НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

До захисту допущено:

Завідувач кафедри

_____ Олександр РОЛІК

«___»_____2025 р.

Дипломний проєкт

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інформаційне забезпечення робототехнічних систем»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Діалогова підсистема інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою»

Виконала:
Студентка IV курсу, групи ІК-13
Чурікова Орина Костянтинівна _____

Керівник:
доцент кафедри ІСТ, к.т.н., доц.
Олійник Володимир Валентинович _____

Рецензент:
доцент кафедри ІПІ, к.т.н., доц.
Лісовиченко Олег Іванович _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2025 р.

ЗАВДАННЯ
на дипломний проєкт студентці
Чуріковій Орині Костянтинівні

1. Тема проєкту «Діалогова підсистема інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою», керівник проєкту Олійник Володимир Валентинович, доцент кафедри ІСТ, к.т.н., затверджені наказом по університету від «23» травня 2023 р. №1705-с

2. Термін подання студентом проєкту: 9 червня 2025 року

3. Вихідні дані до проєкту: діалогова підсистема інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою повинна забезпечувати ідентифікацію користувачів за допомогою унікального ідентифікатора, створення індивідуального профілю з історією взаємодій, вибір тем для завдань, надсилання голосових повідомлень через інтерфейс бота, розпізнавання мовлення користувача та його перетворення на текст. Система має здійснювати якісний та кількісний аналіз граматичних і лексичних помилок, оцінювати відповідність відповіді заданій темі, зберігати виявлені помилки та оцінки користувача, формувати та відображати індивідуальну статистику успішності у вигляді графіків. Крім того, система повинна мати інтуїтивно зрозумілий інтерфейс із кнопками та підказками та забезпечувати обробку голосового повідомлення користувача не довше ніж за 50 секунд, підтримувати розпізнавання української вимови, працювати

стабільно без збоїв і відновлювати сесію у разі переривання. При цьому система повинна гарантувати конфіденційність даних шляхом обмеженого зберігання персональної інформації.

4. Зміст пояснювальної записки: аналіз предметної області, огляд об'єкту дослідження, проблематика, постановка мети, постановка завдання, актуальність розробки, існуючі реалізації мовних платформ, формування вимог, вибір технологій для розпізнавання мовлення та аналізу тексту, клієнт-серверна архітектура, проєктування системи, засоби для реалізації, приклад застосування, результати роботи.

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо): логічна модель даних, діаграма варіантів використання, структурна схема робота асистента, діаграма послідовностей.

6. Дата видачі завдання 5 березня 2025 року

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Аналіз та опис предметної області	17.04.25	
2	Постановка мети, опис задачі та її особливостей	20.04.25	
3	Огляд існуючих реалізацій та підходів до вирішення задачі	25.04.25	
4	Вибір програмної платформи, мови програмування, засобів реалізації	31.04.25	
5	Проєктування архітектури та написання програмного коду	05.05.25	
6	Розгортання та тестування системи	15.05.25	
7	Оформлення пояснювальної записки	21.05.25	

Студентка Орина ЧУРІКОВА

Керівник Володимир ОЛІЙНИК

АНОТАЦІЯ

Діалогова підсистема інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою.

Розмір пояснювальної записки – 69 аркушів, містить 34 рисунка, посилання на 24 літературних джерела, додаток А та 4 конструкторських документів.

Ключові слова: діалогова підсистема, інтелектуальний робот-асистент, розмовні навички українською мовою, модель штучного інтелекту, Python, Whisper, GPT.

Об’єктом дослідження є процес використання та вдосконалення розмовних навичок українською мовою.

Метою роботи є підвищення ефективності процесу використання та вдосконалення розмовних навичок українською мовою за рахунок розробки діалогової підсистеми інтелектуального робота асистента.

До використаних технологій належить автоматична система розпізнавання мовлення Whisper, велика мовна модель GPT-4. Серед програмного забезпечення було використано Python, PostgreSQL, середовище написання програмного коду Visual Studio Code та бібліотеки telebot, openai, psycopg2.

Було розраховано показник точності розпізнавання мовлення WER, що склав 8,8%, що свідчить про високу ефективність Whisper. Також обраховано показники стабільності GPT-моделі — відтворюваність 93% та дисперсія 0,0622, що вказує на послідовну роботу моделі. Використовуючи LLM Evaluation, отримано значення F1-score = 0,82, яке свідчить про збалансовану точність і повноту виявлення мовних помилок.

SUMMARY

The explanatory note is 69 pages long, contains 34 figures, references to 24 literary sources, Appendix A, and 4 design documents.

Keywords: dialogue subsystem, intelligent assistant robot, Ukrainian speaking skills, artificial intelligence model, Python, Whisper, GPT.

The object of the study is the process of using and improving Ukrainian speaking skills.

The aim of the work is to enhance the effectiveness of the process of using and improving Ukrainian speaking skills through the development of a dialogue subsystem of an intelligent assistant robot.

The technologies used include the automatic speech recognition system Whisper and the large language model GPT-4. Among the software tools used were Python, PostgreSQL, the Visual Studio Code development environment, and the libraries telebot, openai, and psycopg2.

The speech recognition accuracy metric WER was calculated and amounted to 8.8%, indicating high efficiency of Whisper. Stability metrics of the GPT model were also calculated — reproducibility of 93% and variance of 0.0622, which points to the model's consistent performance. Using LLM Evaluation, an F1-score of 0.82 was obtained, indicating a balanced precision and recall in detecting language errors.

№ рядка	Формат	Позначення	Найменування	Кіл. аркушів	№ екз.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІК13.300БАК.006 ПЗ	Діалогова підсистема	69		
6			інтелектуального робота-асистента			
7			для вдосконалення розмовних			
8			навичок українською мовою.			
9			Пояснювальна записка			
10						
11	А3	ІК13.300БАК.006 Д1	Діалогова підсистема	1		
12			інтелектуального робота-асистента			
13			для вдосконалення розмовних			
14			навичок українською мовою.			
15			Логічна модель даних			
16						
17	А3	ІК13.300БАК.006 Д2	Діалогова підсистема	1		
18			інтелектуального робота-асистента			
19			для вдосконалення розмовних			
20			навичок українською мовою.			
21			Діаграма варіантів використання			
22						
23	А3	ІК13.300БАК.006 Д3	Діалогова підсистема	1		
24			інтелектуального робота-асистента			
25			для вдосконалення розмовних			
26			навичок українською мовою.			
27			Структурна схема робота-асистента			
28						

Зм.	Лист	№ докум.	Підпис		ІК13.300БАК.006 ТП		
Розробив		Чурікова			Діалогова підсистема інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою. Відомість дипломного проєкту	Літ.	Арк. 1 / Аркушів 2
Перевірив		Олійник				Т	
						КПІ ім. Ігоря Сікорського Група ІК-13	
Затв.							

№ рядка	Формат	Позначення	Найменування	Кіл. аркушів	№ екз.	Примітка
1	А3	ІК13.300БАК.006 Д3	Діалогова підсистема	1		
2			інтелектуального робота-асистента			
3			для вдосконалення розмовних			
4			навичок українською мовою.			
5			Діаграма послідовностей			
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						
28						
29						
30						
31						

Пояснювальна записка
до дипломного проєкту
на тему: «Діалогова підсистема інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою»

Київ – 2025 року

ЗМІСТ

ВСТУП

У сучасному світі володіння українською мовою набуло особливого значення не тільки як засобу комунікації, але й як важливого елементу національної ідентичності та професійного розвитку. Все більше і більше людей свідомо обирають українську мову для користування у побуті, на роботі та у всіх сферах життя. Однак у реаліях багато українців, що тільки починають говорити українською, стикаються зі складнощами. Відсутність або нестача мовної практики призводять до відчуття невпевненості та дискомфорту, що стає перепоною на шляху до використання української як основного інструменту комунікації.

Об’єктом дослідження даної роботи є процес використання та вдосконалення розмовних навичок українською мовою. Він супроводжується низкою проблемних ситуацій, таких як психологічний страх помилок, соціальний тиск, брак мовної практики та іншими факторами, що ускладнюють покращення рівня знань української мови. Дослідження цього процесу та розробка ефективних інструментів його оптимізації є важливим завданням для покращення комунікативних навичок, підвищення впевненості у своїх знаннях та сприяння активному використанню української мови в повсякденному житті [1].

Оскільки, знаходження викладача або реального співрозмовника має певні обмеження щодо фінансових, часових або психологічних факторів, було вирішено реалізувати інтелектуального асистента. Отже предметом дослідження стала діалогова підсистема інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою.

Головною метою проекту є підвищення ефективності процесу використання та вдосконалення розмовних навичок українською мовою за рахунок розробки діалогової підсистеми інтелектуального робота-асистента. Конкретно, це включає створення інтелектуального діалогового агента, що здатний аналізувати мовлення користувача та надавати зворотній зв’язок щодо помилок, включно з персоналізованими порадами.

Для досягнення заданої мети було поставлено ряд завдань, таких як: змістовна постановка цілей та задачі, огляд існуючих підходів та наявних реалізацій, аналіз та вибір технологій розробки, розробка програмного забезпечення та опис його структури, оцінка ефективності роботи агента та результати проведених експериментів.

Одержані результати дослідження та розробки інтелектуального асистента для вдосконалення розмовних навичок українською мовою мають велике практичне значення. Вони можуть бути використані як інструмент для самостійної практики української мови або у навчальних закладах для допомоги студентам та викладачам. Такі результати також можуть стати базисом для подальшого вдосконалення інтелектуальних мовних асистентів, зокрема для розширення їх функціоналу в галузі освіти, професійної комунікації та соціальної адаптації.

Через широку популярність та ефективність використання моделей штучного інтелекту для даної проблеми, було проаналізовано ряд можливих засобів та алгоритмів для їх застосування. У результаті чого, було вирішено взяти технології обробки природньої мови та велику мовну модель, засновану на архітектурі трансформерів як основні інструменти для реалізації підсистеми інтелектуального агента.

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Аналіз предметної області практика та вдосконалення розмовних навичок українською мовою

1.1.1 Загальний огляд об'єкту дослідження

Практика та вдосконалення розмовних навичок українською мовою – це процес, який передбачає активне використання мови в усному спілкуванні з метою покращення лексичного запасу, граматичної правильності та загальної комунікативної впевненості.

Розмовні навички можуть включати різні аспекти: ведення діалогів, монологічні висловлювання, обговорення тем, використання ідіом та фразеологізмів, а також адаптацію до різних стилів спілкування (офіційного, неформального, професійного тощо).

Під час процесу набуття розмовних навичок можуть виникати труднощі, такі як страх помилок, обмежений словниковий запас, неправильна вимова, складнощі з побудовою граматичних конструкцій, а також відстуність конструктивного зворотного зв’язку щодо помилок. Ці фактори можуть уповільнювати прогрес і знижувати мотивацію до вивчення мови. Крім того, відсутність мовного середовища або недостатня практика можуть обмежувати розвиток навичок.

Сучасні підходи до практики мови орієнтовані на активне занурення у мовне середовище через різноманітні інтерактивні методи. Одним з таких підходів є жива комунікація – участь у мовних клубах, дискусіях та рольових іграх, що допомагає подолати мовний бар'єр і швидше засвоювати нову лексику. Цифрові інструменти значно розширили можливості для навчання: мовні додатки, штучний інтелект для тренування діалогів, аудіо- та відеоматеріали створюють ефективне навчальне середовище. У даній роботі будуть розглянуті цифрові інструменти, що покращують доступність та легкість мовної практики.

Процес практики та вдосконалення розмовних навичок українською мовою є важливою складовою ефективного мовленнєвого розвитку та може вимагати системного підходу, регулярної практики та використання сучасних методик.

Дослідження в цій галузі є актуальними, оскільки їх результати сприяють підвищенню рівня компетентності, що є ключовим для соціальної, академічної та професійної інтеграції, особливо у наш час, коли питання національної ідентичності та розвитку української культури є дуже нагальними.

1.1.2 Проблематика практики та вдосконалення розмовних навичок українською мовою

Щоб зрозуміти, які труднощі виникають у процесі опанування розмовних навичок українською мовою, варто детальніше розглянути ключові аспекти цієї проблеми.

Насамперед, багато людей стикаються з психологічним бар'єром – страхом помилитись або виглядати некомпетентно під час спілкування, що призводить до мовчазності та уникнення практики. Особливо це стосується тих, хто знаходиться в російськомовному середовищі та має обмежений доступ до україномовної середи.

Крім того, процес вдосконалення розмовних навичок ускладнюється через відсутність системного підходу до навчання. Багато курсів роблять акцент на граматиці та письмі, тоді як усна практика залишається другорядною. Також існують проблеми з вимовою, обмеженим словниковим запасом і невмінням швидко формулювати думки, що робить спілкування повільним і неефективним.

Одним із можливих рішень цієї проблеми є використання сучасних комунікативних методів навчання, таких як імерсійні програми, залучення штучного інтелекту для тренування. Такі підходи дозволяють імітувати реальні ситуації спілкування та зменшити страх помилок, збільшуючи ефективність шляхом надання конструктивного зворотного зв’язку щодо помилок, а також формуванням персоналізованих порад.

Отже, розробка програмного продукту, який надає інструменти для ефективної практики усного мовлення може суттєво покращити якість вивчення української мови та допомогти подолати комунікативні бар'єри. Проте для реалізації такого продукту необхідно враховувати індивідуальні потреби сучасних

користувачів та використовувати технологічні інновації для забезпечення доступності та ефективності мовної практики.

1.2 Постановка мети проєкту

Щоб надати вирішення описаних проблем, сформулюємо ціль даної роботи. Головною метою даного проєкту є покращення ефективності процесу практики розмовних навичок українською мовою за рахунок розробки діалогової підсистеми інтелектуального робота-асистента. Дана мета буде досягнута за рахунок покращення таких критеріїв як доступність, швидкість, точність та актуальність зворотного зв’язку, актуальність та персоналізація сформованих порад.

1.2.1 Завдання роботи над проєктом

Для досягнення сформульованої мети потрібно поставити ряд конкретних завдань. Отже, сформулюємо їх у наступним чином:

– розкрити суть та актуальність задачі, її особливості;

– провести аналіз існуючих підходів та рішень для вирішення визначеної задачі;

– сформулювати технічне завдання, тобто перелік конкретних вимог до розробки;

– оглянути необхідні технології та засоби розробки, їх принцип роботи;

– обрати платформу та мову програмування для розробки програмного забезпечення;

– розробити структуру програмного коду та навести специфікацію функціоналу окремих її компонентів;

– створити та описати діалогового інтелектуального асистента для мовної практики;

– підготувати необхідні тестові дані та статистику, для подальшої оцінки якості роботи системи;

– проаналізувати ефективність роботи підсистеми та зробити висновки.

Цей комплекс завдань дозволить розробити ефективний інструмент для вдосконалення розмовних навичок українською мовою. Послідовне виконання цих завдань забезпечить системний підхід до розробки, дозволить створити функціональний інструмент для мовної практики та отримати дані про його ефективність. Це, у свою чергу, сприятиме подальшому вдосконаленню методів вивчення української мови та розширенню можливостей для її практичного застосування.

1.3 Опис задачі створення діалогової підсистеми для інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою

Відповідно до поставлених завдань у попередньому пункті, спочатку необхідно окреслити задачу дипломного проєкту.

Задача, що розглядається у даній роботі, полягає в створенні інтелектуальної діалогової системи, яка допомагатиме користувачам набувати практику та удосконалювати розмовні навички українською мовою через інтерактивного діалогового агента. Основна мета полягає у створенні інструменту, здатного аналізувати усне мовлення, виявляти помилки та надавати персоналізовані рекомендації для покращення рівню знань. Ключовим елементом системи стане інтелектуальний механізм, який генеруватиме релевантні теми для мовленнєвих вправ, аналізуватиме голосові повідомлення за критеріями граматичної та лексичної правильності, відповідності заданій темі, а також надаватиме детальний зворотній зв'язок із конкретними прикладами виправлень.

Створена система знайде широке застосування як інструмент для самостійного вдосконалення мовних навичок, так і для підтримки навчального процесу. Інтелектуальний асистент стане ефективним помічником у повсякденній практиці усного мовлення, сприяючи швидкому та якісному опановуванню української мови через регулярну інтерактивну взаємодію та персоналізований підхід до користувачів.

1.3.1 Загальний огляд предмету дослідження

З визначення задачі стає зрозуміло, що предметом дослідження є інтелектуальна діалогова система для вдосконалення розмовних навичок українською мовою, яка реалізується у формі інтерактивного чат-бота. Оскільки традиційні методи навчання мовленню часто потребують присутності викладача або мовного середовища, створення віртуального асистента є оптимальним рішенням для забезпечення доступної та ефективної практики.

Це рішення має ряд суттєвих переваг порівняно з класичними методами навчання.

