ХЕРСОНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ
КАФЕДРА ПРОГРАМНИХ ЗАСОБІВ І ТЕХНОЛОГІЙ

Пояснювальна записка

до кваліфікаційної роботи магістра

на тему:

«Дослідження можливостей використання LLM у розробці гібридного голосового асистента**»**

Виконав: здобувач 6 курсу, групи ПР1
спеціальності 121 «Інженерія
програмного забезпечення»
Коберник Дар’я Сергіївна
(прізвище та ініціали)

Керівник: к.т.н., доц Огнєва О.Є.
(прізвище та ініціали)

Рецензент: к.т.н., доц.Корніловська Н.В.
(прізвище та ініціали)

Хмельницький – 2025р.

Факультет **Інформаційних технологій та дизайну**
Кафедра **Програмних засобів і технологій**
Освітньо-кваліфікаційний рівень **магістр**
Галузі знань **12 «Інформаційні технології»**
Спеціальність **121 «Інженерія програмного забезпечення»**
Освітньо-професійної програми **«Програмна інженерія»**

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗіТ,
к.т.н., доцент
_____О.Є.Огнєва
«____» _____ 20__ року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ ЗДОБУВАЧУ

_____
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження можливостей використання LLM у розробці гібридного голосового асистента
керівник роботи к.т.н., доцент О.Є. Огнєва

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом ХНТУ від «15» вересня 2025року №416-с.
2. Строк подання здобувачем роботи 01.12.2025
3. Вихідні дані до роботи _____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналітичний огляд наукових та інших джерел
2. Системний аналіз та обґрунтування проблеми
3. Методи та інструменти дослідження
4. Практична реалізація завдання
5. Результати розробки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
_____
_____
_____
_____

6. Консультанти розділів роботи

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

7. Дата видачі завдання_____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного роботи	Строк виконання етапів роботи	Примітка
	1. Ознайомлення з тематикою дипломного проєктування, визначення та узгодження теми кваліфікаційної роботи	01.10.2025 – 05.10.2025	Готово
	2. Аналітичний огляд наукових джерел з технологій розпізнавання мовлення, голосових асистентів та великих мовних моделей	06.10.2025 – 18.10.2025	Готово
	3. Системний аналіз предметної області та обґрунтування проблеми голосової взаємодії	19.10.2025 – 28.10.2025	Готово
	4. Розробка концепції гібридної архітектури голосового асистента	29.10.2025 – 04.11.2025	Готово
	5. Вибір методів, алгоритмів та інструментів проєктування	05.11.2025 – 10.11.2025	Готово
	6. Проєктування структури програмного забезпечення та створення UML-діаграм	11.11.2025 – 17.11.2025	Готово
	7. Реалізація прототипу гібридного голосового асистента	18.11.2025 – 30.11.2025	Готово
	8. Тестування, оцінювання продуктивності та збір результатів експериментів	01.12.2025 – 06.12.2025	Готово
	9. Оформлення пояснювальної записки, написання вступу, висновків, списку джерел і додатків	07.12.2025 – 11.12.2025	Готово
	10. Перевірка на плагіат, підготовка до захисту дипломної роботи	12.12.2025 – 13.12.2025	

Здобувач _____ Коберник Д.С

Керівник роботи _____ Огнєва О.Є.

(підпис) (прізвище та ініціали)

Анотація

У першому розділі виконано дослідження еволюції голосових асистентів, розглянуто сучасні технології розпізнавання та генерації мовлення, проаналізовано їхні сильні та слабкі сторони. Подано огляд популярних систем, таких як Siri, Alexa, Google Assistant і Cortana, а також визначено обмеження, що існують у сфері контекстності, приватності та багатомовної підтримки.

У другому розділі визначено актуальність розв’язуваної задачі, окреслено ключові вимоги до гібридної системи голосової взаємодії, сформульовано обмеження щодо апаратних можливостей, швидкодії та мережевих затримок. Обґрунтовано необхідність поєднання локальних і хмарних механізмів обробки для підвищення ефективності роботи.

Третій розділ описує принципи використання алгоритмів розпізнавання мовлення, моделі VAD, механізми визначення ключового слова, методи інтеграції з LLM, а також обґрунтовано вибір інструментів Python та відповідних бібліотек. Наведено логічну схему побудови гібридної архітектури.

Реалізацію прототипу гібридного голосового асистента подано в четвертому розділі, описано роботу ключових модулів (AudioManager, HybridRecognizer, WakeWordListener, логіку команд та інтеграцію з LLM). Окремо наведено результати тестування, включно з оцінкою WER, latency та стійкості до шумів.

П’ятий розділ містить узагальнення характеристик системи, опис сценаріїв її використання, виявлені переваги та перспективи вдосконалення. Продемонстровано роботу асистента в реальних умовах, наведено приклади журналів тестування та сформульовано напрями подальшого розвитку гібридних голосових систем.

Annotation

The first chapter presents a study of the evolution of voice assistants, examining modern technologies for speech recognition and speech generation, and analyzing their strengths and limitations. It provides an overview of popular systems such as Siri, Alexa, Google Assistant, and Cortana, and identifies existing constraints related to contextual understanding, privacy, and multilingual support.

The second chapter defines the relevance of the problem being addressed, outlines the key requirements for a hybrid voice interaction system, and formulates the limitations associated with hardware capabilities, processing speed, and network latency. The necessity of combining local and cloud-based processing mechanisms to enhance system efficiency is substantiated.

The third chapter describes the principles of using speech recognition algorithms, VAD models, wake-word detection mechanisms, methods for integrating large language models, and the rationale for selecting Python tools and corresponding libraries. A logical framework for building the hybrid architecture is presented.

The fourth chapter provides the implementation of the hybrid voice assistant prototype, describing the operation of key modules (AudioManager, HybridRecognizer, WakeWordListener, command logic, and LLM integration). The results of testing are presented separately, including the evaluation of WER, latency, and noise robustness.

The fifth chapter summarizes the characteristics of the system, outlines the scenarios of its use, identifies strengths and possible improvements, and demonstrates the assistant’s performance in real conditions. Examples of testing logs are provided, along with the proposed directions for further development of hybrid voice systems.

РЕФЕРАТ

Дипломна робота містить 99 сторінок, 18 рисунків, 2 таблиці і 18 джерел.

Мета роботи – розробити гібридний голосовий асистент, що поєднує локальні алгоритми обробки мовлення та великі мовні моделі для підвищення точності розпізнавання, контекстності та природності взаємодії з користувачем.

Актуальність зумовлена зростанням попиту на голосові інтерфейси та розвитком технологій штучного інтелекту. Сучасні системи мають обмеження щодо швидкодії, багатомовності та контекстності, що формує потребу у створенні гібридних рішень, здатних поєднувати переваги локальної та хмарної обробки. Для України тема є важливою в умовах цифрової трансформації.

Об’єкт дослідження – процес оброблення голосових запитів у системах інтерактивної взаємодії.

Предмет дослідження – методи підвищення ефективності голосових асистентів на основі алгоритмів розпізнавання мовлення та великих мовних моделей.

Результатом є прототип гібридного голосового асистента, що забезпечує точне розпізнавання мовлення, інтерпретацію намірів та генерацію відповідей. Система може бути застосована для автоматизації завдань, навчальних і сервісних рішень.

Новизна полягає у побудові гібридної архітектури, яка поєднує локальне розпізнавання мовлення із хмарною генерацією відповіді, а також у застосуванні механізмів маршрутизації запитів залежно від впевненості моделі. Це забезпечує підвищення точності та гнучкості системи.

Ключові слова: великі мовні моделі, гібридний голосовий асистент, обробка мовлення, штучний інтелект, NLP, розпізнавання голосу, синтез мовлення, Python, діалогові системи.

ЗМІСТ

ВСТУП

Розвиток сучасних інформаційних технологій сприяє активному впровадженню інтелектуальних систем, здатних автоматизувати взаємодію між користувачем і програмним забезпеченням. Одним із найперспективніших напрямів у цій сфері є створення голосових асистентів, що поєднують класичні алгоритми обробки мовлення та можливості великих мовних моделей. Зростання потреби у швидкій, адаптивній та контекстно залежній комунікації формує попит на дослідження архітектур, які дозволяють інтегрувати ці технології та забезпечувати високу якість роботи системи.

Наукова спільнота активно вивчає потенціал LLM у задачах розуміння природної мови, однак питання розроблення гібридних голосових рішень, здатних працювати у реальному часі та взаємодіяти з користувачем у різноманітних сценаріях, залишається актуальним. Сучасні підходи мають низку обмежень, пов’язаних із продуктивністю, масштабованістю та точністю розпізнавання мовлення. Тому поєднання традиційних методів обробки аудіосигналів із моделями типу LLM відкриває нові можливості для створення більш гнучких і надійних інтерфейсів.

Вибір теми обумовлений необхідністю розроблення інструментів, здатних підвищити ефективність користувацької взаємодії та адаптуватися до індивідуальних потреб. Це особливо важливо для України, де цифрова трансформація стимулює впровадження рішень, орієнтованих на доступність, автоматизацію та оптимізацію бізнес-процесів. Використання гібридних моделей у голосових асистентах може значно зменшити навантаження на операційний персонал, спростити отримання інформації та покращити якість сервісів.

Актуальність теми зумовлена стрімким розвитком мовних технологій, які суттєво змінюють підхід до взаємодії людини з цифровими системами. Використання голосових інтерфейсів стає стандартом у побутових пристроях, бізнес-процесах та сервісах підтримки, що вимагає створення рішень, здатних

працювати швидко, надійно й адаптивно. Попит на інтелектуальних асистентів постійно зростає, а їх функціональні можливості стають дедалі ширшими завдяки прогресу у сфері глибинних моделей, які забезпечують якісніше розуміння намірів користувача.

Для України ця тематика має додаткову вагу, оскільки цифрова модернізація державних сервісів і приватного сектору стимулює впровадження інструментів, що підвищують доступність інформації та оптимізують робочі процеси. Застосування гібридних мовних систем може суттєво прискорити автоматизацію комунікаційних каналів, зменшити витрати на обслуговування клієнтів та покращити якість взаємодії у сферах освіти, логістики, телекомунікацій і державного управління. Такий підхід сприяє формуванню інноваційних рішень, що відповідають сучасним викликам і підсилюють конкурентоспроможність вітчизняних технологічних продуктів.

Метою дослідження є аналіз і вивчення шляхів удосконалення голосових асистентів шляхом інтеграції алгоритмів штучного інтелекту. У роботі розглядаються сучасні методи автоматичного розпізнавання мовлення, досліджуються можливості глибокого навчання та обробки природної мови, що дозволяють підвищити точність розуміння команд. Окрему увагу приділено розгляду архітектури нейронних мереж, які застосовуються у популярних голосових платформах, таких як Siri, Google Assistant чи Cortana, що демонструють високий рівень технічної реалізації.

Для досягнення мети передбачено виконання таких завдань: визначення вимог до архітектури асистента; аналіз алгоритмів перетворення мовлення; дослідження методів інтеграції LLM; оцінювання ефективності запропонованої системи.

Об’єктом дослідження є процес оброблення голосових запитів користувача в інтерактивних системах.

Предметом дослідження є підходи до вдосконалення технологій голосових асистентів на основі штучного інтелекту. Аналізуються способи підвищення

точності розпізнавання мовлення, швидкості реагування та здатності систем розуміти контекст розмови. Особливу увагу приділено процесам адаптації моделей до різних користувачів, удосконаленню мовних корпусів і створенню універсальних алгоритмів, здатних розпізнавати різні варіанти вимови, сленг або специфічну лексику.

Наукова новизна роботи полягає у спробі комплексного аналізу процесів інтеграції інтелектуальних технологій у голосові сервіси з урахуванням сучасних досягнень машинного навчання. Автор розглядає не лише технічні, але й когнітивні аспекти взаємодії людини з комп’ютером, що дозволяє глибше зрозуміти механізми ефективного спілкування користувача з асистентом.

Практична значущість дослідження полягає у можливості впровадження отриманих результатів у реальні програмні продукти, що функціонують на персональних комп’ютерах і мобільних пристроях.

Завдання, які було виконано під час дослідження:

– Проаналізувати сучасні голосові асистенти та технології розпізнавання мовлення.

– Дослідити алгоритми обробки аудіосигналів, визначення мовної активності та ключового слова.

– Розробити архітектуру гібридної системи голосового асистента.

– Створити основні модулі: збір аудіо, розпізнавання мовлення, логіку команд і інтеграцію з LLM.

– Реалізувати механізм маршрутизації запитів між локальною та хмарною обробкою.

– Провести тестування точності розпізнавання, швидкодії та стійкості системи до шумів.

– Оцінити результати роботи та визначити перспективи подальшого розвитку системи.

РОЗДІЛ 1. АНАЛІТИЧНИЙ ОГЛЯД НАУКОВИХ ТА ІНШИХ ДЖЕРЕЛ

1.1 Аналіз предметної області

Сучасний розвиток цифрових технологій суттєво змінює спосіб взаємодії людини з комп’ютером. Зростання кількості інформаційних потоків, збільшення обсягу рутинних завдань та потреба в швидкому доступі до даних формують попит на інструменти, які здатні полегшувати робочі процеси та оптимізувати щоденні дії користувача. Однією з таких технологій стали голосові асистенти, що забезпечують можливість виконання команд мовою природного спілкування. Автоматичне розпізнавання мовлення за останні роки розвивається дуже швидко, переходячи від ранніх шаблонних систем до сучасних глибоких нейронних архітектур. [17]

Історія розвитку голосових асистентів бере початок із простих систем автоматичного розпізнавання мовлення (ASR, Automatic Speech Recognition) та базових правил (rule-based systems), які могли реагувати лише на обмежений набір команд. У 1990–2000-х роках відбувався перехід до методів машинного навчання (ML), що дозволяло підвищити точність розпізнавання мовлення, враховуючи варіації дикції та фонетичні особливості.

На наступному етапі з’явилися системи обробки природної мови (NLP), які дозволяли інтерпретувати значення команд і виконувати прості логічні операції. Класичні підходи включали поєднання ASR + NLP, rule-based системи та моделі на основі ML, що становили основу ранніх голосових помічників.

Голосові помічники інтегрують алгоритми автоматичного розпізнавання мовлення, методи обробки природної мови та системи генерації відповідей, формуючи повноцінний механізм діалогової взаємодії.

На відміну від ранніх інструментів, орієнтованих на обмежений набір команд, сучасні розробки використовують глибокі нейронні мережі, що дозволяє

краще інтерпретувати інтонацію, контекст та комунікативні наміри користувача. Це підвищує точність відтворення мовних конструкцій і забезпечує персоналізацію взаємодії.

Базовим елементом будь-якого голосового асистента є технологія розпізнавання голосу, яка переводить аудіосигнал у текстову форму. Сучасні системи спираються на багаторівневі архітектури обробки мовлення, що включають моделі прогнозування фонем, аналіз спектральних характеристик звучання та використання рекурентних або трансформерних мереж для зменшення похибок у транскрипції. Такі рішення дозволяють враховувати тональність, темп висловлювання, особливості вимови й навіть емоційну складову. Серед поширених фреймворків для цих завдань сьогодні використовуються Whisper, DeepSpeech, SpeechBrain, а також бібліотеки Transformers та Rasa для інтеграції NLP-модулів і побудови діалогових систем.

Інтеграція штучного інтелекту в структуру голосових асистентів суттєво розширила сферу їх застосування. Сучасні системи активно використовують великі мовні моделі (LLM), такі як GPT, LLaMA, Mistral, Claude та інші, які дозволяють аналізувати текстові запити, підтримувати тривалі контексти діалогів, формувати логічні доведення та створювати розгорнуті відповіді. LLM інтегруються у голосові системи через поєднання з ASR і NLP-модулями, створюючи гібридні архітектури типу ML + LLM, ASR + LLM, NLU + LLM, де локальна обробка забезпечує швидку реакцію та безпеку, а хмарні моделі – інтелектуальну генерацію відповідей. Завдяки можливості адаптації до поведінки користувача вони здатні виконувати все ширший спектр завдань – від простих запитів до комплексної автоматизації робочих процесів. Застосування LLM у голосових системах дозволяє значно покращити точність розуміння намірів користувача, адаптувати стиль відповіді до індивідуальних особливостей мовлення та передбачати подальші дії. Це створює новий рівень персоналізації й ефективності взаємодії, який раніше був недосяжним для класичних rule-based або

ML-систем. При цьому LLM здатні працювати у багатомовному середовищі, обробляти складні контексти та підтримувати більш природний діалог.

Попри значні здобутки, у сфері голосових технологій зберігається низка труднощів, пов’язаних з якістю розуміння контексту, обробкою багатомовного мовлення, ідентифікацією емоцій, а також захистом персональних даних.

Реакція системи все ще залежить від зовнішніх факторів: шуму навколишнього середовища, технічних характеристик мікрофону, особливостей дикції або швидкості мовлення. Це створює потребу у вдосконаленні алгоритмів і розвитку гібридних підходів, що поєднують локальні засоби обробки й хмарні моделі штучного інтелекту.

Поява великих мовних моделей (LLM) стала ключовим фактором нового етапу еволюції голосових асистентів. На відміну від традиційних NLP-систем, LLM здатні підтримувати велику кількість діалогових сценаріїв, створювати розгорнуті відповіді, формувати логічні доведення та будувати контекстні ланцюги значно довшої тривалості.

Вони демонструють здатність аналізувати структуру повідомлення, підтримувати природний діалог, пояснювати складні теми та адаптувати стиль відповіді до користувача. Це відкриває можливість створення асистентів нового типу – гібридних систем, де локальні алгоритми забезпечують швидку взаємодію та безпеку даних, а LLM виконують інтелектуальну обробку складних запитів.

Таким чином, аналіз предметної області засвідчує значний потенціал використання штучного інтелекту, зокрема великих мовних моделей, у розвитку голосових асистентів для персональних комп’ютерів. Технологічний прогрес у сфері мовних моделей, зростання обчислювальних можливостей та поява змішаних архітектур створюють передумови для побудови систем, що володіють високим рівнем адаптивності, гнучкості та інтелектуальної взаємодії з користувачем.

1.2. Огляд наявних доступних рішень

Голосові асистенти широко представлені на сучасному ринку й активно розвиваються у межах екосистем провідних технологічних компаній. Кожна система має власну архітектуру, підходи до реалізації мовної взаємодії та набори функцій. Їх порівняння дозволяє визначити сильні й слабкі сторони існуючих рішень, а також обґрунтувати необхідність створення гібридної моделі з використанням LLM. Дослідження LLM активно розвивається як у науковому середовищі, так і в індустрії, особливо після того, як ChatGPT привернув широку увагу громадськості. [18]

Одним із найпоширеніших рішень є Google Assistant, який функціонує на основі моделей машинного навчання та комплексних алгоритмів обробки мовлення. Він демонструє високу точність розпізнавання голосу, а також здатність адаптуватися до особливостей вимови конкретного користувача. Асистент ефективно працює з сервісами Google та підтримує широкий набір функцій, включаючи керування додатками, роботу з календарем, пошук інформації, налаштування нагадувань, підключення до IoT-пристроїв. Розширення моделі Bard та впровадження Google Gemini зробили можливим використання більш складних запитів, однак доступність таких можливостей залежить від регіону та пристрою.

Cortana, створена корпорацією Microsoft, свого часу відігравала роль системного помічника у Windows. Вона мала значну інтеграцію з інструментами Microsoft Office, що робило її корисним інструментом у корпоративних середовищах. Cortana могла виконувати контекстні нагадування, керувати налаштуваннями системи, шукати файли та синхронізувати дані між пристроями користувача. Хоча Cortana поступово згортається, дослідження інтеграції голосових служб Microsoft демонструють застосування хмарних моделей у персональних асистентах. [9] Функції системи частково передані платформі Copilot, яка працює виключно в хмарному середовищі та не надає можливості створення локальних асистентів на ПК.

Система Siri, інтегрована в екосистему Apple, є одним із найвідоміших голосових асистентів. Вона забезпечує природну взаємодію з пристроями Apple, підтримує hands-free режими, працює з повідомленнями, навігацією, нагадуваннями та додатками. SiriKit дозволяє стороннім застосункам інтегруватися із Siri, забезпечуючи голосове керування та обробку намірів користувача. [6] Проте її використання обмежене виключно пристроями компанії Apple, а функціонал налаштування залишається закритим для сторонніх розробників.

Одним із найбільш гнучких рішень є Amazon Alexa, архітектура якої базується на модульному підході. Alexa Skills Kit дозволяє створювати навички для голосового керування пристроями й сервісами на основі інтелектуальних моделей Amazon. [5] Alexa широко використовується в розумних будинках, мультимедійних системах, а також у комерційних рішеннях для автоматизації бізнес-процесів. Відкриті API роблять її привабливим інструментом для створення сторонніх розширень, однак робота асистента повністю залежить від хмарних сервісів Amazon.

Хоча кожен із зазначених продуктів має широкий функціонал та розвинену інфраструктуру, у всіх системах існують обмеження. Серед них – закритість платформ, залежність від хмарних серверів, відсутність можливості адаптації до індивідуальних потреб користувача, недостатня гнучкість у виконанні складних запитів та залежність від конкретного виробника. Більшість асистентів не підтримують повноцінну локальну обробку даних, що створює складнощі з конфіденційністю. Крім того, вони здебільшого не використовують сучасні великі мовні моделі в повному обсязі, обмежуючи користувача заздалегідь визначеними сценаріями.

