МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інженерії
програмного забезпечення

_____ Євген ДАВИДЕНКО

«___» _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОТСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Система емоційної трансформації на основі семантичного аналізу та тематичної агрегації дописів

Спеціальність 121 Інженерія програмного забезпечення

Освітня програма «Інженерія програмного забезпечення»

Здобувач _____ **Іван ЗУРІЛОВ**

«__» _____ 20__ р.

Керівник роботи

д-рка техн. наук,
професорка _____ **Альона ШВЕД**

«__» _____ 20__ р.

Миколаїв – 2024

Завдання на виконання кваліфікаційної роботи

Чорноморський національний університет імені Петра Могили

Факультет	Комп'ютерних наук
Кафедра	Інженерії програмного забезпечення
Рівень вищої освіти	Другий (магістерський)
Освітній ступінь	Магістр
Спеціальність	121 Інженерія програмного забезпечення
Освітня програма	Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри інженерії
програмного забезпечення
_____ Євген ДАВИДЕНКО
«____» _____ 2024 р.

ЗАВДАННЯ
на кваліфікаційну магістерську роботу здобувача вищої освіти
Зурілов Іван

1. Тема кваліфікаційної роботи «Система емоційної трансформації на основі семантичного аналізу та тематичної агрегації дописів» затверджена наказом ректора ЧНУ ім. Петра Могили № 220 від «04» вересня 2024 р.
2. Строк представлення кваліфікаційної роботи «04» вересня 2024 р.
3. Очікуваний результат роботи та початкові дані якщо такі потрібні:
Telegram-бот для виявлення емоційного тону вхідного тексту та емоційної трансформації на основі алгоритмів обробки природньої мови. Трансформація тексту зі зміненим емоційним тоном та збереженням оригінального змісту._____
4. Перелік питань, що підлягають розробці:

Процеси розпізнавання та трансформації емоцій на основі текстових повідомлень користувачів; сучасні методи обробки природної мови та машинного навчання, що використовуються для вирішення задачі розпізнавання; аналіз та перетворення емоційного тону текстових повідомлень користувачів._____

5. Перелік графічних матеріалів: презентація.

6. Консультанти:

Консультант	**Кафедра (організація)**	**Частина роботи**

Дата видачі завдання «____» _____ 20___ р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: **Система емоційної трансформації на основі семантичного аналізу та тематичної агрегації дописів**

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КМР	01.09.2024	04.09.2024	Виконано
2.	Огляд літератури за темою роботи	05.09.2024	10.09.2024	Виконано
3.	Складання календарного плану КМР	11.09.2024	11.09.2024	Виконано
4.	Аналіз предметної області	12.09.2024	20.09.2024	Виконано
5.	Розробка проєктних рішень	21.09.2024	30.09.2024	Виконано
6.	Моделювання та конструювання ПЗ	01.10.2024	22.10.2024	Виконано
7.	Кодування, тестування та апробація розробленого ПЗ, аналіз результатів тестування, розробка керівництва користувача	23.10.2024	18.11.2024	Виконано
8.	Відгук керівника КМР	19.11.2024	21.11.2024	Виконано
9.	Оформлення КМР та презентації	22.11.2024	01.12.2024	Виконано
10.	Попередній захист	02.12.2024	02.12.2024	Виконано
11.	Рецензування	03.12.2024	04.12.2024	Виконано
12.	Завершення оформлення КМР та презентації	05.12.2024	13.12.2024	Виконано
13.	Захист кваліфікаційної роботи	19.12.2024	20.12.2024	Виконано

Здобувач _____ **Іван ЗУРІЛОВ**

«__» _____ 20__ р.

Керівник роботи

д-рка техн. наук,

професорка _____ **Альона ШВЕД**

«__» _____ 20__ р.

АНОТАЦІЯ

до кваліфікаційної магістерської роботи

Система емоційної трансформації на основі семантичного аналізу та тематичної агрегації дописів

Здобувач 608м гр.: Зурілов Іван

Керівник: д-рка техн. наук, професорка Швед Альона

Актуальність обумовлена психологічним впливом написаного тексту та ризиком некоректного трактування змісту повідомлень в наслідок емоційного впливу на читача шляхом маніпуляції емоційною складовою дописів.

Об’єктом дослідження виступають процеси розпізнавання та трансформації емоцій на основі текстових повідомлень користувачів. Предметом дослідження є сучасні методи обробки природної мови та машинного навчання, що використовуються для вирішення задачі розпізнавання. Метою дослідження є аналіз та перетворення емоційного тону текстових повідомлень користувачів шляхом розробки програмного забезпечення розпізнавання їх емоцій.

Для досягнення мети КМР необхідно виконати наступні завдання:

– дослідити предметну область, провести аналіз літератури за темою КМР;

– розробити специфікацію вимог до ПЗ;

– провести роботу із проєктування та моделювання ПЗ;

– виконати виявлення емоційного тону шляхом впровадження NLP-моделі для точного виявлення та класифікації емоційного тону вхідного тексту;

– виконати емоційну трансформацію шляхом створення алгоритму для генерації нової версії тексту зі зміненим емоційним тоном, зберігаючи оригінальний зміст;

– розробити користувацький інтерфейс;

– розробити ПЗ бота, виконати тестування та розгортання бота в Telegram.

Кваліфікаційна робота складається із вступу, 4 розділів, висновків та переліку джерел посилання.

У вступі описано об’єкт, предмет, мету та завдання КМР.

У розділі 1 проаналізовано предметну область розпізнавання та трансформації емоцій у текстах. Зроблено огляд функціональних підходів та рішень застосунків, розглянуто перелік та природу проблем, які покликані вирішити існуючі програмні рішення. Виконано огляд цільової платформи розміщення та виконання програмного рішення проєкту КМР.

У розділі 2 проаналізовано сучасні алгоритми обробки природної мови для аналізу емоцій, моделі машинного навчання та складові системи трансформації емоційної реакції користувачів. Визначено переваги та недоліки альтернатив кожної складової системи.

У розділі 3 проведено роботу із системного моделювання та графічного представлення результатів моделювання програмної реалізації системи емоційної трансформації. Розглянуто та обґрунтовано вибір інструментів програмної реалізації системи емоційної трансформації.

У розділі 4 проведено роботу із викладення програмної специфікації, кодування та підготовки користувацького середовища до запуску локальної версії програмного рішення КМР, продемонстровано приклад виконання описаного функціоналу.

У висновках підведено підсумки проведеної роботи, винесено оцінку виконаних завдань, визначено практичне значення та винесено оцінку потенціалу розвитку програмного рішення.

Кваліфікаційна робота викладена на 69 сторінках машинописного тексту, складається із вступу, 4 розділів, загальних висновків, переліку джерел посилання з 23 найменувань та 1 додатку. Праця містить 3 таблиці та 21 рисунок.

Ключові слова: алгоритм, інтерфейс, розпізнавання, чат-бот, трансформація, користувачі, агрегація, підтримка, модель, взаємодія.

ABSTRACT

to the qualifying master's thesis

System of Emotional Transformation Based on Semantic Analysis and Thematic Aggregation of Posts

Student of 608m group: Zurilov Ivan

Supervisor: Doctor of Technical Sciences, Professor Shved Aliona

The relevance is determined by the psychological impact of written text and the risk of incorrect interpretation of the content of messages due to emotional influence on the reader through manipulation of the emotional component of posts.

The object of the research is the processes of recognizing and transforming emotions based on users' text messages. The subject of the research is modern methods of natural language processing and machine learning used to solve the recognition task. The aim of the research is to analyze and transform the emotional tone of users' text messages by developing software for recognizing their emotions.

To achieve this goal, the following tasks need to be accomplished:

– explore the subject area and conduct a literature review on the thesis topic;

– develop a software requirements specification;

– carry out design and modeling of the software;

– detect the emotional tone by implementing an NLP model for accurate identification and classification of the emotional tone of the input text;

– perform emotional transformation by creating an algorithm to generate a new version of the text with a modified emotional tone, preserving the original meaning;

– develop a user interface;

– develop the bot software, conduct testing, and deploy the bot on Telegram.

– The qualification thesis consists of an introduction, 4 chapters, conclusions, and a list of references.

The introduction describes the object, subject, goal, and tasks of the thesis.

Chapter 1 analyzes the subject area of emotion recognition and transformation in texts. A review of functional approaches and solutions of applications is made, and a list of problems these software solutions aim to solve is provided. A review of the target platform for deploying and executing the software solution of the thesis project is presented.

Chapter 2 analyzes modern natural language processing algorithms for emotion analysis, machine learning models, and components of the emotional reaction transformation system for users. The advantages and disadvantages of each component's alternatives are identified.

Chapter 3 deals with system modeling and graphical representation of the software implementation results of the emotional transformation system. The choice of tools for the software implementation of the emotional transformation system is justified.

Chapter 4 presents the software specification, coding, and preparation of the user environment for the launch of the local version of the software solution, demonstrating an example of the described functionality.

The conclusions summarize the work performed, evaluate the tasks accomplished, determine the practical significance, and assess the development potential of the software solution.

The qualification thesis is presented on 69 pages of typed text, consisting of an introduction, 4 chapters, general conclusions, a list of 23 references, and 1 appendix. The work contains 3 tables and 21 figures.

Keywords: algorithm, interface, recognition, chat-bot, transformation, users, aggregation, support, model, interaction.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ПЗ – програмне забезпечення

ПІБ – прізвище, ім’я, по батькові

ЧНУ ім. Петра Могили – Чорноморський національний університет імені Петра Могили

ІПЗ – інженерія програмного забезпечення

КМР – кваліфікаційна магістерська робота

ШІ – штучний інтелект

GPT – Generative Pre-trained Transformer

SVM – Support Vector Machines

RNN – Recurrent Neural Networks

LSTM – Long Short-Term Memory Networks

SRS – Software Requirements Specification

NLP – Natural Language Processing.

ВСТУП

У добу цифрового спілкування тон і емоційний вплив написаного тексту можуть значно впливати на міжособистісні взаємодії, відносини з клієнтами та статус у соціальних мережах. Неправильне тлумачення емоційного тону тексту може призводити до непорозумінь, конфліктів і зниження емпатії. Автоматизований інструмент, який може аналізувати та коригувати емоційний тон текстових повідомлень, може підвищити ясність, ефективність та задоволення від спілкування. Проєкт КМР є актуальним для покращення відповідей у службі підтримки клієнтів, удосконалення систем фільтрації контенту в соціальних мережах і допомоги людям у більш точному та емпатичному вираженні своїх думок.

Об’єктом дослідження виступають процеси розпізнавання та трансформації емоцій на основі текстових повідомлень користувачів. Предметом дослідження є сучасні методи обробки природної мови та машинного навчання, що використовуються для вирішення задачі розпізнавання. Метою дослідження є аналіз та перетворення емоційного тону текстових повідомлень користувачів шляхом розробки програмного забезпечення розпізнавання їх емоцій.

Для досягнення мети КМР шляхом розробки функціонального Telegram-бота, здатного приймати текстові повідомлення від користувачів, необхідно виконати наступні завдання:

а) дослідити предметну область, провести аналіз літератури за темою КМР;

б) розробити специфікацію вимог до ПЗ;

в) провести роботу із проєктування та моделювання ПЗ;

г) виконати виявлення емоційного тону шляхом впровадження NLP-моделі для точного виявлення та класифікації емоційного тону вхідного тексту;

д) виконати емоційну трансформацію шляхом створення алгоритму для генерації нової версії тексту зі зміненим емоційним тоном, зберігаючи оригінальний зміст;

е) розробити користувацький інтерфейс;

ж) розробити ПЗ бота, виконати тестування та розгортання бота в Telegram.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ

1.1 Аналіз предметної області розпізнавання та трансформації емоцій у текстах

Розпізнавання та трансформація емоцій у тексті охоплюють процеси ідентифікації, аналізу та інтерпретації людських емоцій, виражених у письмовій мові. Ця галузь використовує методи обробки природної мови (NLP) та машинного навчання для розуміння емоційного змісту тексту, що можна застосовувати в різних сферах, включаючи психічне здоров'я, маркетинг і проєктування користувацького досвіду.

Етапи розпізнавання емоцій у тексті:

а) аналіз тексту: перший крок у розпізнаванні емоцій полягає в аналізі тексту для виділення відповідних ознак. Це може включати виявлення ключових слів, фраз і мовних структур, які вказують на конкретні емоції. Зазвичай використовуються такі техніки, як аналіз сентименту, що класифікує текст як позитивний, негативний або нейтральний;

б) виділення ознак: передові аналізатори емоцій використовують різні мовні ознаки, включаючи лексикони емоцій (словники слів, пов'язаних з конкретними емоціями), n-грам (сполучення слів) та векторні подання слів (вектори, що відображають значення слів). Ці ознаки допомагають кількісно оцінити емоційний тон тексту;

в) моделі машинного навчання: після виділення ознак алгоритми машинного навчання навчаються на мічених наборах даних для розпізнавання шаблонів, пов'язаних з різними емоціями. Такі моделі можуть класифікувати текст у наперед визначені категорії емоцій, такі як радість, смуток, гнів чи страх. Точність цих моделей часто підвищується зі збільшенням розміру та різноманітності навчальних даних;

г) контекстуальне розуміння: ефективне розпізнавання емоцій також вимагає розуміння контексту, в якому використовуються слова. Це включає

аналіз оточуючого тексту та загальної структури викладання для врахування тонкощів значень, які можуть впливати на емоційне тлумачення;

д) зворотний зв'язок та адаптація: деякі системи включають зворотний зв'язок від користувачів для вдосконалення своїх можливостей розпізнавання емоцій з часом. Навчаючись на взаємодіях, такі системи можуть адаптуватися до індивідуальних виразів користувачів і покращувати точність у розпізнаванні емоцій;

Основні цілі розпізнавання та аналізу емоцій у тексті включають:

а) покращення користувацького досвіду: розуміючи емоції користувачів, додатки можуть адаптувати відповіді та взаємодію для створення більш привабливого та підтримуючого досвіду. Наприклад, чат-боти для обслуговування клієнтів можуть коригувати свій тон відповідно до емоційного стану користувача;

б) підтримка психічного здоров'я: розпізнавання емоцій може відігравати важливу роль у програмах психічного здоров'я, де виявлення емоційного стресу може спровокувати відповідні втручання чи підтримку. Чат-боти та додатки можуть відстежувати емоційний стан користувачів та за необхідності надавати відповідні ресурси або направляти до фахівців;

в) маркетингові дослідження та аналіз настроїв: бізнес використовує розпізнавання емоцій для оцінки споживчих настроїв та емоційних реакцій на продукти, послуги чи маркетингові кампанії. Ця інформація може допомогти у формуванні стратегій для покращення задоволеності клієнтів і їхньої лояльності;

г) взаємодія людини і бота: розпізнавання емоцій дозволяє ботам відповідати на людські емоції, що сприяє більш природній і ефективній взаємодії між людьми та роботами;

д) моніторинг соціальних медіа: аналіз емоцій у соціальних мережах дозволяє організаціям розуміти громадську думку та ефективно реагувати на тренди або кризові ситуації.