По-перше, система забезпечує можливість регулярної практики в будь-який час без необхідності участі викладача чи співрозмовника.

По-друге, вона дозволяє автоматизувати процес виявлення та аналізу помилок, забезпечуючи об'єктивну оцінку та конструктивний зворотний зв’язок.

По-третє, система надає персоналізований підхід до кожного користувача, зберігаючи історію помилок та пропонуючи поради для усунення слабких місць.

Таким чином, розробка інтелектуального діалогового асистента є ефективним способом вирішення проблеми нестачі розмовної практики, що дозволяє користувачам удосконалювати розмовні навички українською мовою в зручному форматі з персональним підходом і об'єктивною оцінкою результатів.

1.3.2 Особливості створення програмного забезпечення

Для глибшого розуміння задачі розглянемо найважливіші особливості створення такого програмного забезпечення як діалогова підсистема робота-асистента. Першою суттєвою особливістю є комплексність аналізу усного мовлення, яка включає одночасну перевірку граматичної та лексичної правильності, включно з перевіркою на русизми, кальку та відповідність розповіді темі. Крім того, українська мова має низку унікальних лінгвістичних особливостей

(наприклад, сім відмінків, вільний порядок слів у реченні), що ускладнює завдання автоматичного аналізу [2].

Також, однією з проблем задачі є формалізація мовних норм та представлення їх у вигляді алгоритмів, зрозумілих для штучного інтелекту. Оскільки правила мови постійно змінюються і містять багато винятків із правил, створення універсального алгоритму аналізу є об’єктивно складною задачею.

Реалізація системи розпізнання мови (ASR) для української мови у діалоговому асистенті стикається з низкою суттєвих технічних викликів. Першою ключовою проблемою є обмежена доступність якісних навчальних даних для української мови порівняно з більш поширеними мовами. Корпуси текстів та аудіозаписів часто недостатньо великі або недостатньо різноманітні, що обмежує можливості тренування точних моделей. Серед інших проблем: чутливість до фонового шуму, варіативність темпу мовлення, особливості артикуляції та різних вимов [3].

Потокова обробка повідомлень, яку вимагає створення такої підсистеми має високу обчислювальну складність, що створює необхідність пошуку балансу між точністю та швидкодією.

1.3.3 Актуальність розробки інтелектуального діалогового асистента для вдосконалення розмовних навичок українською мовою

Для розуміння чому задача є суттєвою та потребує вирішення саме зараз розглянемо актуальну ситуацію та нагальні питання.

У сучасних умовах, коли Україна відстоює свою незалежність та культурну ідентичність, розвиток української мови набуває особливого значення. Останні роки свідчать про значне зростання інтересу та потреби до вивчення української мови серед громадян України. Однак існуючі методи навчання часто виявляються недостатньо ефективними для розвитку розмовних навичок через брак мовної практики та доступність кваліфікованих викладачів.

Сьогодні, коли мільйони українців змушені перебувати за кордоном через війну, збереження зв'язку з рідною культурою через мову стає особливо актуальним. За даними досліджень, понад 60% української діаспори відзначають труднощі з підтримкою рівня володіння рідною мовою. Водночас всередині країни спостерігається активний процес переходу на українську мову в усіх сферах життя.

Розробка інтелектуального діалогового агента дозволить вирішити такі ключові питання як: забезпечення доступності мовної практики для всіх бажаючих, незалежно від місця проживання чи фінансових можливостей, подолання мовного бар’єру для всіх категорій населення.

Особливої актуальності проєкт набуває в умовах відродження національної ідентичності, інтеграції внутрішньо переміщених осіб, збереження зв’язку з рідною культурою для української діаспори за кордоном.

Сучасні технології штучного інтелекту дозволяють створити інструмент, який забезпечує якісний зворотний зв’язок, доступний у будь який час та в будь якому місці та персоналізованість, що полегшує вирішення низки питань, пов’язаних з проблемами, описаними вище [2].

Таким чином, розробка інтелектуального діалогового агента для вдосконалення розмовних навичок українською мовою є надзвичайно актуальною та соціально важливою задачею, яка відповідає сучасним викликам і сприятиме зміцненню української державності через розвиток мовної культури. Реалізація цього проекту стане важливим кроком у процесі об’єднання українського суспільства та збереження національної ідентичності в умовах сьогодення.

1.4 Існуючі підходи створення інтелектуального діалогового асистента для вдосконалення розмовних навичок

Після формалізації задачі та розуміння суті проблеми необхідно проаналізувати існуючі підходи до створення інтелектуального діалогового асистента для покращення розмовних навичок.

Ця задача належить до класу складних задач обробки природної мови, які вимагають розуміння контексту та генерації відповідей. Одним із провідних підходів є використання великих мовних моделей (Large Language Models), які навчаються на великих корпусах текстів та здатні генерувати відповіді подібні до людських, розпізнавати помилки, адаптувати стиль комунікації та підтримувати діалог у межах заданого контексту. Такі моделі, як GPT, PaLM, LLaMA чи Claude, вже демонструють високі результати у завданнях мовного навчання завдяки своїй здатності не лише відповідати на питання, а й вести навчальний діалог, виправляти помилки, формулювати персоналізовані поради.

Окрім цього, широко застосовуються моделі обробки природної мови (Natural Language Processing), які спеціалізуються на конкретних підзадачах, таких як розпізнавання намірів користувача, класифікація тематики висловлювань, виявлення граматичних та лексичних помилок, побудова структури діалогу.

Іншим підходом є використання діалогових систем на основі сценаріїв (rule-based dialogue systems), які будуються за заздалегідь визначеними правилами та шаблонами. Вони дають змогу реалізувати контрольований навчальний процес із чіткою структурою, але мають обмеження щодо гнучкості та варіативності відповідей.

Імітаційні моделі також відіграють важливу роль у навчанні мовам, оскільки дозволяють створити симульоване середовище для тренування діалогів на різні теми з поступовим ускладненням ситуацій.

1.4.1 Аналіз та вибір оптимального методу

У кожного з розглянутих підходів до створення інтелектуального діалогового асистента для покращення розмовних навичок є свої переваги та недоліки. Вибір оптимального методу залежить від цілей системи, очікуваного рівня адаптивності, гнучкості у веденні діалогу, а також від доступних обчислювальних ресурсів. Наприклад, класичні rule-based системи дозволяють реалізувати чітко

структуровані сценарії навчання, проте вони мають суттєві обмеження у варіативності діалогів та не здатні забезпечити живу, природну комунікацію.

Великі мовні моделі (LLM), навчені на масивних корпусах текстів, можуть генерувати гнучкі, контекстно доречні відповіді, виявляти мовні помилки та забезпечувати ефективний зворотний зв'язок. Завдяки своїм генеративним можливостям LLM дозволяють реалізувати більш природну взаємодію, що значно підвищує мотивацію користувача до мовної практики.

Одним із важливих компонентів системи, орієнтованої на роботу з розмовними навичками, є розпізнавання мовлення, оскільки користувач взаємодіє з асистентом голосом. Для цього використовуються технології автоматичного розпізнавання мовлення (ASR), які перетворюють усне мовлення на текст для подальшої обробки. Сучасні ASR-моделі на базі глибокого навчання, зокрема Whisper від OpenAI або wav2vec від Meta, демонструють високу точність навіть у складних умовах, що дозволяє ефективно інтегрувати голосову взаємодію в інтелектуальну систему навчання [4].

Підсумовуючи, можна стверджувати, що поєднання великої мовної моделі та автоматичного розпізнавання мовлення є оптимальним методом для створення інтелектуального діалогового асистента. Такий гібридний підхід відповідає сучасним технологічним можливостям, дозволяє забезпечити адаптивну та ефективну взаємодію з користувачем та сприяє вдосконаленню розмовних навичок користувача.

1.5 Існуючі реалізації мовних платформ з використанням інтелектуальних діалогових агентів

Після обрання підходу до створення інтелектуального діалогового асистента в попередньому підрозділі, було визначено ряд існуючих реалізацій. В цьому підрозділі будуть оглянуті існуючі реалізації різних підходів, їх сильні сторони та недоліки, щоб врахувати їх під час розробки власної реалізації та проаналізувати поточний стан справ у галузі мовної освіти.

1.5.1 Duolingo

Duolingo – один з найпопулярніших додатків для вивчення мов на сучасному ринку, який використовує технології штучного інтелекту для оптимізації процесу навчання. Особливу увагу слід приділити інтелектуальному агенту Duolingo, з я ким користувачі можуть проводити віртуальні дзвінки для практики розмовних навичок [5].

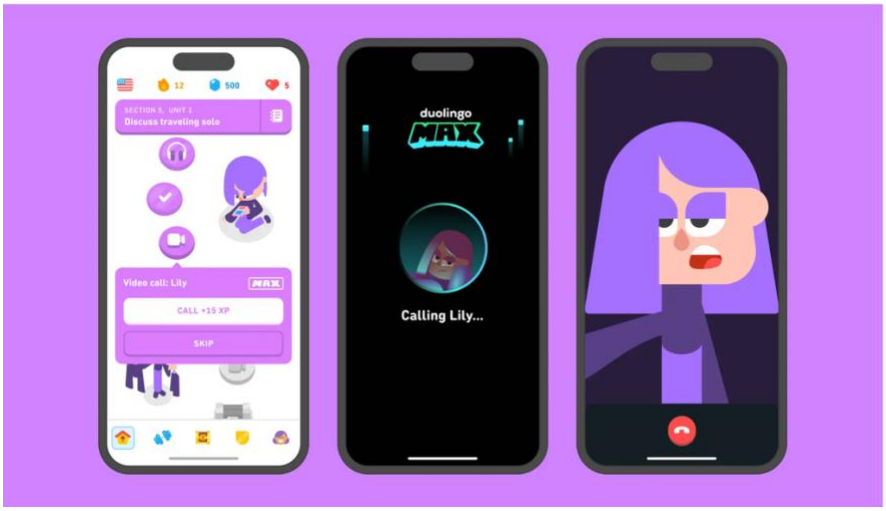


Рисунок 1.1 – Екрани мобільного застосунку Duolingo з викликом віртуального асистента для практики розмовних навичок у форматі відео-дзвінку

Даний віртуальний співрозмовник використовує технології генеративного штучного інтелекту для створення природних діалогів, що імітують реальне спілкування. Під час дзвінку система проводить аналіз вимови користувача, граматичну структуру речень та лексичний запас. Після дзвінку клієнту надається текстовий варіант розмови, де він може перевірити точність почутого віртуальним асистентом.

Недоліком підходу є те, що користувач не отримує зворотного зв’язку щодо своїх помилок, а також на точність почутих асистентом слів можуть впливати зовнішні фактори, такі як шуми та недостатня адаптивність системи до різних акцентів та манери вимови.

Окрім віртуальних дзвінків Duolingo використовує машинне навчання для аналізу прогресу кожного користувача. Алгоритми платформи визначають закономірності засвоєння матеріалу, виявляють найчастіші помилки та підлаштовують програму навчання. Підтримка української мови для інтелектуального агента відсутня.

1.5.2 Babbel

Babbel – популярний додаток для вивчення мов, що виділяється системним підходом до застосування ШІ для оптимізації навчального процесу. Цей додаток фокусується на аналізі вимови користувачів та точній корекції помилок за допомогою технологій машинного навчання та розпізнавання природнього мовлення [6].

Під час виконання вправ студентами, система аналізує інтонаційні патерни, ритм та природність звучання фраз. Після кожної вправи користувач отримує зворотній зв’язок, де вказуються проблемні місця та пропонуються вправи для їх усунення.

Окремою перевагою Babbel є його адаптивна система повторення, яка побудована на алгоритмах прогнозування забування інформації. Система аналізує наскільки швидко користувач запам’ятовує нові слова і автоматично визначає оптимальні моменти для їх повторення, що дозволяє максимізувати ефективність закріплення знань.

Технології Babbel мають певні обмеження, що стосуються якості розпізнавання мовлення, а також відсутності повноцінних віртуальних діалогів з інтелектуальним асистентом. Підтримка української мови для інтелектуального агента відсутня.

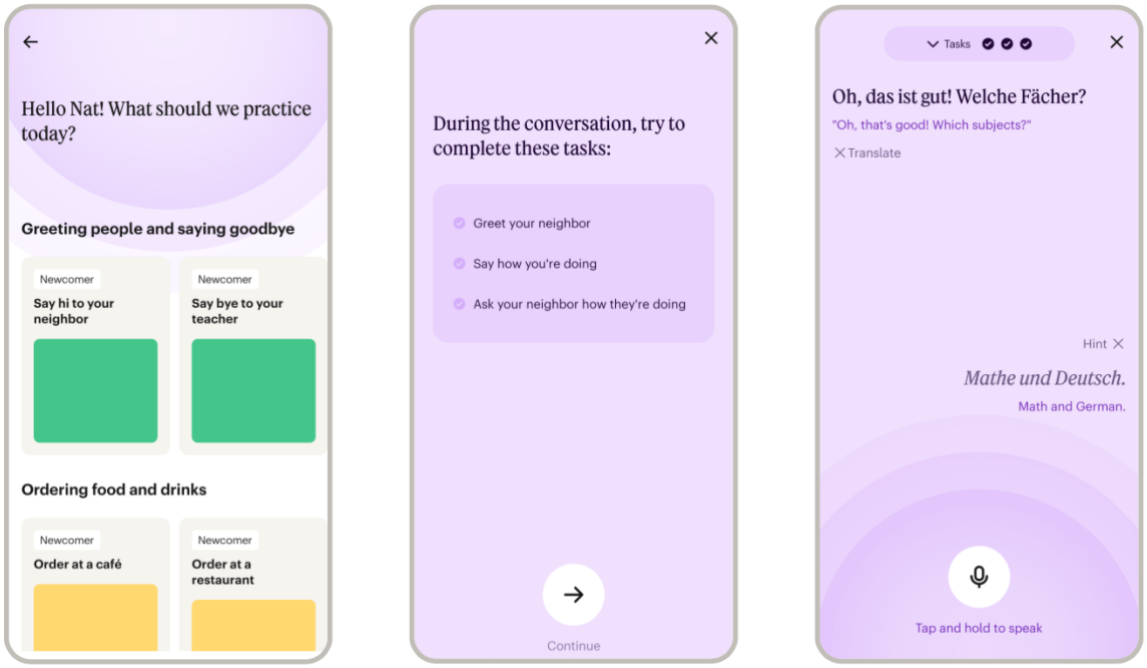


Рисунок 1.2 – Екрани мобільного застосунку Babbel

1.5.3 Memrise

Memrise – додаток для вивчення мов, який акцентується на контекстному засвоєнні лексики, використовуючи відео з носіями мови, що допомагає користувачам розвивати не лише словниковий запас, але й розуміння живої мови.

Ця платформа використовує систему інтервальних повторень, що реалізована за допомогою алгоритмів машинного навчання для визначення підходящих моментів для повторення слів. Штучний інтелект аналізує індивідуальні показники користувача, такі як: швидкість засвоєння слів, частоту помилок, час необхідний для пригадування [7].

Використовуються також технології розпізнавання мови для оцінки вимови, але вони недостатньо ефективно працюють з нестандартним акцентами або зашвидким темпом мовлення. Підтримка української мови для інтелектуального агента відсутня.

Рисунок 1.3 – Екран мобільного застосунку Memrise з чат-ботом

1.5.4 Порівняння власної роботи з наявними

Загалом, всі розглянуті реалізації мають свої переваги та недоліки. Duolingo забезпечує інтерактивність через віртуальні дзвінки з генеративним AI, Babbel пропонує адаптивне повторення матеріалу з акцентом на вимову, а Memrise використовує відео носіїв мови для контекстного засвоєння лексики. Однак усі ці платформи не підтримують повноцінне покращення розмовних навичок саме українською мовою, а також мають обмеження у гнучкості аналізу помилок користувача.

Реалізація, представлена у цій роботі, вирізняється з-поміж наявних рішень низкою характеристик. По-перше, вона зосереджена виключно на вдосконаленні розмовних навичок українською мовою, що є важливим з огляду на майже повну

відсутність підтримки української мови у популярних світових платформах. По-друге, для розпізнавання мовлення буде використовуватись технологія автоматичного розпізнавання мовлення (ASR), яка перетворює голосове повідомлення користувача на текст з урахуванням особливостей української мови.

На наступному етапі цей текст буде проаналізовано за допомогою великої мовної моделі, що дозволить точно і глибоко виявляти граматичні, лексичні та стилістичні помилки у мовленні. На відміну від Duolingo, де користувач отримує лише транскрипцію розмови, запропонована система буде надавати структурований зворотний зв'язок: виділення помилок у мовленні, пояснення суті помилки та персоналізовані рекомендації щодо тем для повторення, що дозволить здійснювати цілеспрямовану корекцію.

Крім того, система передбачає реалізацію модуля "робота над помилками", що дозволить користувачеві переглядати історію своїх помилок за датами, і таким чином стежити за власним прогресом. Такий механізм забезпечить адаптивне навчання, орієнтоване на індивідуальні слабкі сторони кожного користувача, чого не пропонують наявні рішення.

