МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

Даниленко Денис Денисович

Програмний модуль чат бота для оптимізації роботи з документацією/ Chatbot software module to optimize the work with documentation

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Штучний Інтелект

Кваліфікаційна робота

Виконав студент групи КНШІ-41
Д.Д. Даниленко

Науковий керівник
к.т.н. О.П. Адамів

Кваліфікаційну роботу допущено до захисту
«___»_____ 20___ р.

В.о. завідувача кафедри
_____ Н.М. Васильків

ТЕРНОПІЛЬ – 2025

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність 122 – Комп'ютерні науки
освітньо-професійна програма – Штучний Інтелект

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.М. Васильків
«____» _____2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Даниленко Денис Денисович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Програмний модуль чат бота для оптимізації роботи з документацією / Chatbot software module to optimize the work with documentation

керівник роботи О.П.Адамів к.т.н
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 20 грудня 2024 р. № 938.
2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2025 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити:

- Проаналізувати сучасні підходи до автоматизації документообігу, зокрема використання інтелектуальних чат-ботів, що функціонують на базі моделей штучного інтелекту (GPT) для розпізнавання, пошуку та обробки документів;
- Розробити концепцію архітектури чат-бота, орієнтованого на роботу з документами різних форматів у режимі реального часу, з використанням Telegram як основного інтерфейсу взаємодії;
- Побудувати модульну архітектуру системи, що забезпечує індексацію документів, пошук за змістом, генерацію відповідей на запити користувача та інтеграцію з OpenAI API;
- Реалізувати прототип чат-бота, який забезпечує діалогову взаємодію, семантичний пошук у документах та функції експорту відповідей у популярних форматах (.txt, .pdf, .docx);

5. Перелік графічного матеріалу в роботі:

- архітектурна схема чат-бота для обробки документів
- візуальне представлення логічних зв'язків між основними компонентами системи
- схема алгоритму роботи чат-бот

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 20 грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2025 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02. 2025 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 01.04.2025 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2025 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2025 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2025 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2025 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2025 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06. 2025 р.	

Студент_____ Д.Д.Даниленко
(підпис) (прізвище та ініціали)

Керівник кваліфікаційної роботи_____ О.П.Адамів
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи: 51 с., 17 рис., 3 додатки, 27 джерел.

Об’єктом дослідження є інтелектуальні чат-боти, які застосовуються для автоматизації обробки документації в цифрових системах.

Метою роботи є розроблення програмного модуля чат-бота, що використовує сучасні мовні моделі штучного інтелекту (GPT) для взаємодії з користувачем, пошуку, аналізу та генерації відповідей на основі вмісту текстових документів різних форматів.

Методи дослідження: методи обробки природної мови (NLP), використання embedding-векторів і семантичного пошуку, інтеграція з OpenAI API для генерації відповідей; застосування Telegram API для побудови користувацького інтерфейсу та Python-бібліотек для обробки PDF, DOCX та TXT-документів.

Результатом роботи є чат-бот, здатний розпізнавати запити природною мовою, знаходити релевантну інформацію в документах, генерувати стислий змістовний текст і експортувати його у зручному форматі.

Розроблений модуль може бути використаний в організаціях для автоматизації документообігу, зменшення навантаження на персонал та підвищення продуктивності, а також як база для створення інтелектуальних корпоративних асистентів.

Ключові слова: ЧАТ-БОТ, ДОКУМЕНТООБІГ, ШТУЧНИЙ ІНТЕЛЕКТ, GPT, ОБРОБКА ПРИРОДНОЇ МОВИ, TELEGRAM.

ANNOTATION

The bachelor’s thesis report: 51 pages, 17 figures, 3 appendices, 27 sources.

The object of this research is intelligent chatbots used for automating document processing in digital systems.

The purpose of the work is to develop a chatbot software module that utilizes modern large language models (GPT) to interact with users, perform document search, analysis, and generate responses based on the content of various text document formats.

Research methods include natural language processing (NLP), embedding-based semantic search, integration with the OpenAI API for answer generation, use of Telegram API for user interaction, and Python libraries for handling PDF, DOCX, and TXT documents.

The result of the work is a chatbot capable of understanding natural language queries, extracting relevant information from documents, generating concise and informative responses, and exporting them in a user-friendly format.

The developed module can be used in organizations to automate document workflows, reduce staff workload, and increase productivity. It also serves as a foundation for building intelligent corporate assistants.

Keywords: CHATBOT, DOCUMENT WORKFLOW, ARTIFICIAL INTELLIGENCE, GPT, NATURAL LANGUAGE PROCESSING, TELEGRAM.

ЗМІСТ

ВСТУП

У сучасних умовах цифрової трансформації підприємства та організації щодня стикаються з необхідністю ефективного управління великими обсягами документації. Ручна обробка, пошук необхідної інформації та впорядкування документів потребують значних людських ресурсів і часу, що знижує продуктивність роботи та може призводити до помилок. Зростання обсягів даних та потреба в оперативному доступі до інформації висувають нові вимоги до автоматизації процесів документообігу.

Одним із перспективних рішень для оптимізації цих процесів є використання інтелектуальних чат-ботів, що функціонують на основі OpenAI API. Завдяки цій технології можлива реалізація гнучких та потужних алгоритмів аналізу текстів, генерації відповідей та обробки запитів користувачів у режимі реального часу. Використання моделей штучного інтелекту, таких як GPT, дозволяє чат-ботам не лише надавати швидкий доступ до необхідних даних, а й автоматизувати пошук, аналіз і впорядкування інформації. Інтеграція з базами даних, системами управління документами та іншими корпоративними сервісами розширює функціональність таких чат-ботів, дозволяючи їм виконувати завдання з підготовки звітів, витягування ключової інформації з документів та автоматизації рутинних операцій.

Актуальність дослідження обумовлена зростаючою потребою в підвищенні ефективності управління документацією, мінімізації людського фактора в процесі обробки даних та забезпеченні швидкого доступу до інформації. Автоматизація документообігу є критично важливою для компаній, що працюють у сферах юриспруденції, медицини, фінансів, освіти, логістики та інших галузях, де швидкий і точний доступ до документів має вирішальне значення.

Метою цього дослідження є аналіз можливостей використання OpenAI API для реалізації чат-ботів, орієнтованих на обробку документації, визначення

переваг і обмежень такого підходу, а також розробка власного програмного модуля. Основними завданнями дослідження є:

1. аналіз можливостей OpenAI API для розпізнавання, аналізу та генерації текстової інформації.
2. Розробка архітектури чат-бота, що дозволить ефективно обробляти запити користувачів і взаємодіяти з документами.
3. Оцінка продуктивності та ефективності запропонованого рішення в порівнянні з існуючими аналогами.
4. Оптимізація взаємодії користувачів із системою через вдосконалення інтерфейсу та забезпечення високої точності відповідей.

Очікується, що результати дослідження сприятимуть створенню гнучкого та адаптивного рішення для автоматизації роботи з документацією, що дозволить підприємствам зменшити навантаження на співробітників, покращити управління інформацією та підвищити загальну продуктивність.

У межах цього дослідження використовуватимуться такі методи: аналіз і синтез — для вивчення сучасних підходів до автоматизації документообігу та застосування штучного інтелекту; методи програмної інженерії — для розробки архітектури чат-бота і реалізації взаємодії з OpenAI API; експериментальне моделювання — для побудови і тестування функціоналу системи; порівняльний аналіз — для оцінки ефективності створеного рішення; а також методи тестування — для перевірки працездатності, точності відповідей та зручності інтерфейсу.

Об’єктом дослідження виступають процеси обробки, управління та аналізу текстових документів у межах інформаційних систем, а також можливості інтеграції інтелектуальних агентів у корпоративні сервіси для забезпечення швидкого доступу до інформації.

Практичне значення роботи полягає у створенні універсального програмного модуля, який можна адаптувати для потреб різних підприємств і галузей.

1 СТАН ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

У світі бізнесу, де інформація є ключовим активом, підприємства та організації щоденно стикаються з безпрецедентними обсягами різноманітної документації. Цей інформаційний потік, що охоплює широкий спектр ділових паперів – від стратегічно важливих звітів та юридично зобов'язуючих контрактів до складних технічних специфікацій, детальних фінансових виписок та незліченної кількості інших офіційних записів – вимагає не просто формального обліку, а й створення гнучкої та ефективної системи всебічного управління. Основними цілями такого управління є забезпечення миттєвого та безперешкодного доступу до критично важливої інформації для всіх уповноважених співробітників, а також підтримка бездоганного рівня організації всього документообігу, що є фундаментом для злагодженої роботи всієї організації. Традиційні підходи до управління документацією, які часто ґрунтуються на застарілих ручних методах обробки та використанні паперових носіїв, у сучасних умовах можуть виявитися надзвичайно неефективними та часозатратними. Ці процеси неминуче створюють значне, часто непотрібне, навантаження на персонал, відволікаючи цінних співробітників від виконання їхніх основних обов'язків, і, що особливо важливо, значно підвищують ризик виникнення людських помилок, які в діловому світі можуть призвести до серйозних фінансових втрат та репутаційних ризиків для організації.

В епоху безперервної боротьби за оптимізацію всіх аспектів бізнес-процесів, включаючи скорочення витрат, підвищення продуктивності та мінімізацію помилок, автоматизація документообігу перетворюється з бажаної інновації на життєву необхідність для будь-якої амбітної та прогресивної організації, яка прагне зберегти свою конкурентоздатність на ринку. Серед широкого спектру сучасних технологічних інновацій, спеціально розроблених для значного полегшення та оптимізації роботи з документацією, особливе місце та зростаючу популярність заслужено займають інтелектуальні чат-боти. Ці передові програмні агенти, що

функціонують на основі штучного інтелекту, здатні не лише безперешкодно інтегруватися з уже існуючими в організації системами управління документами, але й створювати для кінцевих користувачів інтуїтивно зрозумілий, зручний та надзвичайно ефективний інтерфейс для миттєвого отримання доступу до будь-якої необхідної інформації. Завдяки потужним та постійно вдосконалюваним алгоритмам штучного інтелекту (ШІ) та передовим технологіям обробки природної мови (Natural Language Processing, NLP) [11], чат-боти можуть виконувати широкий спектр складних та інтелектуальних завдань, значно виходячи за рамки простого пошуку за ключовими словами. Вони не лише здатні миттєво знаходити та ретельно аналізувати величезні обсяги різноманітних документів, але й надавати стислі та змістовні витяги з текстів, оперативно відповідати на широкий спектр запитів користувачів, сформульованих природною мовою, а також автоматизувати численні рутинні та монотонні операції, які неминуче супроводжують процеси документообігу, звільняючи таким чином цінний час співробітників для виконання більш творчих та стратегічно важливих завдань.