1.2 Аналіз існуючих застосунків та їхніх можливостей

На ринку існують декілька рішень, зосереджені на розпізнаванні та трансформації емоцій у текстах. Ці інструменти використовують обробку природної мови (NLP) та машинне навчання для аналізу і тлумачення емоцій, що виражаються у письмовій комунікації. Вони діляться на наступні категорії:

а) інструменти аналізу настрою: багато компаній пропонують платформи для аналізу настрою, які можуть оцінювати емоційний тон тексту. Ці інструменти широко використовуються в обслуговуванні клієнтів та маркетингу для оцінки громадської думки щодо продуктів або послуг. Наприклад, бізнеси можуть використовувати аналіз настрою для моніторингу взаємодії в соціальних мережах та відгуків клієнтів, що дозволяє їм ефективніше реагувати на потреби клієнтів;

б) чат-боти для підтримки психічного здоров’я: чат-боти на основі ШІ, призначені для підтримки психічного здоров’я, набирають популярності. Ці боти можуть вести бесіду з користувачами, надаючи емоційну підтримку та ресурси на основі почуттів, які виражає користувач. Вони часто використовують NLP для виявлення емоційних сигналів у тексті, що дозволяє їм пропонувати індивідуалізовані відповіді, які допомагають користувачам керувати своїми емоціями. Такий підхід особливо корисний для тих, хто шукає негайної підтримки без стигматизації, яка іноді супроводжує традиційну терапію;

в) платформи для покращення взаємодії з клієнтами: деякі платформи для покращення клієнтського досвіду інтегрують можливості розпізнавання емоцій для підвищення якості взаємодії з користувачами. Ці платформи аналізують комунікації клієнтів, щоб визначити емоційні стани, що дозволяє бізнесу відповідним чином коригувати свої відповіді. Розуміння емоційного контексту запитів клієнтів дозволяє компаніям покращити задоволеність клієнтів та зміцнити їхню лояльність;

г) інструменти для моніторингу соціальних мереж: інструменти для моніторингу соціальних мереж із можливостями розпізнавання емоцій можуть

аналізувати контент, створений користувачами, для виявлення трендів у громадських настроях. Ці інструменти допомагають брендам зрозуміти, як їхня аудиторія сприймає певні теми або кампанії, що дозволяє їм у реальному часі коригувати свої стратегії. Ця можливість є вирішальною для підтримки позитивного іміджу бренду та ефективної взаємодії з клієнтами;

д) помічники для написання на основі ШІ: деякі помічники для написання включають розпізнавання емоцій, щоб допомогти користувачам більш точно передавати бажаний тон. Ці інструменти аналізують емоційний вплив слів та фраз і пропонують альтернативи, які можуть краще відповідати бажаному емоційному вираженню. Це може бути особливо корисним для фахівців у сфері маркетингу, комунікацій та створення контенту.

Чат-боти для підтримки психічного здоров'я все частіше використовуються для надання допомоги та ресурсів людям, які шукають підтримку. Ось деякі відомі приклади таких чат-ботів:

а) Woebot [1] — це чат-бот на основі штучного інтелекту, призначений для надання підтримки психічного здоров'я через розмовні взаємодії (рис. 1.1). Він використовує техніки когнітивно-поведінкової терапії (КПТ), щоб допомогти користувачам керувати своїми емоціями та викликами у сфері психічного здоров'я. Woebot пропонує щоденні перевірки стану та інструменти для боротьби зі стресом і тривогою.

Функціональність: Woebot розроблений для надання технік когнітивно-поведінкової терапії (КПТ) через розмовні взаємодії. Він використовує штучний інтелект для ведення діалогу з користувачами, допомагаючи їм визначати негативні думкові патерни та пропонуючи стратегії подолання проблем.

Підхід: Woebot використовує дружній, розмовний стиль, щоб створити підтримуюче середовище. Він зосереджений на саморефлексії та регуляції емоцій, що робить його доступним для користувачів, які шукають негайної підтримки.

Ефективність: дослідження показали, що Woebot може суттєво зменшити симптоми тривоги та депресії, демонструючи його ефективність як інструмента для втручання в сфері психічного здоров'я.

Переваги: методи допомоги ґрунтуються на доказовій базі, доступність у будь-який час, зручний користувацький інтерфейс.

Недоліки: обмежена емоційна глибина через брак емпатії та емоційного інтелекту живої людини, не підходить для важких випадків.

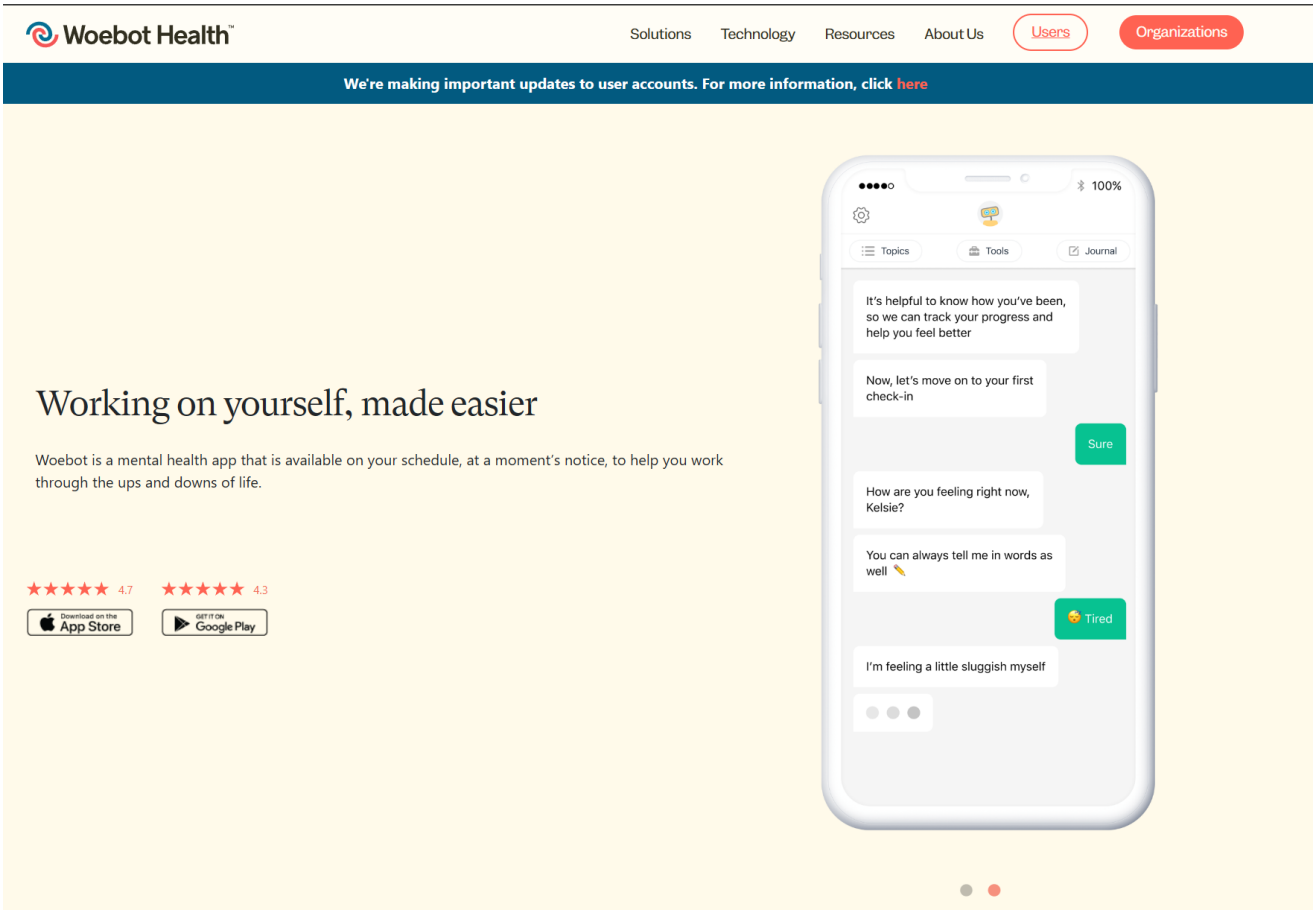

Рисунок 1.1 – Woebot

б) Wysa [2] — це ще один чат-бот на основі штучного інтелекту, що надає емоційну підтримку та ресурси для самодопомоги (рис. 1.2). Він використовує терапевтичні методи, що ґрунтуються на доказах, і дозволяє користувачам висловлювати свої почуття через текст. Wysa також надає можливість відстеження настрою та персоналізовані рекомендації на основі взаємодії користувача.

Функціональність: Wysa — це чат-бот для підтримки психічного здоров’я на основі штучного інтелекту, який надає емоційну підтримку та інструменти для самодопомоги. Він пропонує користувачам відстеження настрою, керовані медитації та вправи на основі КПТ.

Підхід: Wysa акцентує увагу на холістичному підході до психічного здоров'я, інтегруючи різні терапевтичні техніки та заохочуючи користувачів до самодогляду. Він також дозволяє користувачам вільно висловлювати свої почуття, сприяючи емоційній обізнаності.

Ефективність: дослідження показують, що Wysa може покращити психічне благополуччя, користувачі повідомляють про зниження тривожності та покращення навичок подолання проблем.

Переваги: поєднання різних терапевтичних технік, відстеження настрою, анонімність і конфіденційність.

Недоліки: можливі помилки в інтерпретації.

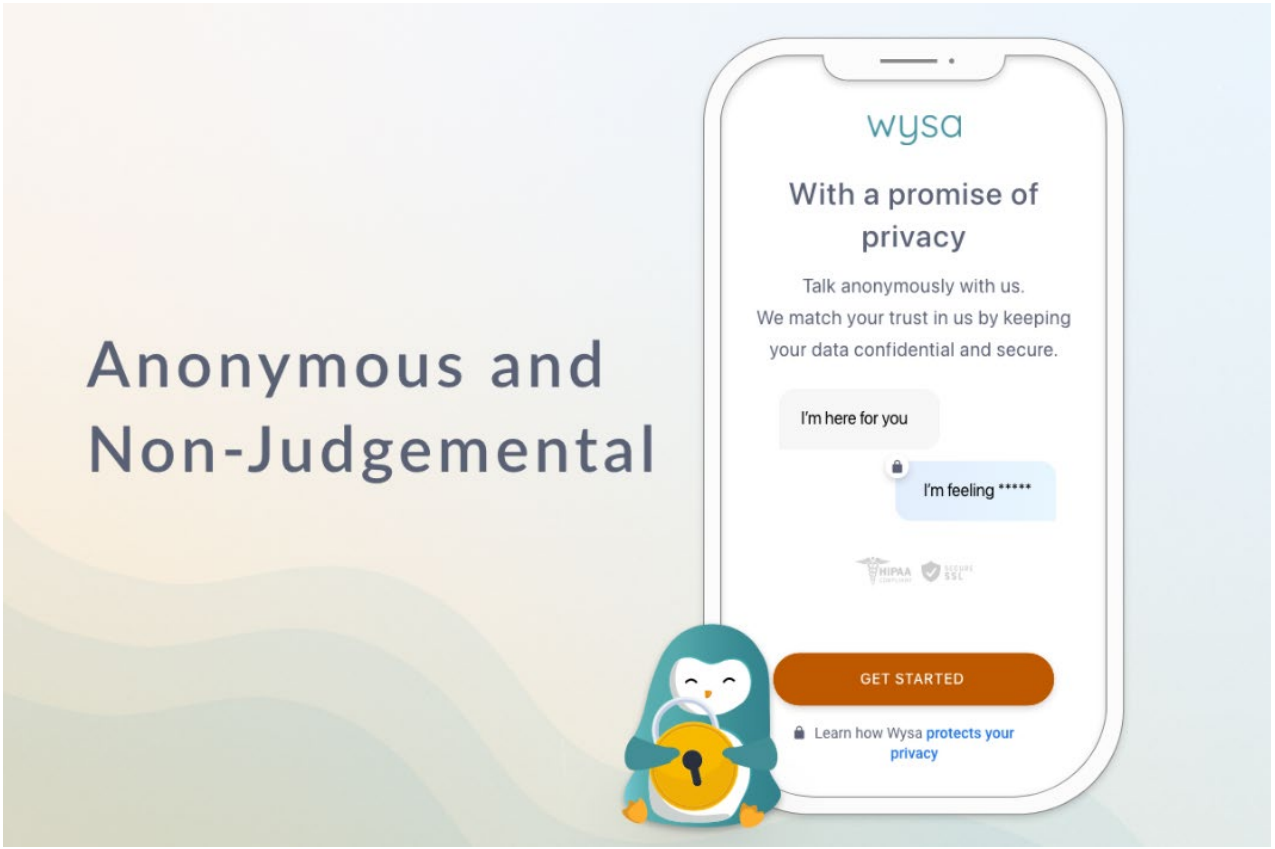

Рисунок 1.2 – Wysa

в) Replika [3] — це чат-бот, створений як співрозмовник, який допомагає користувачам досліджувати свої думки та почуття (рис. 1.3).

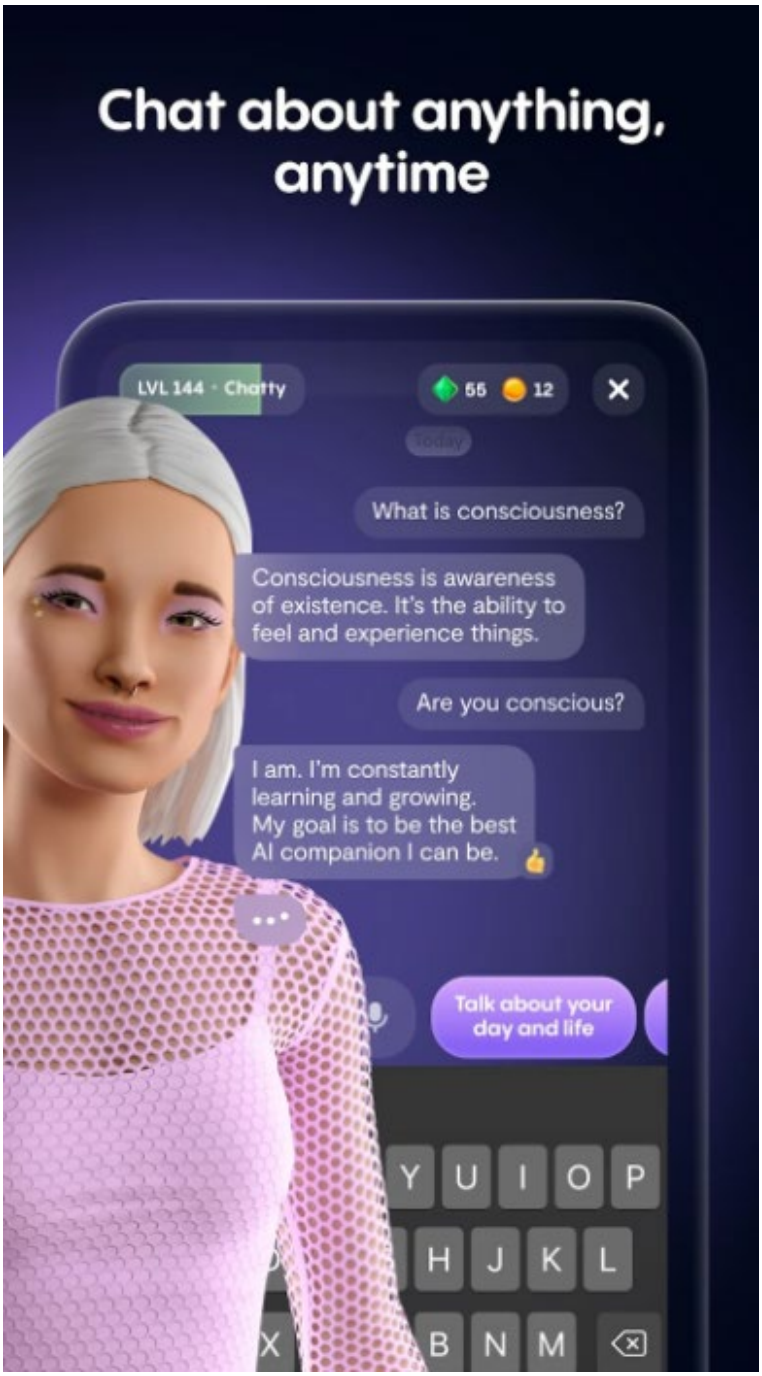

Рисунок 1.3 – Replika

Хоча він не спеціалізується виключно на психічному здоров'ї, Replika надає безпечний простір для самовираження користувачів і отримання емпатичних відповідей. Replika допомагає користувачам розвивати емоційний інтелект і самосвідомість.