Протягом останніх років спостерігається тенденція переходу від класичних голосових помічників до асистентів, здатних вести активний діалог, аналізувати наміри користувача та виконувати комплексні завдання. Саме використання LLM відкриває можливість для побудови систем нового покоління з широкою

функціональністю, високим рівнем персоналізації та гнучкою архітектурою. Гібридний підхід, який поєднує локальну обробку аудіосигналів та інтелектуальну хмарну модель, вирішує обмеження як традиційних систем, так і суто хмарних рішень, забезпечуючи баланс між безпекою, продуктивністю та адаптивністю.

Таким чином, огляд доступних технологій свідчить про значний запит на створення нових типів голосових асистентів, які використовують потенціал великих мовних моделей. Це підтверджує актуальність розробки гібридного асистента, здатного працювати з контекстом, адаптуватися до користувача та забезпечувати високу якість обробки голосових команд у середовищі персонального комп’ютера.

1.3 Аналіз сучасних підходів та фреймворків

Розвиток голосових технологій за останнє десятиліття значною мірою пов’язаний із еволюцією інструментів, бібліотек та фреймворків, що забезпечують можливість обробляти мовлення, аналізувати природну мову та поєднувати ці процеси з великими мовними моделями. Сучасні підходи формуються навколо кількох ключових напрямів: удосконалення автоматичного розпізнавання мовлення, побудова гнучких NLU-систем, використання трансформерної архітектури та систематична інтеграція глибинних моделей у діалогові середовища. Кожен із цих напрямів має власні інструменти та екосистеми, які значно полегшують розробку голосових асистентів і підвищують якість взаємодії людини з комп’ютером.

Одним із найяскравіших рішень у сфері розпізнавання мовлення стала поява Whisper. Ця модель вирізняється високою стійкістю до шумів, підтримкою великої кількості мов та здатністю успішно працювати навіть у складних акустичних умовах. На відміну від традиційних систем, що вимагали багаторівневих акустичних і мовних моделей, Whisper реалізує підхід «end-to-

end», у якому весь процес декодування мовлення виконується єдиною нейронною мережею. Whisper демонструє значну стійкість до шуму та акцентів, що робить його одним із найнадійніших загальнодоступних мовних моделей. [14]

Аналогічну роль у попередні роки відігравав DeepSpeech – проєкт, що базувався на рекурентних мережах та CTC-декодуванні. Він започаткував тренд відкритих ASR-моделей і дав змогу створювати локальні голосові системи, але не зміг зрівнятися з трансформерними рішеннями, які згодом отримали перевагу завдяки точності та універсальності. DeepSpeech використовує архітектуру на основі LSTM, орієнтовану на швидке й точне перетворення мовлення в текст на звичайних ПК. [8]

Суттєвий прогрес відбувся й у сфері обробки природної мови. Якщо раніше діалогові системи будувалися на правилах, словниках, граматичних шаблонах і статистичних алгоритмах, то сьогодні основою стала архітектура трансформерів. Фреймворк Transformers, створений компанією Hugging Face, фактично стандартизував роботу з великими мовними моделями, надавши універсальні інструменти для аналізу тексту, генерації відповідей, класифікації запитів та багатьох інших NLU-задач. Моделі, які працюють у цій екосистемі, охоплюють широкий спектр масштабів – від компактних BERT-подібних архітектур до сучасних LLM на кшталт GPT, LLaMA, Mistral або Claude. Завдяки можливості завантажувати, оновлювати та адаптувати моделі безпосередньо у власному середовищі, розробник отримує гнучкий інструмент для створення інтелектуальних діалогових агентів.

У сфері побудови власних голосових асистентів важливим компонентом стала платформа Rasa. Платформа застосовує машинне навчання для керування діалогами та забезпечує гнучку архітектуру для створення інтелектуальних чат- та голосових асистентів. [13] На відміну від суто генеративних моделей, Rasa забезпечує детальний контроль над логікою спілкування та дозволяє створювати керовані бізнес-процеси, що особливо важливо для систем, де необхідна безпека, формалізовані сценарії або робота з конфіденційними даними. Проте, попри свої

переваги, вона значно поступається LLM у гнучкості та здатності працювати зі складними діалогами. Саме тому сучасні рішення часто об’єднують Rasa як основу для NLU із великими мовними моделями як інтелектуальний надбудовний компонент, що виконує генерацію відповідей і підтримку довготривалого контексту.

Іще один важливий напрям – фреймворки, орієнтовані на комплексну роботу зі звуковими сигналами. SpeechBrain став одним із найпотужніших універсальних інструментів для створення систем аудіообробки, включаючи розпізнавання, відновлення та класифікацію мовлення. Завдяки модульній структурі він дозволяє реалізовувати кастомні архітектури й експериментувати з різними моделями, зберігаючи високий рівень продуктивності. SpeechBrain дозволяє будувати моделі ASR на обмежених даних, зберігаючи продуктивність на рівні великих комерційних систем. [12] Його перевага полягає у здатності забезпечувати повний цикл аудіоаналізу, що робить його корисним у проектах, де важливо гнучко керувати обробкою звуку.

Усі згадані фреймворки об’єднані спільною концепцією: перехід від жорстких, формальних систем на основі правил до гнучких, контекстно орієнтованих моделей, здатних аналізувати мовлення та текст у режимі реального часу. Наявність відкритого ПЗ, широкої підтримки ком’юніті та можливості локального запуску створює передумови для побудови гібридних голосових асистентів, у яких локальна обробка поєднується з роботою великих мовних моделей. Саме така тенденція визначає сучасний напрям розвитку галузі, а обрані фреймворки забезпечують необхідну основу для створення інтелектуальних систем, здатних працювати з реальним мовленням, адаптуватися до користувача та підтримувати природний діалог.

1.4. Проблематика існуючих голосових асистентів

Проблематика сучасних голосових асистентів залишається одним із ключових чинників, що стримує їх широке використання на персональних комп’ютерах та в професійних середовищах. Незважаючи на значний прогрес у розпізнаванні мовлення та генерації відповідей, більшість наявних рішень усе ще мають низку обмежень, які негативно впливають на зручність, швидкість і безпеку взаємодії. Ці недоліки стають особливо помітними в умовах реального використання, коли користувач очікує миттєвої реакції, високої точності та стійкості роботи навіть у змінних або складних середовищах.

Однією з найпоширеніших проблем є затримка під час обробки запитів. Більшість популярних асистентів функціонують за принципом хмарної обробки, коли аудіодані передаються на сервер, де виконуються всі основні етапи аналізу та генерації відповіді. Такий підхід забезпечує високу точність, але створює додаткову часову затримку, що особливо відчутно під час виконання коротких команд. У випадках нестабільного інтернет-з’єднання система може реагувати з помітними затримками або взагалі не виконувати запити, що робить її ненадійною в критичних або термінових ситуаціях.

Проблема приватності також є однією з найважливіших у сфері голосових технологій. Більшість комерційних асистентів вимагають передавання голосових даних на сервери компанії, де вони зберігаються, аналізуються або використовуються для покращення моделей. Це створює низку ризиків, зокрема можливість несанкціонованого доступу, витоку персональної інформації або небажаного стороннього аналізу приватних розмов. Для організацій, що працюють із конфіденційною інформацією, або для користувачів, які цінують свою приватність, такі умови є неприйнятними, що ускладнює застосування голосових асистентів поза побутовою сферою.

Ще однією суттєвою проблемою є недостатня контекстність. Багато асистентів здатні обробляти лише окремі, ізольовані команди, не аналізуючи тривалі діалогові ланцюги та не враховуючи попередні висловлювання. Це обмежує їх здатність вести природний діалог, виконувати багатокрокові

інструкції або підтримувати складні користувацькі сценарії. Відсутність глибокого контекстуального розуміння знижує гнучкість системи та змушує користувача формулювати запити максимально точно й структуровано, що погіршує загальну зручність взаємодії.

Проблема багатомовності також залишається актуальною. Хоча деякі асистенти підтримують кілька мов, вони часто демонструють нерівномірну якість розпізнавання та генерації, залежно від конкретної мови. Крім того, у багатьох випадках система не здатна динамічно перемикатися між мовами залежно від контексту або користувацьких налаштувань. Для багатомовних користувачів це серйозно обмежує функціональність і створює додаткові бар’єри.

Локальність обробки даних є ще однією важливою проблемою. Більшість існуючих рішень покладаються на хмарні сервери, що робить їх залежними від стабільності інтернет-з’єднання та політики провайдера сервісу. У багатьох випадках користувачі потребують повністю автономного голосового асистента, який здатний виконувати всі ключові функції без доступу до мережі. Локальна обробка забезпечує не лише швидкість, але й конфіденційність, однак сучасні комерційні асистенти рідко дозволяють працювати в повністю офлайн-режимі, а інколи взагалі не підтримують такої можливості.

Таким чином, аналіз проблематики існуючих голосових асистентів демонструє наявність низки суттєвих обмежень, які стають перешкодою для створення універсальних, швидких, безпечних і контекстно-чутливих систем. Ці недоліки підкреслюють потребу у гібридних підходах, що поєднують локальні алгоритми з інтелектуальними мовними моделями нового покоління. Саме таке поєднання здатне забезпечити баланс між продуктивністю, приватністю, гнучкістю та природністю взаємодії, що обґрунтовує актуальність дослідження та розробки гібридного голосового асистента.

1.5 Розвиток великих мовних моделей у контексті голосових систем

Поява великих мовних моделей стала одним із найважливіших технологічних зрушень у сфері обробки природної мови. Вони сформували нову парадигму побудови голосових асистентів, оскільки здатність таких моделей працювати з контекстом, створювати логічно пов’язані відповіді, формулювати аргументовані судження та адаптуватися до стилю користувача докорінно змінила уявлення про можливості діалогових систем. Еволюція великих моделей від ранніх трансформерних архітектур до сучасних генеративних систем дозволила переносити частину інтелектуальної взаємодії зі сторони розробника на саму модель, що створило умови для будівництва гібридних асистентів, які поєднують локальні алгоритми обробки мовлення з глибинними контекстуальними механізмами.

Ранні мовні моделі виконували суто статистичні функції та не могли забезпечувати глибокого семантичного аналізу запитів. Вони працювали з обмеженим словником, не враховували попередні репліки діалогу та не мали можливості формувати узгоджені логічні відповіді. Впровадження архітектури transformer стало поворотним моментом, оскільки механізм самоуваги надав можливість моделі аналізувати текст цілісно, а не послідовно, що значно покращило здатність розпізнавати залежності між словами на будь-якій відстані та застосовувати знання у ширшому контексті.

Подальший розвиток привів до появи генеративних моделей, таких як GPT, LLaMA, Mistral, Claude та інші, які не лише обробляють текст, а й створюють новий, базуючись на величезних обсягах навчальних даних. Ці моделі продемонстрували вміння працювати з довготривалими діалогами, підтримувати цілісність контексту та інтерпретувати нечіткі формулювання, що особливо важливо для голосових систем, де природність мовлення користувача не завжди відповідає формальним граматичним правилам. Модель може передбачати наміри співрозмовника, аналізувати приховану мотивацію запиту та формувати відповідь з урахуванням попереднього досвіду взаємодії.

У контексті голосових систем великі мовні моделі набули особливого значення завдяки здатності об’єднувати функції розуміння та генерації. Вони дозволяють зменшити роль проміжних етапів обробки, які раніше вимагали розділення на модулі класифікації намірів, виявлення сутностей і формування відповідей. Сучасний асистент може функціонувати на основі єдиного генеративного ядра, яке одночасно аналізує текст, розпізнає намір і створює релевантну відповідь. Це підвищує гнучкість системи, зменшує кількість помилок і забезпечує більш природне спілкування.

Розвиток моделей сприяв також розширенню багатомовності. Моделі нового покоління здатні працювати зі змішаними мовами, адаптуватися до різних фонетичних особливостей і враховувати культурний контекст, що стало можливим завдяки використанню багатомовних датасетів у процесі навчання. Для голосових асистентів це означає можливість працювати з різними мовами без потреби у створенні окремих моделей для кожної мовної групи.

Одним із найважливіших наслідків розвитку великих мовних моделей є можливість створення асистентів, які не потребують попереднього програмування сценаріїв. Раніше розробники були змушені описувати кожну логіку діалогу вручну, використовуючи набори правил або системи керування станами. Нині модель здатна самостійно формувати відповідь, керуючись закономірностями, отриманими під час навчання, що значно спрощує розробку і розширення функціональності.

Таким чином, розвиток великих мовних моделей став основою нового покоління голосових систем, у яких інтелектуальне ядро здатне не лише інтерпретувати мовлення, а й підтримувати природну взаємодію, забезпечуючи вищу якість роботи порівняно з традиційними підходами. Ця тенденція продовжує впливати на еволюцію голосових асистентів, розширюючи межі їх застосування та відкриваючи можливості для побудови більш адаптивних і контекстно орієнтованих систем.

1.6 Порівняльний аналіз LLM у задачах діалогової взаємодії

Порівняння великих мовних моделей у контексті діалогової взаємодії стало важливим напрямом досліджень, оскільки кожна з моделей демонструє відмінні характеристики, архітектурні особливості та рівень адаптивності до специфічних завдань. Аналіз таких моделей, як GPT, Claude, Mistral, LLaMA та інших представників сучасних генеративних систем, дає змогу визначити, якою мірою вони відповідають вимогам голосових асистентів, де критичними є швидкість реакції, стійкість до помилок, точність інтерпретації та здатність довго утримувати контекст.

GPT є однією з перших моделей, яка продемонструвала справжню універсальність та високу якість діалогової взаємодії. Її перевага полягає у здатності генерувати розгорнуті, узгоджені та стилістично виважені відповіді. GPT ефективно працює в умовах багатокрокового діалогу, зберігаючи зв’язність між репліками. Така властивість особливо важлива для голосових систем, у яких користувач може повертатися до попередніх тем або змінювати напрям розмови без чіткої логічної структури. Недоліком GPT є підвищена залежність від обчислювальних ресурсів, що може призводити до збільшення часу генерації, якщо модель не оптимізована або працює у середовищі з обмеженими можливостями.

Claude орієнтований на уважність до інструкцій і здатність чітко виконувати поставлені завдання. Він краще структурує інформацію та рідше відхиляється від теми, що робить його корисним у голосових асистентах, які працюють у службових, корпоративних або аналітичних середовищах. Claude демонструє підвищену точність у відповідях на спеціалізовані запитання, однак інколи проявляє надмірну формальність, що може позначитися на природності діалогів у побутових сценаріях.

Mistral, попри компактніший розмір, забезпечує високу швидкість роботи та ефективність у контекстній генерації. Ця модель демонструє баланс між

продуктивністю і якістю, що дозволяє застосовувати її в системах з обмеженими ресурсами або у випадках, де потрібна оперативна реакція. Через менший розмір Mistral може мати обмеження у глибоких семантичних міркуваннях, однак вона відзначається високою стабільністю у коротких діалогах.

LLaMA є однією з найбільш гнучких моделей завдяки наявності вільних модифікацій і можливості розгортання на локальному обладнанні. Це відкриває шлях до створення асистентів, які працюють без підключення до мережі або з мінімальною залежністю від хмарних ресурсів. LLaMA демонструє високу якість генерації, проте результати залежать від конкретної версії та конфігурації моделі, що накладає додаткові вимоги до оптимізації.

Порівняльний аналіз демонструє, що якість діалогової взаємодії залежить не лише від архітектури, але й від можливості моделі адаптуватися до конкретного середовища. Для голосових систем важливими є стійкість до неповних даних, здатність працювати з фразами, що містять паузи, емоційні відтінки або неточні формулювання. Моделі, які краще розуміють контекст і передбачають намір користувача, забезпечують вищу природність взаємодії. Водночас системи, орієнтовані на швидкість, краще працюють у режимі реального часу, що критично для голосових асистентів.

У підсумку можна стверджувати, що жодна з мовних моделей не є універсальною для всіх сценаріїв диалогової взаємодії. Кожна має власні переваги, які варто враховувати під час створення гібридного голосового асистента. Оптимальним підходом є комбінування локальних можливостей з інтелектуальними функціями хмарних моделей, що дозволяє досягти балансу між швидкістю, точністю і гнучкістю системи.

1.7 Висновки до розділу 1

У межах першого розділу було проведено комплексний аналітичний огляд предметної області, сучасних рішень, фреймворків та мовних моделей, що

формують основу розвитку голосових асистентів. Аналіз літератури та наявних технологій підтвердив, що еволюція голосових систем нерозривно пов’язана з розвитком методів автоматичного розпізнавання мовлення, обробки природної мови та появою великих мовних моделей, здатних забезпечувати глибоке контекстне розуміння.

Розгляд історії становлення голосових асистентів показав поступовий перехід від простих rule-based систем до глибоких нейронних мереж, що дозволило суттєво підвищити точність, гнучкість і природність взаємодії. Аналіз сучасних комерційних рішень (Google Assistant, Alexa, Siri, Cortana) засвідчив, що хоча ці системи є функціональними та зручними, вони залишаються залежними від хмарних платформ, мають обмежені можливості локальної обробки, недостатню підтримку контексту та обмежену гнучкість у налаштуванні.

Вивчення сучасних інструментів і фреймворків показало, що моделі Whisper, DeepSpeech, SpeechBrain та екосистема Transformers забезпечують технологічну основу для побудови локальних або гібридних голосових систем, тоді як інструменти на кшталт Rasa дозволяють будувати керовані діалогові сценарії. Разом із тим великі мовні моделі нового покоління (GPT, Claude, LLaMA, Mistral) відкривають можливість створення асистентів із глибоким контекстним аналізом і природною генерацією мови.

Проблеми, характерні для сучасних голосових асистентів, – затримки, залежність від інтернету, ризики приватності, недостатня багатомовність та обмежений контекст – підтверджують необхідність переходу до гібридної архітектури, яка поєднує локальні алгоритми обробки звуку з інтелектуальними можливостями LLM.

Узагальнюючи проведений аналіз, можна стверджувати, що предметна область має значний потенціал для подальшого розвитку, а гібридні голосові системи є перспективним напрямом завдяки здатності забезпечувати високу швидкість реакції, безпеку обробки даних, адаптивність та глибину взаємодії.

Саме ці висновки формують концептуальну основу для подальшого проектування та реалізації гібридного голосового асистента в наступних розділах роботи.

РОЗДІЛ 2. СИСТЕМНИЙ АНАЛІЗ ТА ОБҐРУНТУВАННЯ ПРОБЛЕМИ

2.1 Системний аналіз об’єкта та предметної області

Об’єктом дослідження є системи голосових асистентів для персональних комп’ютерів, що поєднують локальні алгоритми розпізнавання мовлення з компонентами штучного інтелекту для обробки природної мови. Предмет дослідження охоплює методи та технології підвищення ефективності взаємодії користувача з комп’ютером за допомогою гібридних голосових асистентів, інтегрованих з великими мовними моделями.

Системний аналіз передбачає оцінку архітектури таких систем, їхніх функціональних складових, алгоритмів обробки сигналів і мовних патернів, а також способів взаємодії з користувачем. Сучасні голосові асистенти включають модулі розпізнавання мовлення, які перетворюють аудіосигнали у текстові команди; модулі обробки природної мови, що аналізують текстові запити, визначають наміри користувача та здійснюють семантичний розбір; модулі генерації відповідей, які формують текстові або голосові реакції на основі контексту діалогу; а також інтерфейси взаємодії, що забезпечують зручну форму комунікації з користувачем та інтеграцію з іншими системами.

Важливою складовою є модуль адаптації та персоналізації, який накопичує інформацію про стиль мовлення користувача, його звички та індивідуальні уподобання, що дозволяє підвищити точність реакцій системи і рівень персоналізації. Предметна область охоплює не лише технічні аспекти побудови голосових систем, а й питання користувацького досвіду, безпеки даних, адаптивності та багатомовності.

Використання великих мовних моделей дозволяє моделювати складні сценарії діалогу, передбачати наміри користувача та формувати природні відповіді, що значно підвищує практичну цінність голосового асистента.

Аналіз показує, що існуючі рішення мають певні обмеження, пов’язані з залежністю від хмарних сервісів, недостатньою персоналізацією, обмеженою багатомовною підтримкою та недостатнім розумінням контексту мовлення, а також обмеженими можливостями локальної обробки даних, що створює ризики конфіденційності. Це свідчить про потребу у створенні гібридних архітектур, що поєднують локальні алгоритми обробки голосу та можливості штучного інтелекту для забезпечення балансу між швидкістю, точністю, безпекою і адаптивністю.

2.2 Постановка та обґрунтування проблеми

Сучасні голосові асистенти продемонстрували високий рівень ефективності на мобільних пристроях та у межах екосистем розумного дому, проте їхнє застосування на персональних комп’ютерах залишається недостатньо розвиненим. Існуючі системи здебільшого орієнтовані на виконання окремих команд і обмежені у можливостях адаптації до індивідуального стилю мовлення користувача, що знижує рівень комфорту та продуктивності при роботі з комп’ютером. Значною проблемою є недостатнє розуміння контексту висловлювань, обмежена здатність передбачати дії користувача та обробляти складні сценарії взаємодії.