Однією з найважливіших та найбільш очевидних переваг використання чат-ботів у сфері документообігу є значне полегшення та прискорення процесу пошуку необхідних документів. Завдяки своїй здатності швидко та ефективно обробляти запити, сформульовані користувачем природною мовою, чат-боти можуть практично миттєво знаходити потрібну інформацію, навіть у випадках, коли користувач не пам'ятає точної назви файлу, дати його створення або його конкретного розташування в системі. Крім того, складні інтелектуальні алгоритми, що лежать в основі роботи чат-ботів, дозволяють їм не просто знаходити відповідні документи, а й здійснювати глибокий контекстний аналіз їхнього змісту, виділяючи ключові факти, важливі дані та основні ідеї. Вони також можуть автоматично здійснювати інтелектуальну класифікацію файлів за різними заданими критеріями, такими як тип документа (наприклад, договір, звіт, рахунок-фактура), дата створення або редагування, відповідальний співробітник чи відділ, а також ефективно контролювати версії документів, автоматично відстежуючи зміни та забезпечуючи користувачам доступ до найновішої та найактуальнішої

інформації, запобігаючи таким чином плутанині та непорозумінням, які можуть виникнути при роботі з застарілими версіями. Усі ці потужні можливості в сукупності сприяють значному поліпшенню загальної організації документопотоків в організації, істотному підвищенню продуктивності праці всіх співробітників, які працюють з документацією, та значному зменшенню кількості випадкових помилок, які можуть виникати в процесах ручної обробки великих обсягів даних.

Сфера практичного застосування чат-ботів у галузі документообігу є надзвичайно широкою, різноманітною та охоплює практично всі існуючі сектори сучасної економіки. У консервативному, але надзвичайно залежному від точності та оперативності обробки інформації юридичному секторі, ці інтелектуальні помічники можуть стати абсолютно незамінними при швидкому пошуку необхідних правових документів, проведенні глибокого аналізу складних багатосторінкових контрактів та автоматизації рутинних процедур обробки різноманітних юридичних запитів, що значно економить час юристів та мінімізує ризик пропуску важливих деталей. У фінансовій сфері, де точність та своєчасність обробки даних мають критичне значення, чат-боти здатні значно оптимізувати процеси обробки платіжних документів, проводити детальний аналіз великих обсягів фінансових транзакцій та автоматично генерувати різноманітні фінансові звіти, надаючи аналітикам та керівництву компанії оперативну та точну інформацію для прийняття обґрунтованих рішень. В галузі охорони здоров'я, де ефективне управління медичною документацією є життєво важливим, чат-боти можуть суттєво спростити ведення електронної медичної документації, автоматизувати процес заповнення стандартизованих форм та карт пацієнтів, а також полегшити обмін необхідною інформацією між різними підрозділами медичного персоналу, підвищуючи таким чином якість та швидкість надання медичної допомоги. У сфері освіти чат-боти можуть ефективно сприяти управлінню величезними масивами навчальних матеріалів, автоматизувати процес перевірки та обробки студентських робіт, звільняючи цінний час викладачів для більш важливих завдань, таких як проведення лекцій та наукові дослідження.

Навіть у складній та динамічній галузі логістики застосування чат-ботів може принести значну практичну користь, допомагаючи оптимізувати процеси обліку накладних, рахунків-фактур, транспортних документів та іншої документації, безпосередньо пов'язаної з вантажоперевезеннями, роблячи цей процес більш прозорим, ефективним та менш схильним до помилок.

Ще однією ключовою та надзвичайно цінною перевагою сучасних чат-ботів, особливо в контексті їхнього використання для управління документацією, є їхня унікальна здатність до безперервного самонавчання та гнучкої адаптації до індивідуальних потреб кожного конкретного користувача та організації в цілому. Завдяки активному використанню передових технологій штучного інтелекту та машинного навчання, ці інтелектуальні системи постійно вдосконалюють свої внутрішні алгоритми обробки інформації, ретельно аналізуючи кожний запит користувача, кожну надану відповідь та кожну отриману зворотну реакцію, що дозволяє їм поступово підвищувати точність своїх відповідей та загальну ефективність роботи з кожною новою взаємодією. Крім того, сучасні чат-боти розробляються з урахуванням можливості їхньої безперешкодної інтеграції та ефективної взаємодії з різноманітними існуючими базами даних та іншими інформаційними системами, що використовуються в організації, об'єднуючи таким чином розрізнені джерела інформації в єдину інтегровану систему та забезпечуючи комплексний та всебічний підхід до управління всією документацією організації, незалежно від її формату чи місця зберігання.

Підсумовуючи всі вищезазначені переваги та можливості, можна з повною упевненістю стверджувати, що автоматизація документообігу за допомогою інтелектуальних чат-ботів є надзвичайно ефективним та перспективним рішенням для сучасних організацій, які прагнуть досягти значного підвищення продуктивності праці своїх співробітників, істотно скоротити час, що витрачається на виконання рутинних та монотонних завдань, та забезпечити швидкий, зручний та безпечний доступ до критично важливої для їхньої діяльності інформації. Успішна інтеграція таких інноваційних інструментів у вже існуючі корпоративні інформаційні системи дозволяє досягти якісно нового рівня в управлінні

інформацією, значно мінімізувати вплив людського фактора на процеси обробки даних та в цілому підвищити рівень організованості та ефективності документопотоків у сучасних компаніях, що, безсумнівно, є ключовим фактором для забезпечення їхньої стійкої конкурентоздатності та подальшого успішного розвитку в умовах швидко мінливого бізнес-середовища.

1.2 Огляд і аналіз існуючих аналогів

Оглядаючи сучасний ринок технологічних інновацій, неможливо не відзначити вражаюче зростання популярності інтелектуальних чат-ботів, які дедалі активніше впроваджуються у різноманітні сфери бізнесу з метою оптимізації рутинних процесів, і документообіг не є винятком. Сьогодні спостерігається справжнє розмаїття таких цифрових помічників, кожен з яких прагне запропонувати користувачам найбільш ефективний та зручний спосіб управління їхньою документацією. У цьому широкому спектрі рішень особливу увагу привертають чат-боти, розроблені спеціально для корпоративного сектору. Ці інтелектуальні інструменти легко та непомітно інтегруються в уже існуючі в компаніях системи управління документами, такі як широко використовувані Microsoft SharePoint та Google Drive, популярні комунікаційні платформи на кшталт Slack, а також у більш складні та комплексні CRM (Customer Relationship Management) та ERP (Enterprise Resource Planning) системи. Їхнє основне призначення полягає у тому, щоб трансформувати традиційно трудомісткі та рутинні операції з документами на швидкі, автоматизовані та ефективні процеси, починаючи від миттєвого пошуку необхідного файлу за лічені секунди та точного контролю за всіма його версіями, закінчуючи автоматичним формуванням різноманітних звітів на основі наявних даних та значним полегшенням обміну інформацією між членами робочої команди, незалежно від їхнього розташування.

Окрему, але не менш важливу нішу на ринку займають чат-боти, в основі яких лежать передові технології штучного інтелекту (ШІ). Такі інноваційні рішення, як, наприклад, IBM Watson Assistant, демонструють вражаючу здатність

не просто оперувати документами на рівні файлів, а й глибоко аналізувати їхній внутрішній зміст, виокремлювати ключові дані, важливі факти та приховані залежності, а також оперативно надавати вичерпні та релевантні відповіді на складні запити користувачів, часто в режимі реального часу. Подібну інтелектуальну гнучкість та потужність пропонує й розробка від Google – Dialogflow, яка дозволяє створювати надзвичайно адаптивні та гнучкі системи для управління документацією, що можуть легко та безболісно інтегруватися з найрізноманітнішими платформами, сервісами та додатками, забезпечуючи таким чином комплексний підхід до роботи з інформацією.

Не менш поширеними та зручними для користувачів є чат-боти, які органічно вписалися в звичне для багатьох робоче середовище – популярні месенджери, такі як Telegram, Microsoft Teams або Slack. Їхня головна та беззаперечна перевага полягає у зручності доступу до необхідної інформації. Користувачі можуть отримувати потрібні їм документи, не покидаючи звичного інтерфейсу месенджера, в якому вони спілкуються з колегами та вирішують поточні робочі питання, що не тільки значно економить їхній дорогоцінний час, але й суттєво спрощує загальний робочий процес. Ці інтелектуальні чат-боти здатні автоматично здійснювати швидкий та ефективний пошук у великих базах даних документів, а в деяких випадках навіть відстежувати поточний статус документів у складних системах документообігу, оперативно надаючи користувачам актуальну інформацію про їхній рух та стан в режимі реального часу.

Крім універсальних рішень, які можуть бути застосовані в різних галузях, на сучасному ринку чітко простежується тенденція до появи все більшої кількості спеціалізованих чат-ботів, які розробляються з урахуванням унікальних потреб та специфіки конкретних галузей промисловості та сфер діяльності. Наприклад, у консервативній, але надзвичайно залежній від точності та оперативності обробки інформації юридичній сфері, все більшої популярності набувають інтелектуальні чат-боти, які здатні автоматизувати складний та трудомісткий процес аналізу великих обсягів контрактів та допомагати юристам у підготовці різноманітної юридичної документації, значно скорочуючи час на рутинні завдання. Яскравим

прикладом такого спеціалізованого рішення є сервіс DoNotPay. У сфері охорони здоров'я чат-боти знаходять своє застосування у веденні електронних медичних записів пацієнтів, значно полегшуючи роботу медичного персоналу та автоматизуючи важливі аспекти комунікації з пацієнтами, підвищуючи таким чином якість та ефективність надання медичних послуг.

Для тих компаній та організацій, які прагнуть отримати максимально кастомізоване та гнучке рішення, що повністю відповідатиме їхнім унікальним потребам та бізнес-процесам, на ринку існують потужні відкриті платформи для створення чат-ботів, такі як Rasa або Botpress. Ці інструменти надають розробникам широку свободу у створенні власних, повністю адаптованих рішень для автоматизації документообігу, дозволяючи їм інтегрувати ці рішення у внутрішні інформаційні системи компанії з урахуванням абсолютно всіх специфічних вимог, технічних обмежень та унікальних бізнес-процесів.

Проте, незважаючи на широке розповсюдження та вражаючу різноманітність існуючих на ринку аналогічних рішень, багато з них все ще мають певні, іноді досить суттєві, недоліки та обмеження. Одним із ключових обмежень, з яким часто стикаються користувачі, є не завжди повна та безпроблемна інтеграція з усіма типами існуючих документів, що може створювати значні незручності для тих, хто у своїй роботі змушений мати справу з різноманітними форматами файлів. Крім того, надзвичайно важливим залишається питання безпеки даних, особливо коли мова йде про роботу з конфіденційною інформацією, і, на жаль, не всі існуючі чат-боти можуть гарантувати необхідний рівень захисту від несанкціонованого доступу та витоку цінних даних. Більшість комерційних та навіть деякі відкриті рішення також потребують значних зусиль та ресурсів на етапі їхнього первинного навчання та подальшого тонкого налаштування, щоб вони могли повноцінно та ефективно адаптуватися до унікальних бізнес-процесів кожної окремої компанії, що може стати суттєвим бар'єром для їхнього впровадження.

