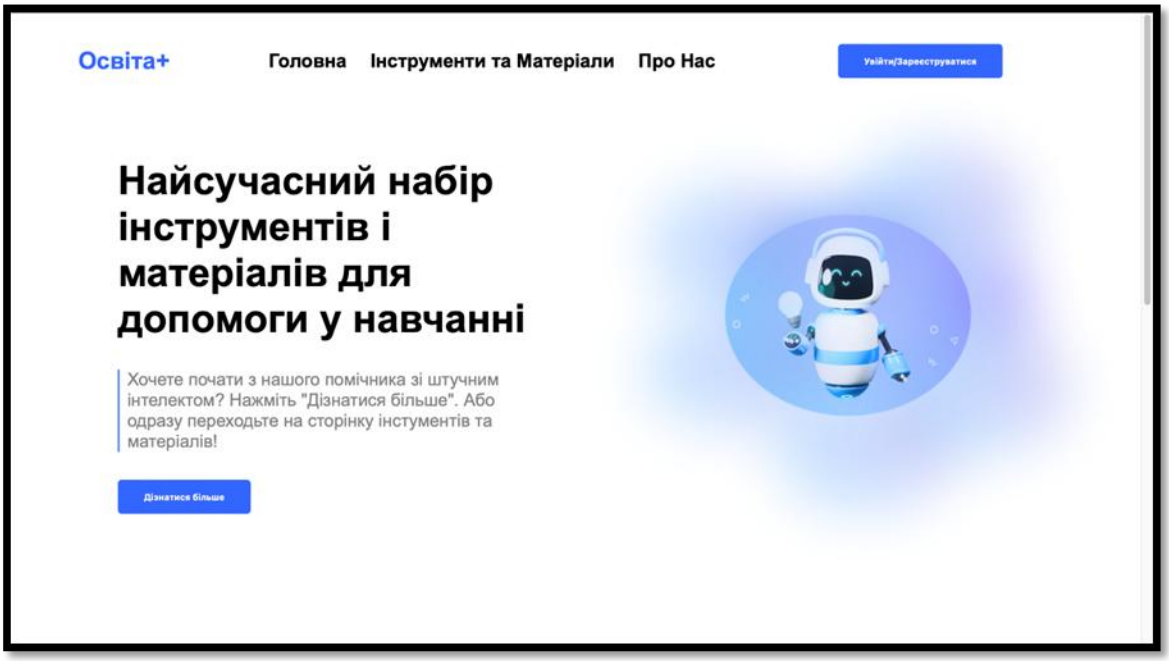

Рисунок 13 – Головна сторінка неавторизованого користувача

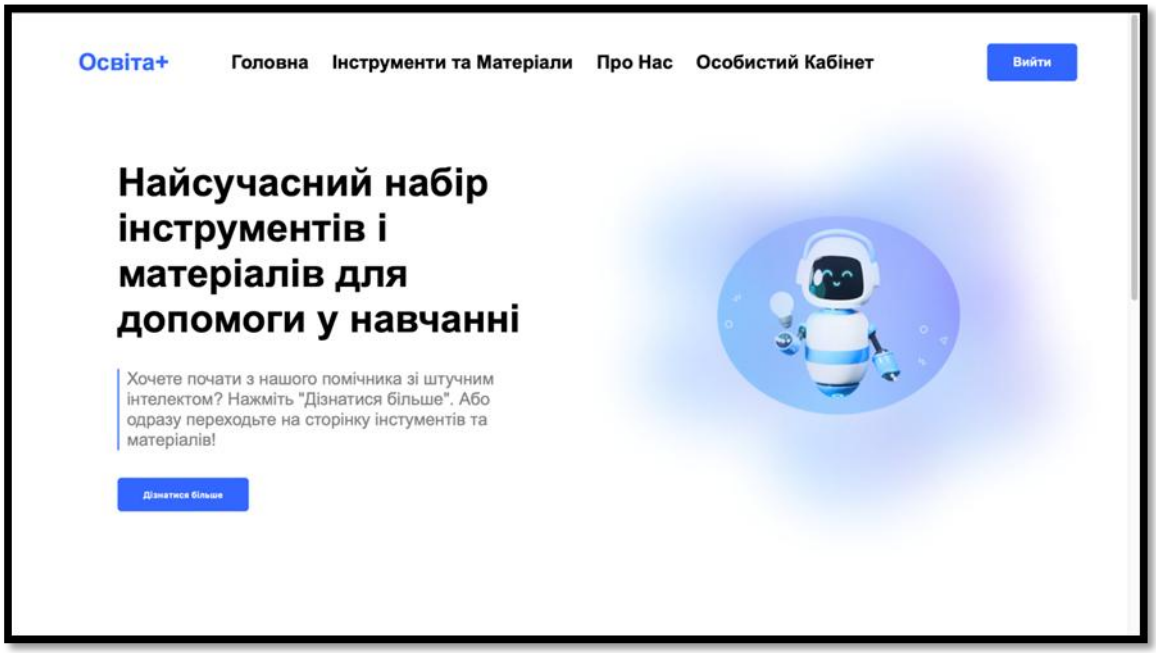

Рисунок 14 – Головна сторінка авторизованого користувача

Наступна секція домашньої сторінки дозволяє користувачам ставити питання безпосередньо через інтегровану систему OpenAI, яка також обізнана щодо контенту самого сайту. Це може бути корисно для отримання відповідей на часті питання або запитань, пов'язаних із функціоналом сайту (див рис. 15).

Рисунок 15 – Головна сторінка (продовження)

На рис. 16-18 наведені фрагменти коду, які забезпечують основний функціонал головної сторінки. На рис. 16 зображений фрагмент HTML-коду, який відповідає за форму, де можна запитати питання до ШІ (як загальне, так і про сам сайт).

```
<div id="AskURquestion">
    <p>Задайте Ваше питання тут і наш ШІ-помічник надасть Вам відповідь</p>
</div>
<section id="FindOutMore">
    <div id="AI">
        <textarea rows="5" cols="20" id="questionInput" style="font-size:
        <button id="AskButton" type="submit">Запитати</button>
        <div id="answer">
            <p id="InstructionsText">Якщо Ваше запитання має вузьку спеціал
        </div>
    </div>
</section>
```

Рисунок 16 – HTML-розмітка форми ШІ-питання на головній сторінці

На рис. 17 зображений відповідний фрагмент jQuery та AJAX коду, який забезпечує обробку введеного питання, перевірку його на пустоту, відправку на сервер для обробки та виведення відповіді або повідомлення про помилку.

```
$("#AskButton").off().click(function(e) {
    e.preventDefault(); // Запобігти відправці за замовчуванням

    var questionText = $("#questionInput").val().trim(); // Trim whitespace from the input

    // Перевірити чи питання пусте
    if (questionText === "") {
        $("#answer").html("Please enter a question to get an answer.").show();
        return; // Вийти з функції якщо питання не було введено
    }

    // Вивести повідомлення про те, що питання опрацовується
    $("#answer").html("Working on it, please wait...").show();

    // Здійснити AJAX запит
    $.ajax({
        url: '/ask',
        type: 'POST',
        contentType: 'application/json',
        data: JSON.stringify({ question: questionText }),
        success: function(data) {
            // Вивести відповідь
            $("#answer").html(data.answer || "Hang on...").show();
        },
        error: function() {
            $("#answer").html("Sorry, there was an error processing your question. Please try again.").show();
        }
    });
});
```

Рисунок 17 – Фрагмент jQuery та AJAX коду для HTML форми на головній сторінці

На рис. 18 зображений відповідний фрагмент Python (Flask) коду, який обробляє запити користувачів до OpenAI API на серверній стороні. Коли на сервер надходить POST запит за адресою /ask, функція ask_question отримує JSON дані з тіла запиту та зчитує питання користувача. Якщо питання не було надано, у логах фіксується помилка і сервер повертає повідомлення про відсутність питання з кодом 400.

Якщо питання є, код виконує запит до OpenAI API, передаючи інформацію про вебсайт Освіта+, таку як опис вебсайту та перелік його розділів. Також ця інформація містить інструкцію відповідати на питання українською мовою. Після отримання відповіді від OpenAI, вона обробляється та очищається від зайвих пробілів.

```python
@app.route('/ask', methods=['POST'])
def ask_question():
    data = request.get_json()
    question = data.get('question', '')

    if not question:
        logging.error('No question provided in the request')
        return jsonify({'error': 'No question provided'}), 400

    try:
        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "system", "content":
                    "You are a helpful assistant for the website Osvita+."
                    "Osvita+ is a website that provides users with various tools "
                    "and resources to answer their questions, including integrations "
                    "with Wolfram Alpha and Google Search (SerpAPI). "
                    "The website has the following sections:\n"
                    "1. Home page: An introductory page to engage users.\n"
                    "2. Ask AI: A feature allowing users to ask questions using OpenAI, "
                    "perfect for FAQ about the website itself.\n"
                    "3. Tools page: Integrates OpenAI with Wolfram Alpha, local PDFs, and SerpAPI.\n"
                    "4. About Us page: Information about the website.\n"
                    "5. Personal dashboard (Особистий кабінет): For authenticated users, "
                    "gives access to latest news and upcoming new features.\n"
                    "Feel free to answer any questions related to the features "
                    "and functionality of Osvita+. Answer the questions in Ukrainian."
                },
                {"role": "user", "content": question},
            ]
        )
        answer = response.choices[0].message.content.strip()
    except Exception as e:
        logging.exception('An error occurred while querying OpenAI')
        answer = "An error occurred: " + str(e)

    return jsonify({'answer': answer})
```

Рисунок 18 – Бекенд-реалізація форми для запитання до ШІ на головній сторінці

У випадку виникнення помилки під час запиту до OpenAI, у логах фіксується виняток, а користувач отримує повідомлення про помилку. Далі,

функція повертає отриману відповідь або повідомлення про помилку у форматі JSON.

5.2.2 Опис сторінки інструментів та матеріалів

На сторінці інструментів представлені інтеграції OpenAI з WolframAlpha для наукових обрахунків, локальними PDF-файлами для роботи з документацією та SerpAPI для пошуку інформації. Це дає користувачам потужні інструменти для навчання і досліджень. Фронтенд інтерфейс представлений на рис. 19 та 20.