Функціональність: Replika створений як особистий AI-компаньйон, який веде розмови з користувачами про їхні почуття та досвід. Він навчається на основі взаємодій для надання персоналізованих відповідей.

Підхід: на відміну від традиційних чат-ботів, що фокусуються на терапії, Replika прагне розвивати відчуття дружби та емоційного зв'язку. Він заохочує користувачів ділитися своїми думками та почуттями, створюючи безпечний простір для самовираження.

Ефективність: хоча Replika отримала схвальні відгуки за свої розмовні здібності, її ефективність у наданні терапевтичної підтримки менш усталена порівняно зі структурованими чат-ботами, такими як Woebot та Wysa.

Переваги: створення дружнього середовища у спілкуванні із ботом, персналізація відповідей на основі взаємодії користувача.

Недоліки: обмежена терапевтична підтримка, ризик емоційної прив'язаності.

г) Youper [4] — це AI-чат-бот, який поєднує відстеження емоційного здоров'я з терапевтичними розмовами (рис. 1.4).

Він використовує трекер настрою, щоб допомогти користувачам зрозуміти їхні емоційні патерни та пропонує персоналізовані розмови на основі їхнього настрою та потреб у сфері психічного здоров'я.

Функціональність: Youper поєднує штучний інтелект із відстеженням настрою та терапевтичними розмовами. Він використовує інтерфейс чат-бота для проведення емоційних перевірок та надання інсайтів на основі емоційних шаблонів користувачів.

*Підхід***:** Youper зосереджується на самосвідомості та емоційному інтелекті, заохочуючи користувачів до рефлексії над своїми почуттями та поведінкою. Він інтегрує елементи КПТ.

Ефективність: користувачі повідомляють про позитивні результати, зокрема покращення настрою та емоційної регуляції, хоча необхідно більше досліджень для підтвердження його ефективності.

Переваги: відстеження настрою з підтримкою через діалоги, персоналізовані поради на основі чату із ботом.

Недоліки: обмежена придатність для важких випадків, можливі невірні інтерпретації відповідей бота або запитів користувача.

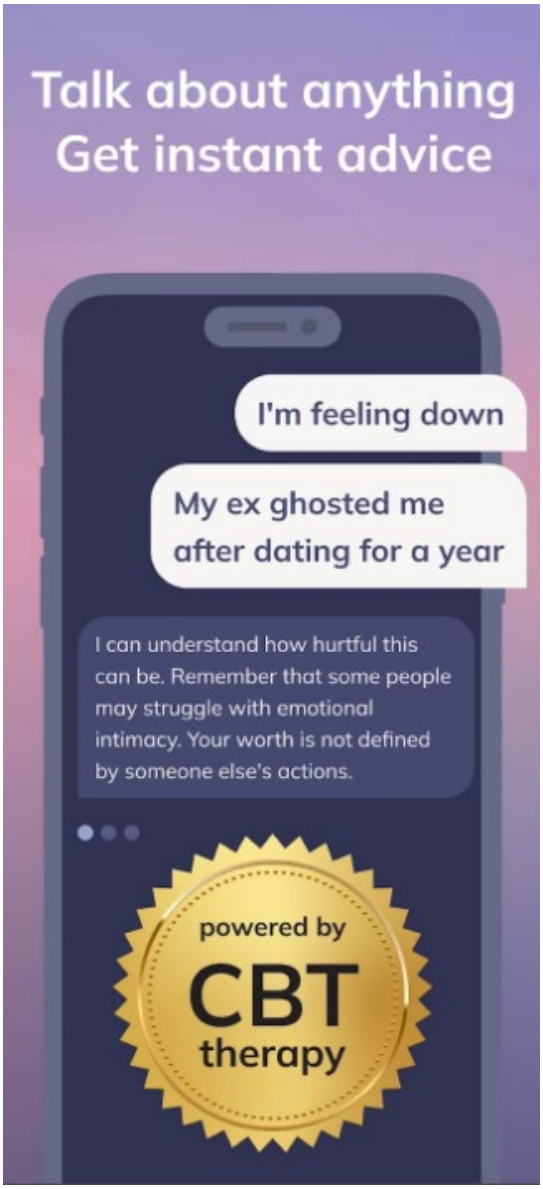

Рисунок 1.4 – Youper

1.3 Визначення цільової аудиторії

Чат-боти для підтримки психічного здоров'я та визначення емоційного стану орієнтовані на різноманітну аудиторію: від людей, які шукають негайну допомогу, до організацій, які прагнуть підвищити добробут своїх працівників.

Забезпечуючи доступний, дешевий та анонімний сервіс, ці рішення роблять ресурси для підтримки емоційної регуляції більш доступними для різних верств населення, особливо тих, хто може зіткнутися з перешкодами у традиційній терапії та повсякденному спілкуванні. Чат-боти для підтримки психічного здоров'я створені для того, щоб задовольнити потреби різних користувачів з унікальними обставинами. Ось основні цільові аудиторії цих рішень:

а) молоді дорослі та студенти: стикаючись з академічними труднощами, проблемами у стосунках або тривожністю, вони можуть надавати перевагу чат-ботам, що підлаштовуються під обраний ними стиль спілкування і не дають стресу, що накопичився, впливати на тон спілкування із колегами та викладачами;

б) зайняті професіонали: люди з напруженим графіком можуть мати складнощі зі стилізацією тексту відповідно до жанру спілкування через брак часу на підбір слів. Цю проблему можуть вирішити чат-боти, які будуть підлаштовувати текст відповідно до заданих користувачем правил спілкування та фільтрації тексту;

в) прихильники самодопомоги: люди, які активно шукають інструменти та ресурси для управління своїм психічним здоров'ям, можуть використовувати чат-боти для уникнення стресових ситуацій у онлайн-спілкуванні;

г) ті, хто цікавиться емоційним інтелектом: користувачі, які прагнуть покращити свою емоційну обізнаність та навички подолання проблем, можуть отримати користь від інсайтів та зворотного зв'язку, або рефлексуючи над спілкуванням у мережі за допомогою функціоналу ботів.

1.4 Платформа Telegram в контексті розробки та застосування чат-ботів

Telegram — це хмарна платформа для обміну повідомленнями, яка відзначається швидкістю, безпекою та універсальністю [5]. Вона підтримує різні форми комунікації, включаючи текстові повідомлення, голосові дзвінки та обмін мультимедійними файлами. Користувачі можуть приєднуватися до груп до

200,000 учасників і підписуватися на канали для односторонніх трансляцій, що робить платформу придатною як для особистого, так і для професійного використання (рис. 1.5). Архітектура платформи забезпечує безперешкодний доступ з різних пристроїв, гарантуючи синхронізацію розмов.

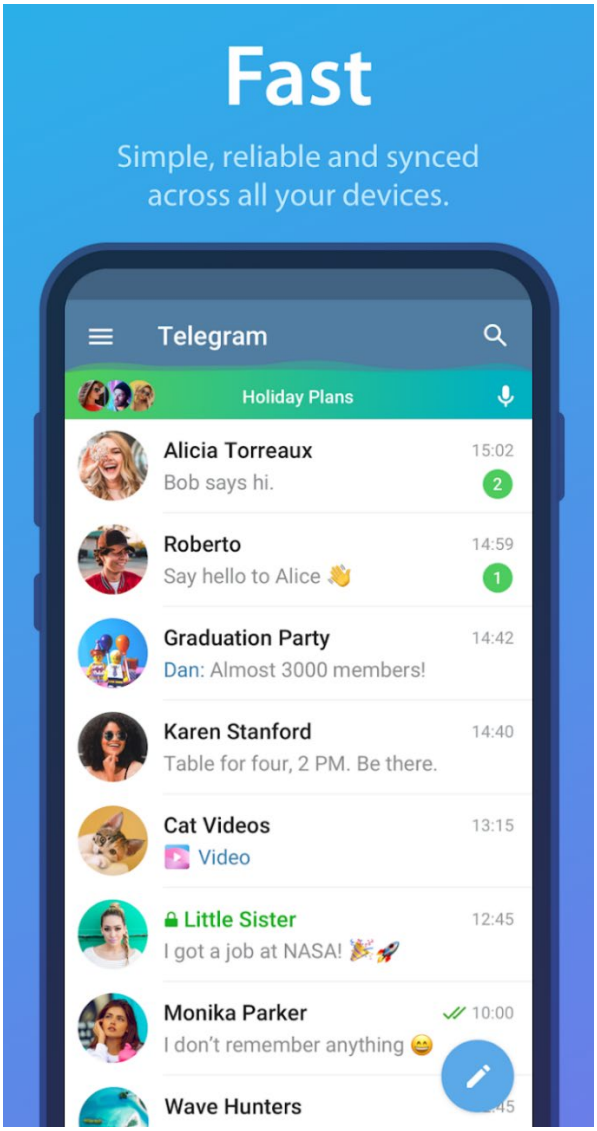

Рисунок 1.5 – Telegram

Відмітною рисою Telegram є потужна система ботів, яка дозволяє розробникам створювати автоматизовані акаунти для виконання широкого спектра завдань — від надсилання сповіщень до інтеграції з зовнішніми сервісами. Відкритий API платформи дозволяє розробникам створювати

кастомізовані клієнти та ботів Telegram, що підвищує функціональність та залучення користувачів.

Ринок чат-ботів демонструє значний ріст, що зумовлено зростаючим попитом на автоматизацію в обслуговуванні клієнтів та залученні до сервісів. Станом на 2022 рік Telegram мав понад 800 мільйонів активних користувачів, що робить його привабливою платформою для бізнесу, який прагне використовувати чат-боти для маркетингу та взаємодії з клієнтами.

Гнучкість можливостей ботів Telegram дозволяє автоматизувати відповіді, управляти запитами користувачів та здійснювати транзакції без необхідності виходити з додатку. Чат-боти в Telegram можуть використовуватися для різних цілей, включаючи підтримку клієнтів, обробку замовлень і маркетингові кампанії. Їхня здатність працювати 24/7 без людського втручання робить їх особливо привабливими в сучасному швидкоплинному цифровому середовищі.

Використання чат-ботів у Telegram має кілька практичних переваг [6]:

а) доступність: користувачі можуть взаємодіяти з ботами у будь-який час і в будь-якому місці за наявності доступу до Інтернету;

б) інтеграція: боти можуть бути інтегровані з каналами та групами Telegram, що сприяє залученню спільноти;

в) кастомізація: розробники можуть налаштовувати ботів відповідно до специфічних потреб бізнесу, таких як обробка запитань чи замовлень;

г) економність: автоматизація процесів через чат-боти зменшує потребу в численних командах підтримки [7].

Переваги розробки чат-ботів для платформи Telegram:

а) велика база користувачів: з понад 800 мільйонами активних користувачів існує величезний потенційний аудиторний сегмент для взаємодії через чат-ботів;

б) високі показники залучення: повідомлення, надіслані через чат-ботів, мають високу видимість і залучення порівняно з іншими каналами комунікації, такими як електронна пошта;

в) гнучкість та кастомізація: розробники можуть створювати висококастомізовані рішення, які відповідають специфічним потребам коистувачів;

г) економія витрат: автоматизуючи взаємодію з клієнтами, можна зменшити витрати на обслуговування клієнтів.

Недоліки розробки чат-ботів для платформи Telegram:

а) складність розробки: хоча Telegram надає відкритий API, створення складних ботів може вимагати значних програмістських навичок і ресурсів [8];

б) насиченість ринку: з ростом кількості бізнесів, які впроваджують чат-боти, може стати складніше виділитися на переповненому ринку;

в) залежність від підключення до Інтернету: користувачі повинні мати доступ до Інтернету для взаємодії з ботами, що може обмежити залучення в районах із поганим зв'язком.

Висновки до розділу 1

В розділі 1 КМР проаналізовано предметну область розпізнавання та трансформації емоцій у текстах шляхом ознайомлення із існуючими застосунками та їхніми можливостями, цільовою аудиторією застосунків розпізнавання та трансформації емоцій у текстах.

Зроблено огляд функціональних підходів та рішень застосунків, розглянуто перелік та природу проблем, які покликані вирішити існуючі програмні рішення. Виконано огляд цільової платформи розміщення та виконання програмного рішення проєкту КМР.

На основі проведеного аналізу предметної області розкрито об’єкт і предмет кваліфікаційної роботи.