Загалом, дана реалізація полягає у поєднанні технологій ASR та великої мовної моделі для глибокого аналізу усного мовлення українською мовою та формування рекомендацій, що робить її унікальною на ринку та відкриває перспективи для розвитку персоналізованого навчання української мови.

1.6 Формулювання конкретних вимог до розробки діалогової підсистеми інтелектуального агента

Після аналізу існуючих підходів та реалізацій на ринку потрібно сформулювати конкретні вимоги до власного програмного забезпечення.

Функціональні вимоги:

– система повинна ідентифікувати користувачів за допомогою ідентифікатора користувача;

– система повинна підтримувати створення індивідуального профілю з історією взаємодій;

– система повинна надавати завдання відповідно до обраної користувачем теми;

– система повинна приймати голосові повідомлення користувача;

– система повинна розпізнавати та транскрибувати голосове повідомлення користувача;

– система повинна передавати транскрипт голосового повідомлення користувача до моделі через API;

– система повинна здійснювати якісний аналіз граматичних та лексичних помилок із прикладами виправлень;

– система повинна визначати чи відповідає відповідь темі завдання;

– система повинна зберігати помилки та оцінки користувача;

– система повинна підтримувати ведення індивідуальної статистики успішності користувача;

– система повинна відображати статистику успішності у вигляді графіку;

– система повинна відображати виявлені граматичні та лексичні помилки з прикладами виправлення у окремому меню;

– система повинна мати інтуїтивно зрозумілий інтерфейс з кнопками та підказками;

– система повинна обробляти помилки та надавати зворотній зв’язок користувачу у випадку помилки;

Нефункціональні вимоги до програмного забезпечення:

– ефективність та швидкість роботи програмного забезпечення;

– система повинна забезпечувати безперервну роботу;

– стійкість та надійність програмного забезпечення;

– інтерфейс системи повинен бути інтуїтивно зрозумілим та зручним для користувачів з різним рівнем технічної підготовки;

– сумісність з різними операційними системами та обладнанням;

– система повинна забезпечувати конфіденційність даних користувачів шляхом обмеженого зберігання даних;

Висновки до розділу 1

Виконавши аналіз предметної області практики та вдосконалення розмовних навичок українською мовою у підрозділі 1.1, було встановлено, що опанування усного мовлення потребує регулярності, конструктивного зворотного зв’язку та доступності інструментів для ефективної практики. Особливу увагу було приділено труднощам, які виникають у процесі навчання: психологічний бар’єр, відсутність україномовного середовища, брак зворотного зв’язку щодо помилок, а також недостатній акцент на саме розмовну практику в більшості існуючих освітніх методиках.

Постановка мети та задач дослідження в підрозділі 1.2 дала змогу сформувати чітке бачення для чого потрібне створення інтелектуального діалогового асистента, який сприятиме покращенню розмовних навичок українською мовою. Було визначено конкретні завдання, реалізація яких дозволяє побудувати функціональну, адаптивну та корисну систему з персоналізованим підходом до навчання.

У підрозділі 1.3 було окреслено особливості розробки програмного забезпечення, його призначення, технічні та лінгвістичні виклики. Зокрема, наголошено на складності автоматичного аналізу українського мовлення через граматичні особливості мови та обмежену кількість якісних навчальних даних для моделей штучного інтелекту. Було обґрунтовано актуальність розробки саме україномовного рішення в умовах підвищеного запиту на вивчення української мови та розвитку української культури.

У підрозділі 1.4 проведено аналіз існуючих підходів до створення діалогових систем: rule-based рішень, імітаційних моделей, NLP-модулів та великих мовних моделей. Обґрунтовано вибір комбінації ASR та великої мовної моделі як оптимального підходу для створення системи, яка здатна аналізувати мовлення

користувача, розпізнавати помилки та генерувати персоналізовані поради для подальшого вдосконалення.

У підрозділі 1.5 проведено порівняння з популярними платформами Duolingo, Babbel та Memrise. Було встановлено, що жодна з них не підтримує повноцінну розмовну практику українською мовою та не надає детального аналізу помилок користувача з рекомендаціями. На відміну від них, запропонована реалізація базується на аналізі усного мовлення українською мовою з подальшим наданням структурованого зворотного зв’язку, що дозволяє ефективно працювати над помилками.

У підрозділі 1.6 було сформульовано технічні вимоги до розроблюваного програмного забезпечення, які включають як функціональні можливості системи, так і нефункціональні характеристики, необхідні для її надійної, безпечної та зручної роботи.

Отже, підсумовуючи зміст першого розділу, було проведено комплексний аналіз предметної області, сформульовано мету та задачі дослідження, описано особливості реалізації інтелектуального діалогового асистента, обґрунтовано вибір оптимального підходу до створення системи, проаналізовано існуючі рішення та сформульовано технічні вимоги. Це дозволило чітко визначити концепцію, архітектуру та напрям подальшої розробки, які будуть розглянуті у наступних розділах даного дипломного проєкт.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

Провівши аналіз предметної області та поставивши конкретну задачу у попередньому розділі, потрібно розглянути технології, алгоритми та засоби програмування, а також окреслити основні поняття, які потрібні для розуміння процесу розробки.

2.1 Розгляд існуючих технологій для розпізнавання природньої мови

Першочерговою задачею у процесі створення інтелектуального діалогового асистента, що працює з голосовими повідомленнями є вибір технології розпізнавання мовлення. Для цього розглянемо основні технології, що використовуються.

Сучасні технології розпізнавання природної мови (NLP) розвиваються стрімкими темпами, пропонуючи різноманітні підходи до обробки текстової та голосової інформації. Серед основних технологій варто виділити статистичні моделі, нейронні мережі та трансформери, кожна з яких має свої особливості застосування. Статистичні методи, такі як N-грами та приховані марковські моделі, були першими кроками в автоматизації обробки мови, вони базуються на аналізі ймовірностей появи слів і фраз у певних контекстах. Ці методи досі використовуються у простих системах, проте їхня ефективність обмежена через нездатність враховувати складні семантичні зв'язки в тексті.

Більш прогресивними є технології на основі нейронних мереж, зокрема рекурентні (RNN) та згорткові (CNN) нейронні мережі, які дозволяють краще обробляти послідовності слів і виявляти залежності між ними. Рекурентні мережі, особливо їхні модифікації з довгостроковою короткочасною пам'яттю (LSTM), ефективні для роботи з текстами, оскільки здатні запам'ятовувати контекст на довших відрізках. Однак їхня обчислювальна складність та проблеми з обробкою дуже довгих послідовностей залишаються суттєвими недоліками.

Трансформерні архітектури, такі як BERT, GPT та їхні аналоги використовують механізми уваги для аналізу залежностей між словами незалежно від їхньої позиції в тексті. Ці моделі демонструють високі результати у завданнях розуміння мови, генерації текстів та аналізу контексту. Вони здатні навчатися на величезних обсягах даних і успішно адаптуються до різних мов, включаючи українську. Проте їхній основний недолік – висока ресурсомісткість, що вимагає потужних обчислювальних систем для навчання та експлуатації [8].

Технології ASR працюють на основі комбінації акустичних та мовних моделей. Акустична модель відповідає за розпізнавання звукових паттернів, тоді як мовна модель покращує результати, враховуючи ймовірність послідовностей слів у мові. Сучасні системи використовують глибокі нейронні мережі, зокрема рекурентні (RNN) та трансформерні архітектури, що дозволяє досягати високої точності розпізнавання.

2.2 Вибір технології для розпізнавання мовлення для діалогового інтелектуального асистента

Після огляду існуючих технологій для розпізнавання мовлення, їх переваг та недоліків потрібно визначити найдоцільніше рішення для реалізації діалогового інтелектуального асистента для вдосконалення розмовних навичок українською мовою.

Найсучаснішим і найбільш перспективним підходом є використання трансформерних архітектур, що були адаптовані до задач розпізнавання мовлення. Моделі типу wav2vec 2.0 (Meta), Whisper (OpenAI) та інших аналогів демонструють високу точність навіть у випадках з низькою якістю запису або фоновим шумом. Whisper, зокрема, вирізняється підтримкою великої кількості мов, включно з українською, що робить її особливо актуальною для даного проєкту. Вона навчається на багатомовних аудіо-даних і текстових транскриптах, що дозволяє враховувати фонетичні особливості української мови, розпізнавати нетипову вимову, змішування з іншими мовами та варіативність темпу мовлення. Крім того,

ці моделі працюють у режимі “end-to-end”, тобто не потребують попередньої побудови окремих мовної й акустичної моделей, що значно спрощує архітектуру проєкту [9].

Для реалізації інтелектуального діалогового асистента технологія ASR на основі трансформерів є оптимальним вибором. Вона дозволяє перетворювати усне мовлення на текст з високою точністю, навіть в умовах нестандартної вимови або шуму. Використання моделі Whisper, яка має відкритий код та доступна для адаптації під українську мову, надає можливість вбудувати цю модель без значних витрат на інфраструктуру.

Таким чином, з огляду на технічні вимоги, специфіку української мови та загальну архітектуру, для реалізації діалогового асистента з аналізом мовлення українською мовою доцільно обрати сучасну ASR-технологію на базі трансформерів, зокрема модель Whisper, яка забезпечує необхідний рівень точності, гнучкості та масштабованості.

2.3 Розгляд існуючих технологій для лінгвістичного аналізу

Наступним етапом після розпізнавання мови у контексті розробки інтелектуального діалогового агента для вдосконалення розмовних навичок є лінгвістичний аналіз повідомлення для виявлення граматичних та лексичних помилок. Розглянемо існуючі технології для аналізу повідомлень.

Одним із ключових етапів автоматизованого лінгвістичного аналізу є морфологічний розбір, який дозволяє визначити частини мови, відмінки, рід, число та інші граматичні категорії. Раніше це здійснювалось за допомогою словникових баз, таких як Hunspell або АОТ, але сучасні системи дедалі частіше використовують глибокі нейронні мережі, які дозволяють ефективніше обробляти нові слова, неформальні конструкції та орфографічні варіації. Наприклад, моделі типу BiLSTM або трансформери можуть враховувати контекст в реченні, що дає змогу уникати помилок при розборі багатозначних слів.

На вищому рівні аналізу - синтаксичному та семантичному застосовуються більш складні технології штучного інтелекту. Наприклад, dependency-parsing (побудова залежностей між словами) або constituency-parsing (визначення фразової структури) використовують глибокі нейронні моделі, які забезпечують точність навіть у мовах зі складним порядком слів, таких як українська. У цій сфері активно використовуються моделі, натреновані за допомогою фреймворків як spaCy, UDPipe, Stanza (від Stanford NLP), а також спеціалізовані українськомовні моделі, наприклад, від lang-uk або models від langmodels.com.

Окремим напрямом лінгвістичного аналізу є виявлення та класифікація граматичних і лексичних помилок. Для цього використовуються як класичні системи на базі правил, так і моделі з навчанням з підкріпленням або напівавтоматичним навчанням на корпусах з анотованими помилками. Такі підходи лежать в основі систем Grammarly, LanguageTool та подібних сервісів, що аналізують текст в реальному часі. В українському сегменті аналогічні задачі вирішуються у проєктах LanguaMetrics, Mova або модулях освітніх платформ типу "Освітній чат-бот".

Великі мовні моделі (Large Language Models, LLM), зокрема GPT (Generative Pre-trained Transformer) та її аналоги, відіграють ключову роль у сучасних технологіях лінгвістичного аналізу. В основі GPT лежить трансформерна архітектура, яка дозволяє моделі ефективно працювати з контекстом довгих текстів, враховуючи взаємозв’язки між словами незалежно від їхнього положення в реченні. Це забезпечується за допомогою механізму самоуваги (self-attention), що дає змогу моделі будувати глибоке семантичне представлення кожного елемента тексту в контексті всього повідомлення [10].

LLM попередньо навчаються на масивних багатомовних корпусах текстів, що дозволяє їм накопичити знання про синтаксис, морфологію, лексику, стилістику та прагматику мовлення. Моделі GPT, PaLM, Claude, LLaMA, а також багатомовні версії BERT (наприклад, mBERT або XLM-R) демонструють виняткову здатність виконувати задачі лінгвістичного аналізу, включаючи визначення частин мови, аналіз залежностей, розпізнавання іменованих сутностей (NER), виявлення

граматичних, стилістичних, семантичних і логічних помилок, а також переформулювання та спрощення текстів. На відміну від традиційних NLP-систем, які покладаються на жорстко задані правила, великі мовні моделі мають здатність до узагальнення та адаптації, що дозволяє їм працювати з неструктурованими, розмовними, діалектичними або помилковими мовними даними [11].

2.4 Вибір технології для лінгвістичного аналізу повідомлень користувачів діалогового інтелектуального асистента

Оглянувши актуальні технології для лінгвістичного аналізу, потрібно визначити підхід, що буде найбільш ефективним у рамках даного проєкту.

У процесі аналізу має бути виявлено граматичні та лексичні помилки, забезпечено їх коректне пояснення та згенеровано персоналізовані поради для користувача. Серед розглянутих у попередньому розділі підходів – від класичних словникових систем і моделей на основі правил до глибоких нейронних мереж і трансформерів – найбільш доцільним для досягнення мети проєкту є використання великої мовної моделі на базі GPT.

Традиційні системи, зокрема Hunspell, AOT або rule-based аналізатори, хоча й здатні виконувати базову морфологічну обробку та виявляти поширені помилки, не мають достатньої гнучкості для аналізу живого мовлення, варіативних синтаксичних конструкцій або контекстно залежних помилок. Вони не враховують індивідуальні стилістичні особливості висловлення, не адаптуються до нетипових форм і мають обмежену здатність до узагальнення.

Моделі на основі глибокого навчання, такі як BiLSTM або Stanza, забезпечують значно вищу точність у задачах синтаксичного та семантичного аналізу, але потребують складного налаштування, масштабування та інтеграції різних компонентів для виконання повноцінної діагностики помилок у тексті.

Натомість великі мовні моделі, такі як GPT, демонструють універсальність і високу ефективність у завданнях лінгвістичного аналізу. Завдяки трансформерній архітектурі GPT здатна враховувати широкий контекст висловлення, що особливо

важливо для виявлення складних граматичних або стилістичних відхилень. Вона не лише визначає частини мови, граматичні функції та синтаксичні зв'язки, але й розпізнає семантичні суперечності, логічні збої у висловленні, русизми, кальки та інші типові помилки. Крім того, GPT може не просто виявити помилку, а й пояснити її суть у доступній формі, запропонувати виправлення, дати приклади аналогічних конструкцій і сформувати рекомендацію щодо повторення відповідної граматичної теми або лексичного блоку [12].

Дана модель штучного інтелекту дозволяє реалізувати повний цикл лінгвістичного аналізу: від розпізнавання мовної структури до генерації пояснень і рекомендацій. На відміну від вузькоспеціалізованих моделей, GPT забезпечує гнучкість, масштабованість і якісний результат без необхідності ручної побудови лінгвістичних правил або великих корпусів помилок для донавчання.

Таким чином, вибір великої мовної моделі GPT для реалізації лінгвістичного аналізу повідомлень у проєкті є обґрунтованим з технологічної, функціональної та методичної точки зору.

2.5 Архітектура системи

Після огляду та вибору технологій для розпізнавання мовлення та лінгвістичного аналізу постає питання вибору архітектури системи, адже вона визначає ключові елементи системи, принципи їхньої взаємодії та загальну організацію структури.

Вибір архітектури грає вирішальну роль у розробці, оскільки вона забезпечує системі структурованість, масштабованість, надійність та зручність підтримки. Існує багато архітектурних підходів, кожен з яких має свої сильні та слабкі сторони. Серед найпоширеніших варіантів: монолітна, мікросервісна, клієнт-серверна, багаторівнева та REST-архітектури. Часто архітектурні моделі поєднуються або базуються одна на одній для ефективної реалізації програмних рішень.

2.5.1 Клієнт-серверна архітектура для діалогової підсистеми інтелектуального робота-асистента

Серед визначених раніше існуючих архітектур найбільш популярною є клієнт-серверна модель. У цій архітектурній системі присутні два ключові елементи: клієнт, який ініціює запит для отримання конкретних даних, і сервер, що обробляє ці запити та надає відповідну інформацію. Взаємодія між клієнтом і сервером здійснюється через інтернет-мережу [13].