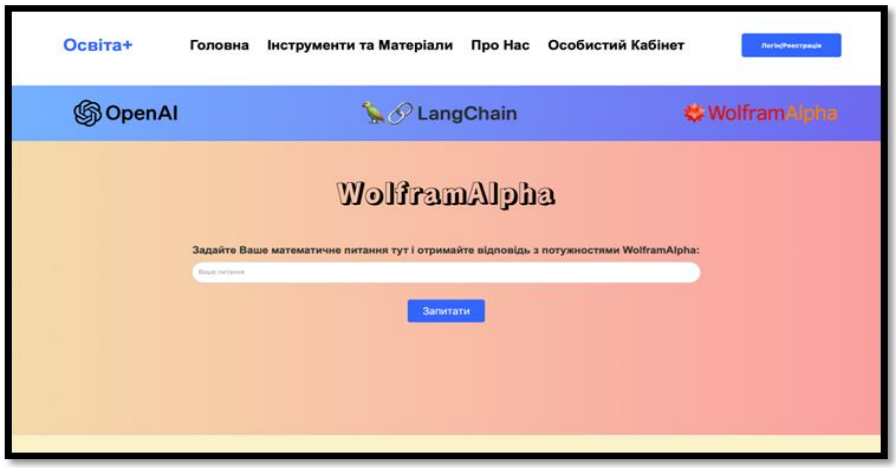

Рисунок 19 – Сторінка інструментів та матеріалів

Рисунок 20 – Сторінка інструментів та матеріалів (продовження)

На рис. 21 наведено код HTML-форми, яка відповідає за роботу з локальними PDF-файлами, а на рис. 22 та 23 □ відповідні фрагменти jQuery та Python (Flask) коду.

```
<section id="CatalogueSection2">
    <div><p id="AImeetsPDF">Local PDF</p></div>
    <div id="catalogueSec2">
        <form id="upload-form" enctype="multipart/form-data">
            <label for="pdf-file">Завантажте Ваш PDF файл і запитуйте про його зміст</label><br>
            <input type="file" id="pdf-file" name="file" required>
            <textarea id="prompt-input" name="prompt" placeholder="Ваше питання" required></textarea>
            <button type="button" onclick="uploadFile()">Завантажити і запитати</button>
        </form>
        <div id="pdf-result" style="display:none;">
            <p id="query-result">Результат запиту буде тут</p>
        </div>
    </div>
</section>
```

Рисунок 21 – Код HTML-форми, яка відповідає за роботу з локальними PDF-файлами

```
$('#pdf-file').on('change', function(event) {
    event.preventDefault();
    var formData = new FormData($('#upload-form')[0]);

    $.ajax({
        url: '/upload',
        type: 'POST',
        data: formData,
        processData: false,
        contentType: false,
        success: function(data) {
            $('#pdf-result').show();
            $('#query-result').text('PDF uploaded successfully. ' + data.answer);
        },
        error: function() {
            $('#pdf-result').show();
            $('#query-result').text('An error occurred while uploading the PDF.');
        }
    });
});
```

Рисунок 22 – Фрагмент jQuery-коду відповідний до HTML-форми, яка відповідає за роботу з локальними PDF-файлами

```python
@app.route('/upload', methods=['POST'])
def upload_file():
    if 'file' not in request.files or 'prompt' not in request.form:
        logging.error('No file part or prompt provided')
        return jsonify({'error': 'No file part or prompt provided'}), 400

    file = request.files['file']
    custom_prompt = request.form['prompt']

    if file.filename == '':
        logging.error('No selected file')
        return jsonify({'error': 'No selected file'}), 400

    try:
        file_path = os.path.join(UPLOAD_FOLDER, file.filename)
        file.save(file_path)
        logging.info(f'File saved to {file_path}')
        text = extract_text_from_pdf(file_path)

        prompt = f"Here is some information that might help: {text}\n\n{custom_prompt}"

        response = query_openai(prompt)

        return jsonify({'text': text, 'answer': response}), 200
    except Exception as e:
        logging.exception('An error occurred while saving the file or querying OpenAI')
        return jsonify({'error': f'An error occurred: {str(e)}'}), 500
```

Рисунок 23 – Фрагмент Python (Flask) коду, відповідний до HTML-форми, яка відповідає за роботу з локальними PDF-файлами

Завдяки HTML-формі і jQuery користувачі можуть завантажити PDF-файл. Після цього запускається AJAX-запит до бекенду. Бекенд на Flask обробляє подачу файлу та запиту, зберігає завантажений файл, витягує з нього текст і поєднує цей текст із спеціальним запитом. Комбінований текст і запит відправляються до OpenAI, і відповідь повертається на фронтенд, де результат відображається користувачу.

5.2.3 Опис інформаційної сторінки "Про Нас" та "Особистий Кабінет"

Сторінка "Про Нас" – це інформаційна сторінка, яка надає інформацію про проєкт Освіта+ (див рис. 24).

Рисунок 24 – Сторінка "Про Нас"

Сторінка має суто інформаційний характер і не виконує будь-якої іншої функції.

Сторінка "Особистий кабінет" (див. рис. 25) також є інформаційною.

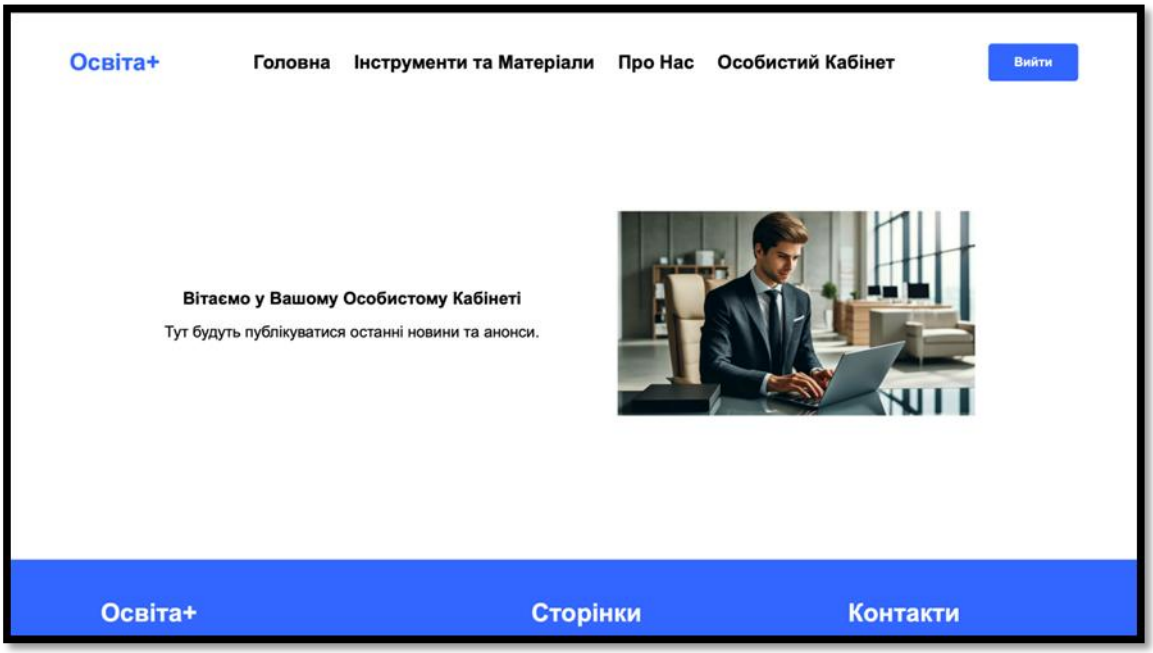

Рисунок 25 – Сторінка "Особистий кабінет"

На рис. 26 наведено повний HTML-код сторінки "Особистий кабінет".

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/index.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/tools.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/about.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/personalSpace.css') }}">
  <title>Освіта+</title>
</head>
<body>
  <iframe src="{{ url_for('header') }}" style="width:100%; border:none;"></iframe>
  <section id="AboutIntro">
    <div id="AboutIntroLeft">
      <p class="WhoWeAre"><b>Вітаємо у Вашому Особистому Кабінеті</b></p>
        <p class="AboutUs">Тут будуть публікуватися останні новини та анонси.
        </p>
    </div>
    <div id="AboutIntroRight">
      <img id="AboutIntroRightImage" src="{{ url_for('static', filename='img/PersonalSpace.png') }}"/>
    </div>
</section>
<footer class="footer">
  <div class="column1">
    <p class="footerBoldText">Освіта+</p>
    <p>Найсучасний набір інструментів і</p>
    <p>матеріалів для допомоги у навчанні</p>
  </div>
  <div class="column2">
    <p class="footerBoldText">Сторінки</p>
    <div class="Links">
      <div ><a class="FooterLink" href="{{ url_for('index') }}" target="_parent">Головна</a></div>
      <div><a class="FooterLink" href="{{ url_for('tools') }}" target="_parent">Інструменти та Матеріали</a></div>
      <div><a class="FooterLink" href="{{ url_for('about') }}" target="_parent">Про Нас</a></div>
      <div><a class="FooterLink" href="{{ url_for('personal_space') }}" target="_parent">Особистий кабінет</a></div>
    </div>
  </div>
  <div class="column3">
    <p class="footerBoldText">Контакти</p>
    <p>artemtsurkan002@gmail.com</p>
  </div>
</footer>
</body>
</html>
```

Рисунок 26 – HTML-код сторінки "Особистий кабінет"

Як можна побачити, тут (як і на більшості інших сторінок сайту) використовується HTML-елемент айфрейм (iframe). Ций елемент дозволяє вставляти один вебсайт в інший. Вони корисні для інтеграції контенту з зовнішніх джерел, таких як відео, карти, форми чи інші веб-ресурси, не виходячи з поточної сторінки. У цьому випадку він використовується для повторюваних заголовків, що дозволяє легко впроваджувати зміни в одному місці без необхідності редагування кожної сторінки окремо.

На рис. 27 наведений повний код CSS-стилей для стилізації сторінки.

```css
:root{
    --blueish-purple: #3366ff;
    --section-gray: #cacfe0;
}

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

html{
    font-family: Arial;
}

#AboutIntro{
    display: flex;
    justify-content: center;
    align-items: center;
    padding: 50px;
    flex-direction: row;
    text-align: center;
}