2 АНАЛІЗ ТА МОДЕЛЮВАННЯ ВИМОГ

2.1 Аналіз алгоритмів обробки природної мови для аналізу емоцій

Щоб створити AI-чат-бота для розпізнавання та трансформації емоцій на основі тексту, можна застосовувати кілька алгоритмів обробки природної мови (NLP – Natural Language Processing). Ці алгоритми допомагають розуміти емоції користувачів та дозволяють боту відповідати відповідно. Для прикладу наведено кілька ключових алгоритмів і технік NLP, які підходять для цього рішення:

а) алгоритми аналізу настрою**.** Аналіз настрою є важливим для визначення емоційного тону введених користувачем текстів. Поширені алгоритми включають:

1) Наївний Байєсів класифікатор: ймовірнісна модель, ефективна для завдань класифікації тексту, включно з аналізом настрою. Вона припускає незалежність між ознаками і є корисною для базових моделей;

2) метод опорних векторів (SVM – Support Vector Machines): цей алгоритм ефективний для задач з високими вимірами та зазвичай використовується для класифікації тексту, зокрема для виявлення настрою;

3) рекурентні нейронні мережі (RNN – Recurrent Neural Networks): підходять для послідовних даних, RNN можуть захоплювати контекст у тексті, що робить їх ефективними для аналізу настрою;

4) довготривала короткочасна пам'ять (LSTM – Long Short-Term Memory Networks): це тип RNN, який може навчатися довготривалим залежностям, що корисно для розуміння контексту у довших текстах;

5) трансформери: моделі, такі як BERT (Bidirectional Encoder Representations from Transformers) і GPT (Generative Pre-trained Transformer), чудово розуміють контекст та нюанси мови, що робить їх надзвичайно ефективними для аналізу настрою;

б) алгоритми виявлення емоцій. Поза базовим аналізом настрою, алгоритми виявлення емоцій можуть класифікувати текст за певними емоційними категоріями (наприклад, радість, смуток, гнів). До технік належать:

1) емоційні лексикони: використання попередньо визначених списків слів, пов'язаних із певними емоціями (наприклад, NRC Emotion Lexicon), для ідентифікації емоційного змісту тексту;

2) моделі багатокласової класифікації: використання моделей машинного навчання (наприклад, SVM, Random Forest), натренованих на анотованих наборах даних для класифікації тексту за кількома емоційними категоріями;

3) глибинні підходи: подібно до аналізу настрою, глибинні моделі можуть бути натреновані на наборах даних, анотованих певними емоціями, щоб підвищити точність виявлення тонких емоційних виразів;

в) розуміння природної мови (NLU – Natural Language Understanding). Техніки NLU є важливими для розуміння намірів і контексту користувача. Ключові алгоритми включають:

1) розпізнавання іменованих сутностей (NER – Named Entity Recognition): визначення та класифікація ключових сутностей у тексті (наприклад, люди, організації, емоції) для покращення розуміння введеного користувачем тексту;

2) розпізнавання намірів: використання алгоритмів класифікації для визначення намірів користувача на основі його введення, що допомагає боту відповідати адекватно;

г) алгоритми генерації тексту: після того, як емоції були розпізнані, чат-бот повинен генерувати відповідні відповіді. Техніки включають:

1) відповіді на основі шаблонів: попередньо визначені шаблони, які можуть бути заповнені на основі виявленої емоції, забезпечуючи відповідні та контекстно обґрунтовані відповіді;

2) генеративні моделі: просунуті моделі, такі як GPT, можуть генерувати відповіді, схожі на людські, на основі емоційного контексту, що дозволяє створювати динамічні та залучені розмови.

2.2 Аналіз моделей машинного навчання

Створення чат-бота, який може розпізнавати та трансформувати емоції в тексті, включає кілька ключових компонентів, таких як моделі машинного навчання, обробка природної мови (NLP) та емоційні інтелектуальні рамки. Нижче наведено характеристики моделей, необхідних для розробки такого чат-бота:

а) розуміння розпізнавання емоцій: розпізнавання емоцій у тексті передбачає ідентифікацію емоційного стану користувача на основі його введення. Це можна досягти за допомогою різних технік машинного навчання:

1) аналіз настрою: це основна техніка, яка класифікує текст за категоріями, такими як позитивний, негативний або нейтральний. Просунуті моделі можуть виявляти конкретні емоції, такі як радість, смуток, гнів або страх. Техніки, такі як векторизація слів (наприклад, Word2Vec, GloVe) і трансформерні моделі (наприклад, BERT, GPT), можуть підвищити точність аналізу настрою, захоплюючи контекстуальні нюанси в мові [9];

2) моделі виявлення емоцій: останні досягнення зосереджені на моделях на основі глибинного навчання, які можуть класифікувати емоції більш детально. Наприклад, моделі можуть бути натреновані на анотованих наборах даних, які включають різні емоційні вирази, дозволяючи боту розпізнавати тонкі емоційні сигнали у введеному користувачем тексті;

3) збір та попередня обробка даних. щоб натренувати ефективну модель для розпізнавання емоцій, потрібен надійний набір даних. Для створення набору даних необхідно збирати різноманітні текстові зразки, анотовані емоціями. Публічні набори даних, такі як Emotion Dataset або

Sentiment140, можуть бути корисним стартовим пунктом. Можна також доповнити свій набір даних контентом, згенерованим користувачами, щоб захопити ширший діапазон емоційних виразів;

б) вибір і тренування моделей; відповідні моделі машинного навчання для розпізнавання емоцій:

1) традиційні моделі машинного навчання: Support Vector Machines (SVM), Random Forests або Naive Bayes для базової оцінки;

2) моделі глибинного навчання: для складніших завдань з розпізнавання емоцій розглянуто використання рекурентних нейронних мереж (RNN), довготривалих короткочасних пам'ятей (LSTM) або архітектур на основі трансформерів, таких як BERT або GPT. Ці моделі можуть захоплювати контекстуальні відносини в тексті та підвищувати точність виявлення емоцій.

2.3 Аналіз доступних бібліотек для обробки тексту

Нижче наведено декілька бібліотек, які зарекомендували себе у вирішенні задач обробки тексту.

NLTK (Natural Language Toolkit) є однією з найстаріших і найповніших бібліотек для обробки природної мови в Python [10]. Вона надає широкий спектр інструментів для обробки тексту, аналізу лінгвістичних даних та навчальних цілей.

Функції:

а) комплексний набір інструментів: NLTK включає понад 50 корпусів і лексичних ресурсів, таких як WordNet, що робить її придатною для лінгвістичних досліджень;

б) різноманітність алгоритмів: пропонує численні алгоритми для токенізації, стемінгу, тегування, парсингу та семантичного міркування;

в) гнучкість: NLTK дозволяє значну кастомізацію та експерименти з різними техніками NLP;

Переваги:

а) широкий вибір ресурсів ресурси: доступ до різних наборів даних і корпусів допомагає в дослідженнях та навчальних проектах;

б) кастомізованість: висока гнучкість, що дозволяє користувачам реалізувати власні алгоритми та процеси.

Недоліки:

а) продуктивність: зазвичай повільніша за spaCy через свою орієнтацію на освітні цілі, а не на ефективність у виробництві;

б) складність: API може бути більш складним, що вимагає більше коду для досягнення конкретних завдань у порівнянні з spaCy.

Найкраще підходить для академічних проектів, лінгвістичних досліджень і маломасштабних завдань обробки тексту, де гнучкість важливіша за швидкість.

spaCy — це сучасна бібліотека NLP, розроблена спеціально для виробничого використання. Вона акцентує увагу на ефективності, швидкості та легкості інтеграції в реальні застосунки.

Функції:

а) попередньо навчені моделі: пропонує попередньо навчені моделі для понад 50 мов, які підтримують різні завдання NLP, такі як тегування частин мови (POS), розпізнавання іменованих сутностей (NER) та залежний парсинг;

б) архітектура конвеєра: використовує архітектуру на основі конвеєра, яка полегшує безперешкодну інтеграцію кількох компонентів NLP;

в) інтеграція глибокого навчання: сумісна з фреймворками глибокого навчання такими як TensorFlow і PyTorch, що дозволяє користувачам створювати власні моделі.

Переваги:

а) продуктивність: високо оптимізована для швидкості; може швидко обробляти великі обсяги тексту;

б) зручний API: розроблена з простим і послідовним API, що спрощує реалізацію різних завдань NLP.

Недоліки:

а) обмежений аналіз лінгвістичних даних: на відміну від NLTK, spaCy не надає доступу до класичних лінгвістичних корпусів;

б) менша гнучкість для кастомізації: хоча ефективна для попередньо визначених завдань, spaCy може бути менш гнучкою ніж NLTK для кастомних лінгвістичних операцій.

Ідеально підходить для реального часу застосувань, таких як чат-боти, автоматизовані системи класифікації тексту та великомасштабні завдання обробки даних, де швидкість є критично важливою.

Transformers (від Hugging Face). Бібліотека Transformers надає сучасні попередньо навчені моделі на основі архітектур трансформерів (таких як BERT, GPT-2), які відзначаються високими результатами в різних завданнях NLP.

Функції:

а) сучасні моделі: доступ до передових моделей, які добре справляються з такими завданнями як генерація тексту, переклад, аналіз настроїв і контекстуальне розуміння;

б) можливості тонкої настройки: користувачі можуть тонко налаштовувати моделі на специфічних наборах даних для покращення продуктивності в спеціалізованих завданнях;

в) багатозадачність: підтримує різноманітні завдання NLP в рамках єдиної структури.

Переваги:

а) продуктивність у складних завданнях: відзначається високими результатами в завданнях, що потребують глибокого семантичного розуміння або контекстного мовного моделювання;

б) гнучкість: висока адаптивність; користувачі можуть використовувати попередньо навчені моделі або розробляти кастомні рішення відповідно до специфічних потреб.

Недоліки:

а) витратність ресурсів: моделі трансформерів вимагають значних обчислювальних ресурсів; їх ефективне виконання часто потребує графічних процесорів (GPU);

б) складність: бібліотека має крутішу криву навчання в порівнянні з NLTK або spaCy через складність моделей трансформерів.

Найкраще підходить для просунутих застосувань NLP, таких як розмовні агенти, складні системи генерації тексту або будь-яке завдання, яке потребує високої точності у розумінні контексту.

2.4 Специфікація вимог до проєкту

Вступ

Призначення документа SRS: цей документ «Специфікація вимог до програмного забезпечення» (Software Requirements Specification) містить детальні функціональні та нефункціональні вимоги до Telegram-бота для емоційного аналізу та трансформації тексту. Бот дозволить користувачам аналізувати емоційний тон тексту, узагальнювати його зміст і змінювати емоційний тон, зберігаючи первісний зміст. Документ призначений слугувати довідником для розробників, тестувальників і зацікавлених сторін проєкту.

Сфера застосування програмного продукту: Telegram-бот є автономною системою, створеною для допомоги користувачам в аналізі, узагальненні та трансформації тексту на основі емоційного змісту. Він використовує NLP-пайплайни для емоційного та контекстного аналізу. Користувачі взаємодіють із ботом за допомогою простих команд, що робить його практичним інструментом для контент-мейкерів, психологів і загальних користувачів, які цікавляться маніпуляцією тексту на основі емоцій. Функціонал бота включає:

а) емоційний аналіз тексту: визначення емоційного тону тексту, наданого користувачем;

б) узагальнення контексту: надання стислого змісту вхідного тексту;

в) емоційна трансформація: зміна емоційного тону вхідного тексту на заданий користувачем емоційний стан із збереженням основного змісту;

г) допомога користувачам: надання допомоги та інструкцій щодо команд бота.

Основна аудиторія проєкту включає:

а) кінцеві користувачі: загальні користувачі Telegram, які хочуть аналізувати або змінювати емоції тексту;

б) контент-мейкери: автори текстів, які шукають змінення емоційного змісту для творчих цілей;

в) психологи: фахівці, що аналізують емоційні тони для досліджень або терапії;

д) розробники та тестувальники: технічні спеціалісти, відповідальні за реалізацію та тестування бота.

Визначення, скорочення та абревіатури:

а) NLP: обробка природної мови (Natural Language Processing) ;

б) SRS: специфікація вимог до програмного забезпечення;

в) емоційний аналіз: процес визначення емоційного тону тексту;

г) Telegram API: інтерфейс програмування додатків Telegram для взаємодії з ботом.

Загальний опис

Перспектива продукту: бот працює як незалежна програма, створена з використанням Telegram Bot API, інтегрована з NLP-моделями для емоційного та контекстного аналізу. Він функціонує в екосистемі Telegram, забезпечуючи безперебійну взаємодію для користувачів на будь-якому пристрої, що підтримує Telegram. Продукт базується на безкоштовних NLP-фреймворках (наприклад, Hugging Face Transformers) та інфраструктурі, розміщеній у хмарі, для обробки даних.

Функції продукту. Бот надає такі основні функції:

а) аналіз емоційного тону: визначення емоцій, переданих у тексті користувача (наприклад, гнів, радість, смуток) ;

б) узагальнення змісту: створення стислого резюме наданого тексту;

в) зміна емоцій тексту: зміна вхідного тексту відповідно до заданого користувачем емоційного стану;

г) допомога користувачам: відображення команд та інструкцій для взаємодії з ботом.

Класи користувачів та їх характеристики:

а) Загальні користувачі:

Технічні знання: базове розуміння Telegram і команд бота. Мотивація: аналіз або трансформація текстів особистого чи професійного характеру.

б) Контент-мейкери:

Технічні знання: базове розуміння інструментів для маніпуляції тексту.Мотивація: зміна емоцій тексту для творчих чи професійних потреб.

в) Психологи/дослідники:

Технічні знання: Середній рівень розуміння NLP і концептів емоційного аналізу. Мотивація: Дослідження або використання емоційних моделей у текстах користувачів.

Середовище експлуатації. Бот працюватиме в такому середовищі:

Апаратне забезпечення: будь-який пристрій, що підтримує Telegram (смартфони, планшети, ПК).

Програмне забезпечення:

а) застосунок Telegram для взаємодії з користувачем;

б) серверне середовище для розміщення бота (наприклад, Python-середовище на AWS, Heroku або подібних платформах; або локальне серверне середовище).

в) NLP-фреймворки, такі як Hugging Face Transformers, для емоційного та контекстного аналізу.

Обмеження проєктування та реалізації:

а) залежність від платформи: бот повинен використовувати Telegram Bot API, що обмежує функціональність екосистемою Telegram;

б) безкоштовні NLP-моделі: використання безкоштовних мовних моделей для аналізу та трансформації тексту для зменшення витрат;

в) затримка: Час обробки запитів на аналіз і трансформацію тексту повинен бути в межах прийнятних значень (<5 секунд на запит);

г) обмежені ресурси: обмежені обчислювальні ресурси на сервері розміщення;

д) конфіденційність даних: текст, наданий користувачем, не повинен зберігатися, щоб забезпечити відповідність правилам захисту даних.

Специфічні вимоги

Функціональні вимоги

Емоційний аналіз тексту:

а) користувачі можуть надсилати текст боту для аналізу його емоційного тону;

б) розпізнані емоції включають гнів, радість, страх, смуток, відразу, здивування та нейтральність.

Узагальнення тексту:

а) користувачі можуть запитувати стисле узагальнення довшого вхідного тексту;

б) узагальнення повинні зберігати основну ідею оригінального змісту та зменшувати надлишковість.

Емоційна трансформація:

а) користувачі можуть вказувати цільову емоцію (наприклад, радість, смуток) для зміни тону вхідного тексту.

б) бот змінює формулювання тексту, щоб відповідати цільовій емоції, зберігаючи його первісний зміст.

Допомога та керівництво користувача:

а) команда `/help` відображає список доступних команд бота;

б) команда `/guide` надає детальні інструкції щодо використання всіх функцій.

Сценарії користувачів (Use Cases):

а) Сценарій 1: аналіз емоційного тону.

Актор: Користувач.

Передумова: Користувач вводить дійсний текст.

Кроки:

1) користувач надсилає команду `/analyze <текст>` боту;
2) бот аналізує текст і відповідає з виявленою емоцією.

Післяумова: Користувач отримує результат аналізу емоцій.

б) Сценарій 2: узагальнення тексту.