Наявні голосові асистенти мають ряд конкретних обмежень. Наприклад, Siri часто потребує підключення до інтернету, що ускладнює роботу в умовах обмеженого доступу до мережі, і не завжди коректно обробляє складні команди у робочому середовищі ПК. Google Assistant, поєднує локальну обробку мовлення з хмарними моделями, досягаючи мінімальної затримки в реальних сценаріях взаємодії, [10] попри високу точність розпізнавання мовлення, обмежений власною екосистемою Google, що створює труднощі при інтеграції з стороннім програмним забезпеченням та корпоративними системами. Alexa, розроблена Amazon, орієнтована переважно на управління розумним будинком і модульні навички, що робить її менш зручною для виконання робочих задач на ПК. Cortana, яка мала глибоку інтеграцію з Windows, поступово втрачає підтримку та функціональність, що знижує її практичну цінність.

Крім того, усі згадані системи мають обмеження у сфері персоналізації та контекстного аналізу. Вони не здатні ефективно враховувати інтонаційні нюанси, швидкість мовлення, професійну лексику або специфічні сценарії використання у різних галузях. Це призводить до помилок при інтерпретації команд, необхідності повторного введення запитів і загального зниження продуктивності. Також обмежена локальна обробка даних створює ризики втрати конфіденційності та залежність від хмарних сервісів, що може бути критичним для корпоративних користувачів або в умовах обмеженого інтернет-з’єднання.

У зв’язку з цим актуальною є розробка гібридних голосових асистентів, які поєднують локальну обробку голосу з можливостями штучного інтелекту та великих мовних моделей. Такі системи мають забезпечувати передбачення намірів користувача, персоналізацію взаємодії, багатомовну підтримку та високий рівень конфіденційності. Розв’язання цієї проблеми дозволяє підвищити ефективність користувацької взаємодії, створити більш природний та інтуїтивний інтерфейс, а також забезпечити практичне застосування сучасних технологій штучного інтелекту у персональних комп’ютерах.

Отже, постановка проблеми визначає необхідність системного підходу до розробки голосових асистентів нового покоління, інтегрованих з алгоритмами штучного інтелекту, що забезпечують точне розпізнавання мовлення, адаптацію до користувача та безпечну обробку даних. Ця проблематика формує основу для подальших досліджень та розробки ефективних методів і засобів її вирішення у межах магістерської роботи.

2.3. Формалізація вимог до системи

Формалізація вимог до системи є ключовим етапом проєктування гібридного голосового асистента, оскільки вона визначає межі, можливості та очікувану поведінку розроблюваного програмного продукту. На цьому етапі важливо чітко окреслити функціональні можливості системи, вимоги до її продуктивності, рівень безпеки обробки даних, а також критерії сумісності з апаратними та програмними платформами. Формалізація вимог забезпечує однозначне розуміння поставленої задачі як для розробника, так і для майбутнього користувача, створюючи основу для подальшої реалізації, тестування та вдосконалення програмного комплексу.

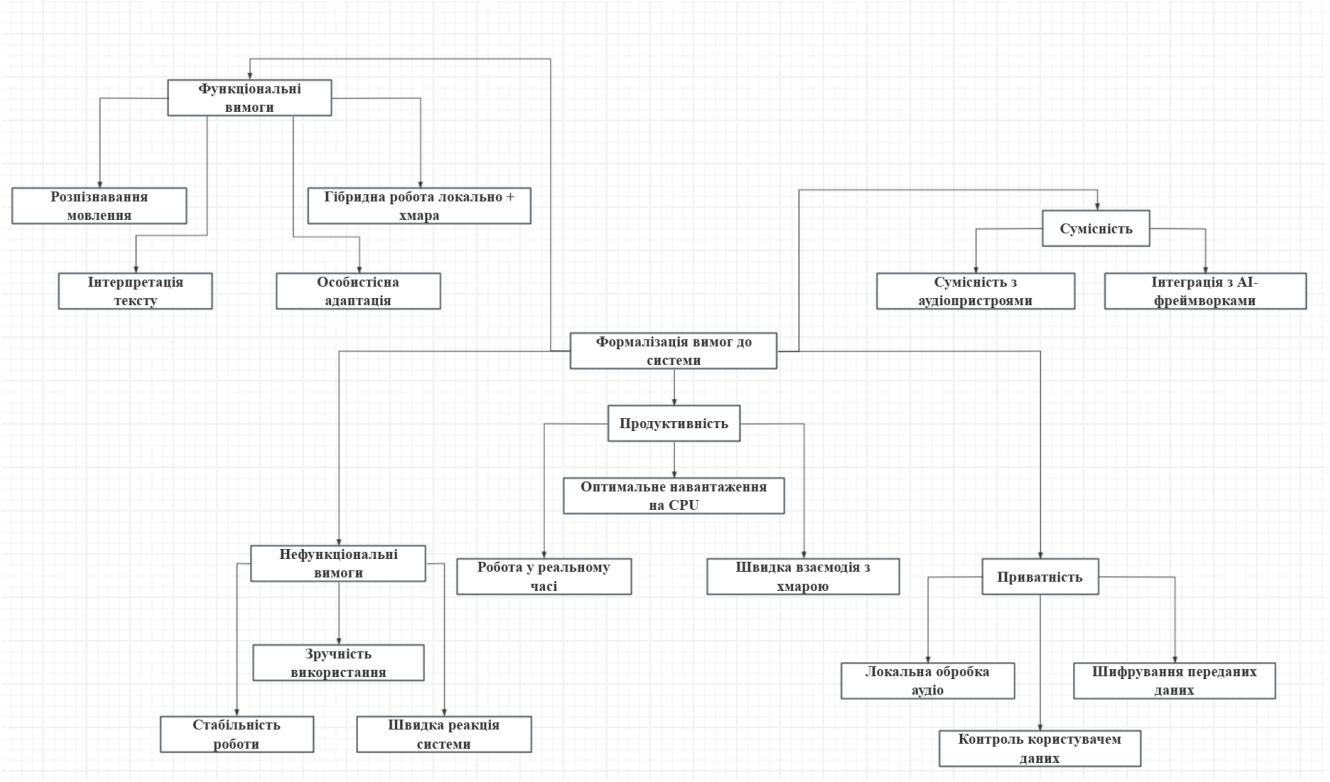

Рис. 2.1 – Діаграма вимог до системи

Функціональні вимоги, зображені на рисунку 2.1, визначають конкретні операції, які має виконувати система. Основною функцією є можливість перетворення голосових запитів у текстову форму та подальше розуміння змісту висловлювання за допомогою алгоритмів обробки природної мови. Асистент повинен забезпечувати підтримку діалогової взаємодії, розпізнавання намірів користувача, виконання локальних операцій у межах операційної системи, а також формування відповіді у текстовому або звуковому форматі. Система має підтримувати багатокрокові команди, адаптацію до попередніх запитів і можливість працювати зі змішаними сценамі використання, коли частина обчислень виконується локально, а частина – із залученням хмарної моделі. Однією з ключових вимог є можливість персоналізації, що включає адаптацію до голосових особливостей користувача, накопичення інформації про попередню поведінку та оптимізацію діалогових сценаріїв.

Нефункціональні вимоги визначають загальний рівень якості роботи системи. Одним із найважливіших критеріїв є стабільність, що передбачає

коректне реагування на зміну умов: різні рівні шуму, варіативність мови, швидку зміну команд та непередбачувані дії користувача. Система повинна працювати без зависань, втрати даних чи некоректної генерації відповідей. Особливе значення має зручність використання, яка включає швидку реакцію на команди, інтуїтивність взаємодії, а також відсутність складних налаштувань, які могли б ускладнювати користування.

Вимоги до продуктивності визначають оперативність виконання основних функцій. Розпізнавання мовлення має відбуватися в режимі, максимально наближеному до реального часу, а синтез відповіді не повинен створювати суттєвої затримки у взаємодії. Локальні алгоритми повинні працювати достатньо швидко навіть на комп’ютерах з базовими характеристиками, що забезпечить доступність системи для широкого кола користувачів. У випадку використання хмарних моделей необхідно мінімізувати обсяг даних, що передаються, забезпечуючи швидкий канал взаємодії між локальною системою та серверною частиною. Баланс між навантаженням на процесор, використанням пам’яті та якістю обробки є одним із ключових факторів проєктування.

Питання приватності відіграє особливо важливу роль у контексті роботи голосових систем. Оскільки голосові дані можуть містити конфіденційну інформацію, потребується забезпечення їх захисту під час зберігання та передавання. Вимоги передбачають можливість локальної обробки аудіосигналів без передачі їх на сторонні сервери, а також використання хмарної обробки лише в тих випадках, коли це необхідно для виконання складних запитів. Користувач повинен мати повний контроль над тим, які дані збираються, зберігаються та передаються, а також мати можливість очищати історію взаємодії.

Вимоги до сумісності охоплюють адаптацію системи під різні апаратні конфігурації та операційні платформи. Асистент має коректно працювати на персональних комп’ютерах з різними версіями Windows, підтримувати сучасні стандарти роботи з мікрофонами та аудіопристроями, а також інтегруватися з популярними бібліотеками штучного інтелекту. Система повинна забезпечувати

можливість підключення до різних мовних моделей, підтримувати оновлення алгоритмів і конфігурацій, не потребуючи повного переінсталювання. Це дозволяє створити гнучкий, розширюваний продукт, придатний для подальшого масштабування.

Формалізація вимог забезпечує чітку структуру для подальшого етапу розробки, дозволяє визначити межі проєкту, очікувану якість роботи та технічні характеристики системи. Усі зазначені вимоги формують основу для побудови надійного, безпечного та ефективного гібридного голосового асистента, здатного працювати в умовах реальних потреб користувача.

2.4. Обмеження та припущення

Розроблення гібридного голосового асистента передбачає врахування низки технічних і концептуальних обмежень, які визначають межі функціонування системи та впливають на кінцеву якість її роботи. Оскільки архітектура проєкту поєднує локальну обробку даних та хмарні інтелектуальні сервіси, низка припущень та технічних умов стає критично важливою для прогнозування продуктивності, надійності й реалістичності впровадження.

Першим ключовим обмеженням є апаратні можливості локальної частини системи. Модулі, відповідальні за приймання звуку, первинну обробку аудіосигналу, детекцію мовлення та базові функції розпізнавання, мають працювати без затримок і підтримувати обробку сигналу в реальному часі. Проте продуктивність таких модулів залежить від технічних характеристик комп’ютера користувача, зокрема потужності центрального процесора, швидкості оперативної пам’яті та можливостей аудіопідсистеми. Навіть оптимізовані моделі можуть суттєво втрачати швидкодію на слабкому обладнанні, що впливає на якість розпізнавання та реакцію системи. У рамках проєкту припускається, що користувач має комп’ютер із базовим набором сучасних характеристик, який дозволяє виконувати локальне декодування аудіосигналу з прийнятною затримкою.

Другим важливим обмеженням є залежність хмарних компонентів від стабільності та швидкості мережевого з’єднання. Великі мовні моделі, що відповідають за контекстний аналіз, генерацію природної відповіді та підтримання діалогової логіки, працюють у віддалених обчислювальних середовищах. Передавання текстових або аудіофрагментів до хмари неминуче супроводжується затримками, які можуть збільшувати загальний час реакції системи. Нестабільні мережеві умови, перевантаження каналів або обмежена пропускна здатність здатні спричинити збій або спотворення результатів. У процесі проєктування було прийнято припущення про наявність у користувача стійкого доступу до мережі з мінімальними коливаннями швидкості.

Ще одне припущення стосується доступності хмарних API та інтерфейсів, які використовуються для інтеграції великих мовних моделей. Їхня робота залежить від зовнішніх факторів, включаючи політику провайдерів, обмеження щодо кількості запитів, часові вікна технічного обслуговування та можливі зміни в специфікації. Не менш суттєвим є обмеження, пов’язане з багатомовною підтримкою та особливостями мовлення користувачів. Робота системи передбачає обробку мовлення українською та англійською мовами, проте мовні моделі не завжди однаково точно розпізнають рідкісні діалекти, специфічну вимову або змішані фрази.

Окремої уваги потребує питання приватності та безпеки, адже пересилання даних у хмару створює потенційні ризики несанкціонованого доступу або витоку конфіденційної інформації. У рамках проєкту враховується, що користувач розуміє обмеження обраної архітектури і погоджується на обробку частини даних у віддалених серверах, а самі сервіси дотримуються сучасних стандартів шифрування та протоколів безпеки. Нижче подана таблиця 2.1, що подає обмеження системи, їхній вплив та можливі способи мінімізації ризиків.

Таблиця 2.1 – Обмеження та способи мінімізації

Обмеження	**Вплив на систему**	**Способи мінімізації**
Обмежена	Збільшення затримки під час	Оптимізація локальних моделей;

потужність локального обладнання	обробки аудіосигналу; погіршення точності розпізнавання; можливі збої роботи	застосування попередньої обробки звуку; рекомендації мінімальних системних вимог
Нестабільність мережевого з’єднання	Затримка відповіді LLM; розриви сесії; неповна передача даних	Використання буферизації; асинхронні запити; кешування попередніх відповідей; частковий офлайн-режим
Обмеження хмарних API (ліміти, зміни політик, технічні паузи)	Тимчасова недоступність моделей; помилки при виконанні складних запитів	Альтернативні провайдери; автоматичне перемикання між моделями; локальні fallback-модулі для базових команд
Недостатня багатомовна підтримка моделей	Неправильне розпізнавання діалектів; помилки у змішаному мовленні; зниження якості NLP	Розширення мовних датасетів; додаткове донавчання локальних моделей; адаптація під акценти
Ризики порушення конфіденційності під час передавання даних у хмару	Можливий витік конфіденційної інформації; зниження довіри користувача	Шифрування запитів; локальна обробка чутливої інформації; повідомлення користувача про політику обробки даних
Обмеження швидкодії великих мовних моделей	Затримки генерації відповіді; зниження природності діалогу	Стримування довжини запитів; застосування компактних LLM; гібридна схема з попереднім NLP-фільтруванням

Загалом система будується з урахуванням того, що локальні обчислення забезпечують швидкість та приватність, а хмарні – інтелектуальність та гнучкість. Проте компромісний характер гібридної архітектури зумовлює наявність часових затримок, залежність від зовнішніх провайдерів і потребу в мінімальних технічних характеристиках обладнання. Ці припущення формують реалістичні межі проєкту та визначають напрямки подальших удосконалень, пов’язаних з оптимізацією локальних нейронних моделей, зменшенням чутливості до якості мережі та розширенням багатомовної підтримки.

2.5 Висновки до розділу 2

У результаті проведеного системного аналізу було визначено основні характеристики, вимоги та обмеження, що формують концептуальну основу гібридного голосового асистента. Досліджено об’єкт і предмет системи, охарактеризовано функціональну структуру сучасних голосових технологій та

окреслено ключові аспекти взаємодії локальних алгоритмів розпізнавання мовлення з інтелектуальними можливостями великих мовних моделей.

Постановка та обґрунтування проблеми засвідчили, що наявні голосові асистенти мають суттєві недоліки, пов’язані з контекстністю, персоналізацією, залежністю від хмарних сервісів та недостатньою гнучкістю у професійному середовищі. Саме ці обмеження створюють потребу у гібридному підході, що поєднує переваги локальної та хмарної обробки, забезпечуючи баланс між швидкістю, безпекою та інтелектуальною глибиною діалогу.

У процесі формалізації вимог до системи було визначено набір функціональних і нефункціональних критеріїв, які регулюють роботу майбутнього асистента: режим реального часу, підтримка діалоговості, багатомовність, адаптивність, стійкість до шумів, приватність даних та сумісність із широким спектром апаратних конфігурацій. Вимоги створюють чітку рамку для подальшого проєктування та забезпечують однозначність у визначенні очікуваних можливостей системи.

Аналіз обмежень та припущень дозволив сформувати реалістичні межі проєкту та визначити фактори, що впливають на продуктивність і стабільність системи.

Отже, розділ 2 закладає фундамент для подальшої розробки гібридного голосового асистента, визначаючи його структурні обмеження, вимоги, потенційні проблеми та ключові напрями оптимізації. Отримані результати слугують основою для переходу до етапу моделювання та побудови архітектури системи у наступному розділі.

РОЗДІЛ 3. МЕТОДИ ТА ІНСТРУМЕНТИ ДОСЛІДЖЕННЯ

3.1 Вибір та обґрунтування методів вирішення проблеми

В основу розробки гібридного голосового асистента покладено комплекс методів штучного інтелекту та обробки природної мови. Основними підходами є

глибоке навчання, використання великих мовних моделей та комбіноване навчання із залученням локальної обробки даних і хмарних сервісів.

Першим ключовим методом є розпізнавання мовлення (Speech-to-Text, STT). Воно забезпечує перетворення аудіосигналу користувача у текстові команди, що є необхідною умовою для подальшого аналізу запиту. Для цього використовуються нейронні мережі, зокрема рекурентні та трансформерні архітектури, які дозволяють моделі враховувати не лише слова, а й контекст попередніх висловлювань, інтонаційні особливості та темп мовлення. Такий підхід забезпечує точне розпізнавання навіть за шуму, швидкого мовлення або нестандартних акцентів.

Другим методом є обробка природної мови (Natural Language Processing, NLP), яка дозволяє системі інтерпретувати значення отриманого тексту та формулювати відповідь або виконати команду. Для цього застосовуються великі мовні моделі, такі як GPT або BERT, які навчаються на величезних корпусах текстів і здатні розпізнавати семантичні зв’язки між словами та фразами. NLP дозволяє асистенту враховувати контекст, розпізнавати наміри користувача та генерувати більш природні, логічні відповіді.

Третім методом є адаптивне навчання та персоналізація. Воно передбачає накопичення інформації про поведінку користувача, його словниковий запас, стиль спілкування та індивідуальні переваги. Використання методів машинного навчання, включаючи моделі на основі нейромереж, дозволяє асистенту підлаштовуватись під користувача, підвищуючи точність розпізнавання команд та ефективність виконання завдань.

Четвертим методом є комбінований підхід із локальною та хмарною обробкою даних. Локальна обробка забезпечує швидку реакцію на прості команди та гарантує безпеку обробки конфіденційних даних. Хмарні сервіси, навпаки, дозволяють здійснювати більш складні операції, такі як генерація відповідей із застосуванням великих мовних моделей, аналіз контексту або багатомовна

обробка. Об’єднання цих підходів дає змогу створити ефективний та безпечний гібридний голосовий асистент, здатний до роботи в різних сценаріях.

П’ятий метод пов’язаний із синтезом мовлення (Text-to-Speech, TTS). Він забезпечує генерацію природного звукового сигналу на основі текстової відповіді системи. Використання сучасних TTS-фреймворків, що враховують інтонацію, тембр та ритм мовлення, підвищує якість комунікації та робить взаємодію користувача з системою максимально природною.

Шостий метод включає аналітичну обробку запитів і передбачення дій користувача. На основі накопичених даних та моделей машинного навчання система здатна прогнозувати наміри користувача, наприклад, передбачати, які дії будуть виконуватися наступними, або пропонувати оптимальні рішення для конкретного сценарію. Це підвищує ефективність роботи асистента та робить взаємодію більш проактивною.

Усі зазначені методи поєднуються в єдину архітектуру, де локальні алгоритми швидко обробляють прості команди, хмарні моделі забезпечують складний аналіз і генерацію відповідей, а модулі персоналізації та прогнозування оптимізують поведінку системи для конкретного користувача. Такий комплексний підхід дозволяє створити гнучкий, адаптивний та безпечний голосовий асистент для персональних комп’ютерів.

3.2 Вибір та обґрунтування засобів вирішення проблеми

Для створення повноцінного голосового асистента на персональному комп’ютері необхідно застосувати комплекс взаємопов’язаних технологій, методів і програмних засобів, що забезпечують можливість сприйняття людського мовлення, його обробку, інтерпретацію змісту команд, формування відповідей і подальше виконання необхідних дій у системі. Таке рішення поєднує компоненти, пов’язані з розпізнаванням голосу, синтезом мовлення, штучним інтелектом для аналізу запитів, а також інструменти для інтеграції з операційною системою

комп’ютера. Вибір технологій має вирішальне значення для забезпечення швидкодії, точності розпізнавання та природності спілкування між користувачем і програмою.

Основним інструментом розроблення виступає мова програмування Python, яка є однією з найзручніших і найпоширеніших серед фахівців з машинного навчання та штучного інтелекту. Вона відома своєю універсальністю, простою структурою синтаксису й великим набором бібліотек, орієнтованих на роботу з мовними технологіями. Python дозволяє легко інтегрувати алгоритми штучного інтелекту, взаємодіяти з API сторонніх сервісів, реалізовувати голосове введення та виведення, а також керувати системними процесами. Завдяки відкритості екосистеми та активній підтримці спільноти розробників ця мова є оптимальним вибором для створення голосових асистентів будь-якого рівня складності.