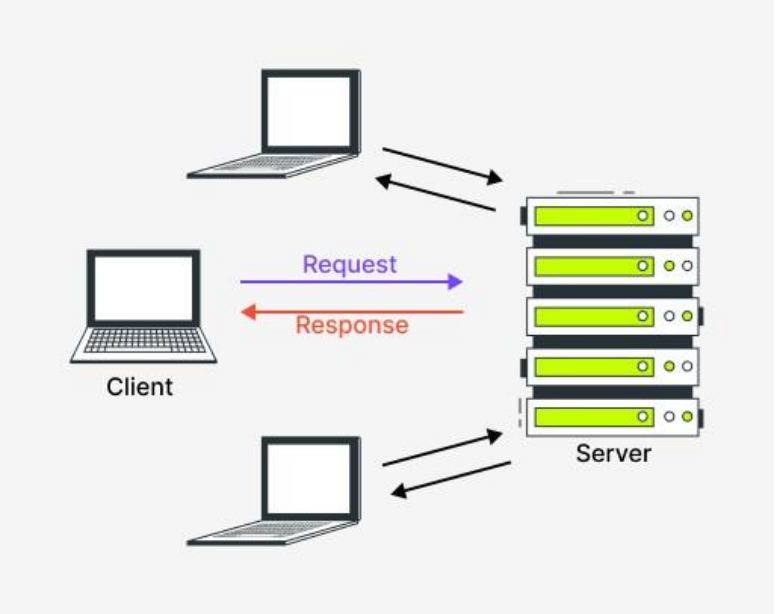


Рисунок 2.1 – Клієнт-серверна архітектура

Під час розробки діалогової підсистеми інтелектуального робота-асистента архітектура системи буде реалізована за трирівневою моделлю, де кожен рівень виконує чітко визначені функції.

Presentation Layer, представлений чат-ботом, відповідає за взаємодію з користувачем через зручний чат-інтерфейс. На цьому рівні буде відбуватись обробка базових команд, прийом голосових повідомлень та текстових відповідей, а також відображення результатів роботи системи. Використання Telegram API як клієнтської частини дозволяє забезпечити крос-платформенність рішення та мінімізувати витрати на розробку власного клієнтського додатку.

Business Logic Layer, що розміщений на серверній частині, є ядром системи та містить усю основну логіку роботи асистента. Тут буде відбуватись аналіз текстових повідомлень за допомогою комбінації API моделей штучного інтелекту

для розпізнавання мовлення та глибокого аналізу змісту, а також основна логіка функціоналу програми.

Data Layer буде реалізований на основі СУБД і відповідає за надійне зберігання всієї необхідної інформації. У базі даних будуть зберігатись транскрипти голосових повідомлень користувачів, детальна інформація про виявлені помилки з їх класифікацією, статистика успішності для кожного користувача, а також історія взаємодії з системою. Використання реляційної бази даних дозволить ефективно організувати структуру даних та забезпечить швидкий доступ до необхідної інформації при формуванні персоналізованих рекомендацій.

Клієнт-серверна архітектура в даному випадку особливо ефективна, оскільки дозволяє розподілити навантаження між клієнтською та серверною частинами. Важкі обчислювальні завдання, такі як аналіз мовлення за допомогою GPT чи генерація графіків, виконуються на сервері, що дозволяє зменшити вимоги до клієнтського пристрою. Одночасно чат-бот забезпечує зручний інтерфейс взаємодії, не вимагаючи від користувача встановлення додаткового програмного забезпечення. Така архітектура також спрощує процес масштабування системи - при збільшенні кількості користувачів можна просто додавати серверні потужності, не вносячи змін у клієнтську частину.

Додатковою перевагою обраної архітектури є підвищена безпека, оскільки всі конфіденційні дані користувачів зберігаються на сервері, а не на клієнтських пристроях. Реалізація трирівневої моделі з чітким розподілом функцій між рівнями забезпечує гнучкість системи та спрощує процес її подальшого розширення новим функціоналом.

2.6 Логічна модель даних діалогової підсистеми інтелектуального робота-асистента

Наступним етапом після визначення архітектури системи є проектування структури бази даних.

На цьому етапі використовується логічна модель даних, яка відображає інформацію про предметну область незалежно від фізичної реалізації. Ця модель включає сутності (наприклад, "Користувач" або "Помилки"), їх атрибути (такі як "Ідентифікатор" чи "Дата створення"), первинні та зовнішні ключі, а також нормалізовані зв’язки між сутностями. На відміну від фізичної моделі, у логічній не вказуються конкретні типи даних, але чітко визначаються всі необхідні взаємозв'язки.

Для візуалізації логічної моделі зазвичай використовують діаграми, які можуть бути представлені в різних нотаціях, таких як нотація Пітера Чена, нотація Мартіна ("вороняча лапка"), нотація Бахмана або нотація Баркера. У даному випадку для побудови діаграми було обрано нотацію Мартіна, оскільки вона дозволяє наочно відображати зв’язки між сутностями, особливо типи "один-до-багатьох".

Логічна модель слугує проміжною ланкою між концептуальним проектуванням, яке базується на бізнес-вимогах, і безпосередньою фізичною реалізацією в СУБД. Вона є основою для подальшої оптимізації бази даних, включаючи денормалізацію, індексацію та інші методи підвищення продуктивності.

Логічна модель даних представлена на кресленику ІК13.300БАК.006 Д1.

2.7 Діаграма послідовностей діалогової підсистеми інтелектуального робота-асистента

Після виокремлення ключових сутностей та побудови логічної моделі даних потрібно визначити основні сценарії поведінки системи та закономірності взаємодії між об’єктами підсистеми.

Одним із ключових елементів візуального моделювання програмної системи є діаграма послідовностей (sequence diagram), яка належить до стандарту UML (Unified Modeling Language) та дозволяє відобразити динаміку взаємодії між компонентами системи у часі [14].

Цей тип діаграм застосовується для опису сценаріїв поведінки системи та дозволяє чітко зрозуміти порядок обміну повідомленнями між об’єктами або підсистемами, що беруть участь у певному процесі [15].

У межах розробки інтелектуального діалогової підсистеми робота-асистента для вдосконалення розмовних навичок українською мовою діаграма послідовностей описує типовий сценарій використання системи: взаємодію між користувачем, чат-ботом, модулем автоматичного розпізнавання мовлення (ASR), великою мовною моделлю (GPT) та базою даних. Діаграма демонструє, як голосове повідомлення, надіслане користувачем, проходить через етапи обробки: конвертація та нормалізація аудіо, трансформація мовлення у текст за допомогою моделі Whisper, аналіз помилок за допомогою GPT-4, повернення оцінки та рекомендацій, а також збереження результатів у базі даних для подальшої роботи.

Побудована діаграма дозволяє візуалізувати логіку послідовного виконання операцій та переконатися, що кожен компонент системи виконує свою функцію відповідно до архітектурного задуму. Зокрема, вона відображає ініціацію взаємодії з боку користувача, обробку повідомлення чат-ботом, виклики зовнішніх API (ASR та GPT), повернення зворотного зв’язку, а також збереження усіх результатів у базі даних.

Таким чином, діаграма послідовностей виконує роль чіткого опису сценарію використання основної функції системи — аналізу мовлення користувача, і є важливим елементом документації, що забезпечує прозорість та зрозумілість внутрішньої логіки роботи діалогової підсистеми інтелектуального асистента.

Розроблена діаграма послідовностей для діалогової підсистеми інтелектуального робота-асистента наведена на кресленику ІК13.300БАК.006 Д4.

2.8 Діаграма варіантів використання діалогової підсистеми інтелектуального робота-асистента

Високорівневий опис функціональності системи на базі сформованих раніше функціональних вимог, є важливим кроком для опису та проектування системи.

Діаграма варіантів використання в UML фокусується на взаємодії між користувачами (акторами) та системою без внутрішньої реалізації. Вона надає зручний графічний спосіб візуалізації основних сценаріїв використання системи, доповнюючи текстовий опис функціональних вимог наочною схемою, яка чітко відображає межі системи та ключові взаємодії.

Основу діаграми складають чотири фундаментальні елементи: актори (зовнішні сутності, які можуть бути як людьми, так і іншими системами), варіанти використання (функціональні можливості, які надає система), межі системи (чітке визначення її кордонів) та зв'язки між ними, що показують характер взаємодії.

Діаграма варіантів використання представлена на кресленику ІА02.220БАК.006 Д2.

2.9 Проектування роботизованої системи інтелектуального асистента для вдосконалення розмовних навичок українською мовою

2.9.1 Структура роботизованої системи

Після вибору методів реалізації та визначення архітектури підсистеми, необхідно перейти до етапу проектування роботизованої системи інтелектуального асистента. Така система повинна забезпечувати не лише інтерактивну взаємодію з користувачем, а й створювати наближене до реального середовище для мовної практики.

Робот-асистент має бути оснащений необхідним набором апаратних компонентів, які забезпечують голосовий ввід, візуальне відображення інформації та зручне керування.

Основними складовими є мікрофон, дисплей та обчислювальний модуль з інтегрованим інтернет-з’єднанням.

Мікрофон відповідає за прийом слів користувача. Важливо, щоб мікрофон мав високу чутливість, підтримував шумозаглушення та міг коректно працювати на відстані без суттєвих втрат якості аудіосигналу.

Дисплей виступає як канал комунікації, надаючи текстовий зворотний зв’язок, відображаючи приклади виправлень, статистику успішності користувача. Він повинен бути сенсорним, що надає можливість взаємодії: користувач зможе обирати теми, переглядати історію помилок та статистику успішності. Розмір екрана повинен забезпечувати зручне сприйняття тексту з відстані близько одного метра.

Центральною частиною системи є обчислювальний модуль, до якого підключаються всі згадані пристрої. Цей модуль виконує передачу інформації на сервер для трансформації голосу у текст за допомогою ASR та на інтегровану LLM-модель для лінгвістичного аналізу, після чого отримує зворотний зв’язок. З'єднання має бути захищеним, передбачено використання HTTPS з авторизацією на основі API-ключів.

Пристрій повинен мати автономне живлення з можливістю роботи від акумулятора або портативного джерела живлення, що дозволяє використовувати його як стаціонарно, так і в мобільному режимі, наприклад, у навчальному класі або вдома.

Вигляд структури такої системи можна побачити на рисунку 2.2.

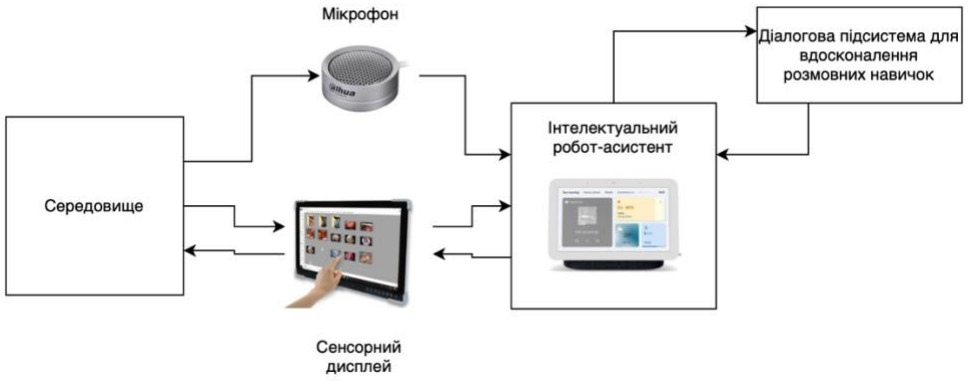


Рисунок 2.2 – Структура роботизованої системи

Така конфігурація забезпечить повноцінну взаємодію з користувачем, створюючи умови для регулярної практики мовлення з автоматичним аналізом та

навчальними рекомендаціями, що є основою ефективного вдосконалення розмовних навичок.

2.9.2 Структурна схема інтелектуального робота-асистента

Наступним кроком після окреслення загальної структури роботизованої системи є створення структурної схеми інтелектуального робота-асистента для вдосконалення розмовних навичок українською мовою.

Блок керування та попередньої обробки відповідає за локальне виконання ресурсомістких операцій, включаючи процесор для фільтрації та підготовки аудіоданих перед передачею, буфер тимчасового зберігання для оптимізації потоків даних та модуль керування інтерфейсом, який синхронізує роботу всіх компонентів.

Блок вводу/виводу включає високочутливий мікрофон з алгоритмами шумозаглушення для чіткого захоплення мовлення та сенсорний дисплей з інтуїтивним інтерфейсом для візуальної комунікації.

Мережева взаємодія реалізована через спеціалізований блок, що складається з Wi-Fi/4G модуля для бездротового підключення, шифрувального процесора з підтримкою сучасних протоколів безпеки (TLS/HTTPS) та API-клієнта, який забезпечує стабільну комунікацію з хмарними сервісами.

Живлення системи організоване за допомогою автономного блока, що включає акумуляторну батарею з розумним контролером заряду, який оптимізує енергоспоживання та забезпечує тривалу роботу без підзарядки.

Ця модульна архітектура дозволяє ефективно поєднувати локальну обробку даних для швидкої реакції та хмарні обчислення, забезпечуючи при цьому мінімальні затримки та високу стабільність роботи. Всі компоненти інтегровані в єдину систему з продуманими інтерфейсами взаємодії, що дозволяє масштабувати функціонал без змін апаратної частини.

Сформована та описана вище структурна схема інтелектуального робота-асистента представлена на кресленику ІА02.220БАК.006 Д3.

Схема враховує сучасні тенденції розподілених обчислень, де основне навантаження лягає на серверну частину, а клієнтський пристрій виконує роль зручного інтерфейсу між користувачем та хмарними алгоритмами обробки мови.

Висновки до розділу 2

Провівши детальний аналіз інформаційного забезпечення діалогової підсистеми інтелектуального асистента, було обґрунтовано вибір сучасних технологій і підходів, необхідних для реалізації системи вдосконалення розмовних навичок українською мовою.

У підрозділі 2.1 було розглянуто основні технології розпізнавання природної мови, включаючи статистичні методи, нейронні мережі та трансформерні архітектури. Було встановлено, що найбільш ефективними у контексті поставленого завдання є трансформерні моделі, здатні працювати з українською мовою.

У підрозділі 2.2 було здійснено обґрунтування вибору технології автоматичного розпізнавання мовлення для проєкту. Найдоцільнішим виявилося використання моделі Whisper, яка підтримує українську мову, демонструє високу точність у складних умовах та не потребує побудови окремих мовної й акустичної моделей, що значно спрощує інтеграцію в систему.

У підрозділі 2.3 було проаналізовано існуючі технології для лінгвістичного аналізу, серед яких – класичні словникові бази, rule-based системи, нейронні мережі та трансформерні моделі.

У підрозділі 2.4, після порівняння розглянутих підходів, було обґрунтовано вибір великої мовної моделі GPT для реалізації повноцінного лінгвістичного аналізу. Ця модель забезпечує високий рівень точності виявлення помилок, пояснення їх суті та формування персоналізованих порад, що є важливою умовою ефективного навчання.

У підрозділі 2.5 розглянуто можливі архітектурні підходи та обґрунтовано доцільність використання клієнт-серверної архітектури для реалізації діалогової

підсистеми, яка дозволяє ефективно розподілити обчислювальне навантаження між сервером і чат-ботом, що слугує інтерфейсом взаємодії з користувачем.

У підрозділі 2.6 було побудовано логічну модель даних, яка відображає основні сутності, їх атрибути та зв’язки, необхідні для зберігання транскриптів, помилок, статистики та історії взаємодії з користувачем.

Підрозділи 2.7 і 2.8 присвячено створенню діаграми компонентів та діаграми варіантів використання, що формалізують внутрішню організацію системи та ілюструють ключові сценарії взаємодії з користувачем.

У підрозділі 2.9 було спроєктовано роботизовану версію асистента, в якій описано основні апаратні компоненти, включаючи мікрофон, дисплей та обчислювальний модуль, що забезпечують взаємодію з користувачем та візуальне відображення інформації.

У підрозділі 2.9.2 представлено структурну схему інтелектуального робота-асистента, де описано функціональні блоки системи: керування, вводу/виводу, обробки даних та живлення.

Отже, у даному розділі було визначено інформаційне середовище та засоби реалізації системи, обґрунтовано вибір відповідних технологій розпізнавання мовлення і лінгвістичного аналізу, спроєктовано архітектуру, логіку зберігання даних, компоненти програмної і апаратної частини, що в комплексі створює основу для програмної реалізації діалогової підсистеми інтелектуального асистента, орієнтованого на вдосконалення розмовних навичок українською мовою.

3 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Після окреслення інформаційного забезпечення проекту з повним розумінням засобів, технологій та архітектурних рішень, які будуть використані у роботі, можна починати програмну реалізацію діалогової підсистеми.

3.1 Засоби для реалізації діалогової підсистеми інтелектуального асистента

3.1.1 Мова програмування Python

Для реалізації продукту було обрано мову програмування Python. Вона має велику кількість бібліотек для роботи зі штучним інтелектом, а також простий та зрозумілий синтаксис, що полегшує розробку та подальшу підтримку коду. Код, написаний на Python може виконуватись на різних операційних системах, що безумовно є великою перевагою.