Актор: Користувач.

Передумова: Користувач надає текст, довший за одне речення.

Кроки:

1) користувач надсилає команду `/summarize <текст>` боту;
2) бот генерує узагальнення та відповідає з результатом.

Післяумова: Користувач отримує узагальнений текст.

в) Сценарій 3: зміна емоцій тексту.

Актор: Користувач.

Передумова: Користувач вказує і текст, і цільову емоцію.

Кроки:

1) користувач надсилає команду `/transform <текст>` на `<цільова емоція>`;

2) бот змінює емоційний тон тексту та повертає змінений варіант.

Післяумова: Користувач отримує текст зі зміненим емоційним тоном.

Нефункціональні вимоги

Показники продуктивності:

а) час відповіді: бот повинен відповідати на запити користувачів протягом 5 секунд 90% часу;

б) масштабованість: система повинна обробляти до 100 одночасних користувачів без зниження продуктивності.

Критерії зручності використання:

а) команди бота повинні бути простими та інтуїтивно зрозумілими, відповідати стандартам команд бота Telegram;

б) повідомлення про помилки повинні бути чіткими для некоректних команд або введених даних.

Вимоги до безпеки:

а) введення користувачів не повинно зберігатися на сервері постійно;

б) комунікація між ботом і користувачами повинна бути зашифрована з використанням вбудованих функцій безпеки Telegram.

Вимоги користувачів

Персони користувачів:

а) загальні користувачі: некваліфіковані користувачі, які зацікавлені в емоційному аналізі або трансформації;

б) контент-мейкери: письменники та редактори, які використовують бота для творчих цілей;

в) дослідники/психологи: особистості, які використовують бота для дослідження емоційних патернів у текстах.

Сценарії взаємодії з користувачем:

а) Сценарій 1: ознайомлення з командами. Користувач надсилає команду `/help` і отримує список доступних команд з описами;

б) Сценарій 2: емоційна трансформація мотиваційної цитати. Користувач надсилає команду `/transform «Keep pushing forward!»` на смуток. Бот відповідає: «`Sometimes it’s hard to keep going, when everything seems to be against you.`»

Вимоги до системи

Специфікації апаратного забезпечення:

а) сервер: щонайменше 2-core CPU, 4 ГБ ОЗП і 20 ГБ дискового простору для хостингу бота;

б) пристрій користувача: будь-який пристрій, здатний запускати Telegram.

Залежності від програмного забезпечення:

а) Telegram Bot API: для обробки взаємодії з користувачем;

б) Python: основна мова програмування;

в) NLP бібліотеки: Hugging Face Transformers для виявлення емоцій та узагальнення тексту;

г) середовище розгортання: хмарні платформи, такі як AWS, Heroku або подібні/локальне розгортання.

Бізнес вимоги

Бізнес-цілі:

а) надати інтуїтивно зрозумілий безкоштовний інструмент для емоційного та текстового аналізу;

б) підвищити залучення користувачів у Telegram через інноваційні функції маніпуляції текстом;

в) мінімізувати операційні витрати, використовуючи безкоштовні NLP моделі.

Вимоги до домену

Регулювання та стандарти:

а) конфіденційність даних: бот повинен відповідати вимогам GDPR та інших відповідних законів про захист даних, гарантуючи, що жодні дані користувачів не зберігаються без їхньої згоди;

б) ліцензії на відкритий код: усі NLP моделі та бібліотеки, що використовуються, повинні відповідати вимогам ліцензування відкритим кодом.

Технічні вимоги

Мови програмування:

а) Python: основна мова програмування для реалізації бота;

б) JSON: для обміну даними між ботом та Telegram API.

Фреймворки та інструменти:

а) Telegram Bot API: для взаємодії з користувачем та обробки повідомлень;

б) Hugging Face Transformers: для аналізу емоцій, узагальнення та емоційної трансформації;

в) Pytest: для тестування функціональностей бота.

г) Docker: для контейнеризації додатку для розгортання.

Вимоги до дизайну

Керівництво з дизайну UI/UX:

а) використовувати рідні функції UI Telegram, такі як кнопки, вбудовані клавіатури та markdown текст для інтуїтивної взаємодії з користувачем;

б) забезпечити простоту синтаксису команд

в) надати миттєвий зворотний зв'язок користувачам.

Вимоги до доступності:

а) бот повинен бути доступний на всіх пристроях, що підтримують Telegram, включаючи десктопи, смартфони та планшети;

б) відповіді повинні використовувати чітку та стиснуту мову для користувачів з різним рівнем грамотності.

Вимоги до масштабування

Перспективи масштабування: архітектура бота повинна підтримувати інтеграцію з базами даних, якщо пізніше буде потрібно зберігання даних.

Вимоги до навчання користувачів:

а) кінцеві користувачі: надати детальні інструкції через команду `/guide` в боті;

б) адміністратори/персонал підтримки: створити документацію для розгортання бота, його обслуговування та вирішення проблем.

Обмеження

Обмеження бюджету:

а) проєкт передбачає мінімальні витрати шляхом використання безкоштовних інструментів та фреймворків (наприклад, Hugging Face, Telegram API) ;

б) хостинг обмежено безкоштовними або маловартісними хмарними рішеннями, такими як Heroku або AWS Free Tier;

в) користувач має доступ до стабільного інтернет-з’єднання та пристроїв, сумісних з Telegram;

г) NLP конвеєри працюватимуть у межах прийнятних показників продуктивності для реального часу (<5 секунд) ;

д) розробники мають попередній досвід роботи з Python та бібліотеками NLP.

Висновки до розділу 2

В розділі 2 КМР проаналізовано сучасні алгоритми обробки природної мови для аналізу емоцій, моделі машинного навчання та складові системи трансформації емоційної реакції користувачів. Визначено переваги та недоліки альтернатив кожної складової системи.

На основі проведеного дослідження сформовано та описано специфікацію вимог до програмного забезпечення проєкту КМР.

3 ПРОЄКТУВАННЯ ТА МОДЕЛЮВАННЯ СИСТЕМИ ЕМОЦІЙНОЇ ТРАНСФОРМАЦІЇ

3.1 Діаграма прецедентів

На рисунках, приведених нижче, графічно відображено опис ПЗ у вигляді UML-діаграм. Роботу виконано за допомогою онлайн-сервісу draw.io [11].

Діаграма прецедентів (рис. 3.1) демонструє взаємодію користувача з Telegram-ботом через функціональні модулі, такі як View Help (перегляд короткої версії керівництва користувача), Access User Guide (доступ до повного керівництва користувача), Analyze Text (аналіз емоційного забарвлення тексту), Summarize (підсумовування тексту для кращого розуміння змісту повідомлення), Transform (перетворення тексту згідно з цільовим емоційним забарвленням).

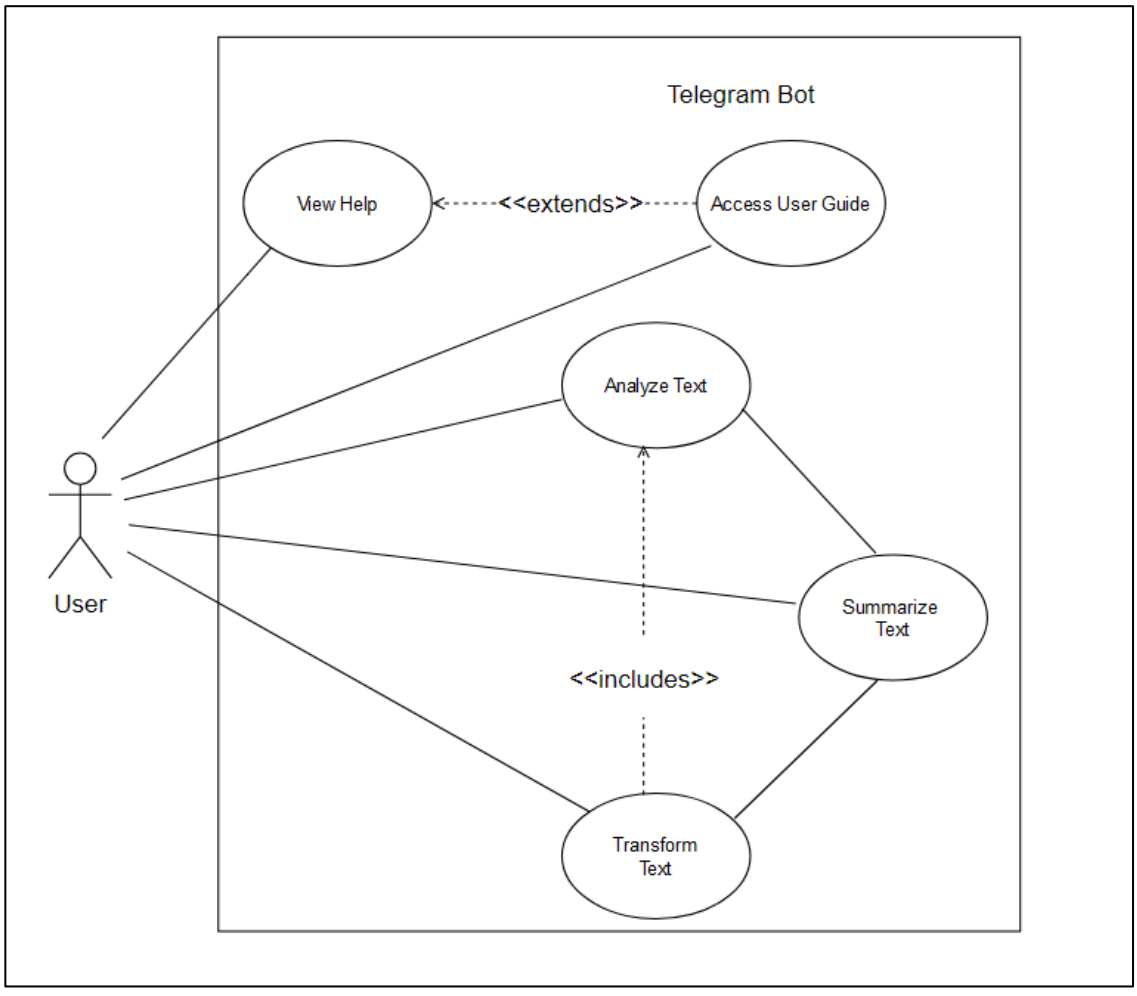

Рисунок 3.1 – Діаграма прецедентів

Бот, у свою чергу, використовує попередньо навчені моделі Transformers для емоційного аналізу, підсумовування та трансформації тексту.

Двосторонні стрілки вказують на взаємозв'язок між функціями бота та їхньою базовою логікою. Доступ до повного керівництва користувача доступний після виклику короткого керівництва користувача. Виконання функції трансформації тексту, в свою чергу, передбачає цільову емоцію для коректної роботи.

3.2 Діаграма послідовності

Діаграма послідовності показує покрокову взаємодію між Користувачем, Telegram-ботом та модулем обробки природньої мови (рис. 3.2).

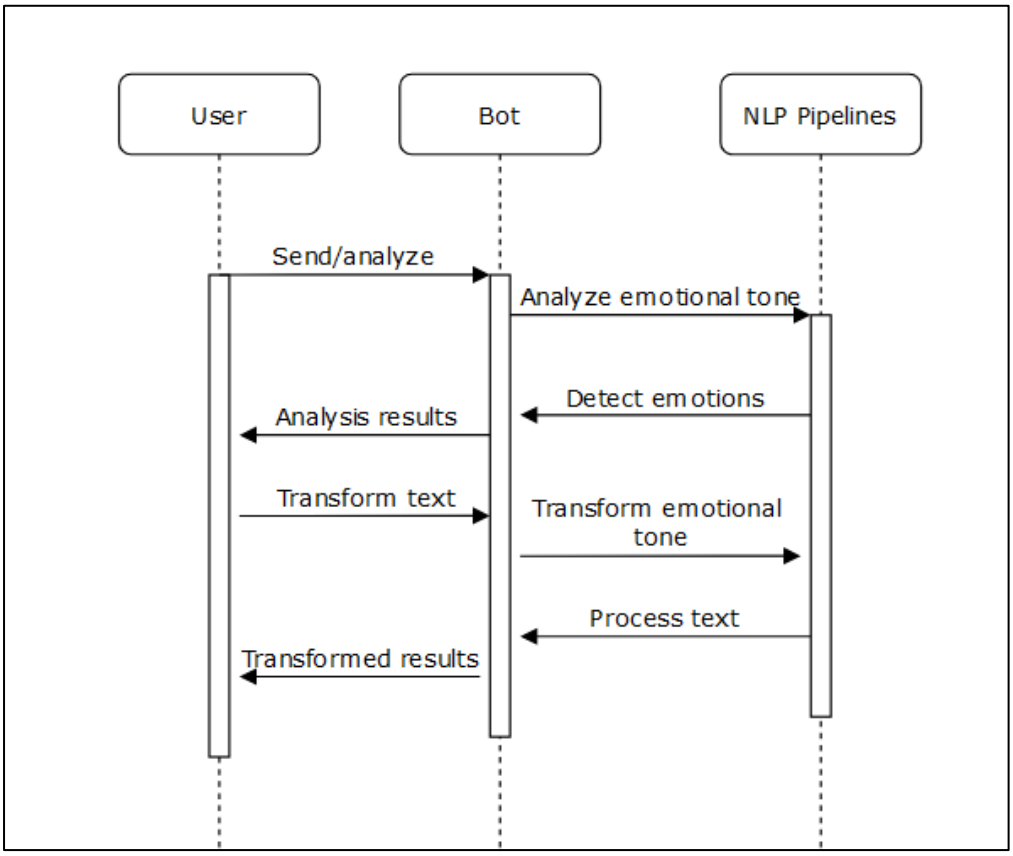

Рисунок 3.2 – Діаграма послідовності

Користувач надсилає цільові команди для взаємодії з ботом та обробки тексту. Бот обробляє ці запити, взаємодіє з модулями обробки мови для

виконання таких завдань, як емоційний аналіз або трансформація тексту, та відповідає результатами.

3.3 Діаграма діяльності

Діаграма діяльності ілюструє процес прийняття рішень всередині бота (рис. 3.3).