#AboutIntroLeft{
    padding: 20px;
}

.WhoWeAre{
    font-size: 22px;
}

.AboutUs{
    padding-top: 18px;
    font-size: 20px;
}

#AboutIntroRight{
    padding: 20px;
}

#AboutIntroRightImage{
    width: 100%;
    max-width: 457px;
    height: auto;
}
```

Рисунок 27 – HTML-код сторінки "Особистий кабінет"

Як можна побачити, у CSS був використаний Flexbox інструмент у CSS, який дозволяє легко та ефективно розташовувати елементи. Flexbox полегшує вирівнювання, розподіл простору та розташування елементів як по горизонталі, так і по вертикалі. Flexbox був використаний не тільки на цій сторінці, але й на інших сторінках сайту.

5.3 Система реєстрації та авторизації Google OAuth

Щоб уникнути ризики безпеки пов'язані зі зберіганням логінів і паролей користувачів, було вирішено реалізувати функцію входу та реєстрації через Google OAuth.

5.3.1 Опис OAuth

OAuth (Open Authorization) — це стандартний протокол авторизації, який дозволяє одному додатку отримати обмежений доступ до ресурсів користувача на іншому додатку без розголошення облікових даних користувача. OAuth забезпечує безпечний та зручний спосіб надання доступу до особистих даних або сервісів без необхідності ділитися паролями та іншими конфіденційними даними. За даними дослідження [9], OAuth є ефективним рішенням для захисту ресурсів користувачів у різних додатках.

Ресурсний власник — це особа, яка володіє ресурсами, до яких запитує доступ клієнтський додаток. Зазвичай, це користувач. Клієнт — це додаток, який запитує доступ до ресурсів, що належать ресурсному власнику. Наприклад, це може бути веб-додаток або мобільний додаток. Сервер авторизації відповідає за авторизацію ресурсного власника та видачу токенів доступу клієнту після успішної авторизації. Сервер ресурсів розміщує ресурси, до яких клієнт хоче отримати доступ, і перевіряє дійсність токенів доступу, виданих сервером авторизації.

Процес авторизації OAuth починається з того, що клієнт звертається до ресурсного власника з проханням дозволити доступ до його ресурсів. Це може бути реалізовано через перенаправлення користувача на сторінку авторизації. Ресурсний власник надає дозвіл клієнту, авторизуючи себе на сервері авторизації. Після цього сервер авторизації видає код авторизації. Клієнт обмінює код авторизації на токен доступу, звернувшись до сервера авторизації. Токен доступу є ключем, який дозволяє клієнту запитувати ресурси на сервері ресурсів. Клієнт використовує токен доступу для запиту ресурсів на сервері ресурсів. Сервер ресурсів перевіряє токен і, якщо він дійсний, надає доступ до запитуваних ресурсів.

Переваги OAuth включають підвищену безпеку, гнучкість та зручність для користувачів. OAuth дозволяє уникати необхідності передачі паролів між додатками, що знижує ризики компрометації облікових даних. У [10]

колективно підкреслюється, що правильна реалізація OAuth значно знижує ризики фішингу, забезпечуючи безпечне поводження з токенами та їх валідацію під час взаємодії клієнта та сервера. Це забезпечує підвищену безпеку для користувачів та дозволяє їм легко керувати та відкликати дозволи, надаючи гнучкий контроль доступу. Користувачі можуть надавати різні рівні доступу до своїх ресурсів для різних додатків. Крім того, користувачі можуть легко керувати доступом до своїх ресурсів та відкликати дозволи в будь-який момент.

OAuth став стандартом де-факто для авторизації в багатьох сучасних веб- та мобільних додатках, забезпечуючи безпечний та ефективний спосіб взаємодії між різними сервісами та платформами.

5.3.2 Налаштування Google OAuth

Для налаштування OAuth спочатку потрібно увійти до Google Cloud Console (https://console.cloud.google.com/apis/) (див. рис. 28) і створити новий проєкт спеціально для нашого додатку.

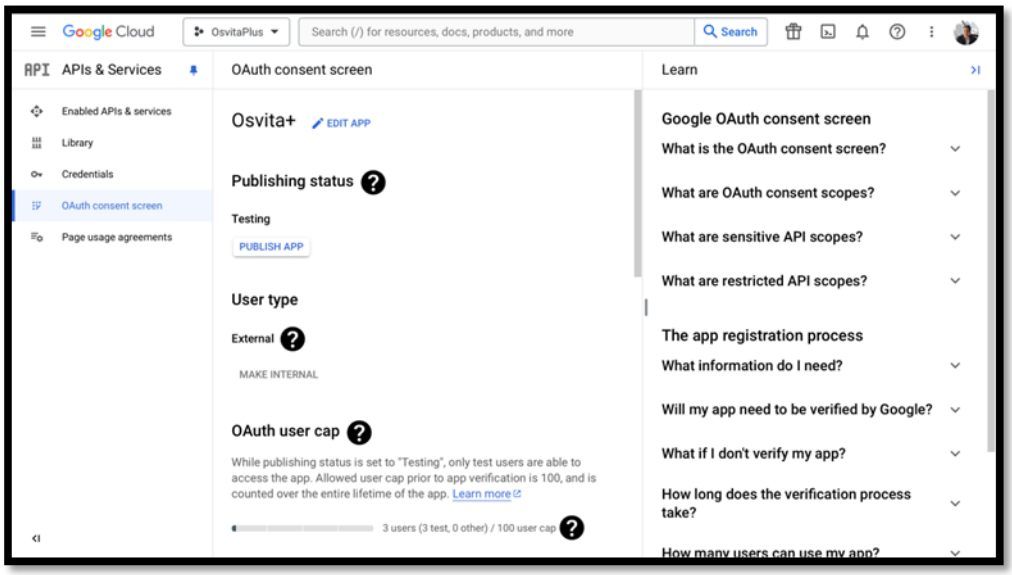

Рисунок 28 – Налаштування Google OAuth у Google Cloud Platform

У межах проєкту необхідно перейти до розділу "APIs & Services" і налаштувати API для нашого додатку. Після цього потрібно налаштувати

екран згоди OAuth, заповнивши необхідну інформацію, таку як назва додатку, дозволені домени та URL політики конфіденційності. Далі створюються облікові дані OAuth 2.0, вибираючи тип додатку "Web application".

Важливо вказати дозволені URL-адреси перенаправлення, які включають URL-адресу, на яку користувачів буде перенаправлено після успішної авторизації. Цей процес надає ідентифікатор клієнта (client ID) та секрет клієнта (client secret), які інтегруються у Flask-додаток для забезпечення безпечної авторизації через облікові записи Google.

5.3.3 Реалізація Google OAuth у фронтенді та бекенді

У процесі реалізації Google OAuth, фронтенд та бекенд тісно взаємодіють для забезпечення безпечної авторизації користувачів.

На фронтенді, інтеграція починається з додавання скрипта для ініціалізації кнопки входу (HTML-реалізація наведена на рис. 29) та обробки відповіді від Google (рис. 30).

```
    <div class="centralPage">
        <a id="personal_cabinet_link" href="{{ url_for('personal_space') }}"
        class="link" target="_parent" style="display: none;">Особистий Кабінет</a>
    </div>
</div>
<div><button class="button LoginRegister">
    <a class="LoginRegister" href="{{ url_for('login') }}">Увійти/Зареєструватися</a></button>
</div>
<div>
    <a class="button Logout" href="{{ url_for('logout') }}" style="display:none; text-decoration: none;">Вийти</a>
</div>
```

Рисунок 29 – HTML-реалізація кнопок входу та виходу на головній сторінці (index.html)

```
$.get('/is-logged-in', function(response) {
    if (response.is_logged_in) {
        $('.LoginRegister').hide();  // Hide Login/Register button
        $('#personal_cabinet_link').css('display', 'block');  // Ex
        $('.Logout').css('display', 'block');  // Explicitly show L
    } else {
        $('.LoginRegister').show();
        $('#personal_cabinet_link').hide();
        $('.Logout').hide();
    }
});

// Logout button click handler
$('.Logout').click(function() {
    $.get('/logout', function() {
        window.location.href = '/';
    });
});
// Check if the user is already authenticated
$.ajax({
    type: "GET",
    url: "/profile",
    success: function (data) {
        if (data.status === 'success') {
            $('.LoginRegister').hide();
            $('#personal_cabinet_link').show();
        }
    },
    error: function (error) {
        console.log('User not authenticated', error);
    }
});
```

Рисунок 30 – Фрагмент jQuery коду головної сторінки (index.js)

Коли користувач натискає кнопку "Увійти/Зареєструватися", браузер перенаправляє його на сторінку авторизації Google, де користувач вводить свої облікові дані. Після успішної авторизації, Google повертає на фронтенд токен ідентифікації (ID token) (див. рис. 31).

На бекенді, цей токен надсилається на сервер для подальшої обробки. Сервер перевіряє дійсність токена, звертаючись до Google для підтвердження автентичності та отримання інформації про користувача.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://accounts.google.com/gsi/client" async defer></script>
</head>
<body>
    <h2>Login with Google</h2>
    <div id="g_id_onload"
         data-client_id="745684469219-jt1t4vl6v9965696pkv0kj1jl900867v.apps.googleusercontent.com"
         data-callback="handleCredentialResponse">
    </div>
    <div class="g_id_signin"
         data-type="standard"
         data-size="large"
         data-theme="outline"
         data-text="sign_in_with"
         data-shape="rectangular"
         data-logo_alignment="left">
    </div>
    <script src="{{ url_for('static', filename='js/index.js') }}"></script>
    <script src="{{ url_for('static', filename='js/header.js') }}"></script>
</body>
</html>
```

Рисунок 31 – Фрагмент HTML коду сторінки авторизації Google (login.html)

Якщо токен є дійсним, сервер зберігає інформацію про користувача в сесії і повертає відповідь на фронтенд, сигналізуючи про успішний вхід. Користувач тепер вважається авторизованим, і йому надається доступ до захищених ресурсів та функціональності додатку (в нашому випадку, це – Особистий Кабінет).

У випадку успішного входу користувача, кнопка "Увійти/ Зареєструватися" зникає, і з'являються посилання на "Особистий Кабінет" та кнопка "Вийти" (див. рис. 30). При натисканні кнопки "Вийти", сесія користувача завершується, і він повертається на головну сторінку.

У Flask-додатку це реалізовано за допомогою бібліотеки authlib.integrations.flask_client (код можна подивитися у додатку А). Коли користувач натискає кнопку входу, фронтенд ініціалізує перенаправлення до Google OAuth через визначений маршрут /login, що налаштовує сервер авторизації та обробляє отримання токену через маршрут /auth.

```
function handleCredentialResponse(response) {
    console.log('Google ID token:', response.credential); // Log the response to see if it's received
    $.ajax({
        type: "POST",
        url: "/auth",
        data: JSON.stringify({ id_token: response.credential }),
        contentType: "application/json",
        success: function (data) {
            console.log('Login successful:', data);
            window.location.href = '/'; // Redirect to homepage
        },
        error: function (error) {
            console.log('Login failed', error);
        }
    });
}

        window.onload = function () {
            google.accounts.id.initialize({
                client_id: '745684469219-jt1t4vl6v9965696pkv0kj1jl900867v.apps.googleusercontent.com',
                callback: handleCredentialResponse
            });
            google.accounts.id.renderButton(
                document.getElementById('buttonDiv'),
                { theme: "outline", size: "large" }
            );
            google.accounts.id.prompt();
        };
```

Рисунок 32 – Фрагмент jQuery коду сторінки авторизації Google (login.js)

Сервер використовує отримані клієнтські облікові дані для встановлення сесії та забезпечення доступу користувача до захищених розділів додатку. Таким чином, взаємодія фронтенду та бекенду забезпечує безпечний та ефективний процес авторизації користувачів через облікові записи Google.

5.4 Веб-хостинг на Heroku

Heroku – це платформа як послуга (PaaS), яка дозволяє розробникам легко розгортати, керувати та масштабувати додатки в хмарі. Вона підтримує широкий спектр мов програмування, включаючи Python, Java, Ruby, Node.js, PHP та багато інших. Однією з головних переваг Heroku є простота використання, що робить її ідеальною платформою для швидкого розгортання та управління веб-проєктами.