3.1.2 Система керування базами даних PostgreSQL

PostgreSQL — це відкрита система керування реляційними базами даних (СКБД), яка поєднує високу продуктивність, надійність та гнучкість. Вона підтримує стандарти SQL, включаючи складні запити, індексацію, транзакції з рівнями ізоляції, а також розширені функції, такі як JSON-об'єкти, геопросторові дані та повнотекстовий пошук. PostgreSQL відома своєю масштабованістю, здатністю обробляти великі обсяги даних і підтримкою паралельних операцій [16].

Для проєкту інтелектуального діалогового асистента PostgreSQL є оптимальним рішенням завдяки таким перевагам: вона забезпечує надійне зберігання даних користувачів (транскрипції, історію помилок, статистику), підтримує складні запити для аналізу та генерації звітів, а також інтегрується з сучасними хмарними технологіями. Крім того, PostgreSQL має вбудовані механізми безпеки, включаючи шифрування та контроль доступу, що критично важливо для конфіденційності даних користувачів.

3.1.3 Середовище розробки Visual Studio Code

Було обрано таке середовище як Visual Studio Code (VSCode), оскільки це потужне інтегроване середовище розробки (IDE).

Visual Studio Code має низку переваг, серед яких: вбудований відлагоджувач Python, підтримка IntelliSense для автодоповнення коду, інтеграція з системами контролю версій, велика кількість корисних рішень для Python, зручний інтерфейс та можливість налаштування робочого простору. Враховуючи перераховані переваги VSCode є оптимальним вибором.

3.1.4 Бібліотеки та фреймвроки

Для реалізації діалогової підсистеми були обрані такі бібліотеки як: Telebot, OpenAI, Whisper, Psycopg2, Asyncio, Datetime, OS.

Telebot – одна з найпопулярніших бібліотек для створення Telegram-ботів на Python. Вона забезпечує взаємодію з клавіатурою та кнопками, обробку команд, повідомлень та голосових файлів, що дозволяє створювати зручний та функціональний інтерфейс для користувача [17].

Whisper – нейромережева бібліотека, яка надає інструменти для автоматичного розпізнавання мовлення (ASR). Вона забезпечує доступ до API для інтеграції моделі Whisper у програму.

OpenAI – бібліотека, що забезпечує інтеграцію штучного інтелекту у програму, а саме моделі GPT-4. Серед основних доступних функцій – текстова обробка (аналіз, генерація, класифікація), обробка зображень та інші [12].

Psycopg2 – бібліотека для роботи з базою даних, що забезпечує інтеграцію з системою управління базами даних PostgreSQL. Вона реалізує специфікацію DB API 2.0 та підтримує всі основні можливості PostgreSQL. Характеризується високою продуктивністю, асинхронністю та детальною обробкою помилок і попереджень [18].

Asyncio – бібліотека, що відповідає за асинхронну обробку задач, що дозволяє ефективно працювати з неблокуючими викликами в Python та забезпечує плавну взаємодію з користувачем.

OS – вбудований модуль Python, що надає інтерфейс взаємодії з операційною системою і дозволяє працювати з файловою системою, отримувати інформацію від середовища та управляти процесами.

Datetime – модуль, що надає класи для роботи з даними та часом у Python. Він дозволяє виконувати операції з датами, часами та часовими інтервалами.

3.2 Розробка програмного інтерфейсу

Програмний інтерфейс для діалогового асистента реалізовано за допомогою бібліотеки telebot, яка забезпечує інтерактивну взаємодію з користувачем через Telegram. Основний функціонал включає обробку текстових команд, голосових повідомлень та відображення результатів аналізу.

Головне меню інтерфейсу надає користувачеві чотири опції: «Обрати тему», «Успішність», «Робота над помилками» та «Фразеологізм дня».

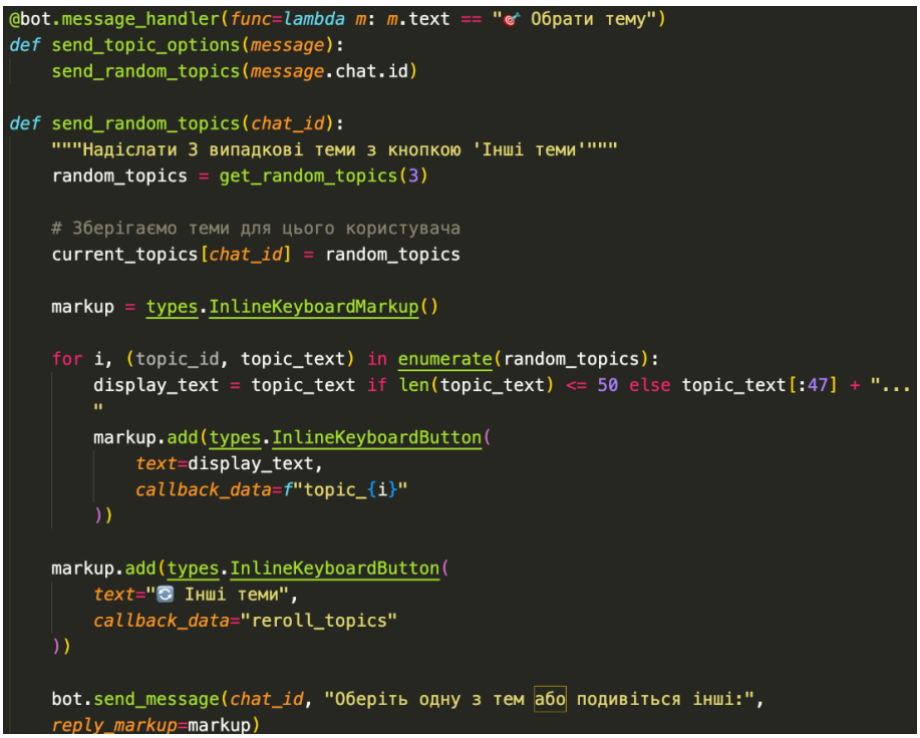
Реалізацію головного меню представлено на рисунку 3.1 .

```
@bot.message_handler(commands=['start'])
def handle_start(message):
    user_id = message.from_user.id
    save_user(user_id)
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    markup.row("🎯 Обрати тему", "📊 Успішність")
    markup.row("🛠 Робота над помилками", "📚 Фразеологізм дня")
    bot.send_message(user_id, "👋 Вітаю! Я допоможу вам вдосконалити розмовну
    українську.", reply_markup=markup)
```

Рисунок 3.1 – Реалізація головного меню

Вибір теми реалізовано через інлайн-клавіатуру з можливістю перезавантаження списку тем (send_random_topics), що забезпечує зручність навігації.

Створення меню «Обрати тему» представлено на рисунку 3.2.

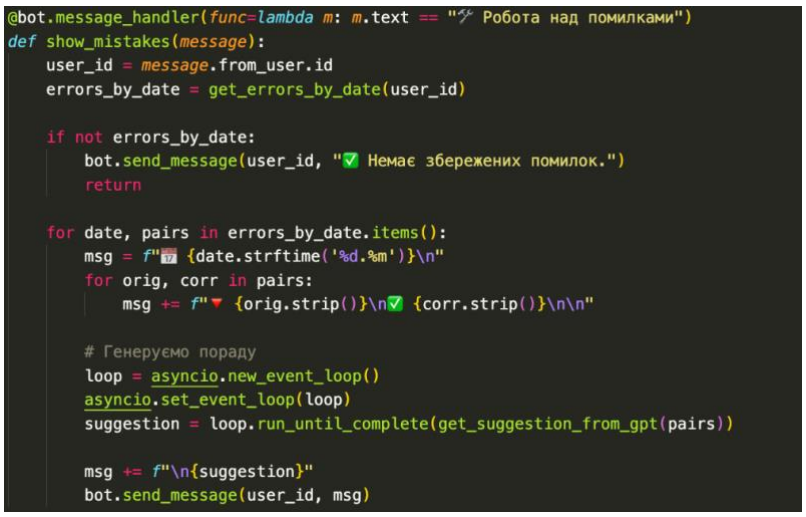


```python
@bot.message_handler(func=lambda m: m.text == "🎯 Обрати тему")
def send_topic_options(message):
    send_random_topics(message.chat.id)

def send_random_topics(chat_id):
    """Надіслати 3 випадкові теми з кнопкою 'Інші теми'"""
    random_topics = get_random_topics(3)

    # Зберігаємо теми для цього користувача
    current_topics[chat_id] = random_topics

    markup = types.InlineKeyboardMarkup()

    for i, (topic_id, topic_text) in enumerate(random_topics):
        display_text = topic_text if len(topic_text) <= 50 else topic_text[:47] + "...
        "
        markup.add(types.InlineKeyboardButton(
            text=display_text,
            callback_data=f"topic_{i}"
        ))

    markup.add(types.InlineKeyboardButton(
        text="🔄 Інші теми",
        callback_data="reroll_topics"
    ))

    bot.send_message(chat_id, "Оберіть одну з тем або подивіться інші:",
    reply_markup=markup)
```

Рисунок 3.2 – Реалізація меню «Обрати тему»

Після вибору теми бот очікує голосове повідомлення, яке обробляється за допомогою моделі Whisper для транскрибації. Для зручності користувача результати аналізу форматується у зрозумілий вигляд з оцінкою, списком помилок та рекомендаціями.

Створення меню «Робота над помилками» представлено на рисунку 3.3.



```python
@bot.message_handler(func=lambda m: m.text == "📌 Робота над помилками")
def show_mistakes(message):
    user_id = message.from_user.id
    errors_by_date = get_errors_by_date(user_id)

    if not errors_by_date:
        bot.send_message(user_id, "✅ Немає збережених помилок.")
        return

    for date, pairs in errors_by_date.items():
        msg = f"📅 {date.strftime('%d.%m')}\n"
        for orig, corr in pairs:
            msg += f"🔻 {orig.strip()}\n✅ {corr.strip()}\n\n"

        # Генеруємо пораду
        loop = asyncio.new_event_loop()
        asyncio.set_event_loop(loop)
        suggestion = loop.run_until_complete(get_suggestion_from_gpt(pairs))

        msg += f"\n{suggestion}"
        bot.send_message(user_id, msg)
```

Рисунок 3.3 – Реалізація меню «Робота над помилками»

Візуалізація прогресу реалізована за допомогою matplotlib, який генерує графік оцінок на основі даних з БД [19].

Створення меню «Успішність» представлено на рисунку 3.4.

```python
@bot.message_handler(func=lambda m: m.text == "📊 Успішність")
def handle_progress(message):
    user_id = message.from_user.id
    chart_today, chart_10days, today_avg, prev_avg = generate_progress_charts(user_id)

    if not chart_today and not chart_10days:
        bot.send_message(user_id, "ℹ️ Недостатньо даних для побудови статистики.")
        return

    # Відправляємо графік за сьогодні (якщо є)
    if chart_today:
        with open(chart_today, 'rb') as photo:
            bot.send_photo(user_id, photo, caption=f"📅 *Середній бал за сьогодні:* {today_avg}/10",
            parse_mode="Markdown")

    # Відправляємо графік за 10 днів (якщо є)
    if chart_10days:
        with open(chart_10days, 'rb') as photo:
            caption = f"📈 *Загальна статистика за 10 днів*"
            if prev_avg > 0:
                caption += f"\n*Середній бал за попередні дні:* {prev_avg}/10"
            bot.send_photo(user_id, photo, caption=caption, parse_mode="Markdown")

    # Якщо є тільки дані за сьогодні, але немає за попередні дні
    if chart_today and not chart_10days and today_avg > 0:
        bot.send_message(user_id, "📊 Поки що є тільки дані за сьогодні. Продовжуйте займатися для більш
        детальної статистики!")

    # Очищаємо файли після відправки
    try:
        if chart_today and os.path.exists(chart_today):
            os.remove(chart_today)
        if chart_10days and os.path.exists(chart_10days):
            os.remove(chart_10days)
    except:
        pass
```

Рисунок 3.4 – Реалізація меню «Успішність»

Інтерфейс також включає додаткове меню «Фразеологізм дня», яке щоденно надає користувачеві український фразеологізм з поясненням та прикладом використання.

Створення меню «Фразеологізм дня» представлено на рисунку 3.5.

```python
@bot.message_handler(func=lambda m: m.text == "📚 Фразеологізм дня")
def send_phraseologism(message):
    user_id = message.from_user.id
    phrase_data = get_todays_phraseologism(user_id)

    if phrase_data:
        _, phrase, meaning, example = phrase_data
        bot.send_message(user_id,
            f"📚 *Фразеологізм дня:*\n\n"
            f"🔹 *{phrase}*\n"
            f"💬 Значення: {meaning}\n"
            f"📝 Приклад: {example}",
            parse_mode="Markdown")
    else:
        bot.send_message(user_id, "✅ Ви вже гуру фразеологізмів!")
```

Рисунок 3.5 – Реалізація меню «Фразеологізм дня»

Загалом, інтерфейс забезпечує зручну та інтерактивну взаємодію та є адаптивним, що дозволяє використовувати інтелектуального діалогового асистента з будь-якого пристрою.

3.3 Налаштування обробки і нормалізація голосового повідомлення користувача

Обробка голосового повідомлення проходить кілька етапів нормалізації для забезпечення якісної транскрипції через Whisper модель.

Спочатку система завантажує голосове повідомлення від користувача через Telegram API, отримуючи файл у форматі OGG, який є стандартним форматом для голосових повідомлень у Telegram, після чого зберігає його локально як "voice.ogg". Далі відбувається критично важливий етап конвертації через ffmpeg, де файл перетворюється з OGG у WAV формат з конкретними параметрами оптимізації.

Параметр "-ar 16000" встановлює частоту дискретизації на 16 кГц, що є оптимальним балансом між якістю звуку та розміром файлу для розпізнавання мовлення, оскільки більшість систем ASR працюють найкраще саме з цією частотою.

Опція "-ac 1" конвертує аудіо в моно режим, видаляючи стерео канали, що спрощує обробку та зменшує обчислювальне навантаження без втрати якості розпізнавання мовлення.

Кодек "pcm_s16le" забезпечує несжате 16-бітне аудіо з little-endian байтовим порядком, що є стандартом для якісної обробки звуку в системах машинного навчання.

Аудіофільтр "highpass=f=80" видаляє низькочастотні шуми нижче 80 Гц, які зазвичай містять фонові шуми, гудіння електромережі та інші артефакти, що можуть заважати розпізнаванню мовлення.

Фільтр "lowpass=f=8000" обрізає високі частоти вище 8 кГц відповідно до теореми Найквіста для частоти дискретизації 16 кГц, видаляючи потенційні високочастотні шуми та артефакти.

Параметр "volume=2.0" подвоює гучність аудіо, що може допомогти при тихому записі, покращуючи відношення сигнал-шум для кращого розпізнавання.

Опції "-y" автоматично перезаписує вихідний файл без запиту підтвердження, а "-loglevel panic" придушує всі повідомлення ffmpeg окрім критичних помилок, забезпечуючи чистий вивід програми. Реалізація налаштування обробки і нормалізації голосового повідомлення користувача представлена на рисунку 3.6.

```python
@bot.message_handler(content_types=['voice'])
def handle_voice_message(message, topic=None):
    user_id = message.from_user.id
    save_user(user_id)
    bot.send_message(user_id, "🔎 Аналізую вашу відповідь...")

    # Завантаження голосового повідомлення
    file_info = bot.get_file(message.voice.file_id)
    file = bot.download_file(file_info.file_path)
    voice_duration = message.voice.duration

    with open("voice.ogg", "wb") as f:
        f.write(file)

    # Обробка звуку через ffmpeg
    ffmpeg_cmd = " ".join([
        "ffmpeg", "-i", "voice.ogg",
        "-ar", "16000", "-ac", "1",
        "-c:a", "pcm_s16le",
        "-af", "highpass=f=80,lowpass=f=8000,volume=2.0",
        "voice.wav",
        "-y", "-loglevel", "panic"
    ])
    os.system(ffmpeg_cmd)
```

Рисунок 3.6 – Реалізація обробки та нормалізації голосового повідомлення

Ця комплексна обробка забезпечує оптимальні умови для точного розпізнавання української мови, мінімізуючи вплив технічних фаткорів запису та максимізуючи якість вхідних даних для системи штучного інтелекту.

3.4 Імплементація та налаштування параметрів моделі Whisper

Імплементація Whisper моделі реалізована через завантаження попередньо навченої моделі розміру "small" на етапі ініціалізації програми, що забезпечує оптимальний баланс між швидкістю обробки та якістю розпізнавання мовлення для real-time застосунків.

Модель "small" містить приблизно 244 мільйони параметрів і забезпечує достатню точність для більшості завдань розпізнавання мовлення при відносно невеликих вимогах до обчислювальних ресурсів та пам'яті, що критично важливо для серверних застосунків з множинними користувачами.

Імплементація моделі Whisper представлена на рисунку 3.7.

```
model = whisper.load_model("small")
```

Рисунок 3.7 – Імплементація моделі Whisper

При виклику методу transcribe передається оброблений WAV файл разом з ключовими параметрами конфігурації, де language="uk" явно вказує модель на український контекст, активуючи спеціалізовані мовні моделі та словники для української мови, що значно покращує точність розпізнавання порівняно з автоматичним визначенням мови.