Проведений ретельний аналіз існуючих на сучасному ринку аналогів чат-ботів для документообігу чітко демонструє, що, незважаючи на значний прогрес, досягнутий у розвитку відповідних технологій, потреба у подальшому

вдосконаленні цих інтелектуальних помічників залишається надзвичайно актуальною. Зокрема, існує значний потенціал для покращення їхньої здатності до гнучкої адаптації до різноманітних та складних бізнес-процесів, підвищення якості розпізнавання текстів, особливо при роботі зі сканованими документами, які часто мають низьку якість зображення, а також для забезпечення ще більш високого рівня безпеки при роботі з конфіденційними документами, що є критично важливим для багатьох організацій. Саме ці напрямки розвитку є ключовими для створення дійсно ефективних, універсальних та надійних чат-ботів, які зможуть повноцінно задовольнити зростаючі потреби сучасного бізнесу в оптимізації процесів документообігу та підвищенні загальної ефективності роботи з інформацією.

1.3 Постановка задачі дослідження

У висококонкурентному та динамічному бізнес-середовищі ефективне управління документацією перестало бути просто бажаною функцією, перетворившись на критично важливий фактор, що безпосередньо впливає на успіх та стійкість будь-якої організації, незалежно від її розміру чи галузевої приналежності. Як вже зазначалося раніше, переважна більшість підприємств та установ щоденно стикаються з необхідністю оперувати значними, часто неконтрольованими, обсягами різноманітних документів – від внутрішніх службових записок та звітів до комерційних пропозицій, складних договорів, технічної документації, фінансових звітів та інших офіційних записів. Цей безперервний потік інформації вимагає не лише ретельного обліку та зберігання, але й значних часових та людських ресурсів для її ефективної обробки, пошуку та використання. Традиційні, часто засновані на паперових носіях або застарілих цифрових системах, методи управління документацією в сучасних умовах все частіше виявляються неефективними та недостатньо гнучкими, що неминуче призводить до затримок у процесах прийняття важливих управлінських рішень, збільшення ймовірності допущення людських помилок при обробці даних, а також

до загального зниження продуктивності праці співробітників, які витрачають непропорційно багато часу на рутинні операції з документами.

В останні десятиліття спостерігається стрімкий розвиток технологій штучного інтелекту (ШІ), і зокрема, активне впровадження інтелектуальних чат-ботів у різноманітні сфери бізнесу з метою автоматизації рутинних та повторюваних процесів. Сфера документообігу не стала винятком, і вже сьогодні на ринку представлено ряд чат-ботів, розроблених для оптимізації роботи з документацією. Ці програмні агенти демонструють значний потенціал у спрощенні таких критично важливих завдань, як швидкий пошук необхідної інформації, автоматизований аналіз великих обсягів тексту та ефективна обробка різноманітних документів. Завдяки своїй здатності до інтеграції з існуючими системами управління документами, корпоративними порталами та іншими інформаційними платформами, чат-боти можуть надавати користувачам миттєвий та зручний доступ до необхідних даних, а також автоматизувати виконання цілого ряду рутинних операцій, пов'язаних з життєвим циклом документів.

Проте, незважаючи на очевидні переваги та зростаючу популярність, проведений детальний огляд та аналіз існуючих на ринку аналогів чітко показує, що сучасні рішення в галузі чат-ботів для документообігу все ще мають певні, іноді досить суттєві, обмеження та недоліки, які перешкоджають їхньому повноцінному та універсальному застосуванню. Зокрема, значна частина існуючих чат-ботів не забезпечує безперешкодної та повноцінної інтеграції з усіма типами електронних документів, що використовуються в організаціях (наприклад, можуть виникати проблеми з обробкою специфічних форматів файлів, застарілих документів або документів, що зберігаються в різних, несумісних системах). Крім того, питання забезпечення належного рівня захисту конфіденційних даних, що містяться в документах, залишається надзвичайно важливим та чутливим, і далеко не всі представлені на ринку чат-боти можуть гарантувати необхідний рівень безпеки від несанкціонованого доступу, витоку або втрати цінної інформації. Більшість існуючих рішень також потребують значних зусиль та витрат часу на етапі їхнього первинного навчання та подальшого тонкого налаштування для того, щоб вони

могли ефективно адаптуватися до специфічних бізнес-процесів конкретної організації, її унікальних робочих та внутрішньої термінології. Нарешті, якість автоматичного розпізнавання текстів, особливо у випадку роботи зі сканованими документами, які часто мають низьку якість зображення, різноманітні дефекти або навіть містять рукописний текст, залишається однією з проблемних зон для багатьох існуючих чат-ботів, що суттєво обмежує їхню ефективність при обробці таких документів.

З огляду на всі вищезазначені проблеми та обмеження, стає очевидною нагальна потреба у розробці та подальшому вдосконаленні чат-ботів для управління документацією, які б ефективно усували існуючі недоліки та забезпечували більш ефективну, безпечну, зручну та універсальну роботу з документацією будь-якого типу та формату. Основною метою даного наукового дослідження є розробка інноваційного програмного модуля чат-бота, який буде спрямований на оптимізацію роботи з документацією в організаціях шляхом значного покращення інтеграції з широким спектром типів документів та існуючих систем їхнього зберігання, суттєвого підвищення якості автоматичного розпізнавання текстів, особливо для складних випадків, забезпечення надійного та багаторівневого захисту конфіденційних даних, а також надання широких можливостей для гнучкого налаштування та адаптації до специфічних потреб та унікальних бізнес-процесів кожного окремого підприємства. Успішна розробка та впровадження такого програмного модуля дозволить значно підвищити загальну ефективність процесів документообігу в організаціях, суттєво зменшити часові витрати співробітників на виконання рутинних завдань, мінімізувати ризик виникнення людських помилок при роботі з документами, що в кінцевому підсумку позитивно вплине на загальну продуктивність праці та підвищить конкурентоздатність організації на сучасному ринку.

Основні завдання дослідження полягають у наступному:

1. Аналіз сучасних підходів до автоматизації документообігу.

Першим кроком є вивчення існуючих рішень для обробки документів, зокрема чат-ботів, що базуються на штучному інтелекті, таких як GPT. Важливо

проаналізувати сучасні технології, включаючи семантичний пошук, інтеграцію з API (наприклад, OpenAI) та методи обробки природної мови (NLP), щоб визначити їх переваги та недоліки.

2. Побудова модульної архітектури системи.

Наступним кроком є реалізація функцій індексації документів різних форматів (PDF, DOCX, TXT), щоб забезпечити їх подальший аналіз. Для ефективного пошуку інформації необхідно використовувати семантичні технології, такі як векторні embedding-моделі. Також важливо інтегрувати систему з OpenAI API, щоб забезпечити якісну генерацію відповідей на запити користувачів.

3. Реалізація прототипу чат-бота.

Після розробки архітектури необхідно створити функціональний прототип чат-бота, який дозволить користувачам завантажувати документи, ставити запитання та отримувати відповіді. Додатково слід реалізувати корисні функції, такі як експорт результатів у популярних форматах (TXT, PDF, DOCX) та можливість переписувати документи відповідно до інструкцій користувача.

4. Тестування та оцінка ефективності системи.

На завершальному етапі потрібно перевірити працездатність чат-бота, використовуючи різні типи документів і запитів. Важливо оцінити точність пошуку, швидкість обробки даних та зручність інтерфейсу. Також слід виявити можливі обмеження системи та запропонувати шляхи її подальшого вдосконалення.

2 АЛГОРИТМІЧНЕ ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

2.1 Розробка архітектури пропонованого рішення

Розділ, присвячений архітектурі програмного рішення, розкриває концептуальне бачення інтелектуального чат-бота, призначеного для оптимізації роботи з документацією в сучасному цифровому середовищі. Це рішення побудовано за модульним принципом, що забезпечує гнучкість, масштабованість і легкість у подальшій підтримці та розвитку системи. Архітектура включає кілька ключових компонентів, кожен з яких виконує чітко визначені функції, спрямовані на покращення ефективності обробки інформації в організаціях.

Центральним елементом архітектури є модуль користувацького інтерфейсу — саме він забезпечує взаємодію між користувачем і системою. В рамках запропонованого рішення роль цього модуля виконує Telegram-бот, що забезпечує зручний і знайомий користувачам спосіб подання запитів у природній мові та отримання відповідей у вигляді текстових повідомлень, гіперпосилань або структурованих блоків даних. Telegram виступає ідеальною платформою для такої взаємодії завдяки широкому розповсюдженню, простоті інтеграції та підтримці як текстових, так і файлових повідомлень. Чат-бот підтримує ведення діалогу з контекстом, що дозволяє користувачеві уточнювати запит, ставити додаткові питання та отримувати логічно зв’язані відповіді.

Після отримання запиту від користувача, його обробкою займається модуль природномовного аналізу, що реалізується через OpenAI API [1]. Цей модуль виконує ключову функцію інтерпретації тексту: визначає намір користувача, виявляє ключові слова й фрази, витягує сутності (такі як назви документів, імена, дати) та формує структурований запит для подальшої обробки.У цьому рішенні застосовуються сучасні мовні моделі від OpenAI, які значно перевершують традиційні інструменти за якістю розуміння мови, здатністю працювати з неоднозначними чи неповними формулюваннями, підтримкою синонімів та

гнучкістю у спілкуванні. На рисунку 2.1 зображено структуру проєкту Telegram-бота

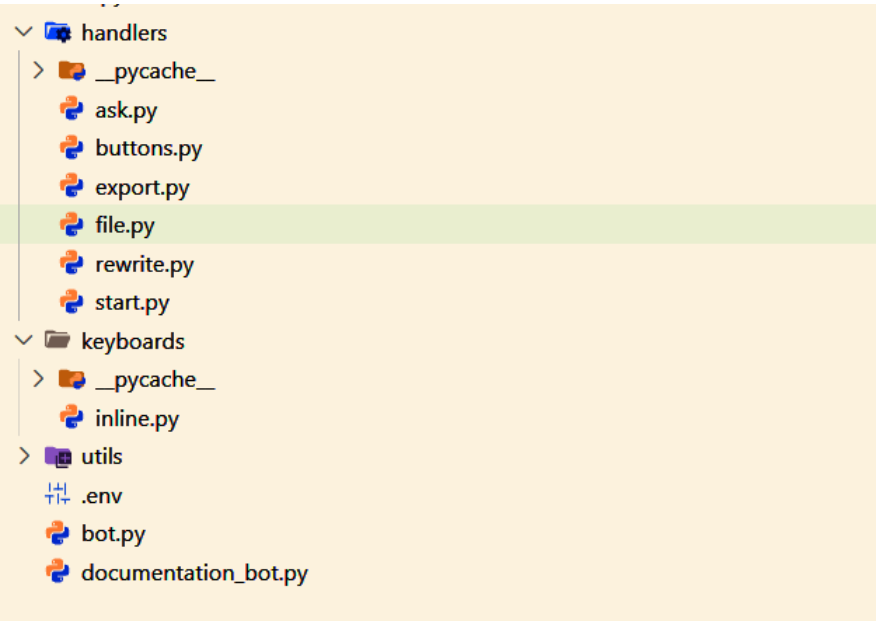

Рисунок 2.1 - Структура проєкту Telegram-бота: основні модулі обробки запитів користувача

Отриманий від NLP-модуля [10] запит передається до модуля пошуку та вилучення інформації, який відповідає за доступ до внутрішніх інформаційних джерел організації. У даному рішенні реалізовано обробку PDF, DOCX та текстових документів з використанням мовної моделі GPT. Користувач може ставити запитання до завантаженого документа, і система формує відповідь, базуючись на його повному вмісті. Модель забезпечує генерацію стислих відповідей, що є релевантними до поставленого запиту. Використання моделей embedding дозволяє реалізувати пошук — тобто знаходження інформації навіть у випадках, коли запит не містить точних формулювань.