Для початку роботи з Heroku потрібно зареєструвати обліковий запис на офіційному сайті (https://www.heroku.com/) (рис. 33). Після реєстрації можна встановити Heroku CLI, що дозволяє взаємодіяти з платформою через командний рядок. Використовуючи Heroku CLI, можна створити новий

проєкт, підключити його до репозиторію на GitHub та розгорнути додаток на сервері Heroku. Платформа автоматично обробляє всі необхідні конфігурації та налаштування, забезпечуючи безперебійне розгортання додатку.

Однією з ключових особливостей Heroku є підтримка масштабування. Можна легко збільшити ресурси, які доступні додатку, додавши більше динамо (dynos) – контейнерів, в яких запускаються додатки. Це дозволяє додатку обробляти більшу кількість запитів та користувачів без необхідності кардинальних змін у коді або архітектурі. Крім того, Heroku пропонує різноманітні додатки (add-ons), які можна легко інтегрувати у проєкт для додаткових функціональних можливостей, таких як бази даних, моніторинг, кешування та багато іншого.

Безпека також є важливим аспектом, який враховує Heroku. Платформа забезпечує автоматичні оновлення та виправлення безпеки, що допомагає захистити додаток від потенційних загроз.

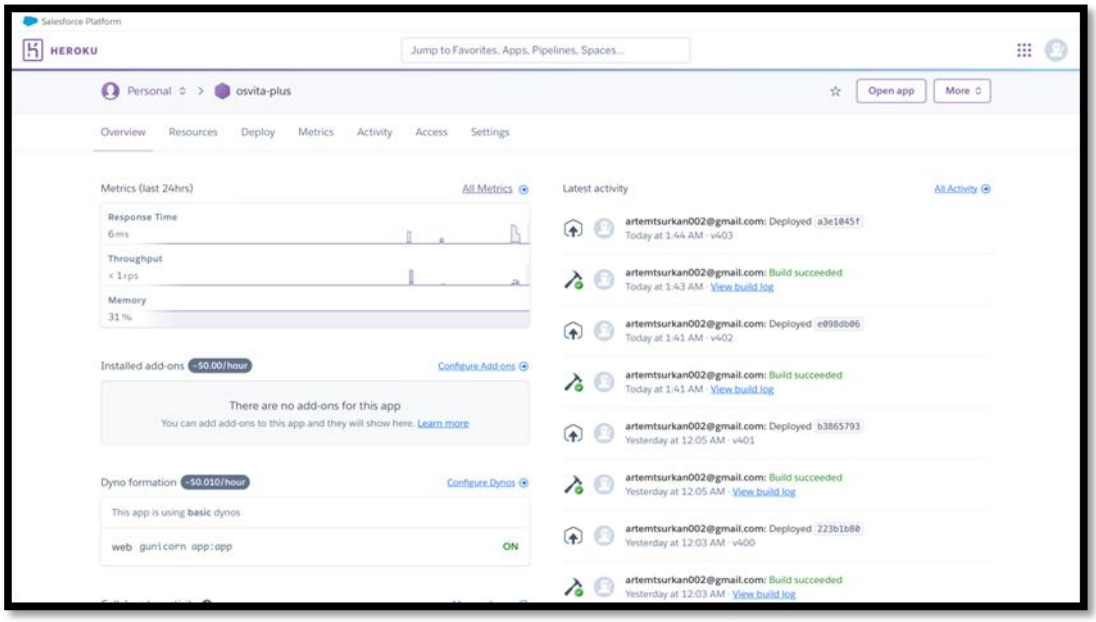

Рисунок 23 – Панель управління додатком на Heroku

Таким чином, Heroku є потужною та гнучкою платформою для розгортання веб-додатків, яка спрощує процес управління інфраструктурою та забезпечує високу надійність і масштабованість проєкту.

ВИСНОВКИ

У цьому проекті було розглянуто створення веб-додатку, який за допомогою Langchain поєднує можливості OpenAI з Wolfram Alpha, веб-пошуком та локальними PDF-документам. Було окреслено функціональні та нефункціональні вимоги до системи.

Розробка веб-додатку для впровадження онлайн-освіти дозволила реалізувати багатофункціональну платформу, яка інтегрує потужні інструменти для обробки та аналізу інформації. Використання технологій OpenAI та Wolfram Alpha забезпечує високий рівень взаємодії користувачів з даними, дозволяючи миттєво отримувати відповіді на складні запитання та аналізувати різні джерела інформації.

Основні результати роботи включають:

1) дослідження предметної області − було проведено аналіз існуючих освітніх платформ та виявлено їхні переваги та недоліки, що дозволило сформулювати вимоги до нової системи, яка поєднує кращі аспекти наявних рішень і додає інноваційні функціональності;
2) вибір програмних засобів − після детального порівняльного аналізу було обрано оптимальні технології для реалізації додатку;
3) проектування архітектури − було спроектовано загальну архітектуру платформи, яка забезпечує інтеграцію різних інструментів та сервісів, що дозволяє користувачам легко орієнтуватися та зручно використовувати всі функціональні можливості платформи.
4) реалізація додатку − було розроблено та протестовано основні функціональні модулі платформи, включаючи інтеграцію з OpenAI для генерації відповідей на запитання, Wolfram Alpha для наукових обчислень, обробку локальних PDF-документів та пошукову систему SerpAPI для отримання актуальної інформації з Інтернету.

5) тестування та оптимізація − проведено тестування додатку на відповідність вимогам та надійності роботи, та виявлені недоліки було виправлено, а додаток оптимізовано для забезпечення високої продуктивності та зручності користування.

Платформа Освіта+ успішно об’єднує інструменти для роботи з локальними PDF-документами, веб-пошуком та іншими джерелами даних, забезпечуючи користувачам зручний і ефективний доступ до необхідної інформації. Реалізація інтеграцій з Google OAuth та хостингом на Heroku дозволила забезпечити безпеку та надійність платформи.

Проєкт досяг поставлених цілей та завдань, запропонувавши інноваційні рішення для покращення процесу навчання та досліджень. Успішне впровадження таких технологій може стати основою для подальшого розвитку інтелектуальних систем в освітньому процесі та інших галузях.

Запропоновані методи та підходи можуть бути використані для подальшого розширення функціональності платформи, включаючи додавання нових інструментів та інтеграцій. Такий підхід сприятиме підвищенню ефективності навчання та доступності знань для широкого кола користувачів.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Матеріали студентської наукової конференції Одеського державного екологічного університету (10-19 квітня 2024 р.). Одеса, 2024.
2. Online Learning Statistics: Market Size and Facts for 2024 [Електронний ресурс]. URL: https://www.skillademia.com/statistics/online-learning-statistics/ (дата звернення: 15.01.2024).
3. Ng, A. Y., & Jordan, M. I. Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. URL: https://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf (дата звернення: 2024).
4. Domingos, P. A Few Useful Things to Know About Machine Learning. URL: https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf (дата звернення: 2024).
5. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. Generative Adversarial Nets. URL: https://arxiv.org/pdf/1406.2661 (дата звернення: 2024).
6. Kingma, D. P., & Welling, M. Auto-Encoding Variational Bayes. URL: https://arxiv.org/pdf/1312.6114 (дата звернення: 2024).
7. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. Attention is All You Need. URL: https://arxiv.org/pdf/1706.03762 (дата звернення: 2024).
8. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... & Amodei, D. Language Models are Few-Shot Learners. URL: https://arxiv.org/pdf/2005.14165 (дата звернення: 2024).
9. Hardt, D. The OAuth 2.0 Authorization Framework. IEEE Security & Privacy, 10(2), 105-110. URL: https://datatracker.ietf.org/doc/html/rfc6749 (дата звернення: 2024).

10. Lodderstedt, T., McGloin, M., & Hunt, P. OAuth 2.0 Threat Model and Security Considerations. Internet Engineering Task Force (IETF). URL: https://www.rfc-editor.org/info/rfc6819 (дата звернення: 2024).

ДОДАТОК А

ВИХІДНИЙ КОД ДОДАТКУ

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/index.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/about.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/personalSpace.css') }}">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <title>Освіта+</title>
  <script src="{{ url_for('static', filename='js/index.js') }}"></script>
  <script src="{{ url_for('static', filename='js/header.js') }}"></script>
</head>
<body id="indexBody">
<!--====================HEADER===========-->
<header>
   <div class="WebsiteTitle">Освіта+</div>
   <div class="CentralPages">
      <div     class="centralPage"><a     href="{{     url_for('index')     }}"     class="link"
target="_parent">Головна</a></div>
      <div     class="centralPage"><a     href="{{     url_for('tools')     }}"     class="link"
target="_parent">Інструменти та Матеріали</a></div>
      <div class="centralPage"><a href="{{ url_for('about') }}" class="link" target="_parent">Про
Нас</a></div>
      <div class="centralPage">
         <a id="personal_cabinet_link" href="{{ url_for('personal_space') }}"
         class="link" target="_parent" style="display: none;">Особистий Кабінет</a>
      </div>
   </div>
   <div><button class="button LoginRegister">
      <a class="LoginRegister" href="{{ url_for('login') }}">Увійти/Зареєструватися</a></button>
   </div>
   <div>
      <a  class="button  Logout"  href="{{  url_for('logout')  }}"  style="display:none;  text-
decoration: none;">Вийти</a>
   </div>
</header>

<!--====================HEADER ENDS - INTRO BEGINS==============-->

 <section class="Intro">
    <div class="Intro-Left">
       <h1 class=boldText>Найсучасний набір інструментів і матеріалів для допомоги у навчанні
</h1>
       <p class="grayText">Хочете почати з нашого помічника зі штучним інтелектом? Нажміть
"Дізнатися більше". Або одразу переходьте на сторінку інстументів та матеріалів!</p>
       <a    href="#AskURquestion"><button    id="findOutMoreButton"    class="button">Дізнатися
більше</button></a>
    </div>
    <div class="Intro-Right">
      <img id="Intro-Image" src="{{ url_for('static', filename='img/Group2-Resize.png') }}"/>
    </div>
 </section>
<!--====================OPEN AI=================-->
<div id="FlexWrapper">
   <div id="AskURquestion">
      <p>Задайте Ваше питання тут і наш ШІ-помічник надасть Вам відповідь</p>
   </div>
   <section id="FindOutMore">
      <div id="AI">
            <textarea    rows="5"    cols="20"    id="questionInput"    style="font-size:    14px;"
placeholder="Задайте Ваше питання тут і наш ШІ-помічник надасть Вам відповідь"></textarea>
            <button id="AskButton" type="submit">Запитати</button>
            <div id="answer">
               <p id="InstructionsText">Якщо Ваше запитання має вузьку спеціалізацію і Ви хочете
використати певний інструмент (наприклад, WolframAlpha) для отримання відповіді - рекомендуємо
перейти  на  сторінку  <a  href="./html/tools.html"  style="text-decoration:none;  color:  gray;
cursor:pointer">Інструментів та Матеріалів.</a></p>
            </div>
      </div>
   </section>
   <!--====================FOOTER=========-->
      <footer class="footer">
         <div class="column1">
            <p class="footerBoldText">Освіта+</p>
            <p>Найсучасний набір інструментів і</p>
            <p>матеріалів для допомоги у навчанні</p>
         </div>
         <div class="column2">
            <p class="footerBoldText">Сторінки</p>
            <div class="Links">
               <div      ><a      class="FooterLink"      href="{{      url_for('index')      }}"
target="_parent">Головна</a></div>
```