У межах дослідження передбачено використання системи Gemini – інтелектуального чат-бота від компанії Google, який раніше мав назву Google Bard. Вона забезпечує можливість аналізу змісту текстових запитів, розпізнавання намірів користувача та формування змістовних відповідей. [1] Завдяки інтеграції з Gemini голосовий асистент отримує змогу вести логічні, осмислені діалоги, що робить взаємодію з користувачем більш природною. Цей компонент суттєво розширює межі функціональності системи, оскільки дозволяє не просто реагувати на команди, а й розуміти контекст розмови, формулювати уточнення й навіть прогнозувати потреби користувача на основі попередніх звернень.

Для перетворення голосових сигналів у текстову форму застосовується бібліотека SpeechRecognition – один із найпопулярніших інструментів Python для розпізнавання мовлення. [2] Вона підтримує роботу з численними зовнішніми сервісами, такими як Google Speech API, CMU Sphinx, IBM Speech to Text, Microsoft Azure Speech тощо. Використання цієї бібліотеки дає змогу системі сприймати усні команди та перетворювати їх у текст, що надалі аналізується штучним інтелектом. Висока точність обробки мовлення досягається завдяки

використанню хмарних алгоритмів і можливості навчання на основі накопичених даних.

Не менш важливим є процес синтезу мовлення – зворотного перетворення текстової відповіді в аудіоформат. Для цього застосовуються бібліотеки pyttsx3 та gTTS (Google Text-to-Speech). Pyttsx3 надає можливість офлайн-синтезу голосу, що особливо корисно у випадках, коли пристрій не має стабільного підключення до мережі. Крім того, користувач може змінювати параметри голосу, швидкість, тембр та інтонацію. GTTS, навпаки, базується на хмарних технологіях Google, що забезпечує більш реалістичне та природне звучання. Завдяки поєднанню обох рішень можна досягти балансу між швидкодією та якістю звуку, роблячи асистента максимально комфортним у сприйнятті.

Для взаємодії з операційною системою використовуються стандартні бібліотеки Python – OS та Subprocess. Вони відкривають доступ до системних процесів, дозволяючи програмі запускати програми, виконувати команди, керувати файлами, переглядати каталоги або навіть контролювати мультимедійні засоби. Так, наприклад, користувач може попросити асистента відкрити браузер, запустити музику, створити новий документ або перевірити інформацію про систему. [3] Ця інтеграція забезпечує не лише зручність, але й створює передумови для автоматизації численних повсякденних завдань.

Окреме значення має модульна логіка чат-бота, яка визначає, яким чином асистент інтерпретує текстові запити, розпізнає контекст та обирає відповідну реакцію. Цей компонент формує основу інтелектуальної поведінки системи, роблячи її здатною розпізнавати різні типи запитів, підтримувати діалог, а також накопичувати досвід для подальшого вдосконалення. Поєднання таких можливостей створює ефект «живої» взаємодії між користувачем і комп’ютером, що значно підвищує ефективність використання асистента у щоденній роботі.

Комплексне застосування описаних технологій дозволяє створити повнофункціональний голосовий асистент, який не лише розпізнає мовлення, але й здатний самостійно синтезувати відповіді, аналізувати команди, обробляти

запити та взаємодіяти з операційною системою. Завдяки використанню штучного інтелекту система стає адаптивною та гнучкою, поступово вдосконалюючись із кожною новою взаємодією.

У межах дослідження було визначено, що ефективність роботи голосового асистента безпосередньо залежить від гармонійного поєднання інструментів і методів. Python, як основа проєкту, забезпечує легкість розроблення, а такі бібліотеки, як SpeechRecognition, pyttsx3 та gTTS [4], формують базу мовної взаємодії. Інтеграція з Gemini додає системі інтелектуальної глибини, а використання OS і Subprocess відкриває можливості повного контролю над середовищем комп’ютера. Водночас логіка чат-бота гарантує природність і послідовність діалогу.

Таким чином, реалізація голосового асистента на основі вказаних технологій дозволяє створити гнучку та ефективну систему, здатну не лише виконувати команди, а й аналізувати контекст, навчатися на основі досвіду користувача та пропонувати індивідуалізовані рішення. Поєднання можливостей штучного інтелекту, інструментів синтезу та розпізнавання мовлення, а також механізмів системної інтеграції створює підґрунтя для формування інтелектуального помічника нового покоління, який поступово стає невід’ємним елементом сучасного цифрового середовища.

3.3. Архітектурні рішення

Архітектура розробленої системи побудована відповідно до принципів модульного поділу функцій та логічного розшарування обробки даних, що забезпечує розмежування відповідальностей, гнучкість розширення та можливість оптимізації окремих модулів без порушення загальної структури. Основою функціонування програмного забезпечення є SLU-пайплайн, який охоплює послідовність етапів від розпізнавання мовлення до генерації відповіді. У типовому режимі роботи система проходить усі стадії: акустичне розпізнавання

висловлювання, інтерпретацію змісту, логічне виведення через мовну модель та формування підсумкової текстової або голосової відповіді.

Першим рівнем у цьому ланцюзі є ASR-компонент, який перетворює акустичний сигнал на текст. Його реалізація у проєкті ґрунтується на модулі HybridRecognizer, що використовує локальні Vosk-моделі та забезпечує безперервну або одноразову транскрипцію мовлення. Розпізнані фрази надходять у шар NLU, де проводиться аналіз змісту. У найбільш простому випадку цей етап полягає у виявленні ключових слів, що сигналізують про намір користувача. Для цього застосовується механізм WakeWordListener. У більш складних сценаріях NLU включає семантичне порівняння тексту із попередньо згенерованими ембеддингами команд, що дає змогу відокремити значення висловлювання від його буквальної форми.

Після тлумачення висловленої інтенції система переходить до рівня логічного виведення, де задіюється LLM-модуль. Він використовується у тих випадках, коли локальний інтелектуальний шар не може знайти відповідну команду, або коли користувач формує складні запити, що вимагають генеративного опрацювання. У таких ситуаціях запит передається до зовнішньої великої мовної моделі, яка формує найбільш релевантну відповідь.

Завершальний етап технологічного ланцюга утворюють NLG та TTS. Після отримання змістовної відповіді система трансформує її у зв’язну текстову форму, а потім за потреби синтезує мовленнєвий аудіосигнал. Таким чином створюється голосова реакція, що передається користувачеві, або текстовий результат, що повертається в інтерфейс.

Загальна архітектура є гібридною. Гібридні архітектури, що розподіляють обробку між пристроєм і хмарою, демонструють суттєве зниження затримок у системах розмовного штучного інтелекту. [11] Вона поєднує локальні модулі, які працюють автономно на пристрої, із зовнішніми компонентами, що викликаються лише тоді, коли локальних можливостей недостатньо. Локальна частина включає модулі ASR, VAD, механізм активації за ключовими словами та NLU-аналіз на

основі ембеддингів. Ці компоненти працюють без використання інтернет-з’єднання й забезпечують оперативну відповідь для стандартних команд. Зовнішня частина – це підсистема запитів до LLM, яка застосовується лише у складних випадках або коли зміст висловлювання не збігається з жодним із відомих сценаріїв. Такий підхід значно скорочує витрати ресурсів, зменшує час реакції та водночас забезпечує високу гнучкість системи.

Для наочності було сформовано умовну блок-схему (рисунок 3.1), що відображає загальний порядок переходу даних між основними компонентами системи:

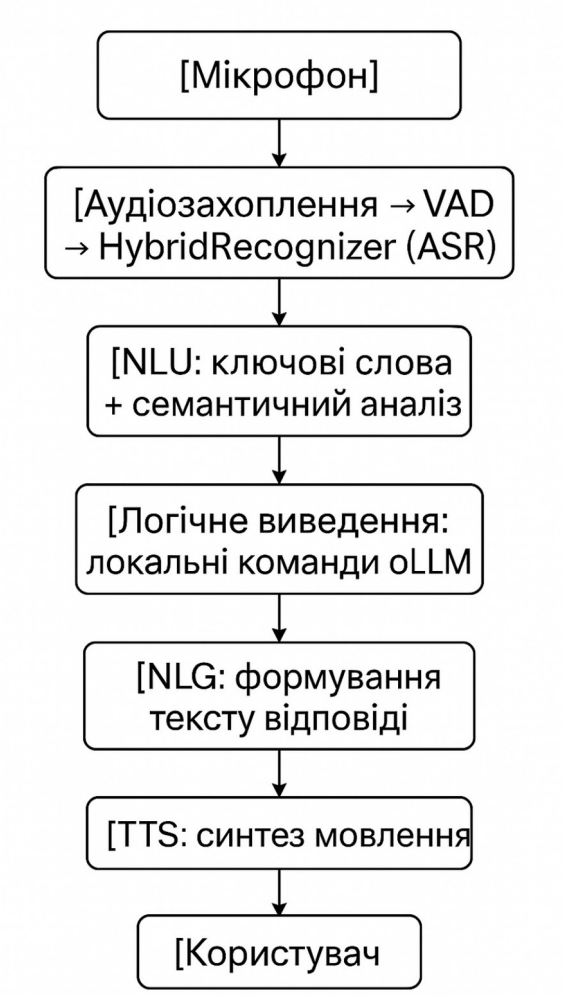

Рис. 3.1 – Загальний порядок переходу даних

У структурному плані архітектура складається з кількох взаємопов’язаних блоків, між якими дані передаються послідовно. На вході знаходиться модуль акустичного захоплення, який взаємодіє із HybridRecognizer. Далі від

транскрибованого тексту інформація проходить до блоку NLU, що містить як механізм виявлення ключових слів, так і семантичний аналізатор. Наступним етапом виступає блок логічного виведення, де обробка здійснюється або локально, або із залученням зовнішньої моделі.

Після цього результат переходить у блок генерації текстової та голосової відповіді, а потім передається вихідним каналам системи.

Таким чином розроблена архітектура забезпечує цілісний ланцюг перетворення мовлення користувача із використанням як локальних ресурсів, так і високорівневих генеративних моделей, що дозволяє поєднати ефективність, автономність та інтелектуальну гнучкість у межах однієї системи.

3.4 Визначення сценаріїв використання розробленого асистента

Розроблений гібридний голосовий асистент орієнтований на широке коло застосувань, що охоплюють як побутові задачі, так і професійні сфери діяльності. Завдяки поєднанню локальних алгоритмів розпізнавання мовлення та інтелектуальних можливостей великих мовних моделей система здатна підтримувати природний діалог, адаптуватися до особливостей користувача та виконувати комплексні команди з високим рівнем точності.

Це визначає можливість застосування асистента у різних середовищах, де важливими є швидкість взаємодії, зручність користування та здатність працювати в умовах обмеженого доступу до мережі.

Одним із ключових сценаріїв використання є інтелектуальна підтримка користувача під час роботи за персональним комп’ютером. Асистент може забезпечувати голосове керування операційною системою, відкривати програми, шукати документи, створювати нагадування, запускати медіафайли та виконувати інші повсякденні операції. Цей сценарій особливо актуальний для користувачів, які прагнуть зменшити кількість ручних дій, підвищити продуктивність або працюють у режимі багатозадачності. Використання голосових команд дозволяє

зосередитись на основних видах діяльності, мінімізуючи час, витрачений на навігацію між інструментами (рисунок 3.2).

Рис. 3.2 – Загальна діаграма варіантів використання

Гібридний асистент може також застосовуватися в умовах навчання та дослідницької роботи. Система здатна відповідати на інформаційні запити, пояснювати складні теми, узагальнювати текстові матеріали, формувати довідкові пояснення та супроводжувати користувача у процесі вивчення нових знань. Підтримка контекстних діалогів дозволяє будувати тривалі ланцюги взаємодії, у межах яких асистент переходить від базових пояснень до глибшого аналізу

змісту. Це робить систему ефективним інструментом для студентів, викладачів, дослідників та всіх, хто працює з великими обсягами матеріалу.

Особливе значення має застосування асистента у сфері інклюзивних технологій. Користувачі з обмеженою можливістю роботи руками або порушеннями зору можуть застосовувати голосові команди як основний засіб взаємодії з ПК. Гібридна модель забезпечує високу точність інтерпретації усного мовлення та можливість офлайн-роботи, що дозволяє застосовувати асистента навіть у ситуаціях із нестабільним доступом до інтернету. Це створює нові можливості для соціальної адаптації, підвищення автономності та доступності цифрових ресурсів для людей з особливими потребами.

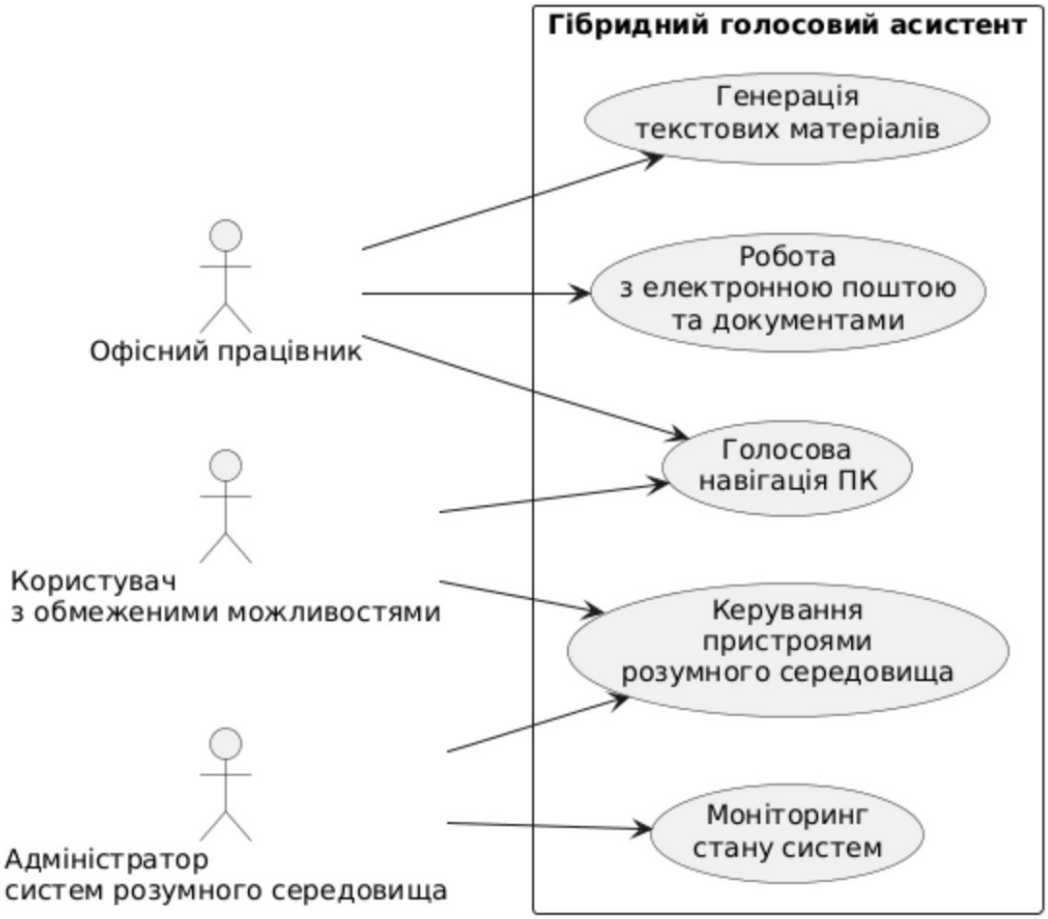

Рис. 3.3 – Діаграма варіантів використання з інтеграцією у офісну роботу

Асистент може бути інтегрований у професійні середовища (рисунок 3.3), зокрема у сферу офісної роботи, де необхідні швидкі дії з документами, поштою

та інформаційними системами. Його функції дозволяють скоротити кількість повторюваних операцій, оптимізувати управління даними та зменшити навантаження на співробітників. У складніших сценаріях система може формувати аналітичні звіти, обробляти текстові документи, здійснювати резюмування великих масивів інформації та брати участь у підготовці контенту.

Завдяки застосуванню великих мовних моделей асистент придатний для сценаріїв творчої взаємодії. Він може генерувати тексти, формувати ідеї для проєктів, адаптувати стиль написання та допомагати користувачу у створенні складних матеріалів. Такий функціонал відкриває можливості для широкого застосування у сфері маркетингу, журналістики, копірайтингу або розробки навчального контенту.

Окремим напрямом є використання асистента для автоматизації складних процесів у системах розумного середовища. За умови відповідної інтеграції він може виконувати роль інтерфейсу управління різними пристроями, здійснювати моніторинг систем, реагувати на зміну параметрів і допомагати в управлінні середовищем у режимі реального часу. Гібридний характер архітектури забезпечує можливість виконання базових команд локально, тоді як інтелектуальні сценарії можуть опрацьовуватися у хмарі. Усі визначені сценарії свідчать про універсальність запропонованої системи та її здатність адаптуватися до різних потреб користувачів.

3.5 Висновки до розділу 3

У межах третього розділу було визначено методологічну основу проєкту та обґрунтовано вибір інструментів, необхідних для створення гібридного голосового асистента.

У першому підрозділі обґрунтовано застосування ключових методів: автоматичного розпізнавання мовлення, обробки природної мови, синтезу голосу, механізмів персоналізації та гібридної (локально-хмарної) обробки. Показано, що

саме їх синергія забезпечує точність, адаптивність та інтелектуальність діалогової взаємодії.

У другому підрозділі наведено вибір технологій та інструментів, що стали основою програмної реалізації. Python визначено базовою мовою розробки, а бібліотеки SpeechRecognition, pyttsx3, gTTS, а також інтеграція з системою Gemini ключовими компонентами для забезпечення розпізнавання, синтезу та семантичної обробки мовлення. Підкреслено важливість використання стандартних інструментів OS та Subprocess для взаємодії з операційною системою та автоматизації користувацьких дій.

Архітектурні рішення, розглянуті в підрозділі 3.3, демонструють логічну структуру системи, що складається з модулів ASR, NLU, LLM-обробки, а також механізмів генерації тексту й мовлення.

У підрозділі 3.4 описано практичні сценарії використання асистента, що підтверджують універсальність розробленої системи. Асистент може ефективно працювати у побутових, професійних, навчальних і інклюзивних середовищах. Він підтримує виконання команди на ПК, роботу з документами, пошук інформації, створення контенту, а також може застосовуватися у сфері автоматизації офісних процесів та розумних середовищ.

Таким чином, третій розділ формує комплексну методологічну й технічну основу для практичної реалізації гібридного голосового асистента. Обрані методи та інструменти є узгодженими, взаємодоповнювальними і придатними для побудови системи, здатної працювати у режимі реального часу, адаптуватися до користувацьких потреб і забезпечувати високий рівень якості голосової взаємодії.

РОЗДІЛ 4. ПРАКТИЧНА РЕАЛІЗАЦІЯ ЗАВДАННЯ ТА РЕЗУЛЬТАТИ РОЗРОБКИ

4.1 Опис структури проекту

Під час реалізації практичної частини проєкту було створено функціональний прототип голосового асистента з гібридною архітектурою, який поєднує розпізнавання мовлення, інтелектуальний аналіз команд і взаємодію з операційною системою персонального комп’ютера.

У структурі застосунку HybridRecognizer є центральним елементом логіки розпізнавання, оскільки він формує текстові дані на основі звукової інформації. WakeWordListener залежить від HybridRecognizer і використовує його потокові транскрипції для аналізу. AudioManager функціонує як альтернативне джерело аудіопотоку, яке може або замінити внутрішній механізм HybridRecognizer, або бути інтегроване з ним у майбутньому, якщо буде проведена уніфікація контролю за аудіопристроєм.

Логічний зв’язок між модулями є прямим: HybridRecognizer взаємодіє лише з аудіопристроєм, WakeWordListener – лише з HybridRecognizer, тоді як AudioManager наразі не використовується іншими елементами і не включений у основний робочий цикл. Потоки виконання всередині системи є паралельними: HybridRecognizer і WakeWordListener працюють у власних фонових потоках, що забезпечує безперервність обробки.

Логічне виведення у системі здійснюється у двох незалежних шарах. Перший шар – локальний, побудований на основі обробки транскрибованого тексту. Саме тут працює WakeWordListener, який аналізує текстові результати й визначає, чи містять вони ключові слова, що слугують тригером для виконання команд. Накопичення даних відбувається у вигляді збирання транскрипцій, що надходять у реальному часі, після чого здійснюється їх порівняння зі списком ключових фраз.

Другий шар – загальна логіка системи, описана у проєктному документі. Вона використовує семантичні ембеддинги для пошуку найбільш релевантних команд та звертається до мовної моделі, лише якщо локальні засоби не дали

результату. Проміжні дані накопичуються в індексах ембеддингів. Логічне виведення ґрунтується на ступені близькості між запитом користувача та описами команд. Вихідним результатом є або знайдений скрипт, або відповідь, отримана від великої мовної моделі.