Параметр fp16=False вимикає використання половинної точності з плаваючою комою, замість цього використовуючи стандартну 32-бітну точність, що може незначно збільшити споживання пам'яті та час обчислень, але потенційно

покращує якість результатів, особливо для складних акустичних умов або менш поширених мов як українська.

Встановлення temperature=0.0 забезпечує детермінований процес декодування без випадковості в генерації тексту, що означає що одне і те ж аудіо завжди буде транскрибовано однаково, що важливо для постійності результатів та можливості відтворення результатів в системі оцінювання мовлення.

Результат транскрипції отримується через індекс 'text' з об'єкта result, який містить розпізнаний текст у вигляді рядка, готового для подальшого аналізу через GPT модель.

Налаштування параметрів моделі Whisper представлені на рисунку 3.8.

```
result = model.transcribe("voice.wav", language="uk", fp16=False, temperature=0.0)
transcript = result['text']

# Збереження транскрипції
save_transcript(user_id, topic, transcript, voice_duration)
```

Рисунок 3.8 – Налаштування моделі Whisper

3.5 Імплементація та налаштування параметрів моделі GPT-4

Імплементація GPT-4 моделі здійснюється через OpenAI API з детально налаштованим промптом для аналізу української мови, де ключовим елементом є структурований системний промпт, який визначає роль асистента як спеціаліста з аналізу українського мовлення та встановлює конкретні критерії оцінювання [12].

Імплементація моделі GPT-4 представлена на рисунку 3.9.

```
BOT_TOKEN = "7548200371:AAE5GPz5wDOjtMs_FSKhJOPVyrPH1vhkGSw"
OPENAI_API_KEY =
"sk-proj-hrPpbD5-c1YNrYafTWanom5iK5QihV7uvSx9W-goEgBIgkUJisGka-9
5qDwUEBTXSGan7jPhLKwHi1kYPS08lEQ0gCbaj0DLXqdfVaVtRYcCkA"
```

Рисунок 3.9 – Імплементація моделі GPT-4

Модель "gpt-4" обрана як найпотужніша версія на момент розробки, що забезпечує високу якість лінгвістичного аналізу та розуміння нюансів української мови, включаючи виявлення русизмів, кальки і граматичних помилок.

Параметр temperature=0.3 встановлений на низькому рівні для забезпечення послідовності та точності оцінювання, оскільки занадто висока температура може призвести до непередбачуваних або непослідовних оцінок, тоді як занадто низька може зробити відповіді надто шаблонними, тому значення 0.3 забезпечує оптимальний баланс.

Налаштування параметрів моделі GPT-4 та промпту представлені на рисунку 3.10.

```
response = openai.ChatCompletion.create(
    model="gpt-4",
    messages=[
        {"role": "system", "content": "Ти помічник, що аналізує мовлення користувача українською мовою."},
        {"role": "user", "content": prompt}
    ],
    temperature=0.3,
)
return response['choices'][0]['message']['content']
```

Рисунок 3.10 – Налаштування параметрів моделі та промпту

Промпт спеціально налаштований для виявлення русизмів та кальки з російської мови, що є важливим для українського контексту, включаючи конкретні приклади помилкових конструкцій та їх правильні українські еквіваленти.

Формування промпту для моделі штучного інтелекту наведене на рисунку 5.13

```
async def analyze_text(transcript, topic):
    prompt = f"""
    Завдання: Проаналізуй український текст, приділи увагу мовним помилкам (граматичні, ле
    кальки з російської). Додатково використовуй стандарти української мови з наведених ре
### Формат відповіді (суворий!):
1. Score: [0-10]  (використовуй саме такий формат виводу оцінки)
   Мовні помилки: список пар у форматі (помилка -> виправлення), наприклад:
   - я іду в школу -> я йду до школи
   - зробити дзвінок -> зателефонувати

2. Relevance: Так/Ні – чи відповідає текст заданій темі
```

Рисунок 3.11 – Формування промпту

Функція також включає асинхронну обробку через asyncio для уникнення блокування основного потоку під час API викликів, що важливо для підтримки responsiveness Telegram бота при одночасній роботі з кількома користувачами.

Реалізація асинхронної обробки відповіді від GPT-4 наведена на рисунку 3.12

```
# GPT-аналіз
loop = asyncio.new_event_loop()
asyncio.set_event_loop(loop)
gpt_response = loop.run_until_complete(analyze_text(transcript, topic))

score, errors, relevance = parse_analysis_response(gpt_response)

print("GPT response:\n", gpt_response)
```

Рисунок 3.12 – Реалізація асинхронної обробки

Результат API виклику обробляється через індексацію choices[0]['message']['content'] для отримання текстової відповіді, яка потім парситься спеціальною функцією parse_analysis_response для виділення структурованих елементів: числової оцінки, списку помилок у форматі "оригінал -> виправлення" та оцінки релевантності відповіді заданій темі, що дозволяє системі надавати детальний зворотний зв'язок користувачеві та зберігати структуровані дані для подальшого аналізу прогресу навчання.

Функція для парсингу відповіді від GPT-4 наведена на рисунку 3.13.

```python
def parse_analysis_response(response_text):
    lines = response_text.strip().split("\n")
    score = 0
    errors = []
    relevance = None

    # Отримання оцінки
    for line in lines:
        if "Score" in line:
            match = re.search(r'\d+', line)
            if match:
                score = int(match.group())

    # Отримання помилок
    capturing_errors = False
    for line in lines:
        if "Мовні помилки" in line or "mistakes" in line.lower():
            capturing_errors = True
            continue
        if capturing_errors:
            if line.strip() == "" or "Relevance" in line:
                break
            # Парсимо рядок у форматі "оригінал -> виправлення"
            if "->" in line:
                parts = [p.strip() for p in line.split("->", 1)]
                if len(parts) == 2:
                    errors.append((parts[0], parts[1]))

    # Отримання релевантності
    for line in lines:
        if "Relevance" in line:
            relevance = line.split(":")[1].strip()

    return score, errors, relevance
```

Рисунок 3.13 – Функція для парсингу відповіді

Функція для надання персоналізованої поради від GPT-4 на основі помилок користувача наведена на рисунку 3.14

```python
async def get_suggestion_from_gpt(error_pairs):
    prompt = f"""
    Ти мовний асистент. На основі цих мовних помилок в українській мові, дай коротку пора
    повторити. Не поясню, просто порадь тему повторення. Формат відповіді: "📌 Порада: п
    без лапок.

    Помилки:
    """ + "\n".join([f"{orig.strip()} -> {corr.strip()}" for orig, corr in error_pairs])

    response = openai.ChatCompletion.create(
        model="gpt-4",
        messages=[{"role": "user", "content": prompt}],
        temperature=0.3
    )
    return response['choices'][0]['message']['content'].strip()
```

Рисунок 3.14 – Функція надання персоналізованих порад

3.6 Приклад застосування

Після повної реалізації функціоналу діалогової підсистеми перейдемо до тестування сценарію використання користувачем.

Для початку роботи потрібно надіслати команду /start у бот, після чого з’являється головне меню та вітальне повідомлення.

Вітальне повідомлення та відображення головного меню наведені на рисунках 3.15 та 3.16.

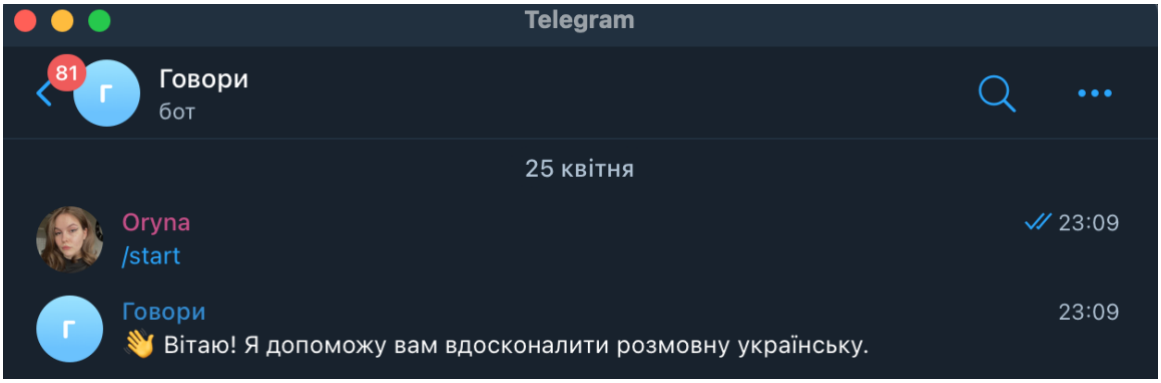


Рисунок 3.15 – Вітальне повідомлення

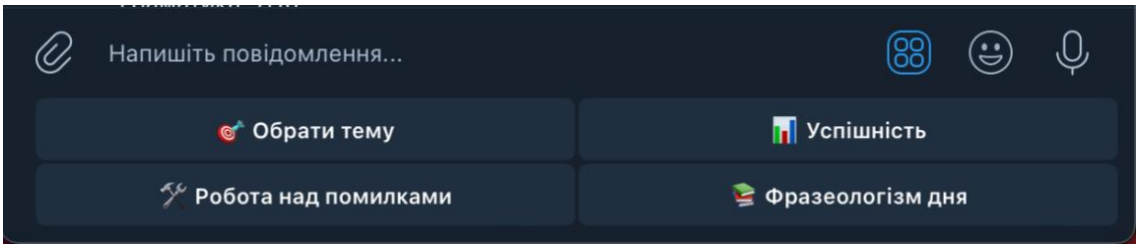


Рисунок 3.16 – Головне меню

Після натискання на кнопку «Обрати тему», з’являється клікабельний список з трьох тем та кнопки «Інші теми», що дозволяє отримати список з інших тем для вибору. Інтерфейс меню «Обрати тему» наведений на рисунку 3.17.

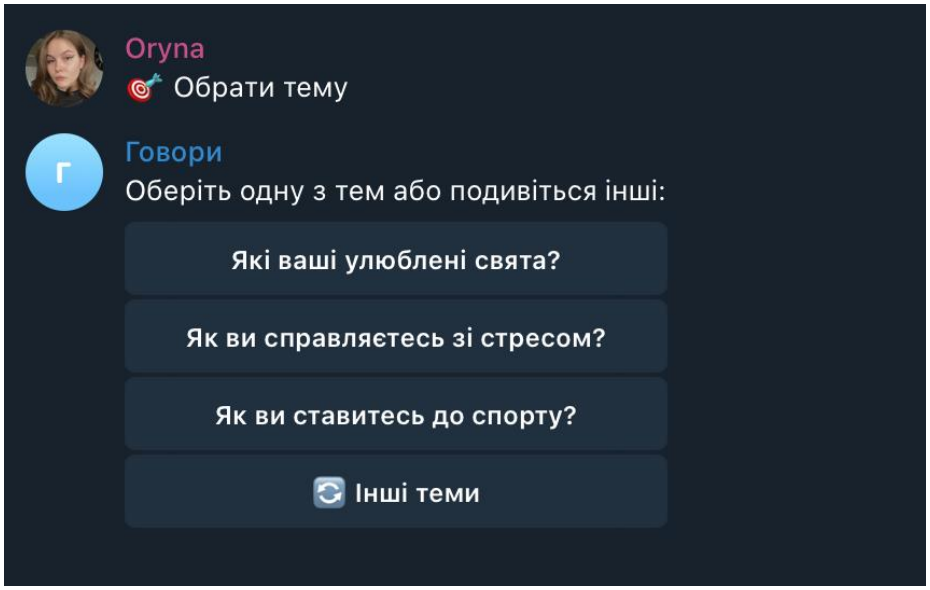


Рисунок 3.17 – Інтерфейс меню «Обрати тему»

Обравши одну з тем, користувач отримає повідомлення із завданням, після чого система очікує на голосове повідомлення з розповіддю.

Після надсилання голосового запису користувач отримує зворотний зв’язок з наведенням припущених помилок та їх правильних варіантів, а також оцінку. Ланцюг описаних дій та результатів наведений на рисунку 3.18

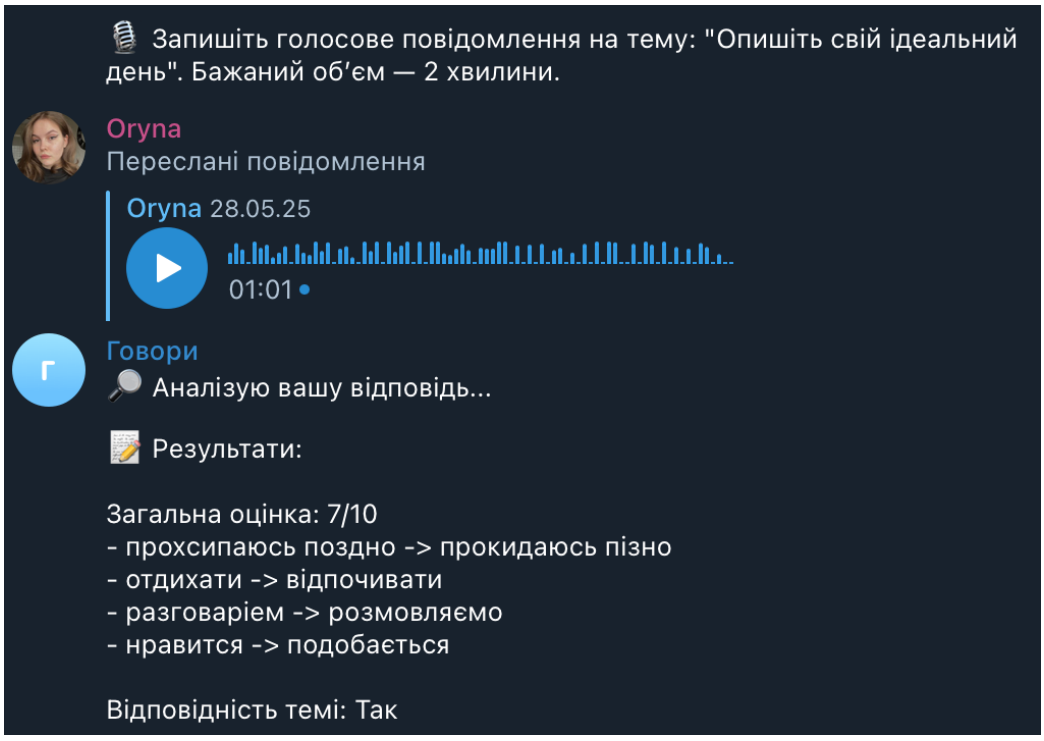


Рисунок 3.18 – Зворотний зв’язок про помилки

Користувач має змогу продивитись допущені ним помилки та виправлені варіанти за датою у меню «Робота над помилками». Діалоговий асистент також формує пораду, аналізуючи помилки користувача.

Приклад роботи над помилками наведений на рисунку 3.19

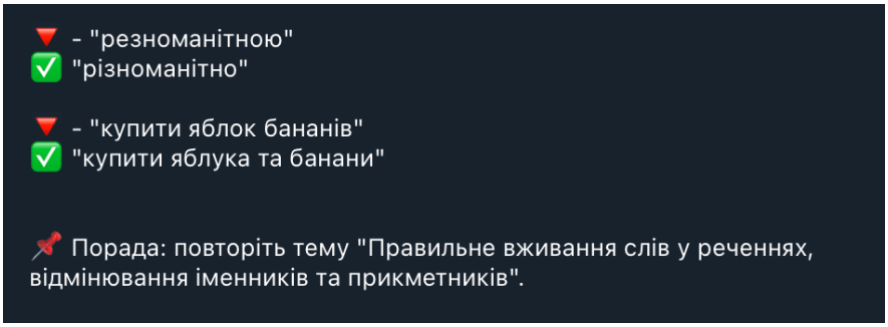


Рисунок 3.19– Інтерфейс роботи над помилками

За допомогою меню «Успішність» користувач може подивитись на свою статистику оцінок, представлену у вигляді двох графіків.

Перший графік демонструє динаміку значення середньої оцінки за сьогоднішній день за часом.

Другий графік демонструє динаміку значення середньої оцінки за останні 10 днів за датою.

Також користувач отримує значення середньої оцінки за ці два проміжки.