```
            <div><a       class="FooterLink"       href="{{       url_for('tools')       }}"
target="_parent">Інструменти та Матеріали</a></div>
            <div><a  class="FooterLink"  href="{{  url_for('about')  }}"  target="_parent">Про
Нас</a></div>
            <div><a     class="FooterLink"     href="{{     url_for('personal_space')     }}"
target="_parent">Особистий кабінет</a></div>
         </div>
      </div>
      <div class="column3">
         <p class="footerBoldText">Контакти</p>
         <p>artemtsurkan002@gmail.com</p>
      </div>
   </footer>
</div>
</body>
</html>
```

index.js

```
$(document).ready(function() {
    // Check if the user is logged in
$.get('/is-logged-in', function(response) {
      if (response.is_logged_in) {
            $('.LoginRegister').hide();  // Hide Login/Register button
            $('#personal_cabinet_link').css('display',  'block');   //  Explicitly  show  Personal
Space link
            $('.Logout').css('display', 'block');  // Explicitly show Logout button
      } else {
            $('.LoginRegister').show();
            $('#personal_cabinet_link').hide();
            $('.Logout').hide();
      }
   });

   // Logout button click handler
   $('.Logout').click(function() {
      $.get('/logout', function() {
            window.location.href = '/';
      });
   });
// Check if the user is already authenticated
$.ajax({
    type: "GET",
    url: "/profile",
    success: function (data) {
        if (data.status === 'success') {
            $('.LoginRegister').hide();
            $('#personal_cabinet_link').show();
        }
    },
    error: function (error) {
        console.log('User not authenticated', error);
    }
});
// Handle AI assistant question submission
$("#AskButton").off().click(function(e) {
   e.preventDefault(); // Запобігти відправці за замовчуванням
   var questionText = $("#questionInput").val().trim(); // Trim whitespace from the input
   // Перевірити чи питання пусте
   if (questionText === "") {
      $("#answer").html("Please enter a question to get an answer.").show();
      return; // Вийти з функції якщо питання не було введено
   }

   // Вивести повідомлення про те, що питання опрацовується
   $("#answer").html("Working on it, please wait...").show();
   // Здійснити AJAX запит
   $.ajax({
      url: '/ask',
      type: 'POST',
      contentType: 'application/json',
      data: JSON.stringify({ question: questionText }),
      success: function(data) {
         // Вивести відповідь
         $("#answer").html(data.answer || "Hang on...").show();
      },
      error: function() {
         $("#answer").html("Sorry,  there  was  an  error  processing  your  question.  Please  try
again.").show();
      }
   });
});
});
```

header.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="{{ url_for('static', filename='css/index.css') }}">
    <link rel="stylesheet" href="{{ url_for('static', filename='css/tools.css') }}">
    <link rel="stylesheet" href="{{ url_for('static', filename='css/about.css') }}">
    <link rel="stylesheet" href="{{ url_for('static', filename='css/personalSpace.css') }}">
```

```
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <title>Освіта+</title>
    <script src="{{ url_for('static', filename='js/header.js') }}"></script>
</head>
    <body>
        <header>
            <div class="WebsiteTitle">Освіта+</div>
            <div class="CentralPages">
               <div   class="centralPage"><a   href="{{   url_for('index')   }}"   class="link"
target="_parent">Головна</a></div>
               <div   class="centralPage"><a   href="{{   url_for('tools')   }}"   class="link"
target="_parent">Інструменти та Матеріали</a></div>
               <div   class="centralPage"><a   href="{{   url_for('about')   }}"   class="link"
target="_parent">Про Нас</a></div>
               <div       class="centralPage"><a       id="personal_cabinet_link"       href="{{
url_for('personal_space')  }}"  class="link"  target="_parent"  style="display:  none;">Особистий
Кабінет</a></div>
            </div>
            <div><button   class="button   LoginRegister"><a   class="LoginRegister"   href="{{
url_for('login') }}" target="_parent">Увійти/Зареєструватися</a></button></div>
            <div><a  class="button  Logout"  href="{{  url_for('logout')  }}"  style="display:none;
text-decoration: none;" target="_parent">Вийти</a></div>
        </header>
</body>
</html>
```

header.js

```
$(document).ready(function() {
    // Check if the user is logged in
    $.get('/is-logged-in', function(response) {
        if (response.is_logged_in) {
            $('.LoginRegister').hide();  // Hide Login/Register button
            $('#personal_cabinet_link').css('display',  'block');   //  Explicitly  show  Personal
Space link
            $('.Logout').css('display', 'block');  // Explicitly show Logout button
        } else {
            $('.LoginRegister').show();
            $('#personal_cabinet_link').hide();
            $('.Logout').hide();
        }
    });

    // Logout button click handler
    $('.Logout').click(function() {
        $.get('/logout', function() {
            window.location.href = '/';
        });
    });
});
```

login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <script src="https://accounts.google.com/gsi/client" async defer></script>
</head>
<body>
    <h2>Login with Google</h2>
    <div id="g_id_onload"
         data-client_id="745684469219-
jt1t4vl6v9965696pkv0kj1jl900867v.apps.googleusercontent.com"
         data-callback="handleCredentialResponse">
    </div>
    <div class="g_id_signin"
         data-type="standard"
         data-size="large"
         data-theme="outline"
         data-text="sign_in_with"
         data-shape="rectangular"
         data-logo_alignment="left">
    </div>
    <script src="{{ url_for('static', filename='js/index.js') }}"></script>
    <script src="{{ url_for('static', filename='js/header.js') }}"></script>
</body>
</html>
```

login.js

```
function handleCredentialResponse(response) {
    console.log('Google  ID  token:',  response.credential);  //  Log  the  response  to  see  if  it's
received
    $.ajax({
        type: "POST",
        url: "/auth",
        data: JSON.stringify({ id_token: response.credential }),
        contentType: "application/json",
        success: function (data) {
            console.log('Login successful:', data);
            window.location.href = '/'; // Redirect to homepage
        },
        error: function (error) {
```

```
            console.log('Login failed', error);
        }
    });
}

        window.onload = function () {
            google.accounts.id.initialize({
                client_id:                                                    '745684469219-
jt1t4vl6v9965696pkv0kj1jl900867v.apps.googleusercontent.com',
                callback: handleCredentialResponse
            });
            google.accounts.id.renderButton(
                document.getElementById('buttonDiv'),
                { theme: "outline", size: "large" }
            );
            google.accounts.id.prompt();
        };
```

tools.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
  <script src="{{ url_for('static', filename='js/tools.js') }}"></script>
  <link rel="stylesheet" href="{{ url_for('static', filename='css/index.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/tools.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/about.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/personalSpace.css') }}">
  <title>Освіта+</title>
  <script src="{{ url_for('static', filename='js/header.js') }}"></script>
  <style>
    @import url('https://fonts.googleapis.com/css2?family=Rampart+One&display=swap');
  </style>
</head>
<body>
    <iframe  src="{{  url_for('header')  }}"  style="width:100%;  border:none;padding-bottom:0;
margin-bottom: -20px;"></iframe>
<section class="banner-section">
    <div        id="banner-img1"><img        id="OpenAILogo"        src="{{        url_for('static',
filename='img/OpenAI_Logo.svg') }}" alt="Open AI Logo"/></div>
    <div       id="banner-img4"><img       id="LangChainLogo"       src="{{       url_for('static',
filename='img/LangChainLogo2.svg') }}" alt="Lang Chain Logo"/></div>
    <div      id="banner-img2"><img      id="WolframAlphaLogo"      src="{{      url_for('static',
filename='img/WolframAlphaLogo.svg') }}" alt="Wolfram Alpha Logo"/></div>
</section>
<div id="FlexWrapper">
<section id="CatalogueSection1">
    <div><p id="AImeetsWolfram">WolframAlpha</p></div>
    <div id="catalogueSec1">
        <form id="scienceQuestionForm">
            <label  for="scienceQuestion">Задайте  Ваше  математичне  питання  тут  і  отримайте
відповідь з потужностями WolframAlpha:</label><br>
            <input  type="text"  id="scienceQuestion"  name="scienceQuestion"  placeholder="Ваше
питання"><br>
            <button type="submit">Запитати</button>
        </form>
        <div id="scienceAnswer">
            <p>Відповідь буде тут</p>
        </div>
    </div>
</section>

<section id="CatalogueSection2">
    <div><p id="AImeetsPDF">Local PDF</p></div>
    <div id="catalogueSec2">
        <form id="upload-form" enctype="multipart/form-data">
            <label for="pdf-file">Завантажте Ваш PDF файл і запитуйте про його зміст</label><br>
            <input type="file" id="pdf-file" name="file" required>
            <textarea     id="prompt-input"     name="prompt"     placeholder="Ваше     питання"
required></textarea>
            <button type="button" onclick="uploadFile()">Завантажити і запитати</button>
        </form>
        <div id="pdf-result" style="display:none;">
            <p id="query-result">Результат запиту буде тут</p>
        </div>
    </div>
</section>

<section id="CatalogueSection3">
    <div><p id="AImeetsGoogleSearch">GoogleSearch</p></div>
    <div id="catalogueSec3">
        <form id="googleQuestionForm">
            <label for="googleQuestion">ШІ який можете шукати надновітню інформацію</label><br>
            <input   type="text"   id="googleQuestion"   name="googleQuestion"   placeholder="Ваше
питання"><br>
            <button type="submit">Запитати</button>
        </form>
        <div id="googleAnswer">
            <p>Відповідь буде тут</p>
        </div>
    </div>
```

```
</section>
<footer class="footer">
    <div class="column1">
       <p class="footerBoldText">Освіта+</p>
       <p>Найсучасний набір інструментів і</p>
       <p>матеріалів для допомоги у навчанні</p>
    </div>
    <div class="column2">
       <p class="footerBoldText">Сторінки</p>
       <div class="Links">
          <div       ><a       class="FooterLink"       href="{{       url_for('index')       }}"
target="_parent">Головна</a></div>
          <div><a  class="FooterLink"  href="{{  url_for('tools')  }}"  target="_parent">Інструменти
та Матеріали</a></div>
          <div><a   class="FooterLink"   href="{{   url_for('about')   }}"   target="_parent">Про
Нас</a></div>
          <div><a       class="FooterLink"       href="{{       url_for('personal_space')       }}"
target="_parent">Особистий кабінет</a></div>
       </div>
    </div>
    <div class="column3">
       <p class="footerBoldText">Контакти</p>
       <p>artemtsurkan002@gmail.com</p>
    </div>
 </footer>
</div>
</body>
</html>
```