Далі за формування відповіді користувачеві відповідає модуль генерації. Його завдання — побудова зрозумілого тексту, який подає знайдену інформацію у стислій, чіткій та коректній формі. Модуль може формувати як короткі відповіді у вигляді окремих речень або абзаців, так і деталізовані зведення змісту документа, витяги з нього чи списки ключових даних. Якщо в процесі обробки запиту виникає

потреба в уточненні, система автоматично ініціює додаткові питання, підтримуючи логіку діалогу з користувачем у Telegram. Взаємодія GPT-моделі з завантаженим документом представлені на рисунку 2.2.

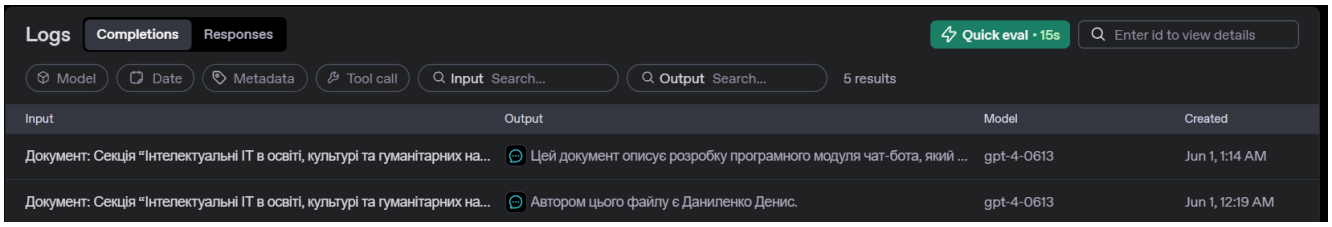

Рисунок 2.2 - Взаємодія GPT-моделі з завантаженим документом

Архітектура проєктованого рішення є модульною, що дозволяє гнучко підходити до реалізації окремих її компонентів, змінювати технології або розширювати функціональність без порушення загальної структури. Використання Telegram як платформи для взаємодії з користувачами, а також OpenAI API [1] як основного інструменту для інтерпретації природної мови, відкриває широкі можливості для реалізації сучасного, ефективного та масштабованого чат-бота, здатного істотно полегшити та пришвидшити роботу з документацією в умовах цифрової трансформації організацій.

2.2 Апаратна архітектура програмного модуля

Для забезпечення стабільної роботи Telegram-бота використовується проста, проте достатньо ефективна апаратна архітектура, яка поєднує персональний комп'ютер (локальний сервер розробника), хмарні API-сервіси та клієнтські пристрої користувачів. Така архітектура дозволяє обробляти документи, надсилати запити до OpenAI, а також підтримувати зв'язок із користувачем через Telegram.

Основна логіка роботи чат-бота виконується на локальному сервері (комп'ютері розробника), де реалізовано обробку PDF та DOCX документів. Чат-бот аналізує текст за допомогою повнотекстового пошуку, виконує пошук за ключовими словами та забезпечує базову інтеграцію з OpenAI API. Генерація embedding-векторів чи складна тематична класифікація документів на даному етапі

не реалізована — відповіді формуються на основі індексованого змісту документів і передаються у вигляді простих текстових повідомлень.

Чат-бот працює через Telegram API, використовуючи webhook для прийому повідомлень. Запити користувачів передаються через Telegram Server на локальний сервер, де чат-бот обробляє їх, формує відповідь і повертає її назад. При цьому Telegram забезпечує базову авторизацію користувачів за їх унікальним user_id, а сам чат-бот може обмежувати доступ лише для дозволених користувачів. Додаткові механізми автентифікації (OAuth, токени, інтеграція з іншими системами) поки не використовуються.

Користувачі можуть взаємодіяти з чат-ботом через будь-який пристрій, на якому встановлений Telegram — смартфон, планшет або комп'ютер. Чат-бот підтримує обробку текстових повідомлень. Архітектурна схема чат-бота представлені на рисунку 2.3.

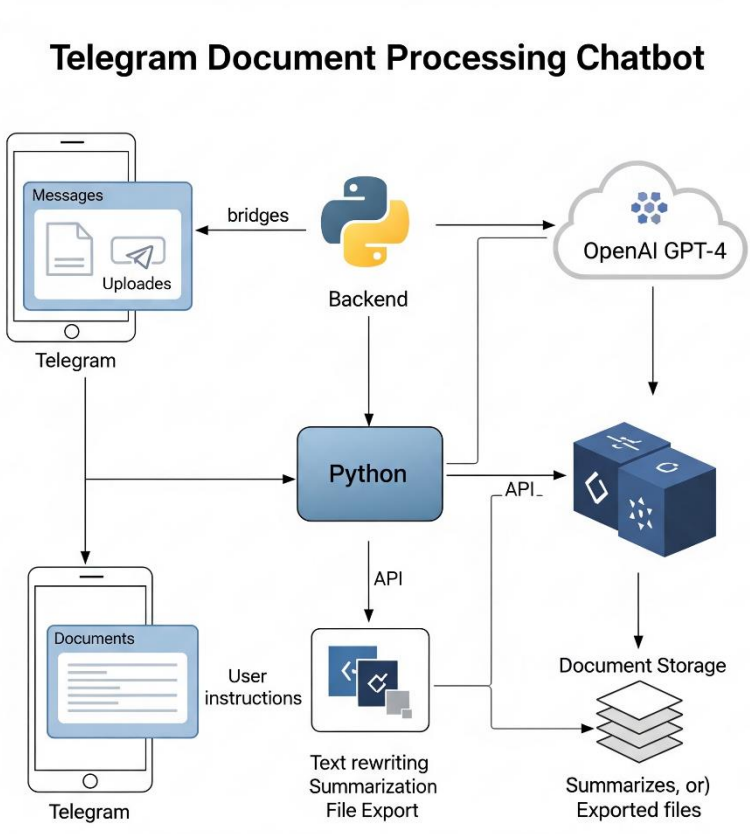

Рисунок 2.3 – Архітектурна схема чат-бота для обробки документів

Таким чином, апаратна архітектура цього рішення є мінімалістичною, але придатною для цілей демонстрації: вона дозволяє працювати з документами, відповідати на запити через OpenAI та забезпечувати зручний доступ до системи через Telegram.

2.3 Програмне забезпечення та алгоритми

У процесі розробки чат-бота для оптимізації роботи з документацією ключовою метою було створення інтелектуального програмного модуля, що забезпечує зручну, швидку та якісну взаємодію користувача з документальними ресурсами. Вибір технологій, підходів та архітектурних рішень здійснювався з урахуванням сучасних вимог до систем автоматизованої обробки природної мови, простоти розгортання, а також ефективності масштабування. Схема алгоритму роботи чат-бота представлені на рисунку 2.4.

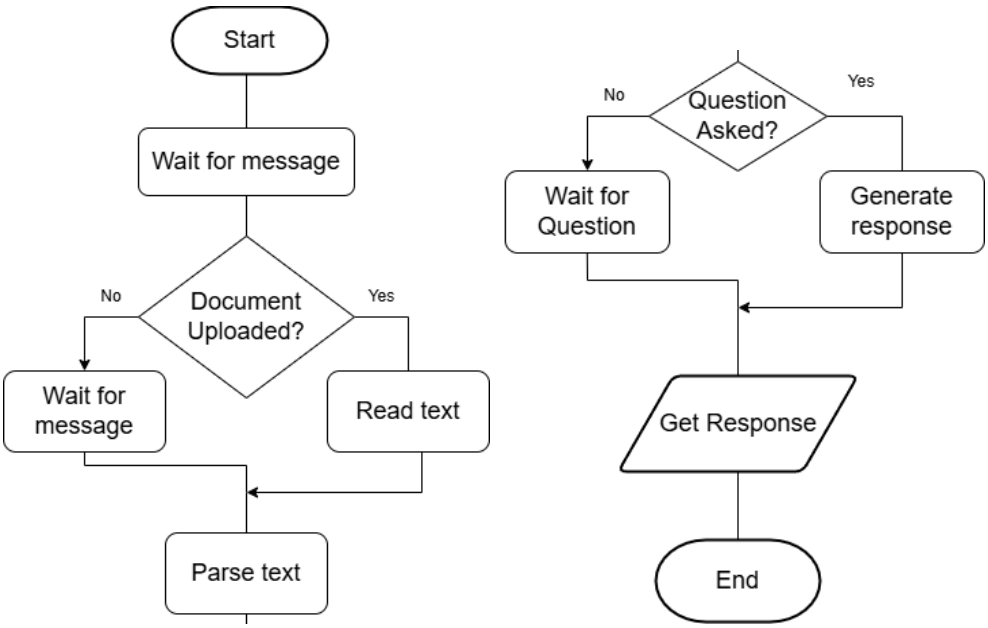

Рисунок 2.4 - Схема алгоритму роботи чат-бота

Основною мовою реалізації програмного забезпечення обрано Python, що є одним із найпопулярніших інструментів у сфері розробки інтелектуальних систем та обробки даних. Гнучкість синтаксису, наявність великої кількості бібліотек і

фреймворків, активна спільнота та підтримка інтеграції з API сторонніх сервісів дозволили ефективно реалізувати складні функції з мінімальними витратами часу. Python також надає можливість кросплатформеного розгортання, що робить систему універсальною та адаптованою до різних операційних середовищ.

У якості користувацького інтерфейсу було реалізовано Telegram-бота, що забезпечує інтерактивну взаємодію з користувачем у популярному месенджері Telegram. Такий підхід дозволив зробити систему максимально доступною та зручною для кінцевого користувача, оскільки Telegram є одним із найпоширеніших каналів спілкування. Для взаємодії з платформою Telegram використано Telegram Bot API [2], який надає всі необхідні інструменти для обробки повідомлень, кнопок, команд та інших елементів інтерфейсу. Це дозволило реалізувати логіку приймання запитів, надсилання відповідей, прикріплення документів і керування діалогом у межах уже знайомого користувачеві середовища. Візуальне представлення логічних зв’язків представлені на рисунку 2.5.