Приклад перегляду меню «Успішність», наведений на рисунку 3.20

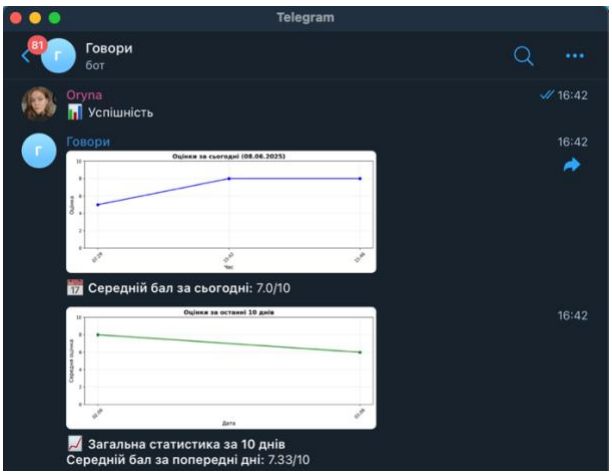


Рисунок 3.20 – Інтерфейс меню «Успішність»

Меню «Фразеологізм дня» надає можливість дізнаватись новий фразеологізм кожного дня. При натисканні на кнопку, користувач отримає унікальний фразеологізм із вказанням його значення та прикладу використання. Фразеологізми змінюються кожного дня та не повторюються. За допомогою цієї функції користувачі можуть збагачувати свій словниковий запас та урізноманітнювати свої розповіді [20].

Приклад перегляду фразеологізму дня наведений на рисунку 3.21

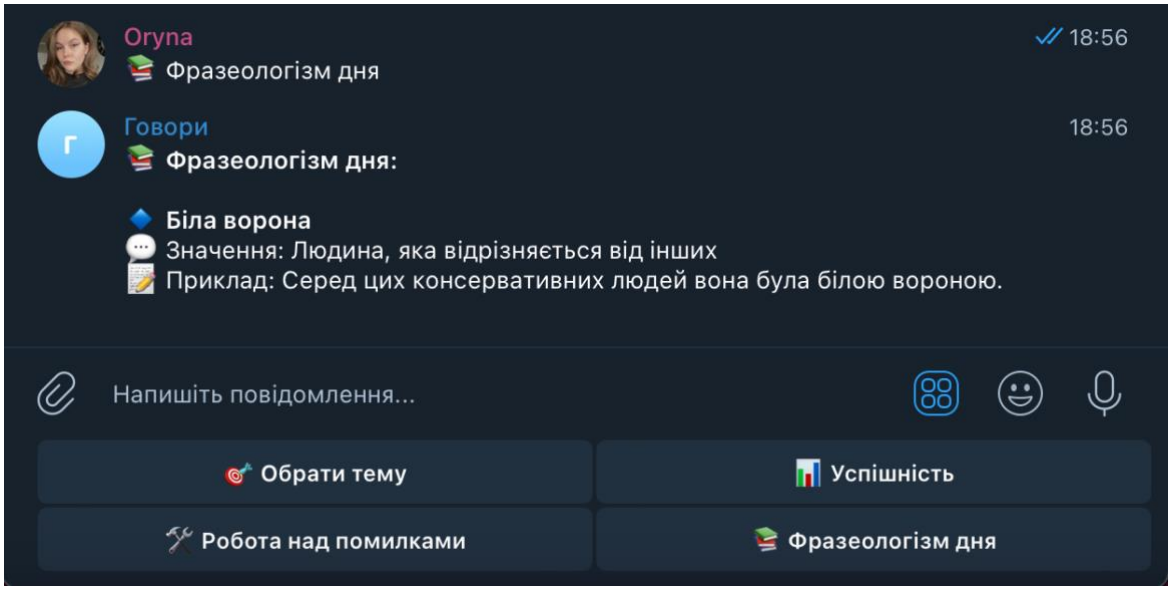


Рисунок 3.21 – Інтерфейс перегляду фразеологізму

3.6 Результати роботи діалогової підсистеми інтелектуального асистента

Після завершення розробки було проведено тестування роботи підсистеми. Відповідно до цього, далі буде детальніше розглянуто результати роботи інтелектуального асистента за метриками.

3.6.1 Метрика WER

Метрика Word Error Rate (WER) є основним інструментом для оцінки якості роботи систем автоматичного розпізнавання мовлення (ASR). Вона вимірює

відсоткове відхилення між текстом, розпізнаним системою, та еталонною транскрипцією, враховуючи три типи помилок: заміни слів, їх пропуски та вставки зайвих слів [21].

Формула розрахунку WER передбачає ділення суми всіх помилок на загальну кількість слів у еталонному тексті з подальшим множенням на 100 для отримання процентного значення.

Ця метрика є стандартом у галузі обробки мовлення через свою об'єктивність і можливість прямого порівняння різних систем.

Для обрахування даної метрики було підготовано 20 еталонних текстів, після чого вони були порівняні з транскриптами, що були сформовані внаслідок надсилання голосових повідомлень у чат-бот.

Обчислення WER відбувається за формулою 4.1

$$WER = \frac{S + D + I}{N} \times 100\% \quad (4.1)$$

S (Substitutions) – кількість замінених слів (коли система помилково розпізнала одне слово замість іншого);

D (Deletion) - кількість пропущених слів;

I (Insertions) – кількість доданих слів, яких немає в оригіналі;

N – загальна кількість слів у еталонному тексті.

За допомогою бібліотеки jiwer було реалізовано скрипт, що автоматично обраховує необхідні параметри за допомогою завантаження 20 пар текстів (вихідний – еталонний) [22].

Програмну реалізацію обчислення метрики наведено на рисунку 4.1

```python
# Завантаження 20 пар: reference + hypothesis
cursor.execute("""
    SELECT t.transcript_text, r.reference_text
    FROM transcripts t
    JOIN reference_texts r ON t.id = r.transcript_id
    LIMIT 20
""")
rows = cursor.fetchall()

# Підрахунок загальних S, D, I, N
total_S = total_D = total_I = total_N = 0

for hypothesis, reference in rows:
    measures = compute_measures(reference, hypothesis)
    total_S += measures['substitutions']
    total_D += measures['deletions']
    total_I += measures['insertions']
    total_N += measures['hits'] + measures['substitutions'] + measures['deletions']

# Розрахунок WER
wer_score = (total_S + total_D + total_I) / total_N

print(f"WER: {wer_score:.3f}")
print(f"Substitutions (S): {total_S}")
print(f"Deletions (D): {total_D}")
print(f"Insertions (I): {total_I}")
print(f"Total words in reference (N): {total_N}")
```

Рисунок 4.1 – Реалізація автоматичного обрахунку WER

Результати обчислення у консолі наведені на рисунку 4.2

```
WER: 0.088
Substitutions (S): 129
Deletions (D): 10
Insertions (I): 11
Total words in reference (N): 1704
```

Рисунок 4.2 – Результати автоматичного обчислення

Отримане значення метрики Word Error Rate (WER) на рівні 8,8% (0,088) свідчить про високу точність роботи системи автоматичного розпізнавання мовлення.

Таке значення вказує на те, що система правильно розпізнає більшість слів у мовленні, допускаючи лише незначну кількість помилок, що включає заміни, пропуски або вставки зайвих слів.

Для багатьох практичних застосувань, особливо в умовах реального часу або при обробці природної розмовної мови, WER у межах 8-10% вважається дуже гарним результатом, який дозволяє забезпечити зрозумілість і придатність тексту для подальшого аналізу чи використання.

Результат WER = 8,8% підтверджує ефективність обраного підходу до розпізнавання мовлення та його придатність для вирішення поставлених завдань у проекті.

3.6.2 Метрики стабільності моделі для визначення помилок

3.6.2.1 Відтворюваність

Однією з метрик стабільності є відтворюваність (Reproducibility). Для її обрахування необхідно визначити відсоток запусків, де кількість помилок не змінилася.

Для обчислення даної метрики одне і теж повідомлення було надіслано боту 15 разів.

Отриманий результат:

$$Reproducibility = \frac{14}{15} \times 100 = 93\%$$

Такий показник свідчить про високу стабільність роботи моделі GPT-4 у задачі виявлення помилок у тексті. Це означає, що при багаторазовому аналізі одного й того самого тексту модель у переважній більшості випадків демонструє послідовні й узгоджені результати.

Такий високий рівень відтворюваності вказує на те, що модель майже завжди правильно ідентифікує одні й ті самі помилки, мінімізуючи випадкові відхилення чи суб’єктивність у оцінках.

Решта 7% можуть бути пов’язані з незначними коливаннями в інтерпретації складних або неоднозначних помилок, що є нормальним для будь-якої NLP-моделі.

3.6.2.2 Дисперсія

Дисперсія (Variance) — це статистична міра, яка показує, наскільки сильно відхиляються результати моделі (наприклад, кількість знайдених помилок) від їх середнього значення при багаторазовому тестуванні одного й того самого тексту. Чим нижча дисперсія, тим стабільніша модель.

Дисперсію було обчислено програмно за допомогою бібліотеки numpy.

За результатами автоматичного обчислення значення дисперсії дорівнює 0,0622, що свідчить про високий рівень стабільності роботи моделі при виявленні помилок у тексті.

Такий низький показник дисперсії означає, що результати моделі майже не коливаються між окремими запусками – кількість знайдених помилок залишається дуже близькою до середнього значення при багаторазовому тестуванні одного й того самого тексту. Це свідчить про передбачувану та надійну поведінку системи, де різниця між окремими вимірами є мінімальною.

На практиці такий рівень дисперсії (менше 0,1) вказує на те, що модель чітко та послідовно ідентифікує помилки, а її робота не залежить від випадкових факторів, що особливо важливо для завдань автоматизованої перевірки текстів, де потрібна точність і стабільність.

Можна зробити висновок, що модель демонструє гарну відтворюваність результатів і є придатною для використання в реальних умовах.

3.6.3 LLM Evaluation для розрахування точності визначення помилок

LLM Evaluation – підхід, за якого одна велика мовна модель (LLM) оцінює результати іншої. Цей метод дозволяє автоматизувати процес перевірки точності, об’єктивності та стабільності роботи інтелектуальних асистентів [23].

Для проведення оцінки було підготовано набір з 20 тестових текстів, кожне з яких було переведене в аудіо повідомлення та надіслане створеному

інтелектуальному агенту. Після чого був сформований запит до великої мовної моделі Claude, що представлений на рисунку 4.3.

```
Ти — експерт з мовної аналітики. Оціни результати роботи
інтелектуального агента:
1. Для кожної знайденої ним помилки вкажи:
   - Чи правильно її ідентифіковано (так/ні)
   - Чи точне виправлення (якщо ні — запропонуй правильний варіант)
2. Вияви пропущені агентом помилки (якщо є).

Формат виводу:
- Правильні результати: [список]
- Помилки агента: [список]
- Пропущені помилки: [список]

1.Текст: "Мій ідеальний день починається с того, що я просипаюсь
позно. Я люблю їсти на сніданок жарені яйця і пить кофе. Потім я йду
гулять в парк, де можно відпочивати під деревами. Днем я смотрю
новий фільм або серіал, тому що це дуже інтересно. Ввечері я
зустрічаюсь з друзями в кафе, де ми їмо піцу і розмовляємо. Якщо
погода хороша, ми йдемо на річку купатись. Перед сном я читаю книгу,
котра мені нравиться. В кінці дня я відчуваю себе повністю
щасливою."
Результати агента: {просипаюсь - прокидаюсь
жарені - смажені
кофе - каву
гулять - гуляти
смотрю - дивлюсь
інтересно - цікаво
нравиться - подобається}
2.Текст: "Моя квартира находиться на п’ятом етажі большого дома. В
```

Рисунок 4.3 – Запит до Claude для проведення LLM Evaluation

Аналіз помилок створеного інтелектуального агента для вдосконалення розмовних навичок був надісланий Claude, після чого на основі відповідей цієї моделі було підраховано кількість вірних знахідок, усі знахідки агента та усі реальні помилки в тексті. Приклад відповіді Claude наведено на рисунку 4.4

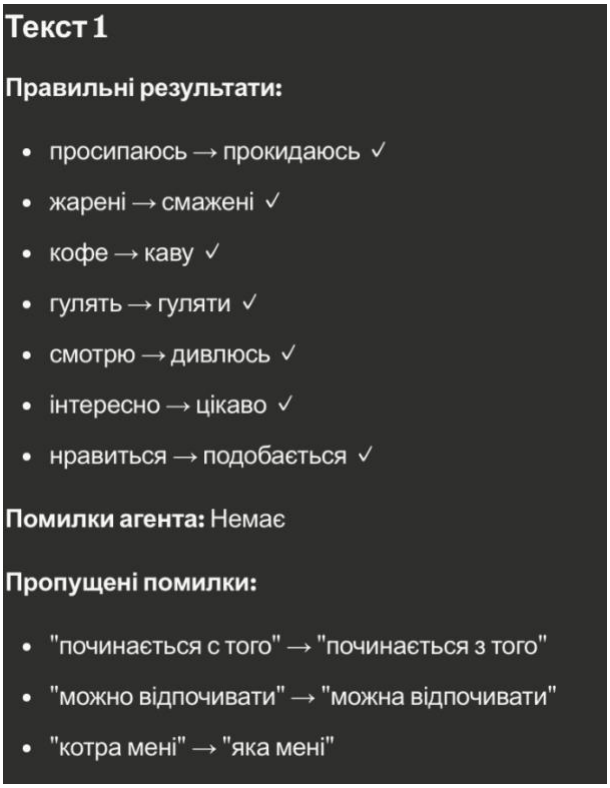


Рисунок 4.4 – Приклад оцінки моделлю Claude

Кількість вірних знахідок, усі знахідки агента та усі реальні помилки в тексті дають змогу обчислити Precision (кількість справжніх помилок), Recall (кількість помилок, знайдених ботом) та F1-score (гармонійне середнє цих значень, що вказує на баланс між ними) [24].

$$Precision = \frac{\text{Кількість вірних знахідок}}{\text{Кількість всіх знайдених помилок}} = \frac{54}{72} = 0{,}76$$

$$Recall = \frac{\text{Кількість вірних знахідок}}{\text{Кількість усіх помилок в тексті}} = \frac{54}{60} = 0{,}90$$

$$F1 = \frac{2 \times (Precision \times Recall)}{(Precision + Recall)} = \frac{2 \times (0{,}76 \times 0{,}90)}{0{,}76 + 0{,}90} \approx 0{,}82$$

Показник F1-score = 0,82 для інтелектуального агента з виявлення текстових помилок свідчить про високу та збалансовану ефективність роботи системи. Цей

показник демонструє, що агент успішно справляється із завданням: він одночасно мінімізує кількість пропущених помилок і хибних спрацьовувань. Значення 0,82 вказує на те, що модель правильно ідентифікує більшість мовних помилок, допускаючи лише незначну кількість неточностей.

3.6.4 Динаміка часу обробки запитів

Під час тестування роботи створеного інтелектуального асистента на двадцяти підготовлених текстах, було розраховано кількість символів кожного повідомлення та час, затрачений на обробку повідомлення окремо для моделі Whisper та загальний витрачений час на формування відповіді.

За допомогою цих даних та коду на Python було побудовано графіки залежностей часу обробки від кількості символів у тексті. Фрагмент коду наведений на рисунку 4.5

```python
import matplotlib.pyplot as plt
import re
import numpy as np
from scipy.stats import linregress

# Читання даних з файлу
with open('/Users/churikovaarina/Desktop/final sem/Диплом/UkrainianSpeakingBot/log2.txt', 'r', encoding='utf-8') as file:
    data = file.read()

# Парсинг даних
pattern = r'Символів: (\d+)\s+Whisper: ([\d.]+)с\s+GPT: ([\d.]+)с\s+Загальний час: ([\d.]+)с'
matches = re.findall(pattern, data)

# Сортування даних за кількістю символів
matches_sorted = sorted(matches, key=lambda x: int(x[0]))
chars, whisper_times, gpt_times, total_times = zip(*[
    (int(m[0]), float(m[1]), float(m[2]), float(m[3])) for m in matches_sorted
])

# Функція для додавання лінії тренду
def add_trendline(ax, x, y, color, label):
    slope, intercept, r_value, _, _ = linregress(x, y)
    line = slope * np.array(x) + intercept
    ax.plot(x, line, '--', color=color, alpha=0.3,
            label=f'{label} тренд (R²={r_value**2:.2f})')

# Створення графіків
plt.figure(figsize=(15, 5))

# Графік 1: Whisper
plt.subplot(1, 3, 1)
plt.plot(chars, whisper_times, 'b-', marker='o', label='Whisper', linewidth=2)
add_trendline(plt.gca(), chars, whisper_times, 'blue', 'Whisper')
plt.title('Час обробки Whisper')
plt.xlabel('Кількість символів')
plt.ylabel('Час (с)')
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend()
```

Рисунок 4.5 – Реалізація автоматичного обчислення та побудови графіків

Кінцеві графіки наведені на рисунках 4.6, 4.7 та 4.8.