tools.js

```
$(document).ready(function() {
    // showing the answer div when the form is submitted
    $('#scienceQuestionForm').on('submit', function(event) {
        event.preventDefault(); // Prevent the form from submitting the traditional way
        $('#scienceAnswer').show(); // Show the answer div
    });

    $("#scienceQuestionForm").submit(function(event) {
        event.preventDefault();
        var questionText = $("#scienceQuestion").val();

        $("#scienceAnswer").text("Loading...");

        $.ajax({
            url: 'https://osvita-plus-047c3f277d12.herokuapp.com/askScience',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify({ question: questionText }),
            success: function(data) {
                $("#scienceAnswer").html(data.answer);
            },
            error: function() {
                $("#scienceAnswer").text("An error occurred. Please try again.");
            }
        });
    });

    // Upload PDF file and send custom prompt
    $('#pdf-file').on('change', function(event) {
        event.preventDefault();
        var formData = new FormData($('#upload-form')[0]);

        $.ajax({
            url: '/upload',
            type: 'POST',
            data: formData,
            processData: false,
            contentType: false,
            success: function(data) {
                $('#pdf-result').show();
                $('#query-result').text('PDF uploaded successfully. ' + data.answer);
            },
            error: function() {
                $('#pdf-result').show();
                $('#query-result').text('An error occurred while uploading the PDF.');
            }
        });
    });

    // Show the query result div when the query button is clicked
    $('#query-button').on('click', function(event) {
        event.preventDefault(); // Prevent the default button behavior
        var queryText = $('#query-input').val();

        $.ajax({
            url: '/query',
            type: 'POST',
            contentType: 'application/json',
            data: JSON.stringify({ query: queryText }),
            success: function(data) {
                $('#query-result').text(data.response);
            },
            error: function() {
                $('#query-result').text('An error occurred while processing the query.');
```

```
            }
        });

        $('#pdf-result').show(); // Show the query result div
    });

    ////////////////////////////////////////////////////////////////////////
    ///////// Google Search /////////Google Search//////////////////////////
    ////////////////////////////////////////////////////////////////////////

        $('#googleQuestionForm').on('submit', function(event) {
            event.preventDefault(); // Prevent the form from submitting the traditional way
            $('#googleAnswer').show(); // Show the answer div
        });

        $("#googleQuestionForm").submit(function(event) {
            event.preventDefault();
            var questionText = $("#googleQuestion").val();

            $("#googleAnswer").text("Loading...");

            $.ajax({
                url: '/askGoogle',
                type: 'POST',
                contentType: 'application/json',
                data: JSON.stringify({ question: questionText }),
                success: function(data) {
                    $("#googleAnswer").html(data.answer);
                },
                error: function() {
                    $("#googleAnswer").text("An error occurred. Please try again.");
                }
            });
        });
        ////////////////////////////////////////////////////////
});

function uploadFile() {
    var formData = new FormData(document.getElementById('upload-form'));

    $.ajax({
        url: '/upload',
        type: 'POST',
        data: formData,
        processData: false,
        contentType: false,
        success: function(data) {
            $('#pdf-result').show();
            $('#query-result').text('PDF uploaded successfully. ' + data.answer);
        },
        error: function() {
            $('#pdf-result').show();
            $('#query-result').text('An error occurred while uploading the PDF.');
        }
    });
}
```

about.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/tools.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/about.css') }}">
  <title>Освіта+</title>
  <script src="{{ url_for('static', filename='js/header.js') }}"></script>
</head>
<body>
  <iframe src="{{ url_for('header') }}" style="width:100%; border:none;"></iframe>

<section id="AboutIntro">
    <div id="AboutIntroLeft">
      <p class="WhoWeAre"><b>Про сайт</b></p>
        <p class="AboutUs">Освіта+ - це освітній веб-сайт створений в рамках бакалаврської
          дипломної роботи студента Одеського Державного Екологічного Університету Цуркана
          Артема Ігоровича.
        </p>
        <p class="AboutUs">
          Освіта+ — це інноваційний вебсайт, який поєднує в собі найкращі інструменти для
забезпечення
          високоякісного освітнього досвіду. Він інтегрує WolframAlpha для складних обчислень,
          Google Docs для створення та спільного редагування документів, Google Search для
швидкого
          пошуку інформації, а також багато інших корисних ресурсів. Використовуючи передові
технології,
          такі як Langchain, OpenAI API, Flask, HTML, CSS і JavaScript, Освіта+ створює зручне та
ефективне
          середовище для навчання. Користувачі можуть легко шукати інформацію, створювати
інтерактивні
          документи, проводити детальний аналіз даних і багато іншого.
        </p>
          <p class="AboutUs">
```

```
          Освіта+ робить процес навчання більш інтерактивним і продуктивним, дозволяючи
адаптувати його
          до індивідуальних потреб кожного учня. Завдяки своєму багатофункціональному підходу,
Освіта+
          стає незамінним інструментом для сучасного навчання.
        </p>
    </div>
    <div id="AboutIntroRight">
      <img  id="AboutIntroRightImage"  src="{{  url_for('static',  filename='img/about-us.jpeg')
}}"/>
    </div>
</section>
<footer class="footer">
  <div class="column1">
     <p class="footerBoldText">Освіта+</p>
     <p>Найсучасний набір інструментів і</p>
     <p>матеріалів для допомоги у навчанні</p>
  </div>
  <div class="column2">
     <p class="footerBoldText">Сторінки</p>
     <div class="Links">
        <div       ><a       class="FooterLink"       href="{{       url_for('index')       }}"
target="_parent">Головна</a></div>
        <div><a class="FooterLink" href="{{ url_for('tools') }}" target="_parent">Інструменти та
Матеріали</a></div>
        <div><a    class="FooterLink"    href="{{    url_for('about')    }}"    target="_parent">Про
Нас</a></div>
        <div><a       class="FooterLink"       href="{{       url_for('personal_space')       }}"
target="_parent">Особистий кабінет</a></div>
     </div>
  </div>
  <div class="column3">
     <p class="footerBoldText">Контакти</p>
     <p>artemtsurkan002@gmail.com</p>
  </div>
</footer>
</body>
</html>
```

personalSpace.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/index.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/tools.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/about.css') }}">
  <link rel="stylesheet" href="{{ url_for('static', filename='css/personalSpace.css') }}">
  <title>Освіта+</title>
</head>
<body>
  <iframe src="{{ url_for('header') }}" style="width:100%; border:none;"></iframe>
  <section id="AboutIntro">
    <div id="AboutIntroLeft">
      <p class="WhoWeAre"><b>Вітаємо у Вашому Особистому Кабінеті</b></p>
        <p class="AboutUs">Тут будуть публікуватися останні новини та анонси.
        </p>
    </div>
    <div id="AboutIntroRight">
      <img id="AboutIntroRightImage" src="{{ url_for('static', filename='img/PersonalSpace.png')
}}"/>
    </div>
</section>
<footer class="footer">
  <div class="column1">
     <p class="footerBoldText">Освіта+</p>
     <p>Найсучасний набір інструментів і</p>
     <p>матеріалів для допомоги у навчанні</p>
  </div>
  <div class="column2">
     <p class="footerBoldText">Сторінки</p>
     <div class="Links">
        <div       ><a       class="FooterLink"       href="{{       url_for('index')       }}"
target="_parent">Головна</a></div>
        <div><a class="FooterLink" href="{{ url_for('tools') }}" target="_parent">Інструменти та
Матеріали</a></div>
        <div><a    class="FooterLink"    href="{{    url_for('about')    }}"    target="_parent">Про
Нас</a></div>
        <div><a       class="FooterLink"       href="{{       url_for('personal_space')       }}"
target="_parent">Особистий кабінет</a></div>
     </div>
  </div>
  <div class="column3">
     <p class="footerBoldText">Контакти</p>
     <p>artemtsurkan002@gmail.com</p>
  </div>
</footer>
</body>
</html>
```

index.css

```
:root{
    --blueish-purple: #3366ff;
```

```
    --section-gray: purple;
}

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}
html{
    font-family: Arial;
}
header{
    display: flex;
    justify-content: space-between;
    align-items:center;
    align-content: center;
    padding: 20px;
    margin: 20px 55px;
    font-weight: bold;
}

header .WebsiteTitle{
    font-size: 1.9em;
    color: var(--blueish-purple);
}
header .CentralPages{
    display: flex;
    justify-content:center;
    flex:1;
    font-size: 1.5em;
}
.link{
    text-decoration: none;
    color: black;
}
.current-page{
    text-decoration: underline;
}

 header .LoginRegister{
    color: white;
    text-decoration: none;
}
.centralPage{
    margin-left: 15px;
    margin-right: 15px;
}
.button{
    background-color: var(--blueish-purple);
    border: none;
    color: white;
    padding: 15px 33px;
    font-weight: bold;
    cursor: pointer;
    border-radius: 5px;
}

.about{
    display: flex;
    margin-left: 30px;
}
.Intro{
    display:flex;
    justify-content:space-between;
    align-items:center;
    position: relative;
    margin-left: 10px;
}
.Intro-Left{
    margin: 30px;
    position: relative;
    top: -45px;
    padding-left: 70px;
    padding-bottom: 30px;
    left: 25px;
}
.boldText{
    padding-bottom: 35px;
    font-size: 3.5vw;
}
.grayText{
    color: gray;
    border-left: 2px solid var(--blueish-purple);
    padding-left: 10px;
    font-size: 1.5vw;
}
#Intro-Image{
width: 700px;
height: 630px;
position: relative;
top: -70px;
left: 20px;
}
```

```
.Intro-Right{
    display: flex;
    margin: 10px;
    padding: 20px;
}
#findOutMoreButton{
    margin-top: 35px;
}
/*========NOTE=======*/
#FlexWrapper{
    display: flex;
    flex-direction: column;
    margin-top: -50px;
    margin-left: auto;
    margin-right: auto;
    justify-content: center !important;
    align-items: center !important;
    align-content: center !important;
}
#FlexWrapper #AskURquestion{
    color: black;
    font-size: 21px;
    font-weight: bold;
    max-width: 100%;
    width: auto;
    margin-bottom: 50px;
    padding-top: 50px;
    text-align: center;
    margin-left: auto;
    margin-right: auto;
}

/*========AI=======*/

#FindOutMore{
    flex: 1;
    /*background-color: var(--section-gray);*/
    color: white;
    font-size: 2em;
    font-weight: bold;
    max-width: 100%;
    width: 100vw;
    margin: 0;
    padding: 0;
    height: 800px;
    justify-content: center !important;
    align-items: center !important;
    align-content: center !important;
    margin-left: auto;
    margin-right: auto;
}

#FindOutMore #AI{
    flex: 1;
    margin-left: auto;
    margin-right: auto;
    justify-content: center !important;
    align-items: center !important;
    align-content: center !important;
}

textarea::placeholder {
    color: grey;
}
#AI #questionInput {
    /*flex: 1;*/
    display:flex;
    width: 800px;
    padding: 10px;
    margin-top: 17px;
    margin-bottom: 20px;
    border: 1px solid var(--blueish-purple);
    border-radius: 4px;
    box-sizing: border-box;
    resize: none;
    overflow: auto;
    margin-left: auto;
    margin-right: auto;
    align-self: center;
}
/* Style the submit button */
#AskButton {
    display: flex;
    width: 10%;
    padding: 10px;
    /*position: relative;
    left: 643px;*/
    border: none;
    border-radius: 4px;
    background-color: var(--blueish-purple);
    color: white;
    cursor: pointer;
    font-size: 16px;
    justify-content: center;
```

```
    margin-left: auto;
    margin-right: auto;

}

#questionForm button:hover {
    background-color: #0056b3;
}

#InstructionsText{
    color: grey;
    font-size: 15px;
    font-weight: normal;
    max-width: 100%;
    margin-bottom: 50px;
    padding-left: 40px;
    padding-right: 40px;
    padding-top: 50px;
    text-align: center;
    flex: 1;
}

/* Style the answer container */
#answer {
    flex: 1;
    padding: 70px;
    overflow: auto;
    width: 800px;
    margin-top: 30px;
    margin-bottom: 100px;
    padding: 20px;
    border: 1px solid var(--blueish-purple);
    border-radius: 4px;
    box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);
    font-size: 15px;
    color:black;
    font-weight: normal;
    height: auto;
    margin-left: auto;
    margin-right: auto;
}

/*=============FOOTER=============*/
footer{
    display: flex;
    padding-top: auto;
    margin-top: auto;
    background-color: var(--blueish-purple);
    align-content: center;
    font-size: 1.5vw;
    color: white;
    height: 320px;
    text-decoration: none;
    width: 100%;
}

.footerBoldText{
    font-weight: bold;
    font-size: 2.2vw;
    margin-bottom: 15px;
}

.Links{
    position:relative;
    left:2px;
}

.FooterLink{
    text-decoration: none;
    color: white;
}

.column1{
    margin: 50px 115px;
}

.column2{
    margin: 50px 70px;
}

.column3{
    margin: 50px 70px;
}
```