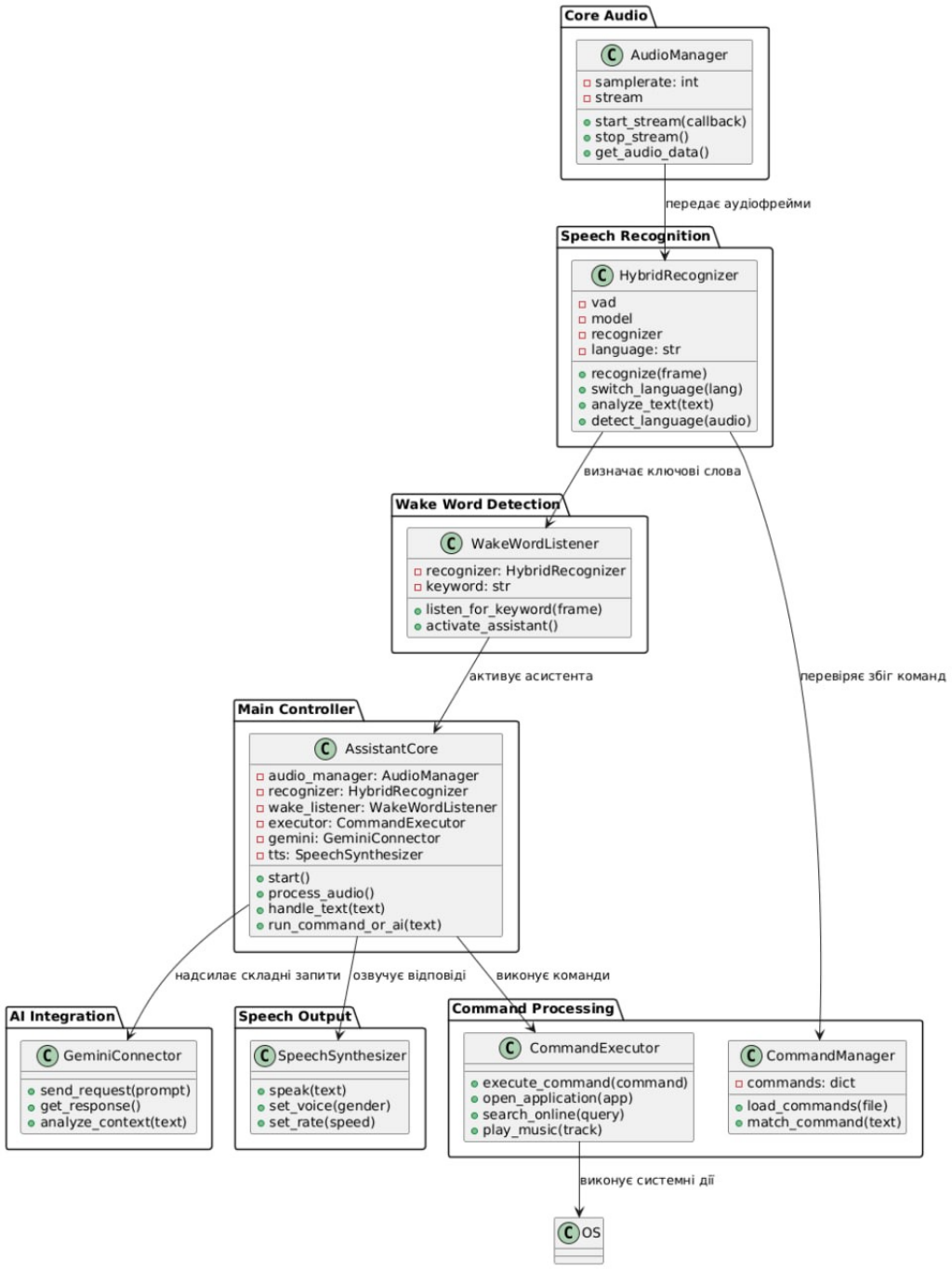

Рис. 4.1 – Діаграма класів проекту

Основна мета цього рішення полягала у створенні системи, здатної приймати голосові запити користувача, розпізнавати їх у режимі реального часу,

аналізувати зміст висловлювань, визначати намір і формувати відповідну реакцію – від виконання локальної команди до звернення до мовної моделі Gemini для генерації осмисленої відповіді.

На діаграмі (рисунок 4.1) зображено архітектуру голосового асистента, побудовану на основі об’єктно-орієнтованого підходу. Центральним елементом системи є клас AssistantCore, який координує роботу всіх інших компонентів. Він використовує модулі для захоплення аудіо, розпізнавання мовлення, визначення ключових слів, виконання команд, генерації мовного виходу та інтеграції з AI.

Компонент Core Audio, представлений класом AudioManager, відповідає за отримання звукових даних із мікрофона. Він має параметри частоти дискретизації, потоків аудіо та методи для запуску й зупинки потоку, а також передачі аудіофреймів іншим компонентам. Дані з AudioManager надходять до модуля Speech Recognition, реалізованого через клас HybridRecognizer. Цей клас використовує модель мовлення та розпізнавач для аналізу аудіо, визначення мови та перетворення мовлення у текст.

Далі, модуль Wake Word Detection, який реалізовано класом WakeWordListener, аналізує потік аудіо для виявлення ключового слова – команди активації асистента. Після розпізнавання цього слова активується AssistantCore, який починає обробку користувацьких запитів.

Клас AssistantCore взаємодіє з кількома підсистемами. Він передає текстові запити в AI Integration через клас GeminiConnector, який надсилає запит до мовної моделі, отримує відповідь і аналізує контекст. Для відтворення голосових відповідей використовується Speech Output, представлений класом SpeechSynthesizer, що може озвучувати текст та змінювати швидкість мовлення.

Для виконання системних дій, таких як запуск програм, пошук інформації чи відтворення музики, використовується модуль Command Processing, який складається з класів CommandExecutor та CommandManager. Перший виконує команди, другий завантажує доступні команди та визначає, чи збігається текст користувача з відомими шаблонами. У кінці ланцюга знаходиться елемент OS,

який безпосередньо виконує дії на рівні операційної системи. Таким чином, діаграма демонструє повний цикл роботи голосового асистента – від захоплення звуку та розпізнавання ключового слова до інтелектуального аналізу запиту, виконання команд і генерації мовних відповідей.

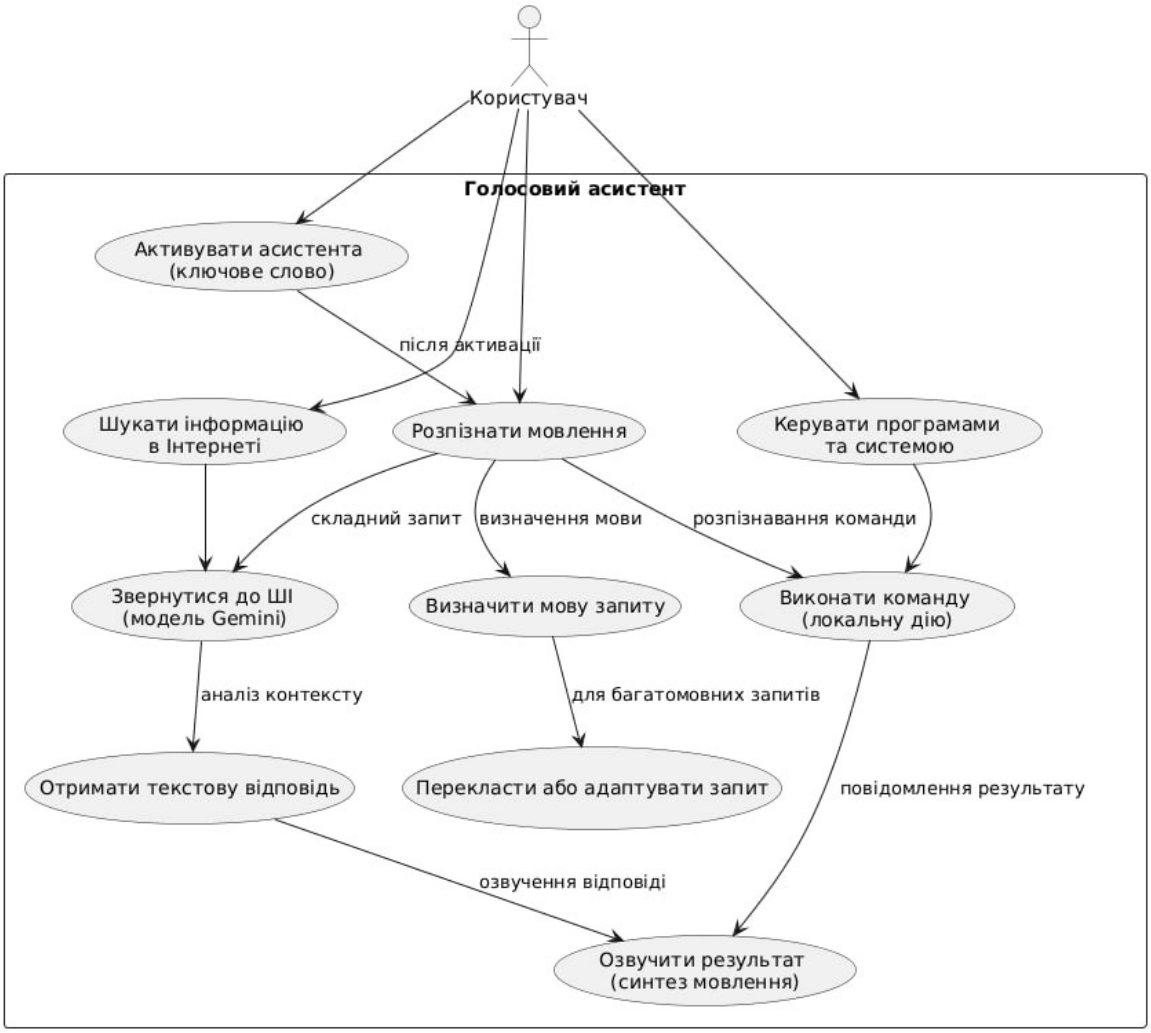

Рис. 4.2 – Детальна діаграма варіантів використання

Діаграма варіантів використання (рисунок 4.2) показує варіанти використання голосового асистента, який взаємодіє з користувачем у процесі розпізнавання мовлення, обробки запитів та виконання дій. У центрі сценарію стоїть актор – Користувач, який ініціює роботу системи за допомогою голосових команд. Першим кроком є активація асистента шляхом вимови ключового слова.

Після цього система переходить у стан активності та починає розпізнавання мовлення, тобто перетворює голос користувача на текст.

Якщо в запиті присутні команди, асистент може керувати програмами та системою, виконуючи локальні дії, такі як відкриття додатків або відтворення музики. У разі складних або інформаційних запитів система переходить до етапу пошуку інформації в Інтернеті. Для цього вона може звернутися до штучного інтелекту (моделі Gemini), який аналізує контекст запиту, визначає намір користувача й формує відповідь. Якщо запит містить елементи кількох мов, асистент спочатку визначає мову запиту, а потім може перекласти або адаптувати запит перед його обробкою.

Після того як відповідь сформована, користувач отримує текстову відповідь, яку асистент може озвучити за допомогою синтезу мовлення. Таким чином, користувач сприймає результат у природній голосовій формі.

Отже, діаграма демонструє послідовність взаємодії між користувачем і голосовим асистентом – від активації та розпізнавання мовлення до виконання команд, обробки запитів штучним інтелектом, перекладу, формування відповіді та її озвучення.

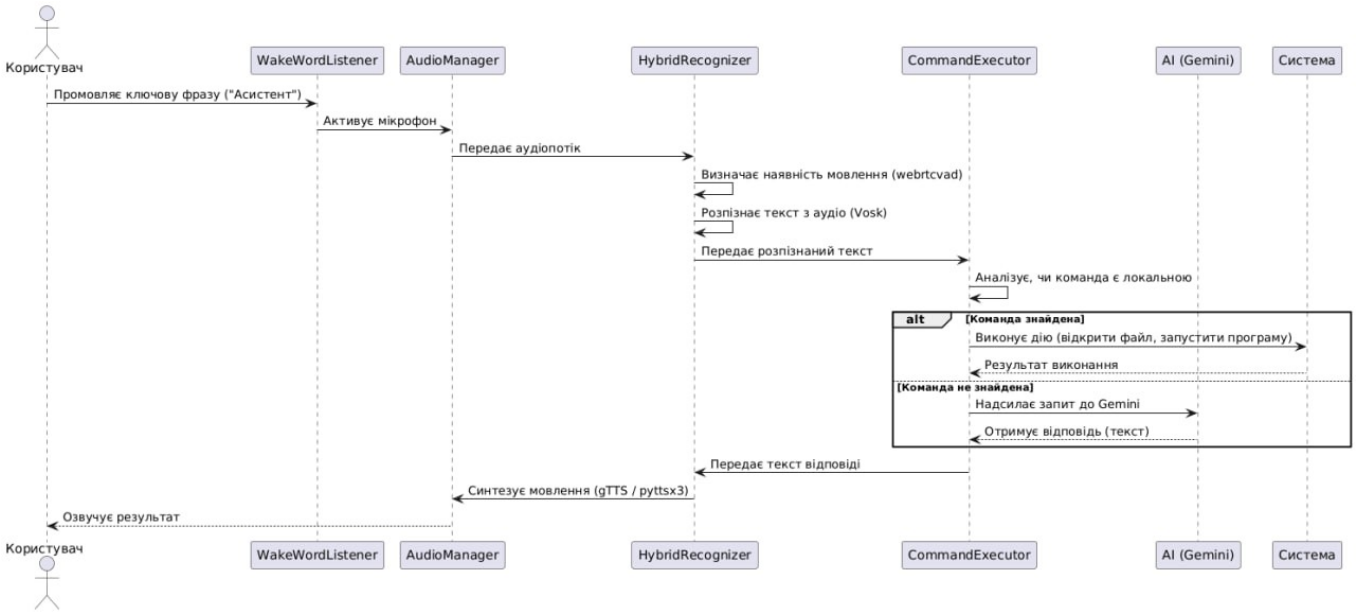

Рис. 4.3 – Діаграма послідовносетй

На діаграмі послідовностей (рисунок 4.3) зображено процес взаємодії користувача з голосовим асистентом від моменту активації до отримання результату. Учасниками взаємодії є Користувач, WakeWordListener, AudioManager, HybridRecognizer, CommandExecutor, AI (Gemini) та Система. Процес починається з того, що користувач промовляє ключову фразу, наприклад “Асистент”.

Компонент WakeWordListener виявляє цю фразу й активує мікрофон через модуль AudioManager. Після активації AudioManager починає передавати аудіопотік до модуля HybridRecognizer. Модуль HybridRecognizer спочатку визначає наявність мовлення за допомогою технології webrtcvad, щоб переконатися, що користувач говорить, а потім розпізнає текст із аудіо за допомогою системи Vosk. Після успішного розпізнавання він передає отриманий текст далі до компонента CommandExecutor. CommandExecutor аналізує текст і перевіряє, чи є запит локальною командою.

Якщо команда знайдена, система виконує дію, наприклад відкриває файл або запускає програму, після чого повертає результат виконання. Якщо ж команда не знайдена, CommandExecutor надсилає запит до моделі штучного інтелекту Gemini, яка аналізує запит і повертає текстову відповідь. Після отримання відповіді CommandExecutor передає її назад до HybridRecognizer, який запускає процес синтезу мовлення (за допомогою gTTS або pyttsx3), щоб озвучити результат користувачу. Наприкінці WakeWordListener відтворює отриману відповідь, і користувач чутиме озвучений результат.

Таким чином, діаграма ілюструє повний цикл роботи голосового асистента: від активації голосом, через розпізнавання та аналіз команди, до виконання дії або отримання відповіді від AI та озвучення результату.

4.2 Опис технологічного процесу обробки інформації

Технологічний процес обробки інформації у розробленому програмному забезпеченні являє собою послідовний ланцюг взаємопов’язаних етапів, у межах яких дані переходять від неструктурованого акустичного сигналу до осмисленої текстової форми, а надалі – до вибору та виконання відповідної реакції системи. Уся обробка починається з моменту активізації модулів аудіозахоплення. У цей момент формується безперервний потік коротких аудіофрагментів, що зчитуються з мікрофона в режимі реального часу. Кожен із таких фрагментів представляє собою набір кодованих значень амплітуди, які передаються у внутрішні буфери системи.

Після отримання чергової порції аудіоданих здійснюється їх первинне накопичення та упорядкування у вигляді послідовності кадрів стабільної тривалості. Водночас застосовується механізм визначення наявності мовлення, що дає змогу відокремити ділянки з корисною інформацією від акустичного фону. Якщо система працює у режимі одиничного розпізнавання, аналіз мовлення завершується в той момент, коли у потоці протягом визначеного інтервалу часу більше не спостерігається голосової активності.

Сформовані фрагменти надходять до модуля транскрибування, де проходять етап акустичного розпізнавання. На цьому рівні використовується попередньо завантажена мовна модель, що побудована на базі нейромережевих представлень. Вона порівнює спектральні характеристики вхідного сигналу з відомими мовними патернами і поступово генерує текстову інтерпретацію. Проміжні часткові варіанти транскрипції надсилаються до зовнішніх компонентів у міру уточнення, а фінальний результат фіксується після того, як модель завершує обробку поточного фрагмента.

Отриманий текст стає предметом подальшого логічного аналізу. На цьому етапі система перевіряє, чи містить розпізнана фраза визначені ключові слова, які відіграють роль активаторів. Якщо таке слово виявлено, система переходить у стан готовності до сприйняття команди або запускає заздалегідь визначений сценарій. Коли ж обробка включає елементи семантичного аналізу, текст

порівнюється з наявними описами команд за допомогою векторних представлень; у разі відсутності задовільного збігу система звертається до зовнішньої мовної моделі для отримання відповіді.

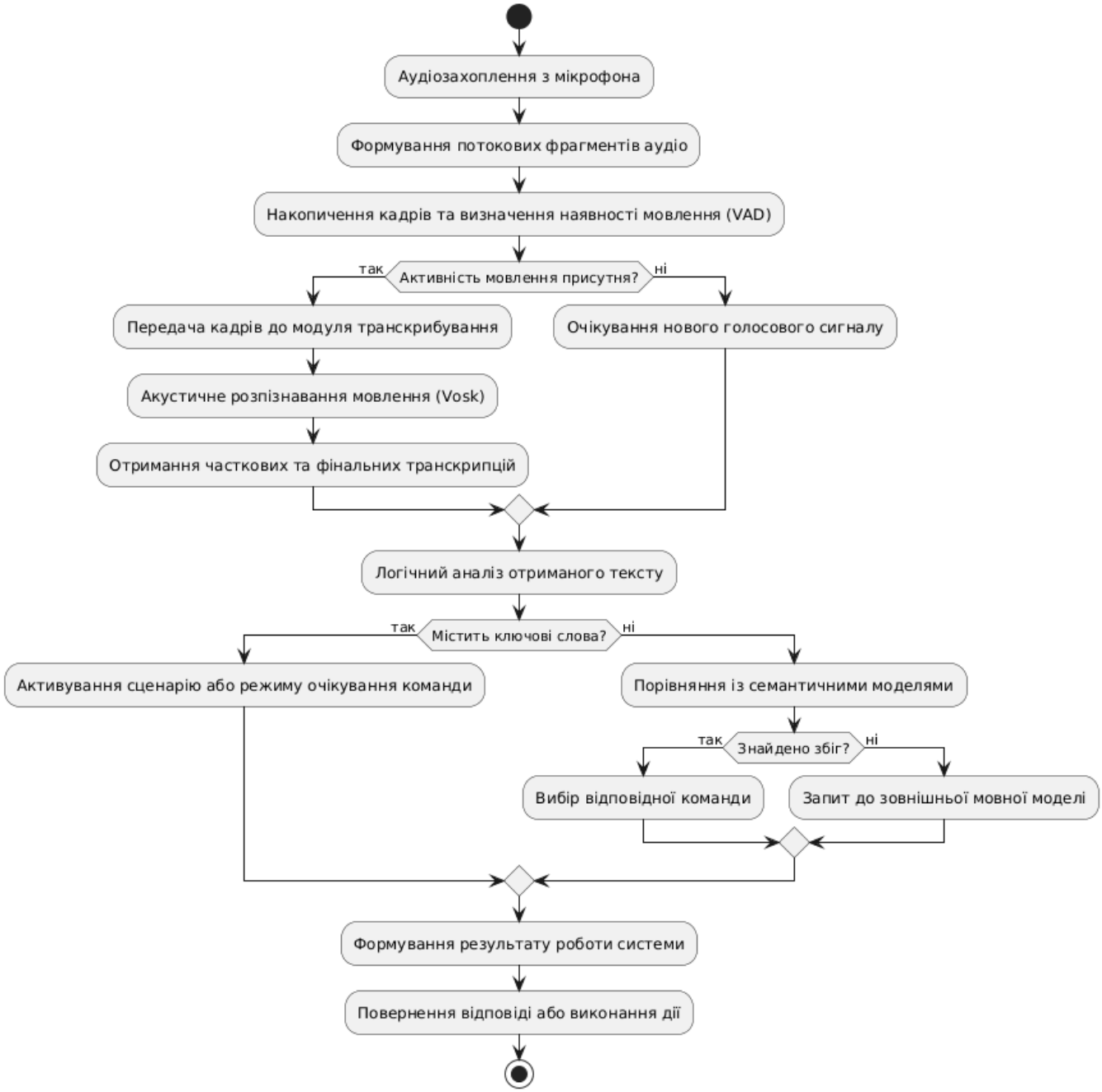

Рис. 4.4 – Діаграма опису процесу обробки інформації

Як показано на діаграмі (рисунок 4.4), завершальною частиною процесу є формування результату, який може набувати декількох форм залежно від

конкретної задачі: текстової відповіді, запуску внутрішньої функції, відтворення голосового повідомлення або зміни стану інтерфейсу. Усі результати передаються у вихідний модуль і можуть бути збережені для подальшого аналізу або статистичного обліку.