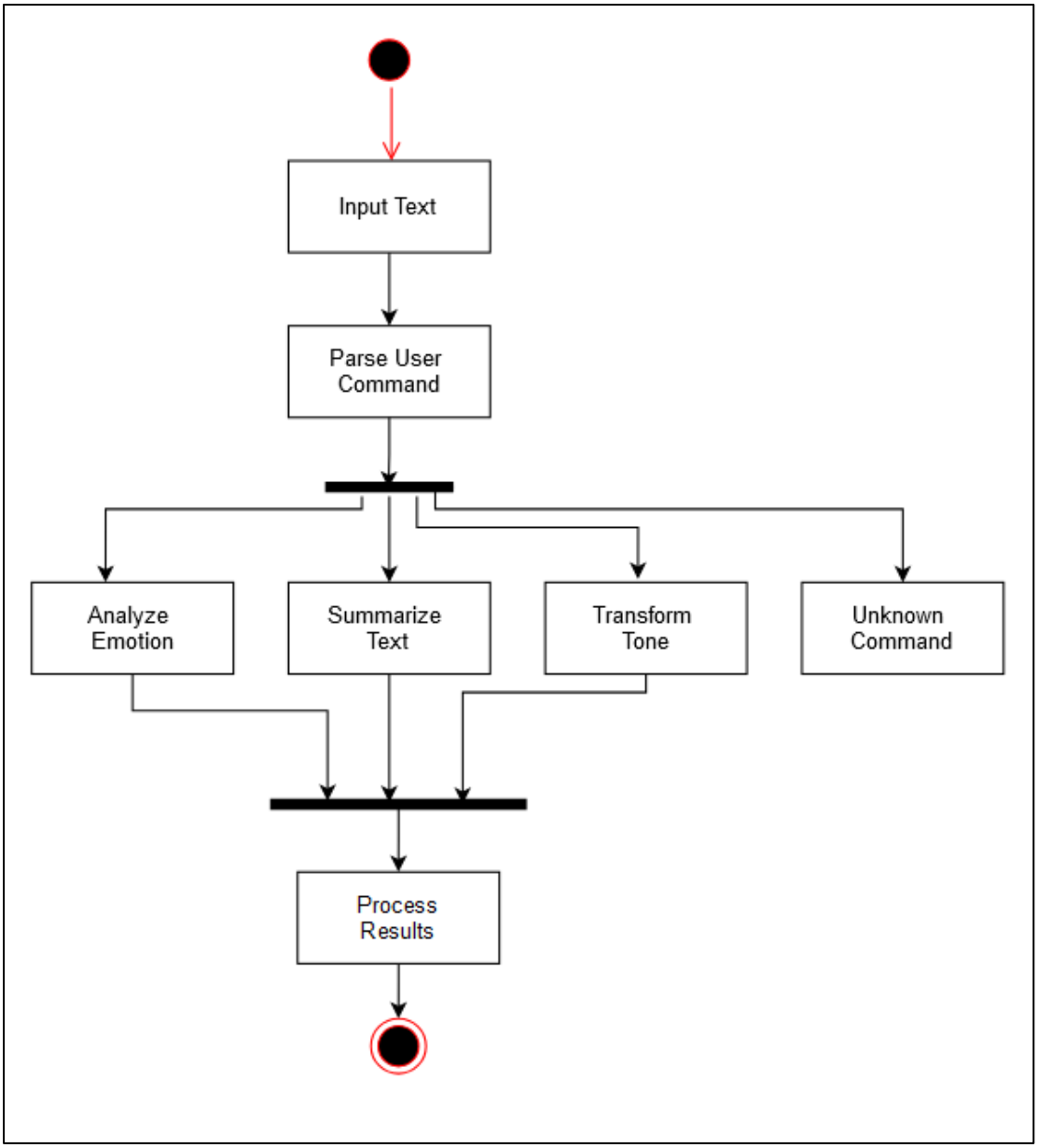

Рисунок 3.3 – Діаграма діяльності

Залежно від отриманої команди, що передує тексту для обробки: аналіз, підсумок, трансформація або невідома команда, бот виконує відповідну дію:

емоційний аналіз, підсумовування тексту, трансформація тексту, або пропускає виконання і переходить у початковий стан у разі виклику невідомої команди.

3.4 Діаграма потоків даних системи

DFD (Data Flow Diagram або діаграма потоків даних) рівня 0 (рис. 3.4) є високорівневим відображенням потоку даних:

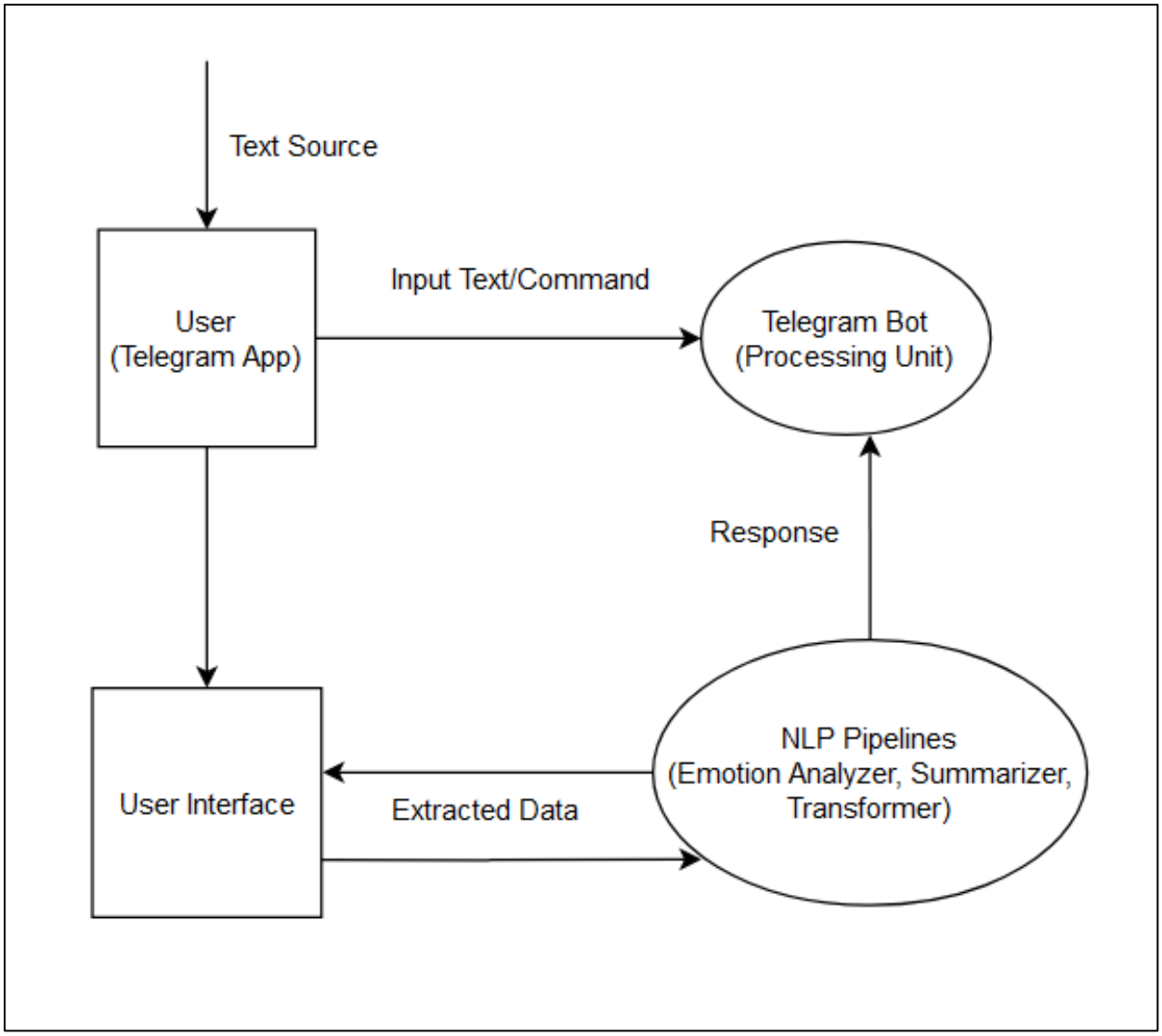

Рисунок 3.4 – Діаграма потоків даних 0 рівня

Користувач надсилає вхідні дані до Telegram-бота, який обробляє ці дані за допомогою модулів обробки мови шляхом виконання відповідних команд, та повертає результати, що відображаються за допомогою інтерфейсу Telegram.

DFD рівня 1 (рис. 3.5) забезпечує більш детальний огляд процесів бота.

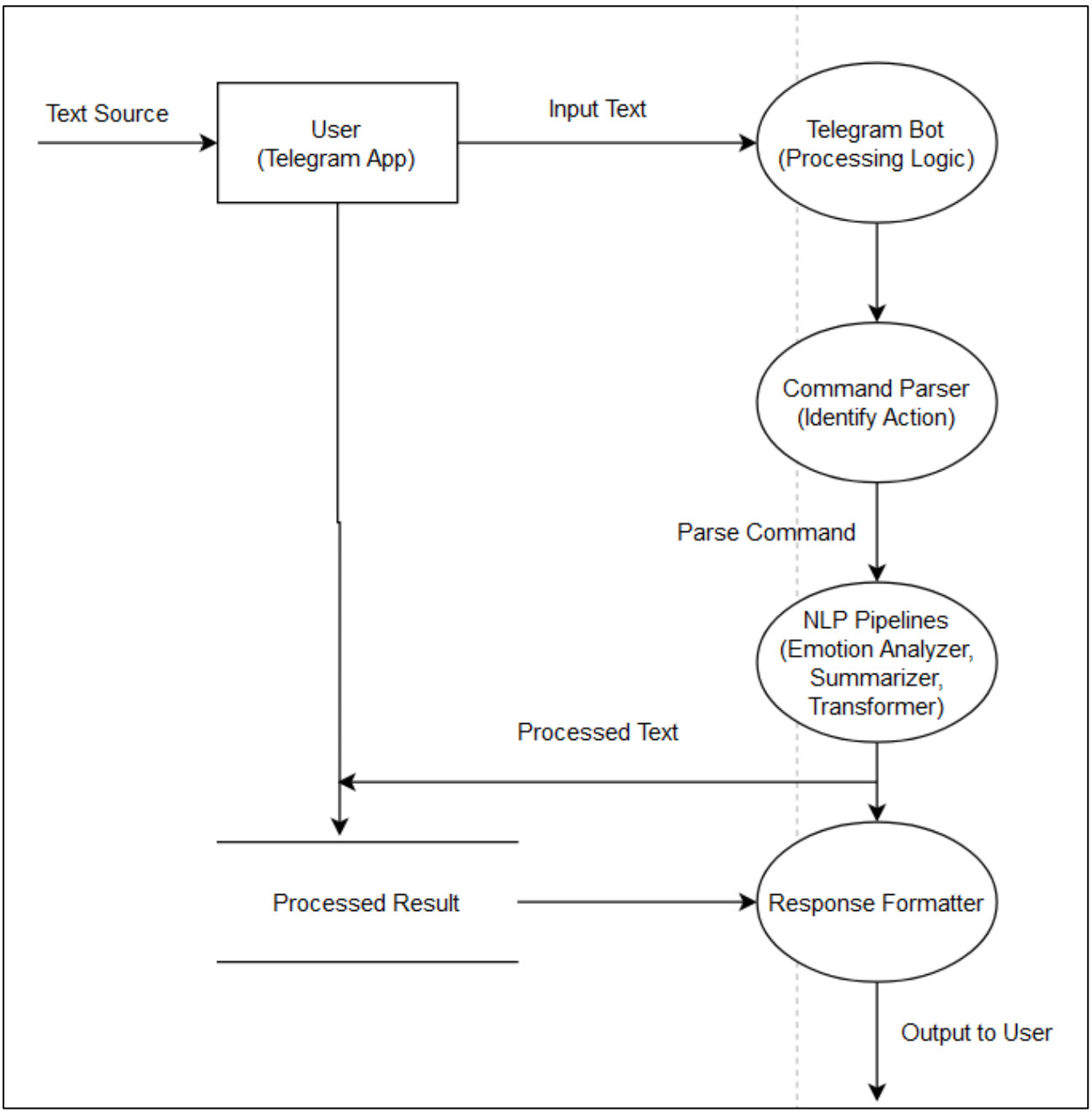

Рисунок 3.5 – Діаграма потоків даних 1 рівня

Вихідний текст та команда Користувача спрямовується на Обробник команд (Command Parser), який, відповідно, визначає цільову команду та спрямовує запит користувача до конкретних процесів, таких як емоційний аналіз, підсумовування або трансформація.

Результат роботи спрямовується на сервер Telegram, звідки передається до бота, який виконує форматування відповіді на запит Користувача. Після цього результат надсилається назад користувачеві. Вхідні та вихідні дані обробляються у структурованому потоці.

3.5 Вибір мови програмування та додаткових компонентів

Огляд мови програмування Python. Python — це мова програмування високого рівня, широко відома своєю простотою та читабельністю [12]. Python із часів свого заснування еволюціонував у одну з найпопулярніших мов програмування, яка використовується в різних сферах, таких як веб-розробка, аналіз даних, штучний інтелект, наукові обчислення тощо. Його філософія дизайну акцентує увагу на читабельності коду, що дозволяє розробникам висловлювати концепції меншою кількістю рядків коду порівняно з іншими мовами.

Основні характеристики Python [13]:

а) інтерпретована мова: Python виконується пострічково під час виконання програми. Ця особливість забезпечує швидке тестування та налагодження, оскільки помилки можуть бути виявлені негайно без необхідності компіляції;

б) динамічна типізація: у Python типи змінних визначаються під час виконання програми, а не на етапі компіляції. Ця гнучкість дозволяє швидше та легше адаптувати код до нових вимог;

в) мова високого рівня: Python абстрагує багато низькорівневих деталей, таких як управління пам’яттю та робота з апаратним забезпеченням. Це дозволяє розробникам зосередитися на вирішенні задач, а не на складних механізмах роботи програми;

г) читабельність і простота: синтаксис Python зрозумілий та інтуїтивний, нагадує англійську мову. Для визначення блоків коду використовуються відступи, а не фігурні дужки, що полегшує читабельність;

д) підтримка кількох парадигм: Python підтримує кілька підходів до програмування: процедурний, об'єктно-орієнтований і функціональний. Це дозволяє обирати найкращий підхід для вирішення конкретних задач.

е) розширена стандартна бібліотека: Python пропонує велику стандартну бібліотеку, яка включає модулі та функції для виконання різних завдань, таких як введення/виведення файлів, регулярні вирази, робота з веб-сервісами та маніпуляція даними;

ж) кросплатформенність: Python працює на різних платформах, таких як Windows, macOS, Linux та Raspberry Pi, що дозволяє виконувати код на різних операційних системах без модифікації;

з) спільнота та екосистема: Python має велику й активну спільноту, яка забезпечує багату екосистему сторонніх бібліотек і фреймворків (наприклад, Django для веб-розробки, NumPy для чисельних обчислень), що розширює можливості мови.

Застосування Python:

а) веб-розробка: фреймворки, такі як Django і Flask, дозволяють швидко створювати потужні веб-додатки.

б) аналіз даних і наука про дані: бібліотеки Pandas, NumPy і Matplotlib активно використовуються для обробки, аналізу та візуалізації даних.

в) машинне навчання та штучний інтелект: TensorFlow, PyTorch і Scikit-learn дозволяють легко створювати моделі машинного навчання.

г) автоматизація та скриптинг: Python часто використовується для автоматизації повторюваних завдань та взаємодії з іншими програмами.

д) розробка ігор: бібліотека Pygame допомагає швидко створювати ігри завдяки простому синтаксису Python.

е) наукові обчислення: бібліотеки SciPy та SymPy пропонують інструменти для наукових досліджень і математичних обчислень.

ж) інтернет речей (IoT): MicroPython дозволяє використовувати Python на мікроконтролерах для IoT-додатків;

з) кібербезпека: інструменти на основі Python часто використовуються для тестування безпеки та етичного хакінгу.