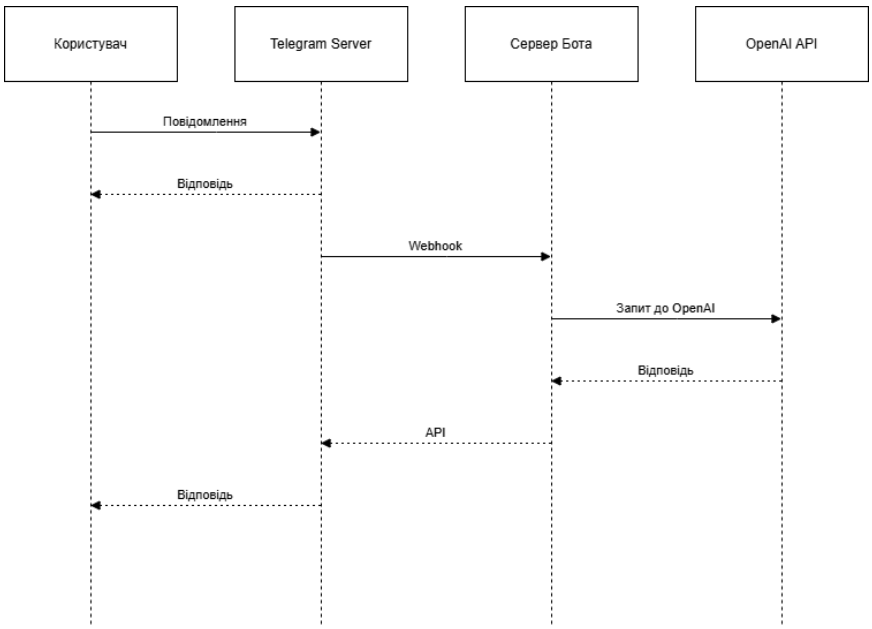

Рисунок 2.5 – Візуальне представлення логічних зв’язків між основними компонентами системи

Ключовим елементом програмного модуля є обробка природної мови, яка реалізована за допомогою OpenAI API [1]. На відміну від класичних бібліотек для обробки тексту рішення, засноване на GPT-моделях, дозволяє досягти високої

точності при мінімальних зусиллях з боку розробника. Використання OpenAI API [1] забезпечує здатність системи інтерпретувати широке коло запитів користувача — від простих запитів на пошук до складних команд із редагування документів. Модель здатна розпізнавати намір користувача, витягувати релевантну інформацію з документів, а також враховувати контекст діалогу, що суттєво покращує природність взаємодії.

GPT-модель використовується для генерації відповідей на запити користувача на основі вмісту документів. GPT забезпечує базову гнучкість у формуванні відповідей завдяки своїй мовній моделі.

Модуль генерації відповіді базується на GPT-моделі, яка формує тексти природною мовою на основі знайденого фрагмента документа. Система генерує динамічні відповіді, що враховують зміст запиту та відповідну частину тексту, проте не використовує збереження діалогу чи шаблонізацію з динамічними змінними (наприклад, дати чи імена). Такий підхід забезпечує базовий рівень "діалоговості" та зручність взаємодії для користувача.

Таким чином, застосування OpenAI API [1] у поєднанні з Telegram Bot API [2] дозволило створити ефективний, інтуїтивно зрозумілий і доступний програмний модуль, який забезпечує інтерактивну роботу з документацією у звичному для користувача середовищі — Telegram.

2.4 Розробка моделі функціоналу чат-бота

Під час створення Telegram-бота основна увага приділялася тому, щоб зробити інструмент максимально зручним для користувача та водночас достатньо «розумним» для роботи з великими текстовими документами. Планувалося реалізувати систему, яка дозволяє автоматично отримувати змістовні відповіді на запитання користувача, без потреби самостійно переглядати весь файл.

Користувач надсилає файл у чат — це може бути PDF, DOCX або текстовий документ. Чат-бот автоматично зчитує вміст і готує його до подальшої обробки.

Уся логіка реалізована так, щоб текст розбивався на логічно завершені частини, з якими можна зручно працювати.

Коли користувач вводить запит, чат-бот не просто шукає ключові слова. Він знаходить відповідні фрагменти тексту за змістом і передає їх мовній моделі (GPT), яка формує відповідь. Це дозволяє отримувати не просто витяги з документа, а змістовні, логічні й стислі відповіді.

Для кращої ефективності система використовує методи семантичного пошуку — текст представлений у вигляді векторів, що дозволяє швидко знаходити потрібну інформацію.

Також передбачається реалізація додаткових функцій, як-от експорт результатів у зручному форматі. Діаграма, яка показує основні модулі чат-бота та їхню взаємодію на рисунок 2.6.

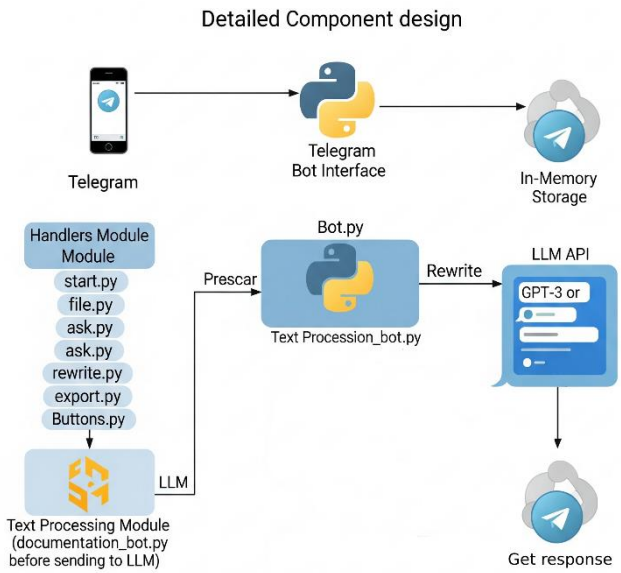

Рисунок 2.6 – Діаграма компонентів модульного чат-бота

Уся система побудована на мові програмування Python з використанням сучасних бібліотек для обробки тексту, створення Telegram-ботів, векторного пошуку та інтеграції з GPT-моделлю. Такий підхід дозволяє зробити рішення не лише технічно ефективним, а й простим у використанні для кінцевого користувача.

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Реалізація програмного забезпечення

У процесі створення чат-бота для платформи Telegram було поставлено завдання реалізувати багатофункціональну систему, здатну ефективно працювати з текстовими документами та надавати користувачам швидкі й релевантні відповіді на природномовні запити. В основі архітектури лежить концепція модульного проєктування: кожен етап роботи бота виконується окремим логічним блоком, що значно полегшує підтримку, розширення та тестування функціоналу.

Однією з головних переваг розробленого рішення є його доступність: користувачу не потрібно встановлювати додаткове програмне забезпечення чи виконувати складні налаштування — достатньо лише мати Telegram. Це робить систему максимально зручною для широкого кола користувачів, у тому числі тих, хто не має технічної підготовки.

Першим кроком взаємодії є передача документів у чат. Користувач надсилає файл у чат-бот, після чого система автоматично визначає його формат та здійснює попередню обробку. Підтримуються найбільш поширені типи документів — PDF, DOCX та TXT, що робить систему універсальною та придатною для використання в різних організаціях. Це все відбувається в модулі file.py, який реалізує окремий функціональний компонент, який відповідає за прийом, обробку та збереження документів, що надсилаються користувачем до Telegram-бота.

Цей модуль виконує ключову роль у забезпеченні доступу до вмісту документів для подальшої аналітики. Мета модуля — зберегти отриманий файл на сервері, обробити його залежно від формату (PDF, DOCX або TXT) та вилучити з нього текстову інформацію. Таким чином, модуль file.py виступає в ролі "текстового екстрактора" — він перетворює фізичний документ на структурований текст, який можна аналізувати. Головна функція handle_file(update, context) викликається, коли користувач надсилає документ у Telegram-бот.

Данна Функція:

- Завантажує файл локально.
- Зберігає його у середину RAM пам’яті чат-бота.
- Визначає тип файлуза розширенням.

Залежно від формату, застосовується відповідний обробник:

- PyMuPDF (через fitz) — для PDF-файлів.
- python-docx — для DOCX-файлів.
- Пряме зчитування — для TXT-файлів.

Витягнутий текст повертається для подальшої індексації або обробки GPT-моделлю (наприклад, у модулі documentation_bot.py). Функція handle_file представлені на рисунку 3.1.

```
async def handle_file(update: Update, context: ContextTypes.DEFAULT_TYPE):
```

Рисунок 3.1 – функція handle_file

Модуль documentation_bot.py відповідає за створення автономного компоненту чат-бота, який здійснює обробку тексту та генерує відповіді на запити користувача. Основна логіка реалізована у вигляді класу DocumentationChatBot можна побачити на рисунку 3.2.

```
def from_text(cls, raw_text: str):
    text_splitter = CharacterTextSplitter(chunk_size=1000, chunk_overlap=100)
    docs = text_splitter.create_documents([raw_text])
    embeddings = OpenAIEmbeddings(openai_api_key=os.getenv("OPENAI_API_KEY"))
    vectorstore = FAISS.from_documents(docs, embeddings)
    retriever = vectorstore.as_retriever()
    llm = OpenAI(temperature=0, openai_api_key=os.getenv("OPENAI_API_KEY"))
    qa_chain = RetrievalQA.from_chain_type(llm=llm, retriever=retriever)
    return cls(qa_chain)
```

Рисунок 3.2 - Фрагмент реалізації пошуку у чат-боті

Після ініціалізації, чат-бот дозволяє ставити запитання природною мовою до певного тексту (наприклад, до контракту, інструкції чи довільного документа) — і отримувати на них узагальнені, логічні відповіді. Модуль побудований на базі

бібліотеки LangChain, що дозволяє ефективно інтегрувати моделі OpenAI для обробки запитів.

За допомогою бібліотеки dotenv, чат-бот автоматично зчитує OpenAI API ключ із .env файлу. Це забезпечує безпечне зберігання конфіденційних даних. Для роботи з великими текстами використовується CharacterTextSplitter, який розбиває текст на частини (чанки) для подальшої індексації. Кожен текстовий фрагмент перетворюється у вектор за допомогою OpenAIEmbeddings. Це дозволяє знаходити релевантні частини документа на основі змісту, а не лише ключових слів. Із використанням FAISS створюється індекс для швидкого пошуку релевантних фрагментів. Модель GPT від OpenAI підключається через RetrievalQA, що дозволяє передавати знайдені фрагменти тексту та формувати відповіді на запитання. Цей метод є зовнішнім інтерфейсом для звернення до чат-бота. Метод Ask Користувач задає запитання, а чат-бот повертає сформовану відповідь на основі аналізу тексту. Можна побачити частину коду методу на рисунку 3.3.

```
def ask(self, question: str) -> str:
    return self.qa_chain.run(question)
```

Рисунок 3.3 – Фрагмент методу ask

Файл .env є важливою частиною безпечної конфігурації проєкту. У ньому зберігаються чутливі дані, які не повинні бути розміщені безпосередньо в коді. Це дозволяє розділяти логіку програми та її налаштування, а також забезпечити більшу безпеку під час розгортання на сервері або публікації коду.

У цьому файлі зазвичай містяться:

- BOT_TOKEN — токен Telegram-бота, отриманий через BotFather, що дозволяє взаємодіяти з Telegram API.
- OPENAI_API_KEY — ключ доступу до GPT-моделі від OpenAI для генерації відповідей.