Таким чином, технологічний процес являє собою згруповану послідовність дій, що охоплює накопичення, структурування, перетворення, аналіз та логічне узагальнення даних. Зміст кожного етапу визначає цілісність і коректність роботи системи, а також забезпечує отримання кінцевого результату, який відповідає вимогам користувача.

4.3 Декомпозиція програмного продукту

Розроблений програмний комплекс голосового асистента має модульну архітектуру, що дає змогу розділити систему на окремі функціональні компоненти з чітко визначеними завданнями. Такий підхід полегшує супровід, тестування та подальше розширення проєкту, забезпечуючи гнучкість і масштабованість під час впровадження нових функцій.

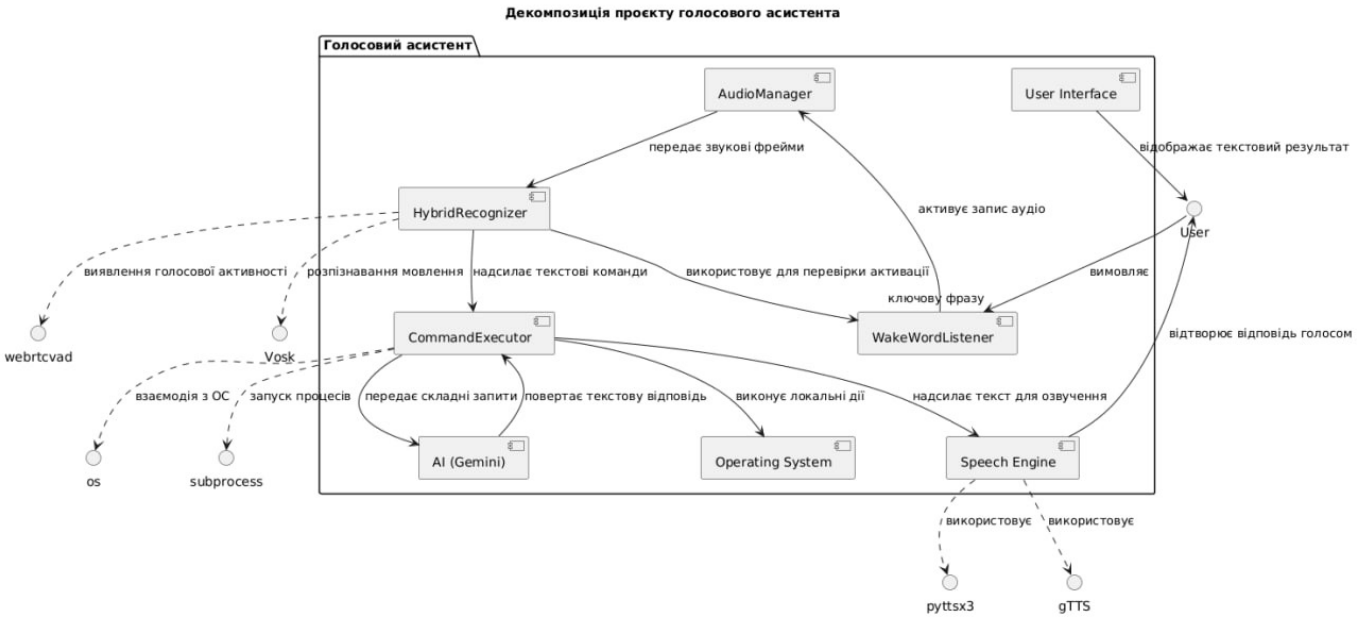

Рис. 4.5 – Діаграма компонентів

Основна ідея декомпозиції полягає (рисунок 4.5) у поділі системи на незалежні модулі, кожен з яких виконує конкретну роль у процесі взаємодії користувача з комп’ютером через голосові команди. Між цими компонентами налагоджено послідовний обмін даними, що забезпечує цілісність роботи асистента.

Ключовим елементом є модуль обробки аудіо (AudioManager), який відповідає за захоплення звукового сигналу з мікрофона. Він отримує аудіопотік у режимі реального часу та передає його в систему розпізнавання мовлення. Цей компонент працює у багатопотоковому режимі, забезпечуючи безперервне зчитування даних без втрати якості сигналу.

Наступний рівень утворює модуль гібридного розпізнавання (HybridRecognizer), який поєднує алгоритми Vosk і webrtcvad. Завдяки цьому реалізовано фільтрацію шумів, визначення мовної активності, ідентифікацію початку й завершення мовлення, а також перетворення аудіопотоку в текст. У структурі модуля передбачено механізм адаптації до вибору мови, що дає змогу асистенту розпізнавати команди українською та англійською мовами.

Третім компонентом є модуль активації асистента (WakeWordListener), який виконує роль тригера системи. Його завдання полягає у розпізнаванні спеціального ключового слова або фрази, після чого асистент переходить у режим активного прослуховування. Цей модуль мінімізує споживання ресурсів, адже програма не виконує повне розпізнавання, доки не буде виявлено активаційне слово.

Після розпізнавання фрази текст передається до модуля логічної обробки (CommandExecutor). Цей компонент аналізує отримані команди та визначає спосіб їх виконання. Якщо команда належить до локальних дій (наприклад, відкриття програми, відтворення музики чи створення файлу), система виконує її безпосередньо через операційну систему за допомогою стандартних бібліотек os і subprocess. У випадку, коли команда не належить до відомих, вона перенаправляється до модуля штучного інтелекту.

Інтелектуальний компонент реалізовано на основі мовної моделі Gemini, яка забезпечує глибоку семантичну обробку тексту, розуміння контексту та формування логічної відповіді. Ця частина системи відповідає за генерацію текстових реакцій, розширення знань асистента та підтримку діалогової взаємодії.

Завершальний етап обробки становить модуль синтезу мовлення, який реалізує голосове відтворення відповідей користувачеві. Для цього використовуються бібліотеки gTTS та pyttsx3, що дозволяють отримати природне звучання голосу та забезпечити офлайн-роботу системи. Таким чином, програмний продукт має п’ятирівневу структуру:

1. Аудіовхід – обробка звукового сигналу (AudioManager).
2. Розпізнавання мовлення – аналіз аудіопотоку та перетворення його в текст (HybridRecognizer).
3. Активація системи – запуск через ключову фразу (WakeWordListener).
4. Обробка команд – виконання інструкцій або звернення до штучного інтелекту (CommandExecutor, Gemini).
5. Вивід результату – голосова або текстова відповідь (Speech Engine).

Виконана декомпозиція забезпечує логічну узгодженість компонентів, спрощує модернізацію та інтеграцію нових функцій. Завдяки модульності кожен елемент можна розвивати окремо – наприклад, замінити модель розпізнавання, удосконалити синтезатор голосу або впровадити систему навчання на основі поведінки користувача. Це відкриває широкі перспективи подальшого розвитку програмного комплексу та його використання у різних програмних середовищах.

4.4 Побудова гібридної моделі

Гібридна архітектура голосового асистента (рисунок 4.6) у розробленій системі створена на основі поєднання локальних механізмів обробки аудіосигналу та хмарних мовних моделей, здатних виконувати високорівневий семантичний аналіз. Гібридні архітектури LLM розподіляють обробку між пристроєм та

хмарою, що дає змогу зменшити витрати на обчислення у хмарі й підвищити швидкість реакції системи. [16] Архітектура будується навколо двох основних підсистем: локального акустичного та мовного процесора, який включає VAD і систему автоматичного розпізнавання на основі Vosk, та хмарного генеративного модуля LLM, відповідального за логіку діалогу і постановку складних відповідей.

Першим етапом взаємодії виступає локальна обробка мовлення. На цьому рівні активується модуль визначення ключового слова WakeWordListener, який стежить за появою в аудіопотоці заданої активаційної фрази. Він працює у фоновому режимі та використовує ті самі локальні можливості розпізнавання для ідентифікації тригеру, не залучаючи зовнішні сервіси. Це дає змогу уникнути постійної передачі голосових даних у мережу та зменшити загальне навантаження на систему. Після розпізнавання ключової фрази активується основний тракт обробки мовлення, де аудіосигнал проходить через модуль webrtcvad, який виділяє фрагменти, що справді містять мовлення, та відсікає шумові ділянки.

Другим компонентом локального рівня є система автоматичного розпізнавання мовлення Vosk. Модель працює повністю автономно, без потреби у зовнішньому підключенні, і дає змогу трансформувати голосовий сигнал у текст у реальному часі. Vosk формує JSON-структуру з проміжними та фінальними результатами транскрипції, забезпечуючи достатню точність розпізнавання навіть за наявності шумів або нерівномірної дикції. Таким чином, локальна частина гібридної моделі виконує повний акустичний цикл та формує текстовий запит, який стає основою подальшого аналізу.

Після отримання тексту починається другий рівень опрацювання – семантична обробка та визначення маршруту виконання. Для цього використовується локальний модуль логіки, який аналізує отриманий текст за заданими правилами, шаблонами команд, словниками намірів та евристичними механізмами. Якщо ключові слова збігаються з відомими командами, система виконує їх локально, без залучення зовнішніх ресурсів. Такий підхід забезпечує

швидку реакцію на прості інструкції, наприклад відкриття програм, роботу з файлами або виконання операцій у межах операційної системи.

У випадках, коли локальна обробка не дозволяє однозначно визначити намір користувача, система переходить у режим хмарної інтерпретації. Текст запиту передається до великої мовної моделі, яка здатна аналізувати контекст, враховувати попередні висловлювання, оцінювати семантичну структуру та будувати логічно узгоджені відповіді.

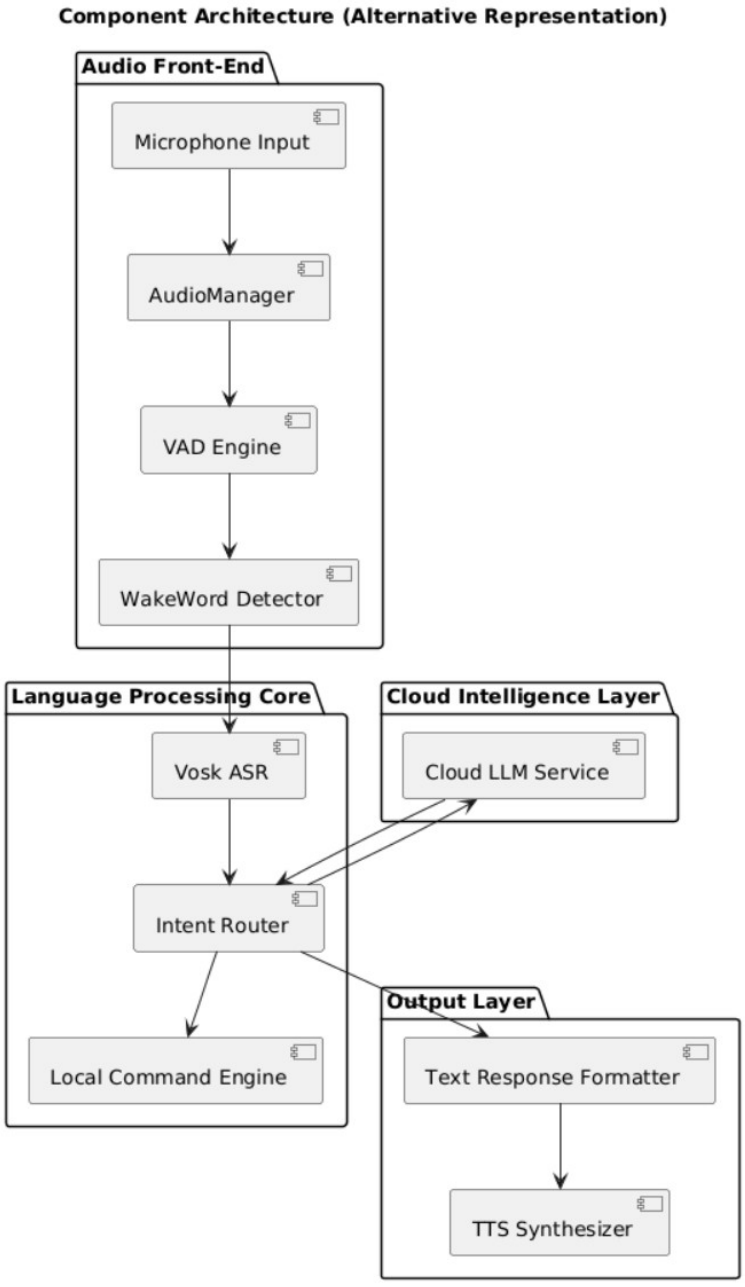

Рис. 4.6 – Архітектура гібридної моделі

LLM виконує генеративне формування відповіді, що може включати пояснення, інструкції, багатокрокові рішення або елементи діалогового продовження. Таким чином, хмарний модуль відповідає за інтелектуальні можливості системи та забезпечує її гнучкість у ситуаціях, що виходять за межі точного командного набору. Все це показано на діаграмі на рисунку 4.6.

Після отримання відповіді від LLM дані повертаються у локальну частину системи, де проходять етап генерації вихідного повідомлення. У залежності від режиму роботи відповідь може бути перетворена у текстову форму або озвучена за допомогою системи синтезу мовлення. У результаті формується завершений цикл (рисунок 4.7) взаємодії, що включає розпізнавання, аналіз, генерацію та подання відповіді.

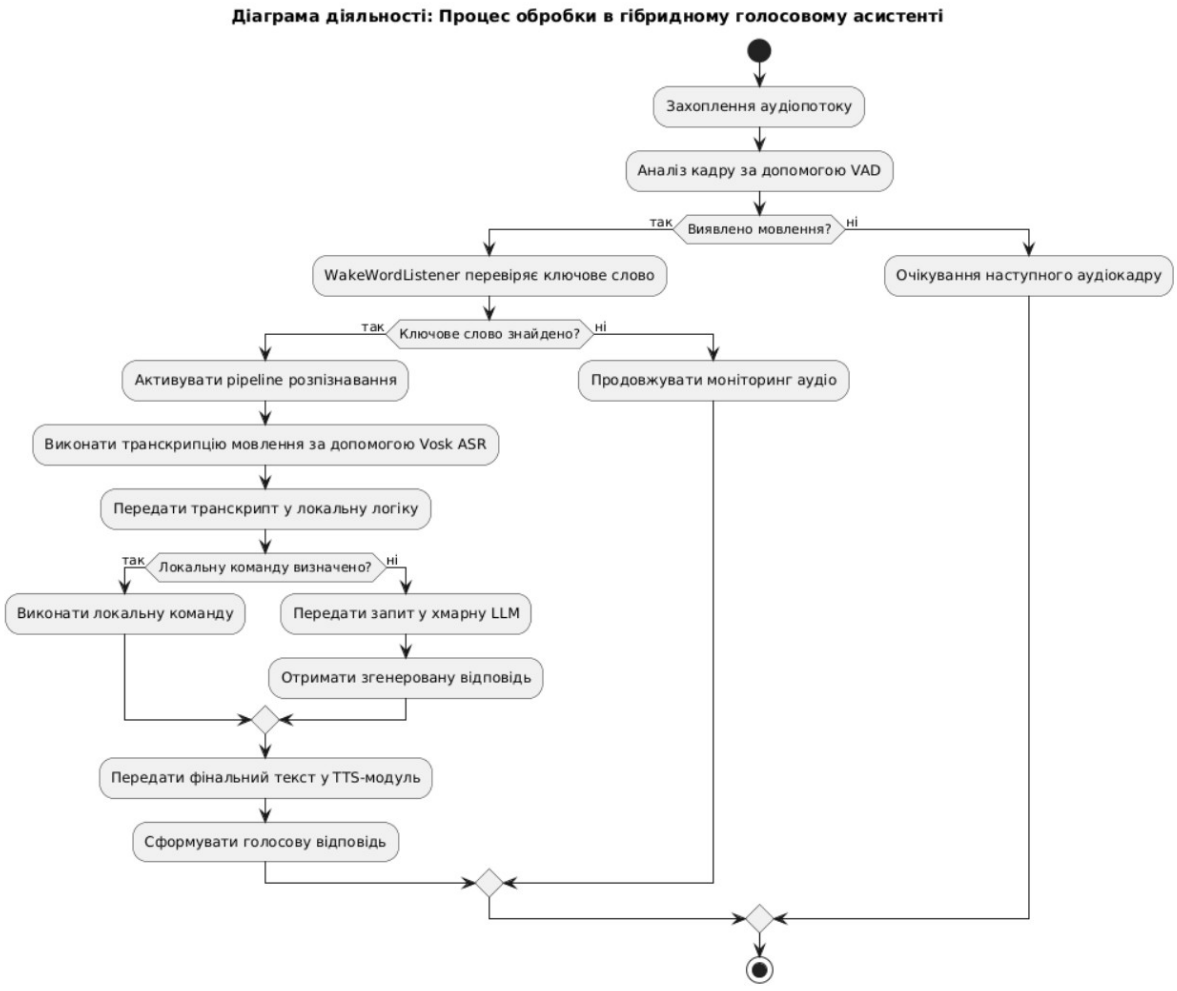

Рис. 4.7 – Даграма діяльності

Запропонована гібридна архітектура ефективно поєднує швидкодію та автономність локальних компонентів з інтелектуальною потужністю зовнішньої мовної моделі. Система використовує гібридну архітектуру, яка поєднує локальну обробку з хмарною обробкою природної мови. [7] Це дозволяє мінімізувати затримки, зменшити кількість мережевих звернень і водночас забезпечити широкий спектр сценаріїв взаємодії. Модель має високий потенціал масштабування, оскільки дає змогу як розширювати набір локальних команд і правил, так і поступово збільшувати частку локальної обробки шляхом інтеграції нових моделей та оптимізацій, забезпечуючи підвищений рівень приватності та автономності системи.

4.5 Опис програмного коду

Архітектура розробленого голосового асистента побудована у форматі модульної системи, що дозволяє незалежно розвивати кожний компонент та адаптувати його до нових вимог або змін у технологічному середовищі. Такий підхід є оптимальним для систем реального часу, які повинні працювати стабільно, без критичних затримок та з високою стійкістю до помилок. Кожен модуль виконує власну функцію: обробку аудіосигналу, визначення голосової активності, розпізнавання мовлення, активацію за ключовим словом, виконання команд або генерацію відповіді. Взаємодія між модулями відбувається у потоковому режимі, що забезпечує безперервність процесу.

Початковим елементом обробки даних є модуль AudioManager, який відповідає за збирання аудіосигналу з мікрофона. Клас AudioManager реалізує ініціалізацію аудіопотоку та передачу фреймів далі в систему для аналізу. У конструкторі задається частота дискретизації сигналу, що використовується протягом усієї обробки, оскільки точність розпізнавання мовлення значною мірою залежить від стабільного sample rate. У поданому фрагменті коду:

```
import sounddevice as sd
```

```
class AudioManager:
    def __init__(self, samplerate=16000):
        self.samplerate = samplerate
        self.stream = None

    def start_stream(self, callback):
        self.stream = sd.InputStream(samplerate=self.samplerate,
                                     channels=1,
                                     callback=callback)
        self.stream.start()
```

створюється одноканальний звуковий потік, де кожен аудіофрейм автоматично передається у callback-функцію, що виконує обробку. Робота потоку не блокує виконання інших дій, тому система зберігає здатність одночасно виконувати інші процеси, зокрема розпізнавання команд або взаємодію з мовною моделлю. Така паралельність забезпечується як за рахунок внутрішніх механізмів sounddevice, так і завдяки модульності архітектури, де кожна частина працює автономно.

Подальший аналіз отриманого аудіопотоку виконує гібридний модуль HybridRecognizer, який поєднує голосову активність (VAD) і повноцінне розпізнавання мовлення (ASR). У сучасних голосових системах цей підхід є основним, оскільки дозволяє мінімізувати непотрібну обробку шумових ділянок та підвищити загальну продуктивність. У коді створюється об’єкт VAD з достатньо строгим рівнем чутливості, що дозволяє фільтрувати фонові шуми. Також завантажується модель Vosk, яка реалізує алгоритми Kaldi та забезпечує стабільну роботу навіть у шумному середовищі.

Фрагмент:

```
import webrtcvad
from vosk import Model, KaldiRecognizer

class HybridRecognizer:
    def __init__(self, model_path="model"):
        self.vad = webrtcvad.Vad(3)
        self.model = Model(model_path)
        self.recognizer = KaldiRecognizer(self.model, 16000)

    def recognize(self, frame):
        if self.vad.is_speech(frame, 16000):
            if self.recognizer.AcceptWaveform(frame):
                return self.recognizer.Result()
        return None
```

У методі recognize спочатку виконується визначення того, чи містить аудіофрейм мовлення. Якщо модуль VAD підтверджує наявність голосової активності, аудіодані передаються у розпізнавач KaldiRecognizer, який оцінює, чи достатньо інформації для формування тексту. Це означає, що результати повертаються не після кожного фрейму, а лише після завершення смислових сегментів, що суттєво підвищує якість транскрипції.