tools.css

```
:root {
    --blueish-purple: #3366ff;
    --section-gray: purple;
}

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
```

```
}

html {
    font-family: Arial;
}

header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 20px;
    margin: 20px 55px;
    font-weight: bold;
}

header .WebsiteTitle {
    font-size: 1.9em;
    color: var(--blueish-purple);
}

header .CentralPages {
    display: flex;
    justify-content: center;
    flex: 1;
    font-size: 1.5em;
}

.link {
    text-decoration: none;
    color: black;
}

.current-page {
    text-decoration: underline;
}

header .LoginRegister {
    color: white;
    text-decoration: none;
}

.centralPage {
    margin-left: 15px;
    margin-right: 15px;
}

.button {
    background-color: var(--blueish-purple);
    border: none;
    color: white;
    padding: 15px 33px;
    font-weight: bold;
    cursor: pointer;
    border-radius: 5px;
    text-align: center;
    display: inline-block;
}

.banner-section {
    background: linear-gradient(to left, #6e69f0, #75b1ff);
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 20px 70px;
    overflow: hidden;
    box-sizing: border-box;
    height: 150px;
}

.banner-section img {
    position: relative;
    top: -23px;
}

.banner-section #WolframAlphaLogo {
    width: 260px;
    height: 45px;
}

.banner-section #LangChainLogo {
    width: 270px;
    height: 55.88px;
}

.banner-section #OpenAILogo {
    width: 238px;
    height: 47.88px;
}

.banner-section #MediaPipeLogo {
    width: 250px;
    height: 67.43px;
}
```

```
#FlexWrapper {
    display: flex;
    flex-direction: column;
    margin-top: -50px;
    margin-left: auto;
    margin-right: auto;
    justify-content: center !important;
    align-items: center !important;
    align-content: center !important;
    width: 100%;
}

/* Section styles */
#CatalogueSection1,
#CatalogueSection2,
#CatalogueSection3 {
    display: flex;
    flex-direction: column;
    font-size: 22px;
    font-weight: bold;
    width: 100%;
    height: auto;
    position: relative;
    text-align: center;
    box-sizing: border-box;
    box-shadow: 1px 2px 4px rgba(0.5, 0.5, 0.5, 0.5);
    justify-content: center;
    align-content: center;
    padding: 50px 20px;
}

#CatalogueSection1 {
    background: linear-gradient(to right, #f5daab, #faa0a0);
    color: black;
}

#CatalogueSection2 {
    background-color: #faf2ca;
    color: black;
    justify-content: flex-start;
}

#CatalogueSection3 {
    background-color: var(--section-gray);
    color: white;
}

/* Section headings */
#AImeetsWolfram,
#AImeetsPDF,
#AImeetsGoogleSearch {
    padding-top: 15px;
    font-family: "Rampart One", sans-serif;
    font-style: normal;
    font-weight: 400;
    font-size: 50px;
    align-content: center;
    justify-content: center;
}

/* Section content containers */
#catalogueSec1,
#catalogueSec2,
#catalogueSec3 {
    flex: 1;
    margin: 20px;
    padding: 20px;
    align-items: center;
    font-size: 18px;
    align-content: center;
    justify-content: center;
}

/* Form styles */
#scienceQuestionForm,
#upload-form,
#googleQuestionForm {
    flex: 1;
    background: transparent;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0);
    margin-bottom: 20px;
    justify-content: center;
    align-content: center;
}

#scienceQuestionForm label,
#upload-form label,
#googleQuestionForm label {
    font-size: 18px;
    color: #333;
    justify-content: center;
    align-content: center;
```

```
}

#scienceQuestionForm input[type=text],
#upload-form input[type="file"],
#googleQuestionForm input[type=text] {
    width: 100%;
    max-width: 500px;
    padding: 10px;
    margin: 10px 0;
    border: 1px solid #ddd;
    border-radius: 20px;
    box-sizing: border-box;
    padding-bottom: 13px;
    margin-bottom: 30px;
    justify-content: center;
    align-content: center;
}

#scienceQuestionForm button,
#upload-form button,
#googleQuestionForm button {
    padding: 10px 25px;
    background-color: var(--blueish-purple);
    color: white;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-size: 18px;
    justify-content: center;
    align-content: center;
    margin-top: 10px;
    text-align: center;
    display: inline-block;
}

#upload-form {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    width: 100%;
}

#upload-form textarea {
    width: 100%;
    max-width: 500px;
    height: 90px;
    margin-bottom: 10px;
    padding: 10px;
    font-size: 16px;
    border: 1px solid #ccc;
    box-sizing: border-box;
    resize: none;
    overflow: auto;
    border-radius: 10px;
}

#upload-form button:hover {
    background-color: #0056b3;
}

#pdf-result {
    margin-top: 20px;
}

#query-result {
    font-size: 18px;
    color: #333;
}

/* Response containers */
#CatalogueSection1 #scienceAnswer,
#CatalogueSection3 #googleAnswer,
#pdf-result {
    padding: 20px;
    background: #ffffff;
    border-radius: 20px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    margin-top: 20px;
    min-height: 50px;
    color: black;
    width: 61%;
    margin: 0 auto;
    justify-content: center;
    align-items: center;
    display: none;
    align-content: center;
}
```

about.css

```
:root{
    --blueish-purple: #3366ff;
    --section-gray: #cacfe0;
}
```

```
* {
        /*outline: 1px solid red;*/
        margin: 0;
        padding: 0;
        box-sizing: border-box;
}

#AboutIntro{
    display: flex;
    padding-left: 50px;
    padding-right: 50px;
    padding-top: 10px;
    padding-bottom: 80px !important;
    margin-bottom: 50px !important;
}

#AboutIntroLeft{
    padding-left: 61px;
    margin-left: 20px;
    margin-right: 30px;
}

.WhoWeAre{
    font-size: 22px;
}

.AboutUs{
    padding-top: 18px;
    font-size: 20px;
    text-align: justify;
}

#AboutIntroRight{
padding-right: 10px;
padding-right: 50px;
margin: 30px;
}

#AboutIntroRightImage{
/*outline: 1px solid black;*/
width: 457px;
height: 353px;
}

/*=============FOOTER=============*/
footer{
    display: flex;
    padding-top: auto;
    margin-top: auto;
    background-color: var(--blueish-purple);
    align-content: center;
    font-size: 1.5vw;
    color: white;
    height: 320px;
    text-decoration: none;
    width: 100%;
}

.footerBoldText{
    font-weight: bold;
    font-size: 2.2vw;
    margin-bottom: 15px;
}

.Links{
    position:relative;
    left:2px;
}

.FooterLink{
    text-decoration: none;
    color: white;
}

.column1{
    margin: 50px 115px;
}

.column2{
    margin: 50px 70px;
}

.column3{
    margin: 50px 70px;
}
```