Завдяки використанню .env всі ці параметри підвантажуються в додаток динамічно, через бібліотеку dotenv, без необхідності жорстко прописувати їх у коді.

Файл ask.py у структурі чат-бота відповідає за обробку текстових запитів користувача після завантаження документа. Це ключовий модуль, який поєднує раніше збережений текст документа з мовною моделлю GPT, щоб сформувати змістовну відповідь.

Модуль ask.py виконує важливу роль у загальній логіці роботи чат-бота — він відповідає за обробку запитів, які користувач надсилає після завантаження документа. Основне його завдання полягає у тому, щоб забезпечити зв'язок між текстовими даними з документа та генерацією осмисленої відповіді за допомогою штучного інтелекту.

Після того як користувач надіслав файл, чат-бот зберігає його вміст у пам'яті сесії (через context.user_data). Коли ж користувач вводить свій запит у чаті, модуль ask.py спершу перевіряє, чи є у системі збережений текст документа. Якщо файл не був попередньо оброблений, чат-бот повідомляє про необхідність спершу завантажити документ.

У випадку, коли текст документа доступний, запускається логіка компонента DocumentationChatBot, який реалізовано як окремий клас. Цей компонент виконує кілька важливих етапів обробки: розбиває документ на менші текстові блоки (фрагменти), формує з них векторне представлення (embedding), а також створює спеціальну структуру для швидкого пошуку релевантних частин тексту. Саме на цьому етапі відбувається векторна індексація, яка дозволяє чат-боту працювати не з усім текстом одночасно, а лише з тими його частинами, які відповідають змісту запиту. Генерація відповіді на запит користувача представлені на рисунку 3.4.

```
answer = bot.ask(user_input)
context.user_data["last_answer"] = answer
await update.message.reply_text(answer, reply_markup=build_keyboard())
```

Рисунок 3.4 – Генерація відповіді на запит користувача

Після цього обрані фрагменти передаються до мовної моделі GPT (від OpenAI), яка генерує відповідь, спираючись як на зміст документа, так і на запит користувача. Завдяки використанню архітектури RetrievalQA, відповідь є логічно

узгодженою та контекстно обґрунтованою. Нарешті, сформована відповідь надсилається користувачу у вигляді звичайного повідомлення в чаті Telegram.

Таким чином, ask.py фактично є посередником між користувачем і штучним інтелектом, який поєднує вже збережений текст документа з мовною моделлю, щоб надати користувачеві максимально точну і корисну інформацію.

Модуль rewrite.py реалізує функціонал переписування тексту, що був отриманий із завантаженого користувачем документа. Основне призначення цього модуля — трансформувати вміст документа відповідно до інструкцій, які задає користувач у Telegram-чаті.

Після того як документ було успішно зчитано та збережено у пам’яті сесії (context.user_data), користувач натискає кнопку і чат-бот переходить в режим очікування текстової інструкції. Це може бути, наприклад: «Скороти до основного», «Зроби формальним» або «Перепиши для молодшої аудиторії».
Коли користувач надсилає інструкцію, функція rewrite_document() формує запит до мовної моделі GPT-4 (через API OpenAI), в якому вказує:

- Повний текст документа.
- Інструкцію, отриману від користувача.
- Побажання переписати документ відповідно до цієї інструкції.

В результаті чат-бот:

– Отримує від GPT новий переписаний варіант документа.

– Надсилає користувачу результат у вигляді повідомлення.

– Додатково генерує оновлений .docx файл, який прикріплюється до повідомлення.

Основна логіка переписування документа представлені на рисунку 3.5.

```
prompt = (
    f"Ось документ:\n{text}\n\n"
    f"Інструкція: {user_input}\n"
    f"Будь ласка, перепиши документ відповідно до інструкції."
)

response = client.chat.completions.create(
    model="gpt-4",
    messages=[{"role": "user", "content": prompt}],
    temperature=0.5
)
```

Рисунок 3.5 - Основна логіка переписування документа

Це дозволяє користувачу одразу отримати не лише переглянуту відповідь, а й готовий документ для подальшого використання.

Модуль inline.py створює інтерактивні кнопки, які з'являються під повідомленнями чат-бота в Telegram. Вони дозволяють користувачу швидко обирати дію без введення тексту вручну.

Модуль export.py відповідає за створення DOCX-файлу на основі останньої відповіді чат-бота та надсилання його користувачу через Telegram.

Основна логіка:

– Бере останню відповідь, збережену в context.user_data["last_answer"].
– Створює новий Word-документ (формат .docx) і вставляє туди текст.
– Генерує файл у пам'яті (BytesIO) без запису на диск.
– Надсилає файл у чат як документ.

Фрагмент модуля export.py представлені на рисунку 3.6.

```
document = Document()
document.add_paragraph(text)
buffer = BytesIO()
document.save(buffer)
buffer.seek(0)

await query.message.reply_document(
    document=buffer,
    filename="last_answer.docx",
    caption="📄 Відповідь у форматі DOCX"
)
```

Рисунок 3.6 – Фрагмент модуля export.py

3.2 Інтерфейс користувача

Інтерфейс користувача розробленого програмного модуля є одним із ключових компонентів системи, адже саме через нього здійснюється безпосередня взаємодія користувача з функціональністю програмного забезпечення. У межах даного проєкту інтерфейс реалізовано у вигляді чат-бота, інтегрованого в

середовище Telegram, що виступає як основний засіб комунікації між користувачем і системою.

Вибір Telegram обумовлений низкою важливих переваг:

- висока поширеність та знайомість середовища серед користувачів, що знижує поріг входу;
- зручність у використанні на різних платформах (смартфон, ПК, планшет);
- відсутність необхідності встановлення окремого програмного забезпечення;
- гнучкість реалізації інтерфейсу за допомогою текстових команд, кнопок, меню, інлайн-відповідей тощо;
- підтримка мультимедійних даних, включно з файлами, зображеннями, документами.

Основним принципом, покладеним в основу інтерфейсу, є "мінімізація зусиль користувача": система повинна дозволити досягати бажаного результату — наприклад, отримати відповідь з документа, експортувати її або замінити вхідний файл — за допомогою мінімальної кількості простих, інтуїтивно зрозумілих дій.

Для покращення зручності взаємодії реалізовано контекстно-залежну поведінку чат-бота. Зокрема:

- після завантаження документа бот автоматично інформує користувача про успішну обробку та пропонує ввести запитання;
- якщо запитання надійшло до завантаження файлу, бот коректно повідомляє про необхідність спочатку надати документ;
- у випадку помилки (наприклад, неправильного формату файлу) система надає зрозуміле пояснення та список підтримуваних форматів.

Кожна дія користувача супроводжується відповідною реакцією системи у вигляді повідомлень-підказок, що полегшує навігацію та знижує ймовірність помилок або непорозумінь. Таким чином, взаємодія з ботом є інтуїтивною, передбачуваною та адаптивною до дій користувача.

Графічне представлення інтерфейсу та сценаріїв взаємодії користувача з чат-ботом наведено на рисунку 3.7, де демонструються ключові етапи обробки файлу, введення запиту та отримання відповіді.

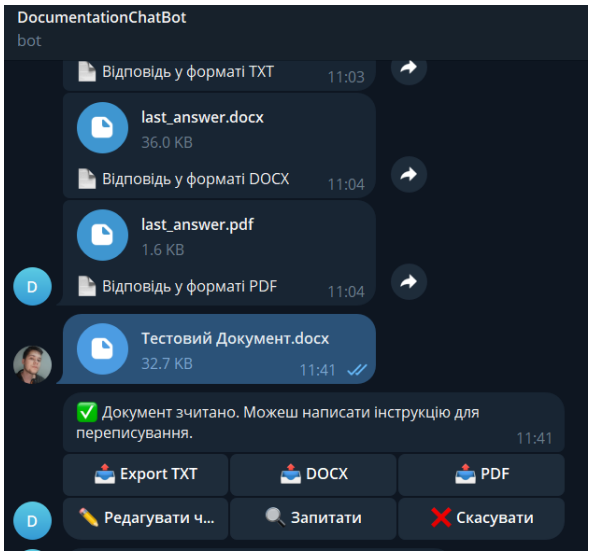

Рисунок 3.7 – Інтерфейс користувача

Інтерфейс чат-бота реалізовано з урахуванням кількох основних елементів, що забезпечують його функціональність та зручність у використанні:

– Текстові повідомлення: використовуються для інформування користувача про статус операцій, результат запиту або помилку. Вони короткі, лаконічні, зрозумілі. Наприклад: «Файл успішно завантажено. Тепер ви можете поставити питання».

– Інлайн-кнопки (inline keyboard): застосовуються після обробки запиту для вибору формату експорту відповіді (TXT, DOCX, PDF). Це забезпечує швидку взаємодію без потреби вводити додаткові команди вручну.

– Обробка файлів: користувач може надсилати документи у форматах .txt, .docx, .pdf. Чат-бот автоматично ідентифікує формат, зчитує вміст, сегментує його для подальшої індексації та використовує для відповіді на питання.

– Командна навігація: хоча інтерфейс переважно подієвий (реактивний), реалізовано також базові команди: /start — запуск взаємодії, /help — коротка інструкція користувача.

Користувацький інтерфейс відповідає принципам доступності та зручності:

- Всі дії зведено до простих кроків: надіслати файл, поставити запитання, обрати формат відповіді.
- Не вимагається знання специфічних команд.
- Користувач отримує своєчасні підказки і повідомлення про помилки.

Також інтерфейс є повністю україномовним, що важливо для цільової аудиторії, та забезпечує адаптацію до потреб як новачків, так і досвідчених користувачів.

– Користувач надсилає файл з текстом документа (наприклад, дипломної роботи у форматі PDF).

– Чат-бот відповідає: «Файл отримано. Ви можете ставити питання щодо його вмісту».

– Користувач запитує: «Яка мета дослідження, описана в документі?».

– Чат-бот аналізує індексований вміст та повертає коротку, релевантну відповідь.

– Користувач натискає бажаний варіант і отримує файл.

Telegram-бот повністю працює у мобільному форматі, що забезпечує високу мобільність користувача. Користувач може працювати з документами і ставити запити буквально «на ходу», не маючи при собі комп’ютера чи спеціалізованого програмного забезпечення. Крім того, завдяки простому API Telegram інтерфейс легко масштабувати або адаптувати для підтримки додаткових мов, інтеграції із зовнішніми сервісами чи базами даних.

3.3 Тестування розробленої програми

Впровадження будь-якого нового програмного рішення в робочі процеси організації вимагає абсолютної впевненості у його надійності та відповідності поставленим завданням. Для розробленого програмного модуля чат-бота, призначеного для значного спрощення та оптимізації роботи з документацією через звичний інтерфейс месенджера Telegram, етап тестування став не просто

формальною процедурою, а критично важливим кроком на шляху до його потенційного успішного впровадження.

Цей всебічний процес дозволив вийти за рамки простого підтвердження наявності заявленого функціоналу, надавши можливість глибоко оцінити загальну стабільність системи, її здатність витримувати різноманітні навантаження, а також, що не менш важливо, оцінити зручність та інтуїтивність її використання з точки зору кінцевого користувача в умовах, що максимально відтворюють реальну повсякденну роботу. На рисунку 3.8 зображено, як експортуються відповіді чат-бота в текстові документи.