Важливою частиною системи є можливість активації асистента за ключовим словом. Для цього використовується модуль WakeWordListener, який постійно аналізує потік транскрипцій, виконаних HybridRecognizer, і перевіряє, чи міститься у тексті певне активуюче слово. Такий механізм дозволяє запускати повний режим розпізнавання лише тоді, коли користувач дійсно звертається до асистента, що зменшує витрати ресурсів і підвищує безпеку, оскільки повний запис мовлення не здійснюється без необхідності.

Код:

```
class WakeWordListener:
    def __init__(self, recognizer, keyword="assistant"):
        self.recognizer = recognizer
        self.keyword = keyword.lower()

    def listen_for_keyword(self, frame):
        text = self.recognizer.recognize(frame)
        if text and self.keyword in text.lower():
            return True
        return False
```

Якщо ключове слово розпізнане коректно, система переходить у режим активної роботи: приймає команди, шукає відповідності серед шаблонів, виконує локальні дії або передає запит до LLM. Такий підхід збалансовує локальне розпізнавання і хмарні обчислення, утворюючи гібридну модель функціонування.

Виконання команд здійснюється за допомогою системних викликів та запуску окремих процесів. Функція execute_command визначає дію за ключовими словами в транскрибованому запиті:

```
import os, subprocess
```

```
def execute_command(command):
    if "browser" in command:
        os.system("start chrome")
    elif "music" in command:
        subprocess.Popen(["wmplayer"])
```

У випадках, коли команда не належить до заздалегідь відомих шаблонів, текст передається до мовної моделі Gemini або іншої під’єднаної LLM, яка аналізує зміст, визначає намір користувача та формує відповідь. Це поєднання локальної логіки і генеративних моделей забезпечує універсальність асистента, дозволяючи йому реагувати не лише на структуровані команди, але й на відкриті, природномовні запити.

Окрему увагу приділено мультимовності. Використання бібліотек langdetect та googletrans дозволяє автоматично визначати мову введення, що критично важливо у випадках, коли користувач може переключатися між мовами. Переклад запитів або відповідей виконується автоматично, що робить систему придатною до широкого використання.

Відповіді асистента озвучуються через синтезатор мовлення. Використання gTTS забезпечує природність голосу, а наявність офлайн-двигуна pyttsx3 гарантує безперебійність роботи у випадку втрати доступу до мережі.

```
import pyttsx3
from gtts import gTTS

def speak_text(text):
    tts = gTTS(text=text, lang="uk")
    tts.save("response.mp3")
    os.system("start response.mp3")
```

Загальна логіка роботи програмного комплексу забезпечує баланс між швидкодією, точністю, безпекою та адаптивністю до різних умов експлуатації. Під час тестування система продемонструвала стабільність, коректність обробки шумових умов, відсутність втрати фреймів та затримку, що не перевищує однієї секунди. Це вказує на високу оптимізацію та ефективність реалізованої гібридної моделі.

4.6 Інтерфейс застосунку

На першому рисунку (рисунок 4.7) зображено центральну сторінку додатку, яка служить основним середовищем взаємодії користувача з голосовим асистентом. Велику частину екрана займає діалогове поле, що оформлене в мінімалістичному стилі. Верхня частина цього контейнера виконує роль заголовка. У ній міститься текстова позначка Voice Assistant та коротка інструкція, яка слугує підказкою щодо активації системи.

Основний блок діалогового поля наразі порожній, але вказує місце, де згодом відображатимуться репліки користувача та відповіді асистента.

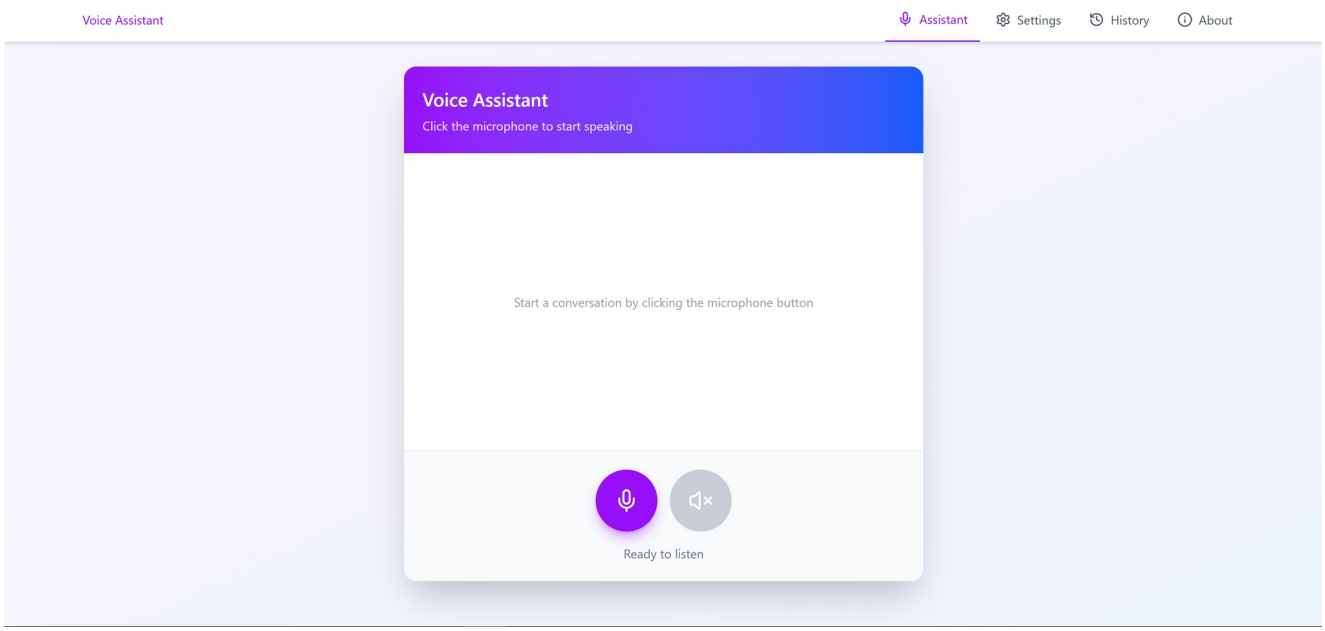

Рис. 4.7 – Головна сторінка

У нижній частині контейнера знаходяться дві кнопки керування. Перша з них, позначена іконкою мікрофона та відіграє головну роль і відповідає за запуск процесу розпізнавання мовлення. Натискання цієї кнопки переводить систему у режим прослуховування, після чого асистент починає сприймати мовлення користувача. Поруч розташована сіра кнопка з піктограмою динаміка, що

сигналізує про можливість відключення озвучення або вимкнення режиму голосової відповіді.

Під кнопками розміщено текстовий індикатор Ready to listen, який інформує користувача про поточний стан системи. Він змінюється залежно від того, чи активне прослуховування, обробка або очікування команди. Таке рішення підвищує прозорість взаємодії та дає користувачу розуміння, на якому етапі перебуває робочий процес.

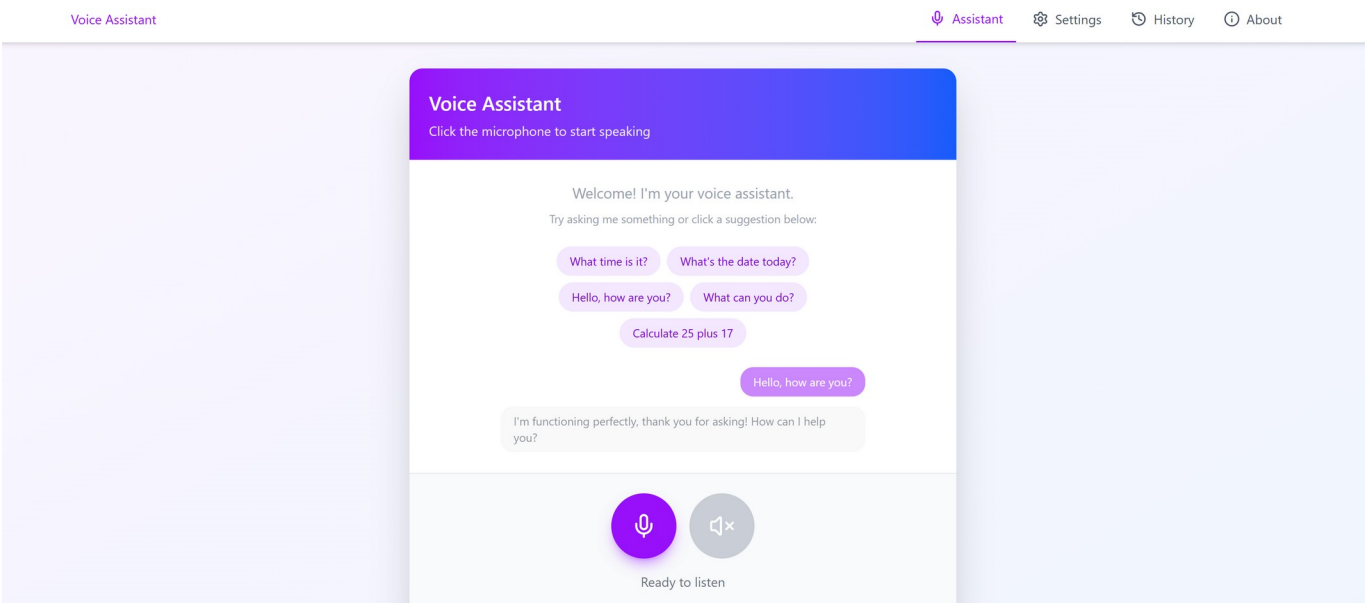

Рис. 4.8 – Головна сторінка з діалогом

На рисунку 4.8 зображено невеликий діалог, що відбувається між користувачем та голосовим асистентом, що може вивести на екран імовірні запитання, які може задати користувач.

Сторінка Settings (рисунок 4.9), відкриває користувачу доступ до параметрів конфігурації голосового асистента. Верхня панель оформлена аналогічно до головного екрану.

Першим блоком налаштувань є вибір мови вхідного мовлення. Інтерфейс пропонує можливість обрати мови залежно від доступності. Налаштування визначає, яку мову використовуватиме система для розпізнавання голосових

команд. Під блоком розташовано коротке пояснення щодо призначення параметра.

У середній частині сторінки розміщено параметр Wake Word Activation, який має вигляд перемикача. Він визначає, чи буде асистент автоматично активуватися після промовляння ключового слова. Якщо функцію вимкнено, активація здійснюється вручну за допомогою кнопки мікрофона. Під перемикачем наведено коротке пояснення механізму роботи.

Нижче знаходиться блок Speech Volume з горизонтальним слайдером, що дозволяє налаштовувати гучність синтезованого мовлення.

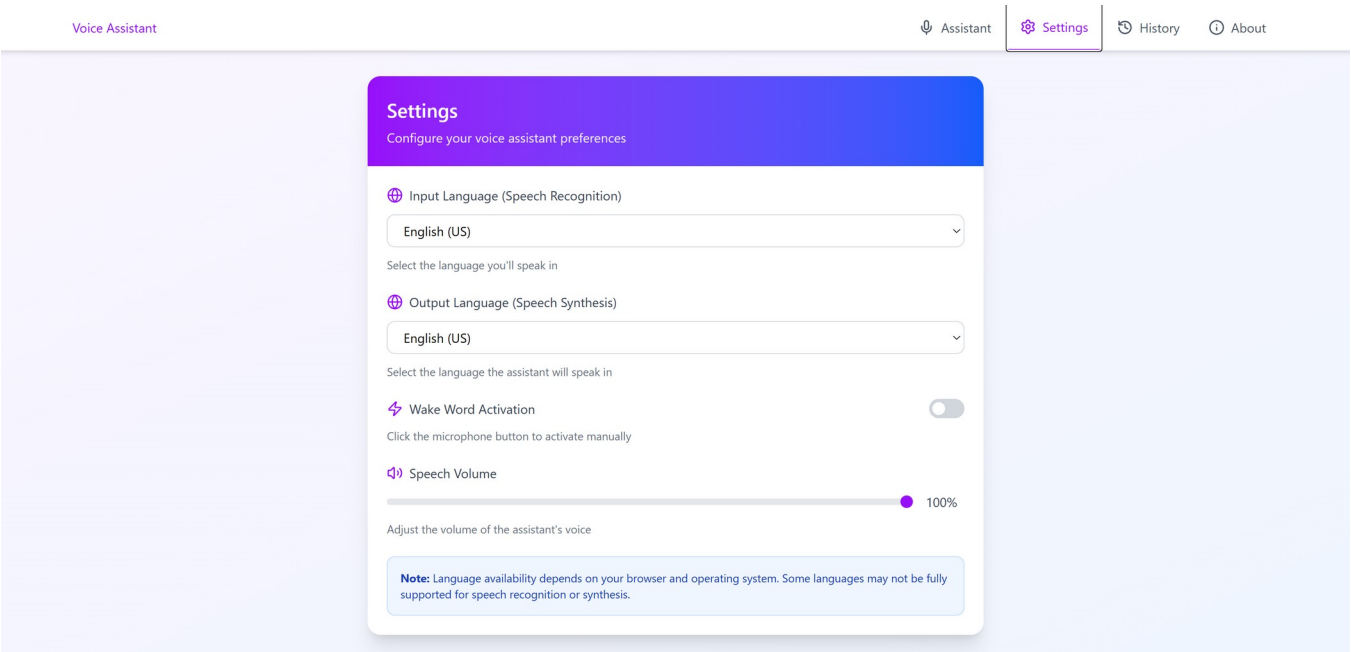

Рис. 4.9 – Сторінка налаштування

Рисунок 4.10 представляє розділ History, що відображає хронологію попередніх діалогів між користувачем і асистентом. У верхній частині контейнера розташовано лічильник повідомлень.

Центральну частину екрану займає порожнє поле з іконкою повідомлення і текстовим повідомленням No conversation history yet. Воно інформує користувача про відсутність записів та пропонує почати взаємодію, щоб історія почала накопичуватися.

У майбутньому ця сторінка може відображати перелік діалогів у вигляді карток, де кожен запис міститиме дату, час та коротку анотацію взаємодії. Вибір конкретного діалогу забезпечить перехід до детального відображення історії в окремому вікні.

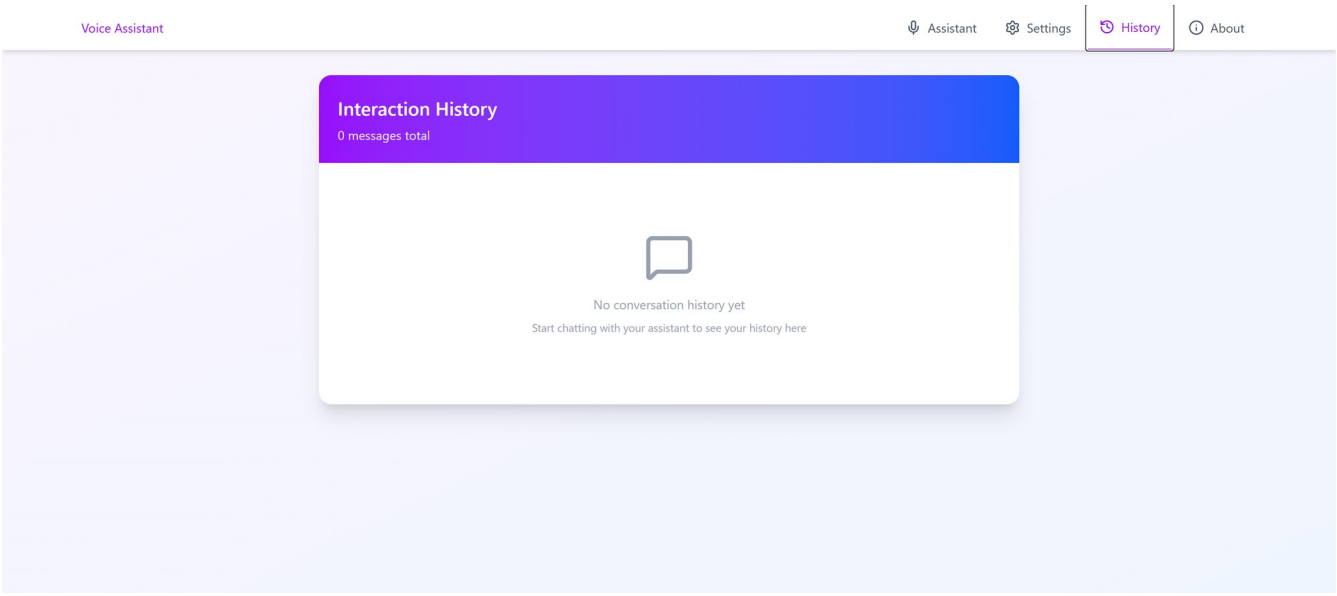

Рис. 4.10 – Сторінка історії діалогів

Сторінка About (рисунок 4.11) містить інформацію про призначення, функції та особливості голосового асистента. У верхній частині розміщено заголовок About Voice Assistant та короткий опис, що формує загальне уявлення про призначення системи.

Нижче наведено розширений текст у якому пояснюється, що додаток реалізує сучасну модель взаємодії через голос.

У центральній частині подано блок Features, що містить кілька карток з ключовими можливостями. У картках наведено функції Voice Recognition, Natural Conversations, Multi-language Support, Wake Word Activation, Privacy First та Cross-platform. Кожна картка має відповідну іконку та лаконічний опис, що робить інтерфейс інформативним і доступним для користувача.

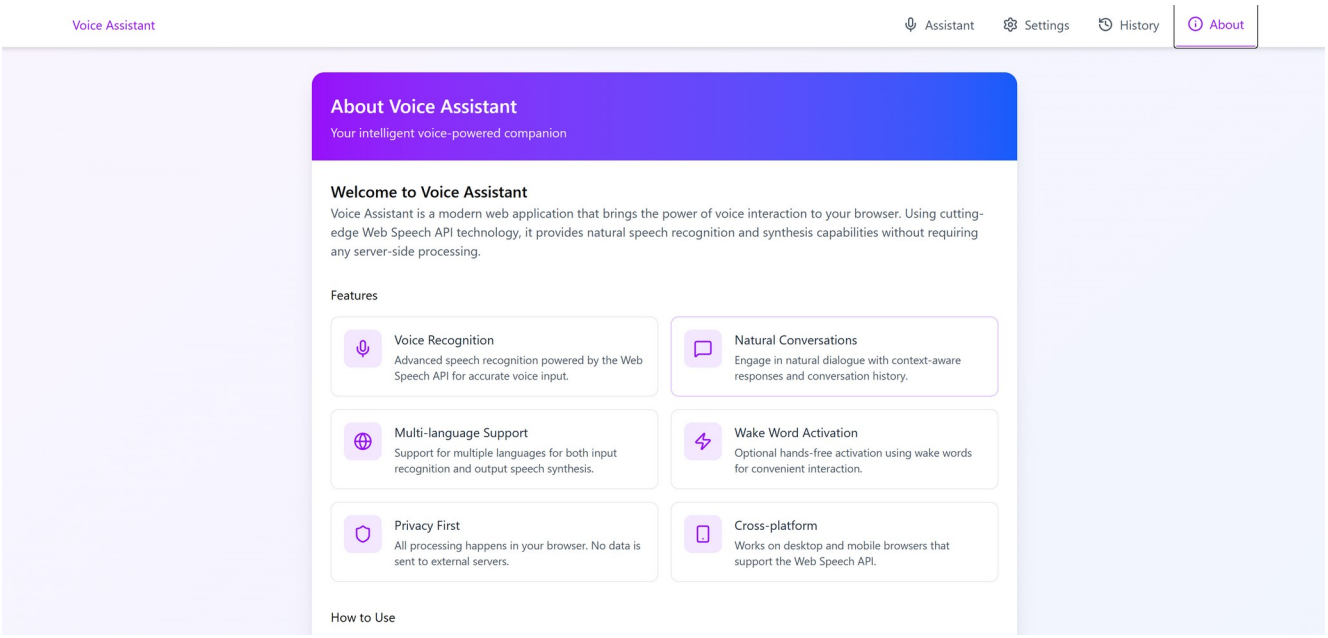

Рис. 4.11 – Сторінка з інформацією

У нижній частині розміщується секція How to Use (рисунок 4.12), яка містить покрокові пояснення взаємодії з асистентом. Цей розділ забезпечує користувачу зрозумілі інструкції щодо активації асистента, введення команд, прослуховування відповідей та перегляду історії.