personalSpace.css

```
:root{
    --blueish-purple: #3366ff;
    --section-gray: #cacfe0;
}
* {
    margin: 0;
```

```
    padding: 0;
    box-sizing: border-box;
}
html{
    font-family: Arial;
}

#AboutIntro{
    display: flex;
    justify-content: center;
    align-items: center;
    padding: 50px;
    flex-direction: row;
    text-align: center;
}
#AboutIntroLeft{
    padding: 20px;
}
.WhoWeAre{
    font-size: 22px;
}
.AboutUs{
    padding-top: 18px;
    font-size: 20px;
}
#AboutIntroRight{
    padding: 20px;
}

#AboutIntroRightImage{
    width: 100%;
    max-width: 457px;
    height: auto;
}
app.py
import json
import logging
from flask import Flask, render_template, request, jsonify, redirect, url_for, session
from flask_cors import CORS
import os
from authlib.integrations.flask_client import OAuth
from openai import OpenAI
from langchain.utilities.wolfram_alpha import WolframAlphaAPIWrapper
from langchain_community.document_loaders import PyPDFLoader
from langchain_community.utilities import SerpAPIWrapper
from werkzeug.middleware.proxy_fix import ProxyFix
from flask_session import Session
import os
import base64

# Configuring logging
logging.basicConfig(level=logging.DEBUG)

UPLOAD_FOLDER = '/tmp/uploads'
os.makedirs(UPLOAD_FOLDER, exist_ok=True)

# Retrieving API keys from environment variables
api_key = os.getenv("OPENAI_API_KEY")
wolfram_app_id = os.getenv("WOLFRAM_ALPHA_APPID")
serpapi_api_key = os.getenv("SERPAPI_API_KEY")
google_client_id = os.getenv("GOOGLE_CLIENT_ID")
google_client_secret = os.getenv("GOOGLE_CLIENT_SECRET")

if not all([api_key, wolfram_app_id, serpapi_api_key, google_client_id, google_client_secret]):
    logging.error("One or more environment variables are missing.")
    raise ValueError("One or more environment variables are missing.")

# Initializing OpenAI
client = OpenAI(api_key=api_key)
wolfram = WolframAlphaAPIWrapper()
search = SerpAPIWrapper(serpapi_api_key=serpapi_api_key)

# Initializing Flask app
app = Flask(__name__)
app.secret_key = os.getenv("FLASK_SECRET_KEY", "supersecretkey")
CORS(app, supports_credentials=True, resources={r"/ask": {"origins": "*"}})

app.config['SESSION_TYPE'] = 'filesystem'
app.config['SESSION_COOKIE_SECURE'] = True
app.config['SESSION_COOKIE_HTTPONLY'] = True
app.config['SESSION_COOKIE_SAMESITE'] = 'Lax'
Session(app)

app.wsgi_app = ProxyFix(app.wsgi_app, x_for=1, x_proto=1, x_host=1, x_port=1, x_prefix=1)

# Setting up OAuth
oauth = OAuth(app)
# Setting OAuth with Google's discovery document
oauth.register(
    name='google',
    server_metadata_url='https://accounts.google.com/.well-known/openid-configuration',
    client_id=google_client_id,
    client_secret=google_client_secret,
    client_kwargs={'scope': 'openid email profile'}
```

```
)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/about')
def about():
    return render_template('about.html')

@app.route('/personal-space')
def personal_space():
    return render_template('personalSpace.html')

@app.route('/tools')
def tools():
    return render_template('tools.html')

@app.route('/header')
def header():
    return render_template('header.html')

'''=======OpenAI API direct========================================='''
@app.route('/ask', methods=['POST'])
def ask_question():
    data = request.get_json()
    question = data.get('question', '')

    if not question:
        logging.error('No question provided in the request')
        return jsonify({'error': 'No question provided'}), 400

    try:
        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "system", "content":
                    "You are a helpful assistant for the website Osvita+."
                    "Osvita+ is a website that provides users with various tools "
                    "and resources to answer their questions, including integrations "
                    "with Wolfram Alpha and Google Search (SerpAPI). "
                    "The website has the following sections:\n"
                    "1. Home page: An introductory page to engage users.\n"
                    "2. Ask AI: A feature allowing users to ask questions using OpenAI, "
                    "perfect for FAQ about the website itself.\n"
                    "3.  Tools  page:  Integrates  OpenAI  with  Wolfram  Alpha,  local  PDFs,  and
SerpAPI.\n"
                    "4. About Us page: Information about the website.\n"
                    "5. Personal dashboard (Особистий кабінет): For authenticated users, "
                    "gives access to latest news and upcoming new features.\n"
                    "Feel free to answer any questions related to the features "
                    "and functionality of Osvita+. Answer the questions in Ukrainian."
                },
                {"role": "user", "content": question},
            ]
        )
        answer = response.choices[0].message.content.strip()
    except Exception as e:
        logging.exception('An error occurred while querying OpenAI')
        answer = "An error occurred: " + str(e)

    return jsonify({'answer': answer})

'''=======Wolfram Alpha========================================='''
@app.route('/askScience', methods=['POST'])
def ask_science_question():
    data = request.get_json()
    question = data.get('question', '')

    if not question:
        logging.error('No question provided in the request')
        return jsonify({'error': 'No question provided'}), 400

    try:
        # Getting the response from Wolfram Alpha
        wolfram_answer = wolfram.run(question)

        # Using OpenAI to process the response from Wolfram Alpha
        openai_prompt = f"The following is a response from Wolfram Alpha about the question
'{question}':\n\n{wolfram_answer}\n\nCan you explain this in simpler terms or provide additional
relevant information? Answer in Ukrainian"

        openai_response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "system", "content": "You are a helpful assistant."},
                {"role": "user", "content": openai_prompt},
            ]
        )
        openai_answer = openai_response.choices[0].message.content.strip()

        # Combining the responses
        combined_answer = f"Wolfram Alpha Response: {wolfram_answer}\n\nOpenAI Enhanced
Explanation: {openai_answer}"
```

```
    except Exception as e:
        logging.exception('An error occurred while querying Wolfram Alpha or OpenAI')
        combined_answer = "An error occurred: " + str(e)

    return jsonify({'answer': combined_answer})

'''=======Local PDF========================================='''
@app.route('/upload', methods=['POST'])
def upload_file():
    if 'file' not in request.files or 'prompt' not in request.form:
        logging.error('No file part or prompt provided')
        return jsonify({'error': 'No file part or prompt provided'}), 400

    file = request.files['file']
    custom_prompt = request.form['prompt']

    if file.filename == '':
        logging.error('No selected file')
        return jsonify({'error': 'No selected file'}), 400

    try:
        file_path = os.path.join(UPLOAD_FOLDER, file.filename)
        file.save(file_path)
        logging.info(f'File saved to {file_path}')
        text = extract_text_from_pdf(file_path)

        prompt = f"Here is some information that might help: {text}\n\n{custom_prompt}"

        response = query_openai(prompt)

        return jsonify({'text': text, 'answer': response}), 200
    except Exception as e:
        logging.exception('An error occurred while saving the file or querying OpenAI')
        return jsonify({'error': f'An error occurred: {str(e)}'}), 500

@app.route('/query', methods=['POST'])
def query():
    data = request.json
    query_text = data.get('query', '')

    if not query_text:
        logging.error('No query text provided')
        return jsonify({'error': 'No query text provided'}), 400

    try:
        response = query_openai(query_text)
        return jsonify({'response': response}), 200
    except Exception as e:
        logging.exception('An error occurred while querying OpenAI')
        return jsonify({'error': f'An error occurred: {str(e)}'}), 500

def extract_text_from_pdf(pdf_path):
    try:
        loader = PyPDFLoader(pdf_path)
        documents = loader.load()  # Loading the document
        text = " ".join([doc.page_content for doc in documents])  # Extracting text
        return text
    except Exception as e:
        logging.exception('An error occurred while extracting text from PDF')
        raise

def query_openai(prompt):
    try:
        response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "system", "content": "You are a helpful assistant."},
                {"role": "user", "content": prompt},
            ]
        )
        return response.choices[0].message.content.strip()
    except Exception as e:
        logging.exception('An error occurred while querying OpenAI')
        raise

'''=======SerpAPI==========================================='''
@app.route('/askGoogle', methods=['POST'])
def ask_google_question():
    data = request.get_json()
    question = data.get('question', '')

    if not question:
        logging.error('No question provided in the request')
        return jsonify({'error': 'No question provided'}), 400

    try:
        # Getting the response from SerpAPI
        search_results = search.run(question)
        logging.debug(f"Raw search results: {search_results}")

        # Using OpenAI to process the response from SerpAPI
```

```
        openai_prompt = f"The following is a response from a Google Search about the question '{question}':\n\n{search_results}\n\nCan you summarize and explain this in simpler terms? Answer in Ukrainian."

        openai_response = client.chat.completions.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "system", "content": "You are a helpful assistant."},
                {"role": "user", "content": openai_prompt},
            ]
        )
        openai_answer = openai_response.choices[0].message.content.strip()

        # Combining the responses
        combined_answer = f"Google Search Results: {search_results}\n\nOpenAI Enhanced Explanation: {openai_answer}"

    except Exception as e:
        logging.exception('An error occurred while querying SerpAPI or OpenAI')
        combined_answer = "An error occurred: " + str(e)

    return jsonify({'answer': combined_answer})

'''=======Google Auth========================================'''
@app.route('/login')
def login():
    # Generating a random nonce
    nonce = base64.urlsafe_b64encode(os.urandom(30)).decode('utf-8')
    session['nonce'] = nonce  # Storing nonce in session for later validation

    redirect_uri = url_for('auth', _external=True)
    return oauth.google.authorize_redirect(redirect_uri, nonce=nonce)

@app.route('/auth')
def auth():
    try:
        token = oauth.google.authorize_access_token()
        nonce = session.pop('nonce', None)  # Retrieving and removing the nonce from session
        resp = oauth.google.parse_id_token(token, nonce=nonce)

        if resp:
            session['user'] = {
                'name': resp.get('name'),
                'email': resp.get('email'),
                'profile_picture': resp.get('picture')
            }
            logging.info(f"User {resp.get('email')} logged in successfully.")
            return redirect(url_for('index'))
        else:
            logging.error("Failed to verify ID token.")
            return redirect(url_for('login_error'))

    except Exception as e:
        logging.exception("Authentication failed.")
        return jsonify({'error': str(e)}), 500

@app.route('/logout')
def logout():
    session.pop('user', None)
    return redirect(url_for('index'))

@app.route('/is-logged-in')
def is_logged_in():
    return jsonify({'is_logged_in': 'user' in session})


if __name__ == '__main__':
    logging.basicConfig(level=logging.DEBUG)
    app.run(debug=True)
```

requirements.txt

```
Flask-Cors==4.0.0
Flask==3.0.2
Jinja2==3.1.3
MarkupSafe==2.1.5
PyPDF2==3.0.1
PyYAML==6.0.1
SQLAlchemy==2.0.29
Werkzeug==3.0.1
aiohttp==3.9.3
aiosignal==1.3.1
annotated-types==0.6.0
anyio==4.3.0
attrs==23.2.0
blinker==1.7.0
certifi==2024.2.2
chardet==5.2.0
charset-normalizer==3.3.2
click-datetime==0.2
click==8.1.7
dataclasses-json==0.6.4
distro==1.9.0
et-xmlfile==1.1.0
frozenlist==1.4.1
```

```
gunicorn==21.2.0
h11==0.14.0
httpcore==1.0.5
httpx==0.27.0
idna==3.6
itsdangerous==2.1.2
jaraco.context==4.3.0
jsonpatch==1.33
jsonpointer==2.4
langchain-community==0.0.31
langchain-core==0.1.37
langchain-openai==0.1.1
langchain-text-splitters==0.0.1
langchain==0.1.13
langsmith==0.1.38
marshmallow==3.21.1
more-itertools==10.2.0
multidict==6.0.5
mypy-extensions==1.0.0
numpy==1.26.2
openai==1.14.3
opencv-python==4.8.1.78
openpyxl==3.1.2
orjson==3.10.0
packaging==23.2
pillow==10.3.0
protobuf==4.25.1
pyHS100==0.3.5.2
pydantic==2.6.4
pydantic_core==2.16.3
regex==2023.12.25
reportlab==4.2.0
requests==2.31.0
setuptools==69.0.2
six==1.16.0
sniffio==1.3.1
tenacity==8.2.3
tiktoken==0.6.0
tqdm==4.66.2
typing-inspect==0.9.0
typing_extensions==4.10.0
urllib3==2.2.1
wolframalpha==5.0.0
xmltodict==0.13.0
yarl==1.9.4
pypdf==4.2.0
google-search-results==2.4.2
appwrite==1.2.0
authlib==1.3.0
cffi==1.16.0
cryptography==42.0.7
pycparser==2.22
Flask-Session==0.8.0

Procfile
web: gunicorn app:app
runtime.txt
python-3.11.6
```