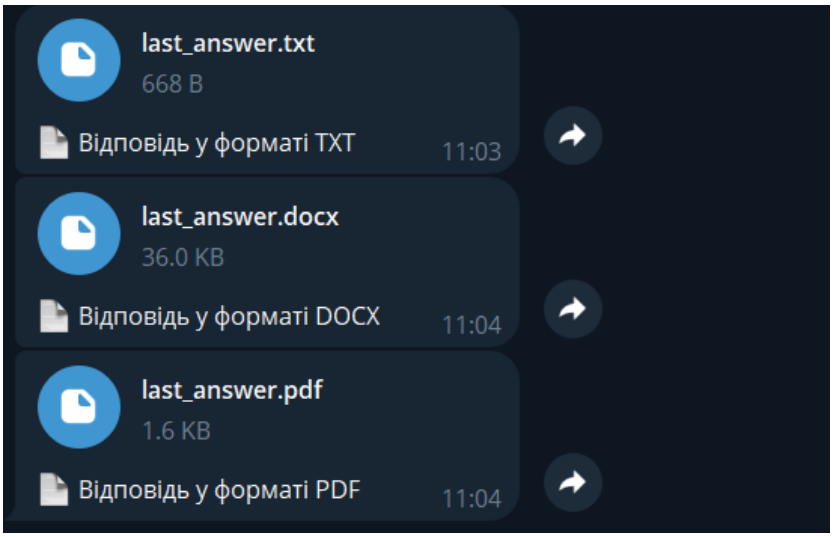

Рисунок 3.8 – Експорт відповіді бота

Цілі, які ставилися перед командою тестування, були багатогранними та охоплювали різні аспекти функціонування чат-бота. Першочерговим завданням було надзвичайно ретельне вивчення правильності виконання його базових функцій. Це виходило далеко за межі простої перевірки можливості успішно завантажити документ у одному з підтримуваних форматів – будь то простий текстовий файл (.txt), документ Word (.docx) чи поширений формат PDF (.pdf). На рисунку 3.9 зображено, як чат-бот приймає документ.

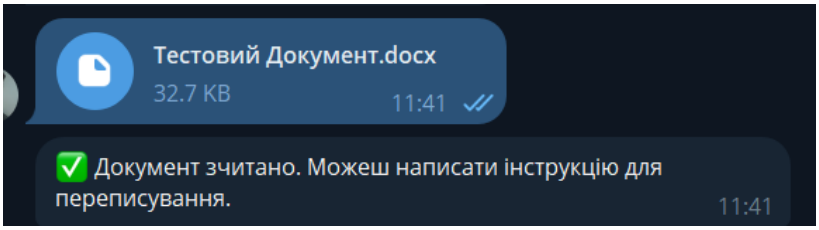

Рисунок 3.9 - Надсилання документа в чат-бот Telegram

Другою, не менш важливою метою тестування була перевірка коректної та злагодженої взаємодії між усіма компонентами системи. Оскільки розроблений чат-бот не є монолітним додатком, а побудований за модульною архітектурою, особливу увагу було приділено тестуванню взаємодії окремих частин. До складу системи входять окремі модулі, які відповідають за завантаження та обробку документів, індексацію тексту, аналіз природної мови (NLP), а також модуль, що забезпечує зв’язок з Telegram API. Враховуючи їхню взаємозалежність, життєво важливо було переконатися у стабільності, надійності та узгодженості роботи кожного з них у процесі повного циклу обробки запиту.

У процесі тестування перевірялася не лише функціональність окремих модулів, але й правильність та швидкість передачі даних між ними, обробка граничних ситуацій, а також відстеження і коректна обробка помилок, що можуть виникнути на етапах взаємодії. Особливу увагу було приділено питанням узгодженості структури даних, що передаються між модулями, відповідності форматів, збереженню логіки діалогу з користувачем та коректному реагуванню системи на всі типи вхідних дій.

Одним із головних завдань цього етапу було виявлення як явних помилок (збоїв, відмов, помилок обробки), так і менш очевидної або нетипової поведінки системи в неочікуваних сценаріях. Це включало, наприклад, ситуації, коли користувач надсилає пошкоджений файл, вводить запит без попереднього завантаження документа, або ставить надзвичайно загальне або некоректне запитання. Тестування дозволило також виявити низку випадків неоптимального використання ресурсів або надмірної затримки при обробці великих документів, що сприяло подальшому вдосконаленню алгоритмів.

Окремим завданням було виявлення логічних прогалин у функціональності, які не завжди призводять до помилок, але можуть викликати некоректну поведінку системи в окремих умовах. Це, наприклад, випадки, коли бот не повідомляє про відсутність результатів або не підказує користувачеві наступні кроки. Такі аспекти є критично важливими з точки зору користувацького досвіду, оскільки впливають на сприйняття системи як цілісної та надійної.

На рисунку 3.10 проілюстровано один із типових сценаріїв взаємодії користувача з ботом, де показано процес формування запиту, надання відповіді, а також можливість подальшого експорту результатів у зручному для користувача форматі. Такий підхід до візуалізації допомагає краще зрозуміти логіку роботи чат-бота в реальних умовах і демонструє, як саме функціональні модулі системи взаємодіють між собою у відповідь на дії користувача.

Рисунок 3.10 - Формулювання запиту та відповідь GPT

Особливу увагу в процесі тестування було приділено перевірці обробки некоректного вводу та поведінки системи у нештатних ситуаціях. У реальному світі користувачі не завжди діють за інструкцією, вони можуть вводити непередбачувані дані, використовувати нестандартні формулювання запитів або опинятися в нестандартних ситуаціях – наприклад, спроба завантажити пошкоджений або

некоректно сформований файл, або ж ситуація, коли велика кількість користувачів одночасно надсилає велику кількість запитів. Тестування мало показати, наскільки стійкою та "живучою" є система до таких нетипових сценаріїв, чи обробляє помилки, що при цьому виникають, і чи надає користувачеві зрозумілі та інформативні повідомлення про проблему, а не просто "падає". Це забезпечує надійність функціонування модуля та позитивний досвід взаємодії для кінцевого користувача навіть у складних умовах. На рисунку 3.11 зображено лог запиту до модульного чат-бота."

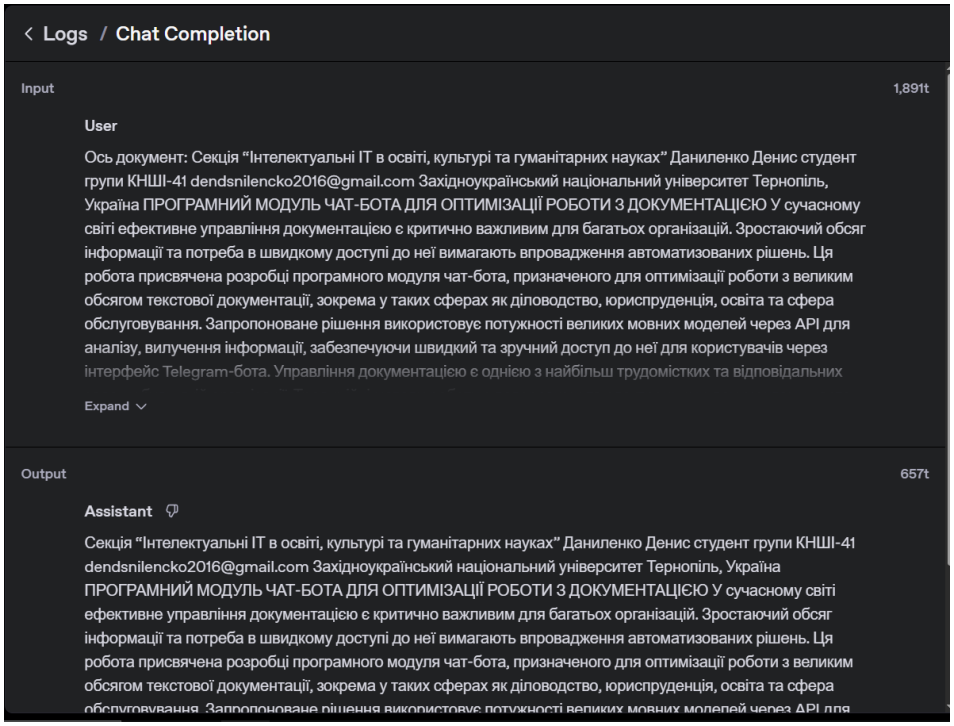

Рисунок 3.11 – Лог запиту

Нарешті, однією з ключових цілей тестування була оцінка швидкодії системи за різного обсягу вхідних даних і навантаження. Оскільки ефективність чат-бота напряму залежить від швидкості обробки документів та генерації відповідей, було проведено перевірку залежності часу відгуку від розміру й складності файлів, а також від кількості одночасних користувачів. Це дозволило об’єктивно оцінити масштабованість системи та її готовність до роботи в умовах реального навантаження.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було розроблено програмний модуль Telegram-бота, призначений для полегшення роботи з текстовими документами. Основною метою системи є забезпечення зручного доступу до вмісту документів за допомогою інтегрованого в месенджер Telegram інтерфейсу, що відповідає загальній меті дослідження.

1. Під час дослідження було комплексно проаналізовано предметну область, сформульовано вимоги до функціональності чат-бота та обґрунтовано вибір відповідних інструментів розробки.

2. У рамках проєктування було розроблено архітектуру чат-бота. Ця архітектура дозволяє ефективно обробляти вхідні документи, включаючи PDF, DOCX та текстові файли, та використовувати можливості моделі GPT для формування змістовних відповідей та переписування тексту.

3. Результатом реалізації став функціональний чат-бот, який здатен:

- приймати текстові, PDF і DOCX документи від користувача;
- витягувати текст з цих документів;
- інтерпретувати запити користувача, знаходити відповідні фрагменти за змістом за допомогою методів семантичного пошуку та надавати на них текстові відповіді, сформовані на основі вмісту документа з використанням GPT-моделі;
- працювати у середовищі Telegram без потреби в окремому застосунку;
- надавати додаткові функції, такі як експорт оброблених результатів у зручному форматі (TXT, DOCX, PDF).

4. Проведене тестування засвідчило, що створений чат-бот стабільно виконує основні завдання. Завдяки простому та інтуїтивно зрозумілому інтерфейсу, користувачі можуть взаємодіяти із системою у знайомому середовищі месенджера, що значно знижує бар’єр входу та оптимізує їхню взаємодію з системою.