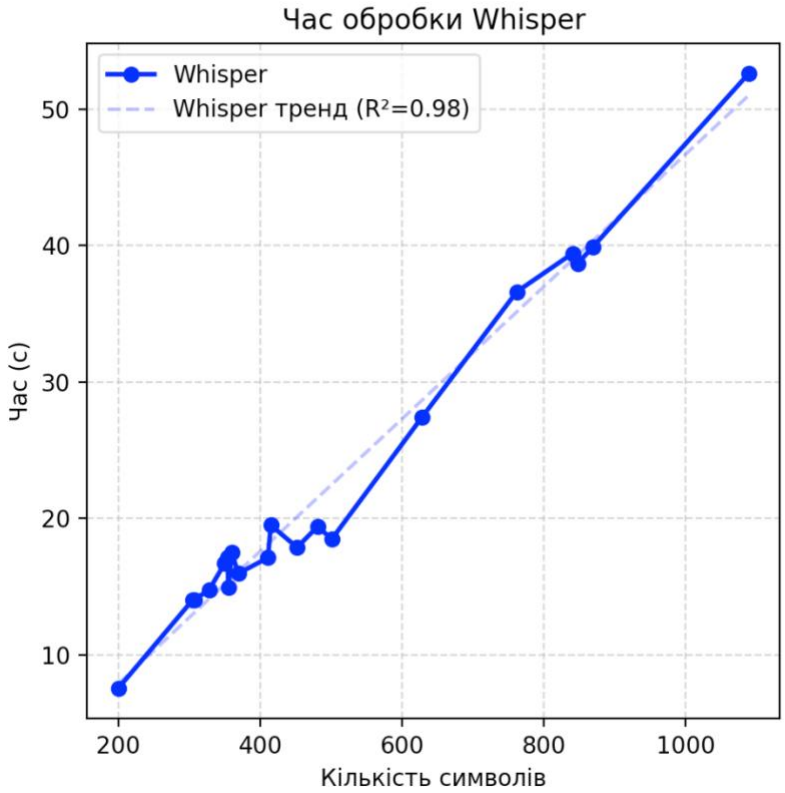


Рисунок 4.6 – Графік залежності часу обробки від кількості символів для моделі Whisper

Графік часу обробки Whisper демонструє чітку лінійну залежність між обсягом тексту та часом виконання, що підтверджується високим коефіцієнтом детермінації $R^2=0.98$.

Це свідчить про стабільну та передбачувану продуктивність системи розпізнавання мовлення, де кожен додатковий символ тексту пропорційно збільшує час обробки.

Така закономірність дозволяє точно прогнозувати витрати часу для текстів будь-якої довжини, що є особливо важливим при обробці великих обсягів даних. Висока стабільність роботи Whisper робить його надійним інструментом для задач транскрибації.

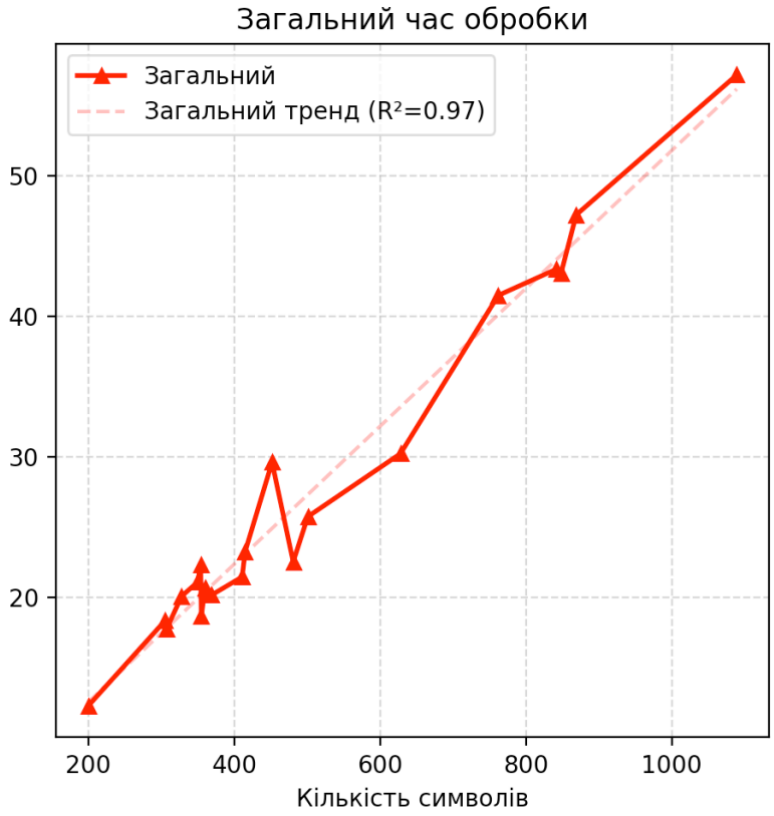


Рисунок 4.7 – Графік залежності загального часу обробки від кількості символів

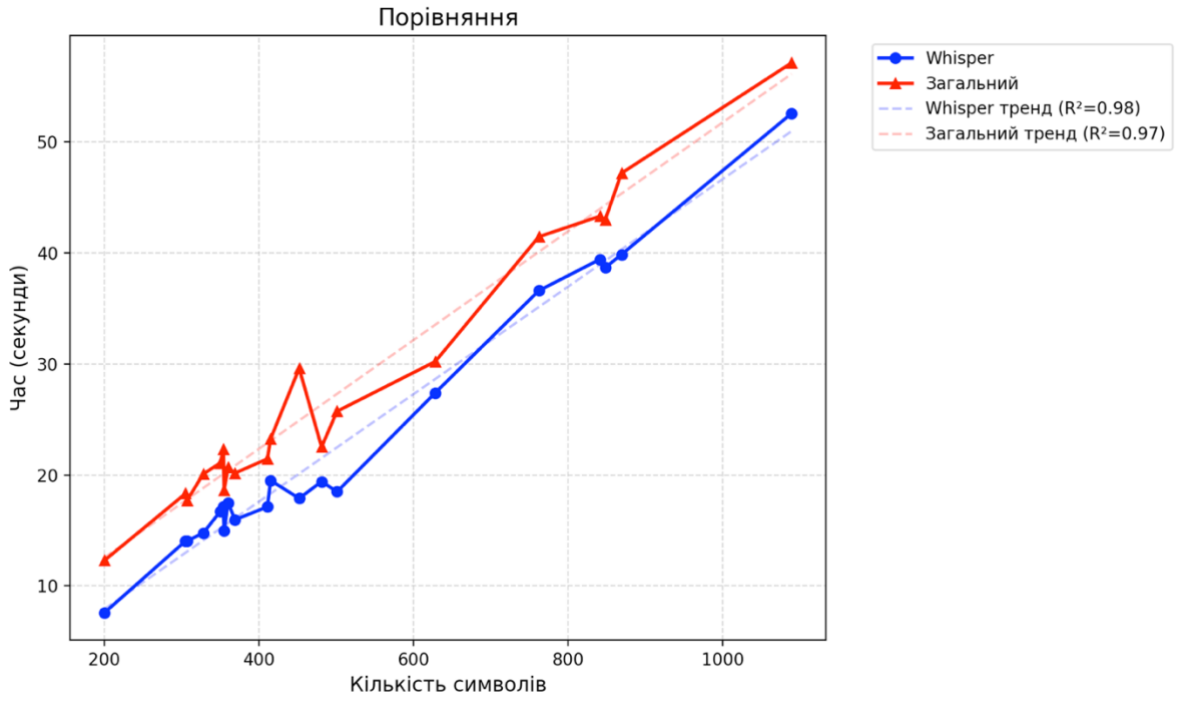


Рисунок 4.8 – Зведений графік залежності часу обробки від кількості символів

Аналіз загального часу обробки підтверджує домінуючий вплив етапу Whisper на продуктивність всієї системи, про що свідчить майже ідентичний лінійний тренд (R^2=0.97).

Загальний час роботи практично повністю визначається етапом розпізнавання мовлення. Така динаміка дозволяє зробити висновок, що для прискорення роботи всієї системи оптимізаційні зусилля слід зосередити саме на компоненті Whisper, тоді як етап аналізу GPT вже демонструє оптимальні показники. Стабільність загального часу обробки без аномальних викидів свідчить про надійність системи в цілому та її здатність ефективно обробляти тексти різного обсягу.

Висновки до розділу 3

У третьому розділі було реалізовано програмну частину інтелектуального діалогового асистента, призначеного для вдосконалення розмовних навичок українською мовою.

У підрозділі 3.1 здійснено обґрунтований вибір засобів реалізації: основною мовою програмування обрано Python завдяки її сумісності з інструментами штучного інтелекту, для керування базами даних використано PostgreSQL як надійне та гнучке рішення, середовищем розробки стала Visual Studio Code, а також підібрано набір бібліотек, зокрема Telebot, OpenAI, Whisper, Psycopg2, що забезпечили реалізацію повного циклу обробки мовлення.

У підрозділі 3.2 описано побудову інтерфейсу Telegram-бота, реалізованого через бібліотеку Telebot. Інтерфейс включає головне меню з чотирма основними функціями: вибір теми, перегляд успішності, робота над помилками та фразеологізм дня, що забезпечує зручну взаємодію з користувачем.

У підрозділі 3.3 описано процес обробки та нормалізації голосового повідомлення перед розпізнаванням: від завантаження голосу у форматі OGG до конвертації у WAV з фільтрацією шумів, зміною гучності та приведенням до стандартів ASR.

У підрозділі 3.4 описано імплементацію та налаштування моделі Whisper, зокрема її параметри, як-от вибір моделі "small", встановлення мови “uk” та точні параметри декодування, що забезпечують високу якість розпізнавання українського мовлення.

У підрозділі 3.5 реалізовано GPT-4 як основний модуль для лінгвістичного аналізу. Описано структуру промпту, параметри виклику API та функції асинхронної обробки, що забезпечує стабільність, послідовність і якість результатів аналізу мовлення.

У підрозділі 3.6 описано сценарій використання асистента користувачем, починаючи з надсилання голосового повідомлення та завершуючи отриманням аналізу з оцінкою, рекомендаціями та збереженням результатів для подальшої роботи.

У підрозділі 3.7 здійснено тестування системи. Було розраховано показник точності розпізнавання мовлення WER, що склав 8,8%, що свідчить про високу ефективність Whisper. Також обраховано показники стабільності GPT-моделі — відтворюваність 93% та дисперсія 0,0622, що вказує на послідовну роботу моделі. Використовуючи LLM Evaluation, отримано значення F1-score = 0,82, яке свідчить про збалансовану точність і повноту виявлення мовних помилок.

Окрім того, у підрозділі 3.7.4 проведено аналіз часу обробки, який показав, що загальна продуктивність системи переважно залежить від компонента Whisper, демонструючи лінійну залежність між довжиною тексту та часом обробки.

Отже, в межах третього розділу було повністю реалізовано та протестовано діалогову підсистему інтелектуального асистента, яка успішно поєднує автоматичне розпізнавання мовлення, аналіз помилок за допомогою великої мовної моделі та зручний інтерфейс у Telegram, що підтверджує технічну спроможність системи до повноцінної практики та вдосконалення розмовних навичок українською мовою.

ВИСНОВКИ

У відповідності до мети, поставленої у підрозділі 1.2, що полягала у покращенні ефективності процесу практики розмовних навичок українською мовою через розробку діалогової підсистеми інтелектуального робота-асистента, у межах дипломного проєкту було реалізовано повний цикл розробки програмного продукту: від формалізації до створення функціональної системи та оцінки її ефективності.

У розділі 1 було проведено аналіз предметної області, виявлено ключові проблеми у практиці українського мовлення та обґрунтовано необхідність створення цифрового інструменту.

У підрозділі 1.3 було описано особливості задачі та розробки, зокрема технічні та лінгвістичні складнощі, пов’язані з обробкою української мови, а також обґрунтовано актуальність реалізації україномовного рішення.

У підрозділі 1.4 було проаналізовано існуючі підходи до створення діалогових систем і здійснено вибір оптимального — комбінації технологій ASR та великої мовної моделі GPT.

Проведене у підрозділі 1.5 порівняння з популярними платформами Duolingo, Babbel та Memrise засвідчило відсутність підтримки повноцінної розмовної практики українською у наявних рішеннях, що підкреслило унікальність обраного підходу. Було сформульовано конкретні технічні вимоги до системи. У другому розділі було детально проаналізовано інформаційне забезпечення, що включає вибір сучасних технологій для розпізнавання мовлення, лінгвістичного аналізу, а також обґрунтовано доцільність використання моделі Whisper для ASR і GPT — для виявлення помилок та формування персоналізованих порад.

У підрозділах 2.5–2.6 спроєктовано архітектуру системи та логічну модель даних, що забезпечує стабільну роботу та масштабованість. Розроблені діаграми компонентів і варіантів використання деталізують структуру системи та сценарії взаємодії. Окрему увагу було приділено проєктуванню роботизованої реалізації

асистента, що дозволяє створити апаратний прототип з мікрофоном, сенсорним дисплеєм та автономним живленням, який забезпечує повноцінну взаємодію користувача з системою.

У третьому розділі представлено безпосередню реалізацію програмного забезпечення: описано архітектуру чат-бота, обробку повідомлень, взаємодію з API ASR і GPT, зберігання транскриптів і помилок у базі даних, реалізацію інтерфейсу користувача, функції статистики, модуля “робота над помилками”, а також алгоритм генерації персоналізованих порад.

Експериментальна перевірка підтвердила ефективність запропонованого рішення у виявленні та поясненні помилок у мовленні користувача, що свідчить про високу якість розробленої системи.

Отже, поставлену мету було досягнуто: розроблено діалогову підсистему інтелектуального робота-асистента, що забезпечує доступну та ефективну мовну практику, персоналізований зворотний зв’язок та сприяє вдосконаленню розмовних навичок українською мовою. Отримані результати мають значну практичну цінність і можуть бути використані в освітніх, професійних та побутових сферах для покращення рівню володіння українською мовою та заохочення до її використання у всіх сферах життя.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Потреби і проблеми українців, які перейшли або переходять на українську мову. URL: https://bazilik.media/potreby-i-problemy-ukraintsiv-iaki-perejshly-abo-perekhodiat-na-ukrainsku-movu/

2. Artificial Intelligence as an asset to language learning in Europe. URL: https://school-education.ec.europa.eu/en/discover/news/artificial-intelligence-asset-language-learning-europe

3. Oliinyk V. Data augmentation with foreign language content in text classification using machine learning / Oliinyk V., Osadcha K. // Адаптивні системи автоматичного управління: міжвідомчий науково-технічний збірник. – 2020. Vol. 1, №36. – P. 51-59.

4. OpenAI Whisper - документація. URL: https://openai.com/research/whisper

5. Duolingo – офіційний сайт. URL: https://www.postgresql.org/docs/

6. Babbel – офіційний сайт. URL: https://www.babbel.com

7. Memrise – офіційний сайт. URL: https://www.memrise.com

8. Oliinyk, V. Low-resource text classification using cross-lingual models for bullying detection in the ukrainian language / V. Oliinyk, I. Matviichuk // Адаптивні системи автоматичного управління: міжвідомчий науково-технічний збірник. – 2023. – № 1 (42). – С. 87-100.

9. Українська мова. URL: https://esu.com.ua/article-68157

10. Oliinyk, V. A comparative study of task formulations for detecting propaganda using Large Language Models/ Oliinyk V., Zakharchyn N. // Адаптивні системи автоматичного управління: міжвідомчий науково-технічний збірник. – 2025. – № 2 (47).

11. Aslanova V., Oliinyk V. Psychological support assistant based on fine-tuned LLaMA 3 model // The International Conference on Security, Fault Tolerance, Intelligence ICSFTI2024 (June 07, 2024, Kyiv, Ukraine), page 1-13. URL: https://icsfti-proc.kpi.ua/article/view/309532 (application date: 08.06.2025)

12. GPT-4 – офіційний сайт. URL: https://openai.com/gpt-4

13. Client-Server Model. URL: https://www.geeksforgeeks.org/client-server-model/

14. The Unified Modeling Language. URL: https://www.uml-diagrams.org

15. Sequence Diagram досі жива? URL: https://e5.ua/uk/blogpost-2/sequence-diagram-dosi-zhiva/

16. PostgreSQL – офіційна документація. URL: https://www.postgresql.org/docs/

17. Telegram Bot API – документація. URL: https://core.telegram.org/bots/api

18. Psycopg2 – документація. URL: https://www.psycopg.org/docs/

19. Matplotlib – документація. URL: https://matplotlib.org/stable/contents.html

20. Фразеологізми. URL: https://ukr-mova.in.ua/library/frazeologizmu/

21. What is WER? What does Word Error Rate mean? URL: https://www.rev.com/resources/what-is-wer-what-does-word-error-rate-mean

22. Jiwer - документація. URL: https://github.com/jitsi/jiwer

23. LLM Evaluation Metrics: The Ultimate LLM Evaluation Guide/ URL: https://www.confident-ai.com/blog/llm-evaluation-metrics-everything-you-need-for-llm-evaluation

24. Understanding Precision, Recall and F1 Score Metrics. URL: https://medium.com/@piyushkashyap045/understanding-precision-recall-and-f1-score-metrics-ea219b908093