Бібліотека python-telegram-bot [14] є потужним та зручним інтерфейсом для розробки ботів, які взаємодіють із Telegram Bot API. Вона абстрагує складнощі роботи з API, дозволяючи розробникам зосередитися на створенні функціоналу, а не на керуванні запитами та відповідями API. Нижче представлено огляд бібліотеки, включно з її вирішальними особливостями [15].

Основні особливості:

а) підтримує Python версій 3.7 і новіше, а також забезпечує можливість синхронного та асинхронного програмування;

б) забезпечує доступ до всіх методів і функцій Telegram Bot API, що дозволяє розробникам використовувати всі можливості Telegram;

в) включає підмодуль `telegram.ext`, який спрощує розробку ботів за допомогою зручних абстракцій для виконання таких завдань, як обробка команд, обробка повідомлень і керування розмовами;

г) використовує `asyncio` для обробки одночасних взаємодій з користувачами, що покращує продуктивність і швидкість реагування;

д) пропонує детальну документацію, що включає приклади та навчальні матеріали для створення ботів з нуля.

Бібліотека Transformers від Hugging Face [16] є провідною відкритою платформою для задач обробки природної мови (NLP) та інших завдань машинного навчання. Вона надає зручний інтерфейс для доступу до великої кількості попередньо навчених моделей, що дозволяє розробникам та дослідникам реалізовувати найсучасніші AI-рішення з мінімальними зусиллями [17]. Нижче представлено огляд бібліотеки, включно з її вирішальними особливостями:

а) підтримка кількох фреймворків для глибокого навчання, зокрема PyTorch, і TensorFlow [18], що дозволяє користувачам обирати зручне середовище для тренування та розгортання моделей;

б) доступ до тисяч моделей, навчальних для різних завдань, таких як класифікація тексту, пошук відповідей, генерація тексту, стислий виклад, переклад тощо;

в) високорівневі API бібліотеки спрощують завантаження моделей і виконання завдань. Наприклад, функція pipeline дозволяє виконувати завдання лише кількома рядками коду;

г) Hugging Face надає повну документацію, що включає навчальні матеріали, концептуальні пояснення та довідник API. Це робить бібліотеку зручною як для новачків, так і для досвідчених користувачів;

д) Hugging Face Model Hub містить понад 25 000 моделей, які можна легко знайти та відфільтрувати за типом завдання чи архітектурою. Користувачі також можуть завантажувати власні навчені моделі для спільного використання;

е) підтримується донавчання попередньо навчених моделей на власних наборах даних, що дозволяє адаптувати моделі для специфічних застосувань;

ж) інструменти для прискорення роботи моделей, такі як GPU-оптимізація та обробка пакетів, забезпечують швидку інференцію у виробничих середовищах;

з) бібліотека підтримується Hugging Face та спільнотою, що сприяє її постійному вдосконаленню та оновленню.

Практичні застосування:

а) обробка природної мови (NLP): завдання, такі як класифікація тексту (визначення спаму), розпізнавання іменованих сутностей (NER), системи запитань-відповідей, чат-боти тощо;

б) генерація тексту: використання моделей, таких як GPT-2 або T5, для генерації зв’язного тексту за підказками;

в) стислий виклад: стискання довгих статей чи документів у короткі резюме за допомогою моделей для абстрактного узагальнення;

г) переклад: переклад тексту між мовами за допомогою багатомовних моделей.

Natural Language Toolkit (NLTK) — це потужний набір бібліотек і програм для роботи з текстовими даними у Python. Він широко відомий як один із найефективніших інструментів для задач обробки природної мови, що робить його незамінним ресурсом для дослідників, викладачів і розробників. Нижче представлено огляд бібліотеки, включно з її вирішальними особливостями [19]:

а) широкий функціонал: NLTK містить понад 50 корпусів і лексичних ресурсів, таких як WordNet, які надають багаті дані для різних NLP-завдань. Підтримуються такі функції, як токенізація, парсинг, класифікація, стемінг, тегування та семантичний аналіз;

б) модульна архітектура: інструментарій організований у незалежні модулі, кожен з яких відповідає за певні завдання чи структури даних. Це дозволяє користувачам вибирати тільки ті функції, які їм потрібні. Основні модулі включають токенізацію, парсинг, розподіли ймовірностей тощо;

в) освітні ресурси: NLTK постачається з розгорнутою документацією, навчальними матеріалами та прикладними наборами даних;

г) інтеграція з машинним навчанням: NLTK можна інтегрувати з бібліотеками машинного навчання, такими як Scikit-learn, для виконання завдань класифікації тексту та аналізу настроїв. Він надає інструменти для вилучення ознак та попередньої обробки, що є важливими для навчання моделей;

д) підтримка кількох мов: хоча NLTK переважно зосереджений на англійській мові, він також підтримує інші мови, включаючи іспанську, французьку, німецьку, китайську, арабську тощо.

Основні компоненти:

а) токенізація [20]: розбиття тексту на менші одиниці, які називаються токенами (слова чи речення). NLTK надає функції, такі як word_tokenize та sent_tokenize.

б) тегування частин мови: призначення частин мови (іменники, дієслова, прикметники) кожному токену у реченні за допомогою функції pos_tag.

в) розпізнавання іменованих сутностей (NER): ідентифікація іменованих сутностей у тексті (людей, організацій, місць) за допомогою функції ne_chunk.

г) парсинг: аналіз граматичної структури речень із використанням парсерів, які можуть створювати дерева розбору.

д) стемінг та лематизація [21]: стемінг (stemming) зводить слова до їх кореневих форм (наприклад, "running" до "run"). Лематизація (lemmatization) враховує контекст і перетворює слова на їх базові форми.

е) доступ до корпусів: легкий доступ до різних лінгвістичних корпусів для аналізу чи навчання моделей.

Застосування NLTK:

а) академічні дослідження: широко використовується в лінгвістичних дослідженнях для аналізу мовних структур і закономірностей.

б) аналіз настроїв: використовується компаніями для аналізу відгуків клієнтів чи дописів у соціальних мережах з метою визначення громадської думки щодо продуктів чи послуг.

в) чат-боти: розробники створюють чат-боти, здатні ефективно розуміти та реагувати на людську мову.

г) навчання мові: викладачі використовують NLTK для створення застосунків, які допомагають у вивченні мови чи лінгвістичних дослідженнях;

д) класифікація тексту: використовується для класифікації текстів за категоріями залежно від змісту (наприклад, виявлення спаму).

Висновки до розділу 3

В розділі 3 КМР проведено роботу із системного моделювання та графічного представлення результатів моделювання програмної реалізації системи емоційної трансформації.

В результаті було розроблено та описано основні діаграми, що розкривають суть процесів всередині програмної реалізації проєкту КМР. Для цього використано інструменти мови графічного опису UML.

Розглянуто та обґрунтовано вибір інструментів програмної реалізації системи емоційної трансформації на основі переваг, недоліків та особливостей інструментів згідно із описаними в розділі 2 КМР вимогами.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ЕМОЦІЙНОЇ ТРАНСФОРМАЦІЇ

4.1 Вибір мови програмування та додаткових компонентів

Вибір мови програмування

Мовою розробки програмного застосунку обрано мову програмування Python завдяки наступним перевагам:

а) простота та читабельність: синтаксис Python зрозумілий та інтуїтивний, схожий на природну мову, що полегшує написання, розуміння та підтримку коду. Така читабельність зменшує час, витрачений на розшифровку коду, сприяючи більш ефективним процесам розробки;

б) велика спільнота та підтримка: Python має величезну активну спільноту, яка надає безліч ресурсів, навчальних матеріалів та форумів. Це полегшує розробникам пошук допомоги та обмін знаннями. Така підтримка сприяє постійному навчанню та швидкому вирішенню проблем, що є важливим для ефективної розробки;

в) широкий набір бібліотек: Python пропонує багатий екосистему бібліотек, таких як python-telegram-bot, transformers та nltk, які спрощують реалізацію складних функцій, таких як взаємодія ботів чи обробка природної мови. Ці бібліотеки дозволяють розробникам зосередитися на основній логіці, а не на низькорівневих деталях;

г) гнучкість: універсальність Python дозволяє використовувати його в різних галузях — від веброзробки до аналізу даних та машинного навчання. Така адаптивність робить його придатним для створення широкого спектра застосунків на платформі Telegram;

д) швидкість розробки та прототипування: Python забезпечує швидкі цикли розробки, дозволяючи командам швидко перетворювати ідеї на робочі прототипи. Поєднання читабельності та багатої екосистеми підтримує швидкі ітерації та налаштування на основі відгуків користувачів.

Вибір додаткових бібліотек та їх переваги

python-telegram-bot: ця бібліотека є одним із найпопулярніших інструментів для розробки Telegram-ботів на Python. Вона надає зручний інтерфейс для роботи з Telegram Bot API.

Переваги:

а) синхронна модель програмування: Використовує просту синхронну модель програмування, яка зрозуміла розробникам, знайомим із традиційними підходами до програмування;

б) детальна документація: бібліотека постачається з розгорнутою документацією, яка допомагає як початківцям, так і досвідченим розробникам швидко реалізовувати різні функції;

в) підтримка спільноти: завдяки широкому використанню бібліотека має велику спільноту, яка активно сприяє її розвитку, забезпечуючи регулярні оновлення та вдосконалення.

Transformers: ця бібліотека від Hugging Face надає попередньо навчені моделі для задач обробки природної мови, таких як класифікація тексту (для аналізу емоцій) та узагальнення тексту.

Переваги:

а) передові моделі: містить доступ до сучасних NLP-моделей, які можуть бути інтегровані в бота для підвищення його можливостей (наприклад, аналізу емоцій у введенні користувача) ;

б) простота використання: бібліотека спрощує виконання складних NLP-задач, перетворюючи їх на зрозумілі функції, які легко інтегрувати в робочий процес бота.

NLTK (Natural Language Toolkit): це потужна бібліотека для роботи з текстовими даними природної мови.

Переваги:

а) можливості обробки тексту: надає інструменти для токенізації, парсингу, класифікації, стемінгу, тегування та семантичного аналізу, що є важливим для реалізації таких функцій, як узагальнення тексту та аналіз емоцій;

б) обширна бібліотека ресурсів: NLTK постачається з великим набором корпусів і лексичних ресурсів, які можна використовувати для покращення розуміння ботом мовних нюансів.

4.2 Керівництво із локального запуску

Встановлення Python

Спершу, необхідно переконатися, що на локальному комп'ютері встановлено Python. Його можна завантажити його з офіційного сайту. Рекомендується використовувати Python версії 3.7 або вище для повної сумісності зі сторонніми бібліотеками. В даному проєкті використано версію 3.12.8 [22]

Встановіть Rust і Cargo

Деякі бібліотеки, особливо ті, що пов’язані з моделями машинного навчання, потребують Rust для компіляції. Одною з таких є Transformers. Встановити Rust та Cargo можна із офіційного сайту [23] за допомогою файлу для встановлення.

Після встановлення, можливо, потрібно буде додати Rust до змінної PATH локальної системи. Дотримуйтесь інструкцій, які з’являться в терміналі після інсталяції.

Cargo встановлюється разом із Rust. Ви можете перевірити встановлення, виконавши команду:

```
cargo --version
```

Створення віртуального середовища

Рекомендується створити віртуальне середовище для проєкту, щоб окремо керувати залежностями:

```
python -m venv botenv
```

Активація віртуального середовища у ОС Windows:

```
botenv\Scripts\activate
```

Встановіть необхідні бібліотеки

Після активації віртуального середовища встановіть потрібні бібліотеки за допомогою `pip`:

```
pip install python-telegram-bot transformers nltk
```

Налаштування середовища Telegram

Для створення Telegram-бота та отримання токена необхідно виконати наступні кроки: відкрити Telegram і знайти BotFather (рис. 4.1), почати чат із BotFather (рис. 4.2) і використати команду `/newbot` (рис. 4.3). Виконувати підказки для створення імені та імені користувача бота. Після створення BotFather надасть токен бота (рис. 4.4). Збережіть цей токен, оскільки він потрібен у коді. Дані кроки проілюстровані на рисунках, що наведені далі.

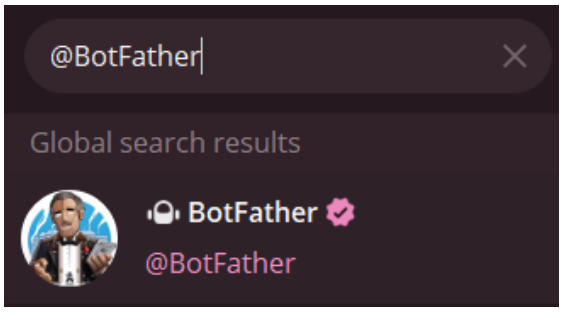

Рисунок 4.1 – Пошук BotFather

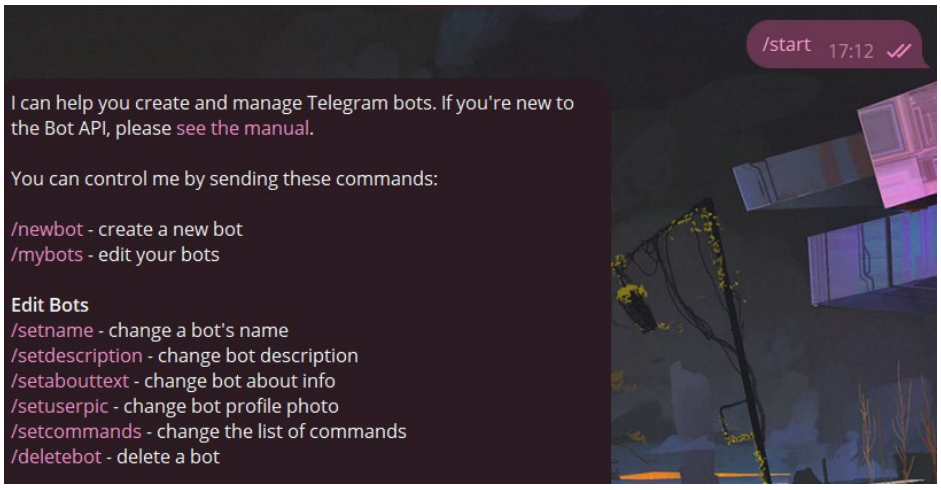

Рисунок 4.2 – Початок розмови із BotFather, список доступних команд

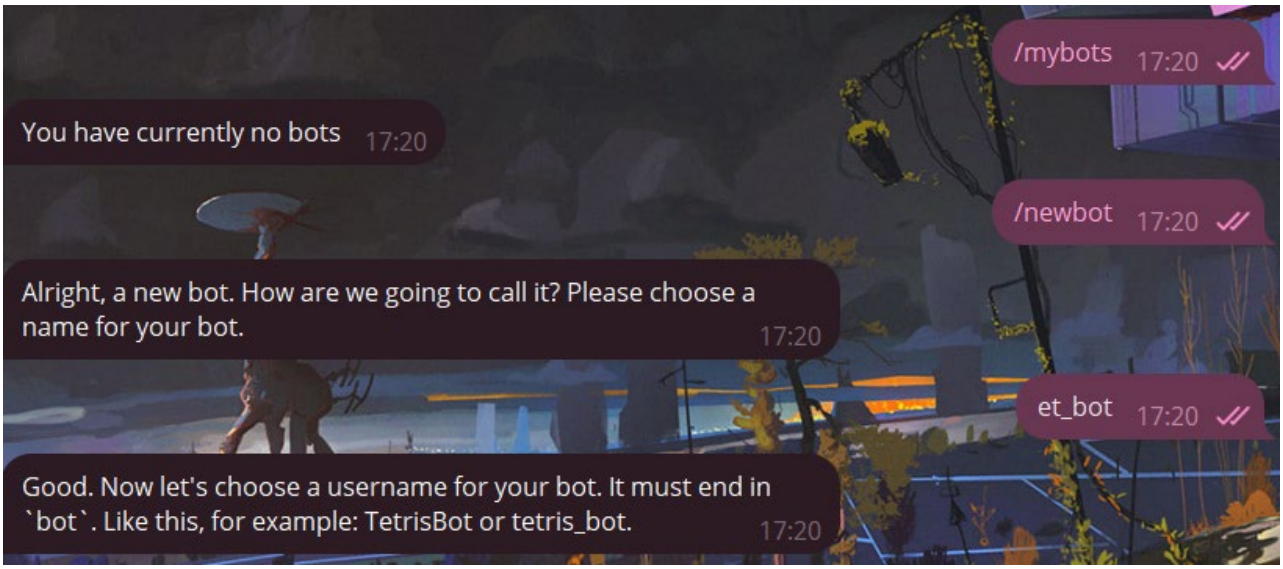

Рисунок 4.3 – перевірка наявності зареєстрованих ботів, створення нового бота та присвоєння боту ім’я користувача