Таким чином, розроблений програмний модуль є гнучким та адаптивним рішенням для автоматизації роботи з документацією, що підтверджує практичну значущість роботи та сприяє підвищенню ефективності управління інформацією.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. OpenAI. OpenAI API Documentation [Електронний ресурс]. URL: https://platform.openai.com/docs (дата звернення: 17.05.2025).
2. Telegram. Telegram Bot API [Електронний ресурс]. URL: https://core.telegram.org/bots/api (дата звернення: 17.05.2025).
3. LangChain. Documentation [Електронний ресурс]. URL: https://docs.langchain.com (дата звернення: 17.05.2025).
4. FAISS. Facebook AI Similarity Search [Електронний ресурс]. URL: https://github.com/facebookresearch/faiss (дата звернення: 17.05.2025).
5. PyMuPDF. Documentation [Електронний ресурс].URL: https://pymupdf.readthedocs.io (дата звернення: 17.05.2025)
6. Python-docx. Documentation [Електронний ресурс]. URL: https://python-docx.readthedocs.io (дата звернення: 17.05.2025).
7. Python Software Foundation. Python Language Reference [Електронний ресурс]. URL: https://docs.python.org/3/ (дата звернення: 17.05.2025).
8. Rasa. Open Source Conversational AI [Електронний ресурс]. URL: https://rasa.com (дата звернення: 17.05.2025).
9. Botpress. Conversational AI Platform [Електронний ресурс]. URL: https://botpress.com (дата звернення: 17.05.2025).
10. IBM. Watson Assistant [Електронний ресурс]. URL: https://www.ibm.com/cloud/watson-assistant (дата звернення: 17.05.2025).
11. DoNotPay – The World's First Robot Lawyer [Електронний ресурс]. URL: https://donotpay.com (дата звернення: 17.05.2025).
12. Vaswani A., Shazeer N., Parmar N. та ін. Attention Is All You Need. arXiv:1706.03762 [Електронний ресурс]. URL: https://arxiv.org/abs/1706.03762 (дата звернення: 17.05.2025).
13. Brown T., Mann B., Ryder N. та ін. Language Models are Few-Shot Learners. arXiv:2005.14165 [Електронний ресурс]. URL: https://arxiv.org/abs/2005.14165 (дата звернення: 17.05.2025).

14. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers. arXiv:1810.04805 [Електронний ресурс]. URL: https://arxiv.org/abs/1810.04805 (дата звернення: 17.05.2025).

15. Jurafsky D., Martin J.H. Speech and Language Processing. 3rd ed. Pearson, 2020. 1024 p.

16. Mitchell T.M. Machine Learning. McGraw-Hill, 1997. 432 p.

17. Романов П.О. Розробка Telegram-ботів мовою Python : навч. посіб. Одеса: ОНПУ, 2021. 130 с.

18. Білик І.В. Комп’ютерний зір: методи та засоби. Тернопіль: ТНТУ, 2021. 160 с.

19. Котляр М.І. Інформаційні системи документообігу : навч. посіб. Харків: ХНУРЕ, 2020. 192 с.

20. Сидоренко А.В. Глибинне навчання в застосуваннях : комп’ютерне бачення та NLP. Київ: КНУ, 2022. 148 с.

21. Ruder S. An Overview of Word Embeddings and their Connection to Distributional Semantic Models [Електронний ресурс]. URL: https://ruder.io/word-embeddings (дата звернення: 17.05.2025).

22. Hugging Face. Transformers Documentation [Електронний ресурс]. URL: https://huggingface.co/docs/transformers (дата звернення: 17.05.2025).

23. Microsoft. Azure Cognitive Services [Електронний ресурс]. URL: https://azure.microsoft.com/en-us/services/cognitive-services (дата звернення: 17.05.2025).

24. Google Cloud. Natural Language API [Електронний ресурс]. URL: https://cloud.google.com/natural-language (дата звернення: 17.05.2025).

25. Mikolov T., Sutskever I., Chen K. та ін. Distributed Representations of Words and Phrases and their Compositionality. arXiv:1310.4546 [Електронний ресурс]. URL: https://arxiv.org/abs/1310.4546 (дата звернення: 17.05.2025).

26. Даниленко Д.Д., Програмний модуль чат-бота для оптимізації роботи з документацією. *Інтелектуальні інформаційні технології в прикладних дослідженнях (IITAR-2025):* матеріали VIII Міжнародної студентської науково -

технічної конференції / Тернопіль: Тернопільський національний технічний університет ім. І.Пулюя. – м. Тернопіль, 24–25 квітня 2025 р. – С. 150–151.

27. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп’яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп’ютерні науки» спеціальності 122 «Комп’ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

Дотаток А

Векторний пошук та генерації відповідей

```
from langchain.text_splitter import CharacterTextSplitter
from langchain_openai import OpenAIEmbeddings
from langchain_community.vectorstores import FAISS
from langchain.chains import RetrievalQA
from langchain_community.llms import OpenAI

from dotenv import load_dotenv
import os

load_dotenv()
openai_api_key = os.getenv("OPENAI_API_KEY")

embeddings = OpenAIEmbeddings(openai_api_key=openai_api_key)

class DocumentationChatBot:
    def __init__(self, qa_chain):
        self.qa_chain = qa_chain

    def ask(self, question: str) -> str:
        return self.qa_chain.run(question)

    @classmethod
    def from_text(cls, raw_text: str):
        text_splitter = CharacterTextSplitter(chunk_size=1000, chunk_overlap=100)
        docs = text_splitter.create_documents([raw_text])
        embeddings = OpenAIEmbeddings(openai_api_key=os.getenv("OPENAI_API_KEY"))
        vectorstore = FAISS.from_documents(docs, embeddings)
        retriever = vectorstore.as_retriever()
        llm = OpenAI(temperature=0, openai_api_key=os.getenv("OPENAI_API_KEY"))
        qa_chain = RetrievalQA.from_chain_type(llm=llm, retriever=retriever)
        return cls(qa_chain)
```

Додаток Б

Переписування документа за інструкцією користувача

```
from telegram import Update
from telegram.ext import ContextTypes
from openai import OpenAI
import os
from keyboards.inline import build_keyboard
from docx import Document
from io import BytesIO

async def rewrite_document(update: Update, context: ContextTypes.DEFAULT_TYPE, user_input: str):
    try:
        client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))
        text = context.user_data.get("doc_text")
        if not text:
            await update.message.reply_text("❗ Спочатку надішли файл.")
            return
        prompt = (
            f"Ось документ:\n{text}\n\n"
            f"Інструкція: {user_input}\n"
            f"Будь ласка, перепиши документ відповідно до інструкції."
        )
        response = client.chat.completions.create(
            model="gpt-4",
            messages=[{"role": "user", "content": prompt}],
            temperature=0.5
        )
        rewritten = response.choices[0].message.content
        context.user_data["last_answer"] = rewritten

        await update.message.reply_text(f"✏️ {rewritten}", reply_markup=build_keyboard())

        document = Document()
        document.add_paragraph(rewritten)
        buffer = BytesIO()
        document.save(buffer)
        buffer.seek(0)

        await update.message.reply_document(
            document=buffer,
            filename="rewritten_document.docx",
            caption="📄 Оновлений документ"
        )

    except Exception as e:
        await update.message.reply_text("❌ Помилка переписування.")
        print(f"[REWRITE ERROR] {e}")
```

Додаток В

Копії публікованих результатів

Західноукраїнський національний університет

Факультет комп'ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління

ЗБІРНИК ТЕЗ ДОПОВІДЕЙ

Студентської науково-практичної конференції

ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРИКЛАДНИХ ДОСЛІДЖЕННЯХ

(IITAR-2025)

27-29 травня 2025 року

Тернопіль

2025

Даниленко Денис
студент групи КНШІ-41
dendsnilencko2016@gmail.com
Західноукраїнський національний університет
Тернопіль, Україна

ПРОГРАМНИЙ МОДУЛЬ ЧАТ-БОТА ДЛЯ ОПТИМІЗАЦІЇ РОБОТИ З ДОКУМЕНТАЦІЄЮ

У сучасному світі ефективне управління документацією є критично важливим для багатьох організацій. Зростаючий обсяг інформації та потреба в швидкому доступі до неї вимагають впровадження автоматизованих рішень. Ця робота присвячена розробці програмного модуля чат-бота, призначеного для оптимізації роботи з великим обсягом текстової документації, зокрема у таких сферах як діловодство, юриспруденція, освіта та сфера обслуговування. Запропоноване рішення використовує потужності великих мовних моделей через API для аналізу, вилучення інформації, забезпечуючи швидкий та зручний доступ до неї для користувачів через інтерфейс Telegram-бота.

Управління документацією є однією з найбільш трудомістких та відповідальних задач у будь-якій організації. Традиційні методи роботи з паперовими та електронними документами часто виявляються неефективними, що призводить до втрат часу та ресурсів. Застосування сучасних технологій, зокрема великих мовних моделей (LLM) через API, та чат-ботів відкриває нові можливості для автоматизації цих процесів, значно підвищуючи ефективність взаємодії з інформацією. Метою даного дослідження є розробка та впровадження програмного модуля чат-бота, здатного ефективно працювати з великими обсягами текстових документів, спрощуючи доступ до необхідної інформації та автоматизуючи рутинні операції за допомогою зручного інтерфейсу Telegram-бота [3].

Розроблений програмний модуль чат-бота базується на архітектурі, яка передбачає взаємодію з користувачем через месенджер Telegram та інтеграцію з існуючими базами даних документів. Ключовим елементом є використання API великої мовної моделі для розуміння запитів користувача та обробки тексту.

Під час тестування розробленого програмного модуля на реальних документах було використано можливості великої мовної моделі (LLM) [1]. Отримані результати засвідчили високу ефективність системи у виконанні ключових завдань: витягу релевантної інформації, формуванні відповідей на запити користувача та забезпеченні зручного інтерфейсу через Telegram-бот. Основу роботи системи становить узгоджена взаємодія кількох програмних компонентів, кожен з яких виконує спеціалізовану функцію у процесі обробки текстової документації. Структура та призначення основних компонентів (таблиця 1).

Таблиця 1 – Оцінка точності моделі для категорії “Електроніка”

Компонент / Технологія	Опис застосування
Telegram Bot API	Отримання повідомлень та документів від користувача, надсилання відповідей
OpenAI ChatGPT API	Генерація відповідей на запити користувача, формування резюме, витяг ключової інформації
LangChain	Побудова логіки обробки запитів, організація ланцюжків роботи з документами та LLM
FAISS / Chroma	Індексація фрагментів документа та пошук релевантних за векторною схожістю
PyMuPDF / python-docx	Обробка PDF та DOCX-документів, конвертація у текстовий формат

Розроблений програмний модуль чат-бота для оптимізації роботи з документацією є ефективним інструментом, здатним значно спростити доступ до інформації та автоматизувати рутинні процеси. Запропоноване рішення, реалізоване через Telegram-бота [3] з використанням потужностей великих мовних моделей через API, може бути використане в різних галузях, де необхідна автоматизована обробка текстової інформації, зокрема в діловодстві, юриспруденції, освіті та сфері обслуговування.

Список використаних джерел

1. Jurafsky, D., & Martin, J. H. (2023). *Speech and Language Processing* (3rd ed. draft), pp. 777–832.
2. VanderPlas, J. T. (2016). Python Data Science Handbook: Essential Tools for Working with Data. O'Reilly Media pp. 305–325.
3. Documentation for python-telegram-bot: https://python-telegram-bot.org/
4. Documentation for OpenAI API https://platform.openai.com/docs/api-reference