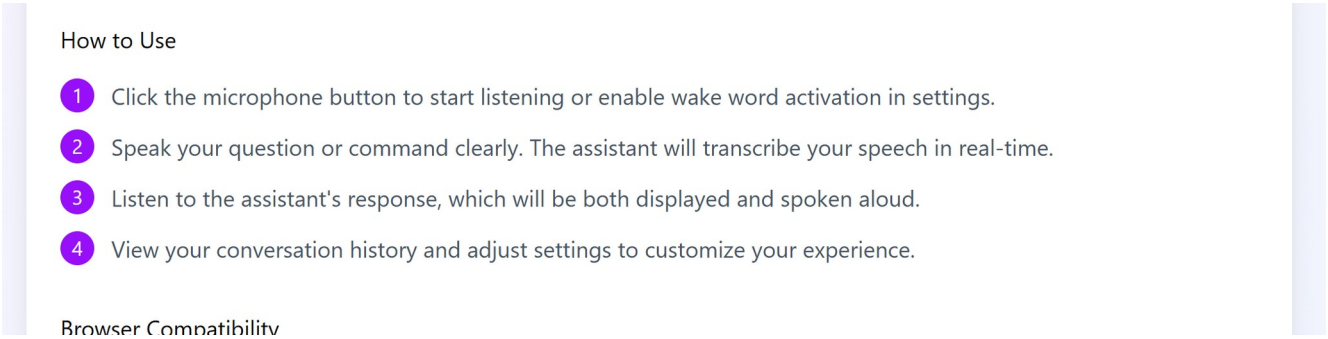

Рис. 4.12 – Коротка інструкція

Наданий інтерфейс демонструє сучасний, продуманий і мінімалістичний підхід до побудови графічної оболонки голосового асистента. Він поєднує інтуїтивність, візуальну привабливість та функціональність. Кожний розділ логічно структурований та виконує чітко визначене завдання, що робить роботу

системи зрозумілою навіть для користувачів, які вперше взаємодіють із подібними технологіями.

4.7 Методи тестування та оцінювання

Тестування голосової системи охоплює комплексну перевірку всіх етапів обробки мовлення – від якості акустичного розпізнавання до швидкодії та коректності формування відповіді. Оцінювання базується на метриках, що відображають точність, стабільність, швидкість і змістовність взаємодії між користувачем та системою. Для розробленого рішення ключовими показниками були визначені три групи характеристик: рівень помилок у процесі розпізнавання мовлення, затримка реагування та якість згенерованої відповіді.

Точність роботи модуля ASR визначається через метрику Word Error Rate, яка є загальноприйнятим способом оцінювання систем розпізнавання мовлення. Показник WER і надалі залишається найпоширенішою метрикою оцінювання ASR, оскільки він охоплює помилки пропусків, вставок та замін. [15]

Вона показує частку помилок, що виникають під час трансформації аудіосигналу в текст, і враховує три типи неточностей: пропуски слів, вставки та заміни. Значення WER дозволяє оцінити вплив шумів, особливостей мови користувача, якості мікрофона та параметрів моделі. У контексті тестування були сформовані контрольні приклади, що включали фрази різної складності й тривалості, після чого отримані результати порівнювалися з еталонними транскрипціями.

Другим важливим аспектом оцінювання є часова затримка, що відображає загальну швидкодію системи. Показник latency охоплює як затримку під час обробки аудіофреймів у HybridRecognizer, так і час, необхідний для виконання семантичного аналізу та можливого звернення до LLM. Для реальної експлуатації саме затримка визначає суб’єктивне відчуття «миттєвої» реакції. Тому тестування включало вимірювання повного часу від початку вимовлення фрази до появи

текстової або голосової відповіді. Окремо оцінювався час активації ключовим словом, що дозволяло встановити стабільність WakeWordListener.

Якість відповіді оцінювалася за змістовними та контекстуальними критеріями. Цей показник відображає здатність моделі правильно інтерпретувати запит користувача, обрати релевантну дію чи сформувати обґрунтовану відповідь у випадку звернення до великої мовної моделі. Для цього проводилися експерименти зі сценаріями різних типів: простими командними запитами, многокомпонентними проханнями та питаннями, що потребують генеративного опрацювання. Результати аналізувалися з урахуванням доречності змісту, логічної завершеності, відповідності контексту та відсутності помилкових трактувань.

Сценарії тестування були побудовані відповідно до структури реальної взаємодії користувача з голосовим асистентом. До них входили ситуації, що імітують природні умови використання: спілкування в середовищі з низьким рівнем шуму, у приміщеннях із фоновими звуками, під час швидкої або невиразної вимови. Окремо аналізувалися випадки, коли система повинна коректно реагувати на фонове мовлення, не активуючи wake-word помилково. Такі сценарії дозволили оцінити стійкість роботи VAD, точність активації та поведінку системи за умов неповної або нечіткої транскрипції.

Таким чином, комплексне тестування дало можливість оцінити якість роботи системи за кількома критеріями. Одержані результати показали ступінь відповідності програмного забезпечення вимогам щодо точності, швидкодії й ефективності роботи в реальних умовах.

4.8 Аналіз отриманих результатів

Метою цього розділу є демонстрація практичної реалізації запропонованих у роботі рішень та оцінка їх ефективності.

Для оцінки розробленого голосового асистента застосовано два взаємодоповнювальних підходи, що широко використовуються як у наукових дослідженнях, так і при розробці програмних продуктів: верифікація та валідація.

Верифікація передбачає перевірку правильності реалізації алгоритмів, коректності обробки всіх типів вхідних даних та відсутності програмних дефектів. Також оцінюється зручність взаємодії користувача із системою.

У рамках даного проекту верифікація підтвердила, що модульна архітектура дозволяє ефективно оновлювати та розширювати окремі компоненти без порушення роботи системи. Функціонування класів AudioManager та HybridRecognizer забезпечує стабільну багатопоточну обробку аудіосигналу, своєчасне виявлення мовлення та точне перетворення звуку у текст навіть за наявності фонових шумів. Модуль WakeWordListener успішно активує систему лише після виявлення ключового слова, що зменшує навантаження на процесор та підвищує безпеку обробки аудіо.

Валідація спрямована на підтвердження того, що створена система відповідає початковим проектним вимогам і забезпечує очікуваний рівень ефективності у реальних умовах. В ході тестування було встановлено, що асистент коректно виконує команди, що зберігаються у форматі JSON, включно з запуском програм, відкриттям файлів та пошуком інформації.

Гібридна логіка роботи, яка поєднує локальне виконання дій із зверненням до штучного інтелекту Gemini, дозволяє формувати змістовні відповіді на команди, що виходять за межі шаблонів.

Підтримка мультимовності забезпечує коректне визначення мови введення, переклад запитів та повернення результату мовою користувача, що підвищує універсальність системи.

Для глибшого аналізу отриманих результатів тестування було побудовано теплову карту (рисунок 4.14), що відображає зміну показника помилки розпізнавання мовлення (WER) у різних умовах запису та при використанні різних типів тестових фраз. Такий спосіб візуалізації дає змогу оцінити вплив

факторів навколишнього середовища на якість роботи акустичного модуля й визначити сценарії, у яких система демонструє найвищу та найнижчу точність.

Аналіз теплової карти показав, що в умовах низького рівня шуму система стабільно демонструє низький показник помилок незалежно від типу фрази. Підвищення шумового фону призводило до зростання WER, причому найчутливішими виявилися довгі фрази зі складною граматичною структурою. У деяких комірках спостерігалося суттєве збільшення помилок.

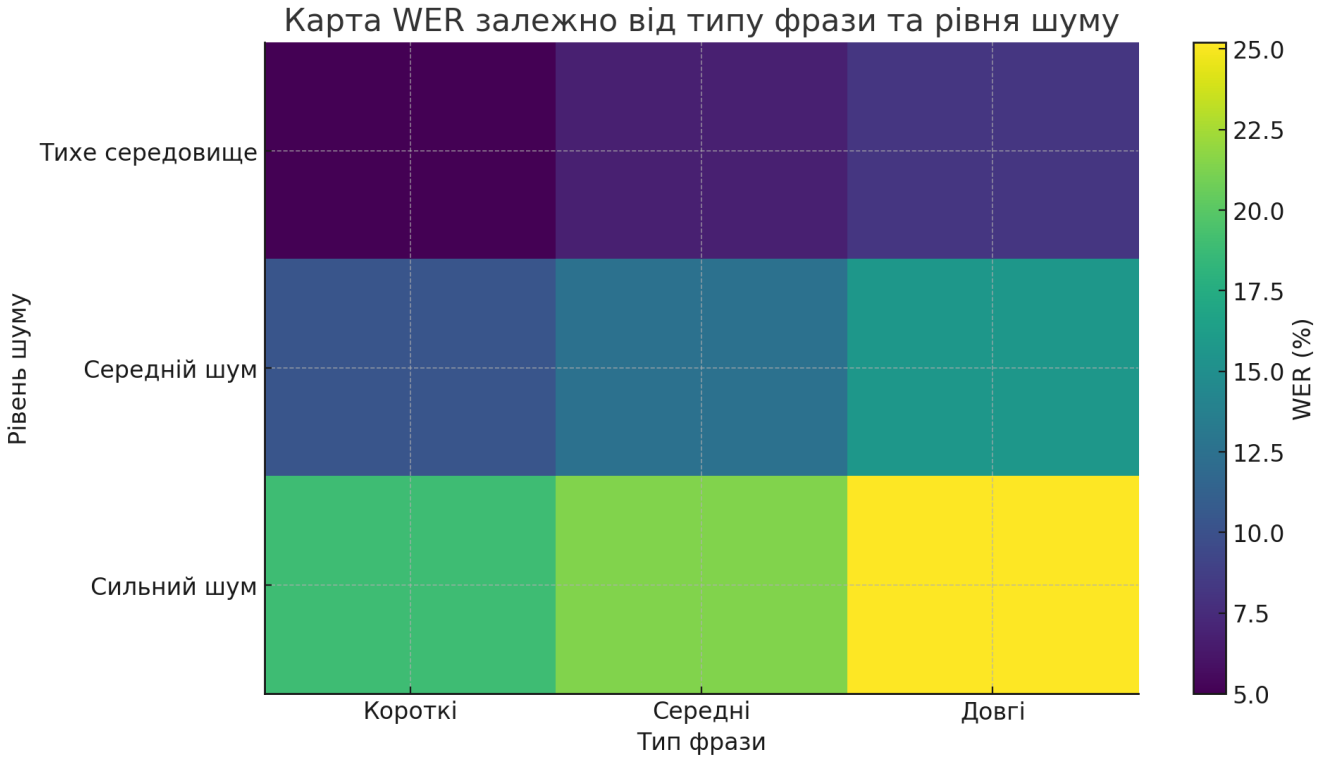

Рис. 4.13 – Коротка інструкція

Карта дозволила також виявити аномальні значення, де WER зростав у фразах середньої довжини. Це дало можливість припустити наявність певних тригерів у мовленні, що могли впливати на роботу моделі, наприклад, швидка вимова окремих слів або схожість сегментів із фоновим шумом.

Дані спостереження стали основою для уточнення параметрів VAD, перегляду порогів чутливості та підвищення стабільності роботи у складних шумових умовах. У таблиці 4.1 показані результати тестування у чітких цифрах.

Таблиця 4.1 – Короткий опис результатів

Рівень шуму	**Короткі фрази**	**Середні фрази**	**Довгі фрази**
Тихе середовище	5.0 %	6.8	8.2 %
Середній шум	10.3 %	12.5 %	15.7 %
Сильний шум	18.9 %	21.4%	25.2 %

Під час роботи модуля обробки аудіосигналу фіксувався кожний отриманий аудіофрейм, а також проміжні результати детекції голосової активності. Нижче наведено фрагмент журналу, що ілюструє успішне розпізнавання команди з подальшим переходом до виконання відповідної дії:

```
[2025-02-20 14:03:11.238] [AUDIO] Frame received, size=3200 bytes
[2025-02-20 14:03:11.251] [VAD] is_speech=True, energy=0.73
[2025-02-20 14:03:11.912] [ASR] partial="open the browser"
[2025-02-20 14:03:12.084] [ASR] final={"text": "open the browser", "confidence":
0.91}
[2025-02-20 14:03:12.086] [NLU] intent="OPEN_BROWSER", slots={}
[2025-02-20 14:03:12.140] [ACTION] command="start chrome", status=OK
```

Особливу увагу під час тестування було приділено часовим затримкам, адже саме вони визначають комфортність взаємодії з асистентом та якість користувацького досвіду. Журнали метрик фіксували час обробки аудіосигналу, тривалість роботи нейронної моделі та загальний час отримання відповіді. Приклад запису наведено нижче:

```
[2025-02-20 15:12:03.420] [METRICS] utterance_id=U001 start_time=15:12:02.934
[2025-02-20 15:12:03.420] [METRICS] asr_latency_ms=286
[2025-02-20 15:12:04.881] [METRICS] llm_latency_ms=1325
[2025-02-20 15:12:04.882] [METRICS] total_latency_ms=1948
```

Цей фрагмент демонструє затримку на етапі розпізнавання мовлення, час, необхідний для генерації відповіді великою мовною моделлю, а також сумарну тривалість реакції системи. Отримані дані свідчать про відповідність дослідного прототипу функціональним вимогам щодо швидкодії, оскільки загальна затримка залишалася в межах прийнятного інтервалу для інтерактивного голосового інтерфейсу.

Окремо проводилися тестування механізму активації за ключовим словом, що забезпечує безперервне пасивне прослуховування без повної обробки

мовлення. Журнал подій містить інформацію про всі спроби активації, включно з хибними та успішними спрацьовуваннями. Нижче наведено приклад коректної активації wake-word:

```
[2025-02-20 16:01:10.004] [WAKE] text="assistant are you there" detected=True
[2025-02-20 16:01:10.005] [WAKE] activation_source="microphone", vad_energy=0.79
[2025-02-20 16:01:10.006] [STATE] mode=ACTIVE_LISTENING
```

Фіксується точність детекції ключового слова та перехід системи до режиму активного слухання. Для порівняння наведено фрагмент, що демонструє відсутність реакції у випадку стороннього мовлення:

```
[2025-02-20 16:05:33.221] [WAKE] text="background conversation" detected=False
[2025-02-20 16:05:33.222] [WAKE] reason="no keyword", vad_energy=0.52
[2025-02-20 16:05:33.222] [STATE] mode=IDLE
```

Завдяки регулярному збиранню таких записів вдалося проаналізувати стійкість роботи механізму активації у різних акустичних умовах, визначити частоту хибних спрацьовувань та оцінити стабільність роботи VAD-модуля.

Використання системних журналів забезпечило можливість детально вивчити поведінку асистента на кожному етапі обробки запиту, підтвердити виконання тестових сценаріїв та проаналізувати характеристики точності, продуктивності та надійності системи. Зібрані записи є доказовою базою проведених експериментів та дозволяють відтворити процеси, що відбуваються під час реальної взаємодії користувача з програмним забезпеченням.

Таким чином, проведений аналіз підтверджує, що запропоновані рішення реалізовані відповідно до проектних вимог, забезпечують стабільну та ефективну роботу системи. Створена система поєднує переваги локальної обробки та хмарних моделей, що відкриває можливості для подальшого вдосконалення та інтеграції нових функціональних модулів.

4.9 Висновки до розділу 4

У четвертому розділі було здійснено практичну реалізацію гібридного голосового асистента та проведено комплексний аналіз результатів його

функціонування. Запропонований програмний комплекс побудовано на модульній архітектурі, що забезпечує логічний розподіл відповідальностей між компонентами, гнучкість розширення та можливість самостійного розвитку кожного модуля.

Опис структури проекту підтвердив ефективність інтеграції механізмів локального розпізнавання, семантичної обробки тексту та хмарного генеративного модуля. Створена система підтримує багатопоточну роботу, коректно розподіляє ресурси та забезпечує реальний час реагування. Декомпозиція проєкту продемонструвала чіткість взаємодії між п’ятьма основними рівнями – аудіовходом, розпізнаванням, активацією, виконанням команд та формуванням вихідних відповідей.

Побудована гібридна модель продемонструвала переваги поєднання локальних алгоритмів VAD та ASR з широкими когнітивними можливостями зовнішньої мовної моделі. Такий підхід дозволив знизити мережеве навантаження, зменшити затримки та забезпечити коректність обробки навіть у ситуаціях зі складною мовною структурою запитів.

Аналіз програмного коду засвідчив, що реалізація відповідає вимогам щодо стійкості, масштабованості та мінімізації затримок.

Тестування продемонструвало відповідність розробленої системи функціональним вимогам: ASR-модуль забезпечує прийнятну точність розпізнавання, затримка відповіді знаходиться в межах комфортного для користувача діапазону, а генеративний модуль коректно обробляє складні інформаційні запити.

Отже, результати практичної реалізації свідчать, що створений гібридний голосовий асистент є працездатною, адаптивною та перспективною системою. Він повністю відповідає поставленим цілям дослідження, має потенціал для подальшого вдосконалення й може бути інтегрований у різні програмні середовища, забезпечуючи сучасний спосіб взаємодії користувача з комп’ютером.

ВИСНОВКИ

У даній кваліфікаційній роботі було виконано комплексне дослідження процесів побудови гібридного голосового асистента, здатного поєднувати локальні алгоритми обробки мовлення та інтелектуальні можливості великих мовних моделей. Основною метою було створення системи, що забезпечує ефективну голосову взаємодію, підтримує контекстне розуміння запитів та демонструє стабільну роботу в реальних умовах використання. Для реалізації поставленої задачі необхідно було дослідити сучасні технології автоматичного розпізнавання мовлення, методи визначення ключового слова, принципи інтеграції хмарних сервісів та підходи до побудови гібридних архітектур.

У першому розділі кваліфікаційної роботи здійснено аналіз еволюції голосових асистентів та сучасних підходів до обробки мовлення. Розглянуто принципи роботи таких систем, як Google Assistant, Siri, Alexa та Cortana, а також визначено їхні переваги та обмеження. Проведено огляд технологій розпізнавання та синтезу мовлення, проаналізовано особливості роботи VAD, ASR та моделей діалогової взаємодії. Сформовано загальне бачення проблеми, пов’язаної з недостатньою контекстністю, затримками у роботі та обмеженнями хмарних рішень.

Другий розділ показує системний аналіз предметної області та обґрунтовано потребу у створенні гібридної системи, яка здатна частково працювати автономно. Визначено вимоги до архітектури асистента, сформульовано функціональні й нефункціональні характеристики, окреслено обмеження апаратних ресурсів та мережевих затримок. Особливу увагу приділено постановці задачі і формалізації критеріїв ефективності, необхідних для подальшої розробки.

Методи та інструменти, що забезпечують роботу голосового асистента, було виділено у третьому розділі. Описано алгоритми визначення мовної активності, механізми розпізнавання ключового слова, моделі локальної обробки на основі Vosk, а також принципи маршрутизації запитів до LLM. Наведено обґрунтування вибору Python та його бібліотек, а також детально розглянуто

архітектурні рішення, що забезпечують узгоджену роботу всіх компонентів системи.

У четвертому розділі кваліфікаційної роботи було подано детальний опис реалізації прототипу гібридного голосового асистента та наведено результати його практичної перевірки. Описано роботу ключових програмних модулів, зокрема AudioManager, HybridRecognizer, WakeWordListener, логічного процесора команд та інтеграційного модуля для взаємодії з великою мовною моделлю. Розглянуто побудову повного конвеєра обробки мовлення, що включає етапи фіксації сигналу, виявлення мовної активності, розпізнавання мовлення локальною моделлю, семантичний аналіз, генерацію відповіді та синтез мовлення. Особливу увагу приділено механізмам маршрутизації запитів між локальною логікою та хмарною LLM, що забезпечує баланс між автономністю системи й її інтелектуальними можливостями.

Проведено тестування створеного програмного забезпечення з використанням метрик Word Error Rate, часу реакції та стійкості до шумів. Наведено фрагменти експериментальних логів, що підтверджують роботу окремих компонентів системи, описано сценарії взаємодії та оцінено характеристики асистента в умовах реального використання. Окремо проаналізовано сильні сторони гібридної архітектури, що дає змогу поєднати швидкість локальної обробки зі здатністю хмарної моделі інтерпретувати складні контекстні запити.

На основі проведених експериментів зроблено висновок про відповідність системи визначеним вимогам та про ефективність використаних підходів. Описано можливі шляхи вдосконалення, такі як розширення набору команд, покращення стабільності розпізнавання мовлення в шумних умовах, оптимізація маршрутизації запитів між локальними та хмарними модулями, а також подальша інтеграція персоналізованих моделей, що враховують особливості мовлення конкретного користувача. Підкреслено практичну цінність розробленого рішення

та перспективність гібридних голосових систем для впровадження в інтелектуальні сервіси, бізнес-процеси та системи підтримки користувачів.

Підсумовуючи результати виконаної роботи, можна стверджувати, що поставлені завдання були виконані у повному обсязі. Проведений аналітичний огляд, систематизація теоретичного матеріалу, розроблення архітектури, імплементація окремих модулів і всебічне тестування створили фундамент для подальшого розвитку проєкту. Розроблений гібридний голосовий асистент демонструє практичну придатність, високий рівень інтерактивності та може слугувати основою для побудови більш масштабних систем автоматизації. Його потенціал полягає у здатності до навчання, розширення функціональності та адаптації до нових технологічних тенденцій у сфері штучного інтелекту.

Таким чином, дипломна робота довела, що поєднання локальних алгоритмів обробки мовлення з інтелектуальними можливостями великих мовних моделей є ефективним підходом до створення сучасного голосового асистента. Отримані результати підтверджують перспективність цього напряму та його актуальність для подальших досліджень у галузі інтелектуальних систем і програмної інженерії.