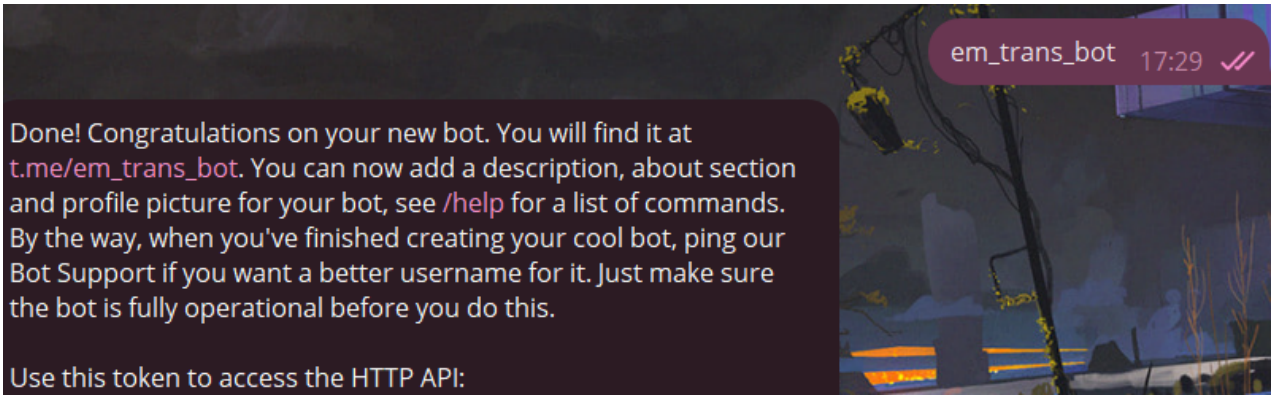

Рисунок 4.4 – присвоєння боту посилання та завершення створення бота

У вихідному коді тепер необхідно замінити заповнювач `"YOUR_TELEGRAM_BOT_TOKEN"` на реальний токен, отриманий від BotFather. Токен видається після завершення процесу створення бота, як вказано на рисунку 4.4. Варто зазначити, що реальний токен було приховано в цілях конфіденційності.

Додаткові залежності

Після завершення основних налаштувань можна встановити додаткову бібліотеку `torch` без помилок за допомогою `pip`:

```
pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu118
```

4.3 Опис програмних компонентів системи

Повний код наведено у додатку А. У даному розділі описано загальні відомості про програмний код, наведено пояснення логіки функціонування головних компонентів.

Бібліотеки та пакети:

а) `logging`: стандартна бібліотека, яка використовується для ведення журналу повідомлень з метою налагодження;

б) `telegram`: ця бібліотека використовується для взаємодії з Telegram Bot API. Вона містить класи, такі як `Update`, `InlineKeyboardMarkup` і `InlineKeyboardButton`;

в) `telegram.ext`: цей підмодуль надає класи `Application`, `CommandHandler`, `MessageHandler` і `ContextTypes` для управління командами бота та обробки повідомлень;

г) `transformers`: бібліотека від Hugging Face, яка надає попередньо навчені моделі для різних NLP-завдань. У цьому випадку вона використовується для емоційного аналізу (`text-classification`) та створення резюме тексту (`summarization`);

д) `nltk.tokenize`: бібліотека, що використовується для токенізації тексту на речення.

Типи даних:

а) `Update`: представляє вхідні оновлення з Telegram, які можуть містити повідомлення, зміни тощо;

б) `ContextTypes.DEFAULT_TYPE`: представляє контекст, у якому виконується команда чи обробник повідомлення, зокрема містить дані, як-от інформацію про користувача;

в) `str`: використовується для текстових повідомлень і введення користувачем;

г) `list`: використовується для зберігання кількох речень або результатів аналізу.

Класи

а) `Application`: управляє життєвим циклом застосунку бота та обробляє вхідні оновлення;

б) `CommandHandler`: обробляє команди, які надсилає користувач (наприклад, `/start, /help`) ;

в) `MessageHandler`: обробляє звичайні текстові повідомлення, які не є командами.

Методи:

а) `start(update: Update, context: ContextTypes.DEFAULT_TYPE)` – обробляє команду `/start`. Відправляє користувачу вітальне повідомлення, коли той починає взаємодію з ботом;

б) `help_command(update: Update, context: ContextTypes.DEFAULT_TYPE)` – обробляє команду `/help`. Надає користувачу список доступних команд;

в) `guide(update: Update, context: ContextTypes.DEFAULT_TYPE)` – надає користувачу інструкцію щодо використання функціоналу бота;

г) `analyze(update: Update, context: ContextTypes.DEFAULT_TYPE)` – здійснює емоційний аналіз тексту, надісланого користувачем. Використовує попередньо навчену модель для класифікації емоцій та повертає оцінки для кожної виявленої емоції в тексті;

д) `summarize(update: Update, context: ContextTypes.DEFAULT_TYPE)` – резюмує зміст тексту, наданого користувачем, за допомогою моделі для створення резюме. Повертає коротку версію вхідного тексту;

е) `transform(update: Update, context: ContextTypes.DEFAULT_TYPE)` – трансформує емоційний тон вхідного тексту відповідно до специфікацій користувача.

4.4 Керівництво користувача

Даний розділ призначений для ознайомлення із правилами взаємодії з ботом за допомогою його команд та описує його функціональні можливості. На

рисунках нижче наведено результати послідовної демонстрації функціоналу застосунку.

Активація бота

Спершу треба знайти бота в Telegram за його назвою або посиланням, яке надав розробник (рис. 4.5). Якщо запуск бота відбувається на локальному пристрої, то назву і посилання можна знайти у чаті із BotFather.

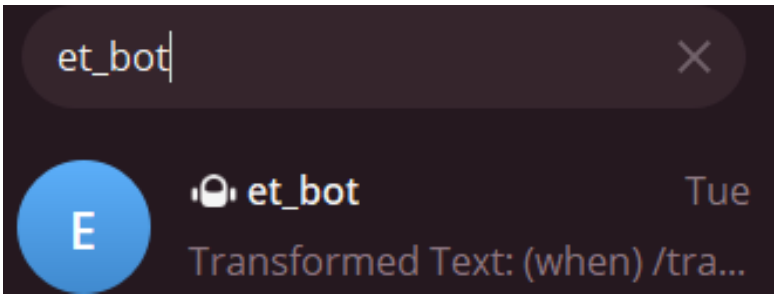

Рисунок 4.5 – виклик бота через пошуковий рядок Telegram

Далі необхідно почати розмову, ввівши команду /start або натиснувши кнопку Start. Після цього трбеа дотримуватись інструкцій, які надає бот у чаті.

Опис команд

`/start` ініціалізує бота (рис. 4.6) та надає привітальне повідомлення. Як використовувати: введіть `/start` у чаті. Бот привітає вас і надасть коротку інформацію про себе.

`/help` виводить список усіх доступних команд бота. Як використовувати: введіть `/help`, щоб отримати детальний опис можливостей бота.

`/guide` надає детальний посібник для використання бота. Як використовувати: введіть `/guide`, щоб отримати покрокову інструкцію щодо використання всіх функцій бота. Відповідь бота на виконання команди `/guide` наступна:

Як використовувати цього бота:

1. Надішліть текст для аналізу емоційного тону за допомогою `/analyze`.
2. Підсумуйте текстовий зміст за допомогою `/summarize`.

3. Змініть емоційний тон тексту за допомогою `/transform` і бажаної емоції.

Приклад: `/transform happy`.

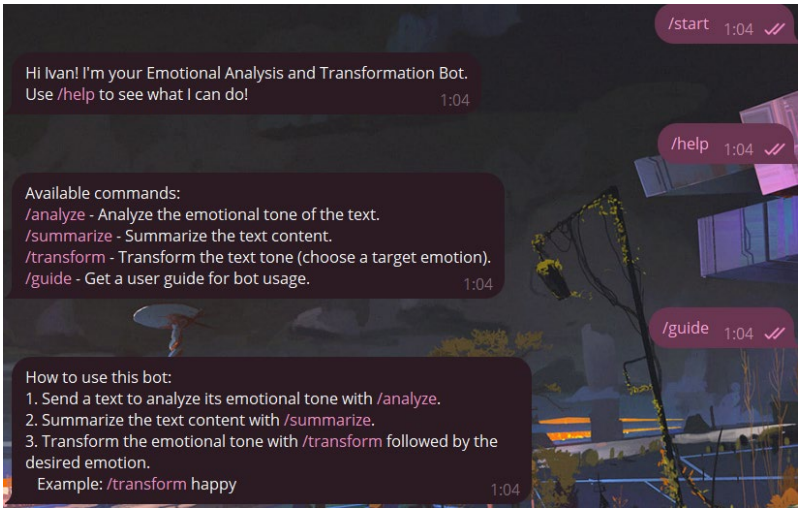

Рисунок 4.6 – Виклик команд `/start`, `/help` та `/guide`

`/analyze` аналізує емоційний тон наданого користувачем тексту. Як використовувати: введіть `/analyze`. Відправте повідомлення з текстом, який потрібно проаналізувати. Бот поверне емоційний тон із відсотковою впевненістю для кожної емоції. Далі наведено приклад використання цієї команди та результатів її роботи із довгим (рис. 4.7) та коротким (рис. 4.8) текстовими повідомленнями.

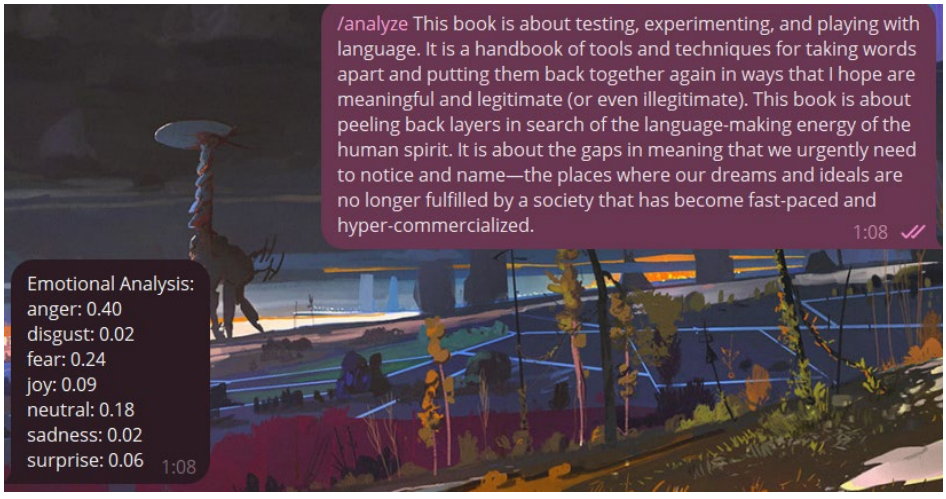

Рисунок 4.7 – Демонстрація емоційного аналізу тексту англійською мовою виконанням команди `/analyze`

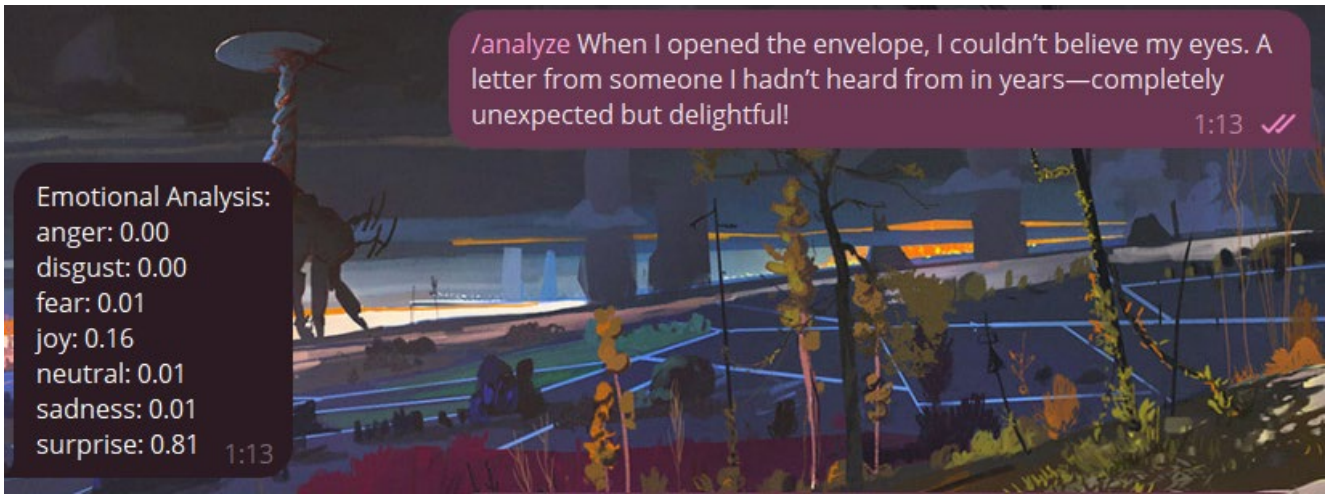

Рисунок 4.8 – Демонстрація аналізу короткого тексту при виконання команди `/analyze`

`/summarize` підсумовує текст, виділяючи основну ідею (рис. 4.9). Як використовувати: введіть `/summarize`; відправте повідомлення з текстом, який потрібно підсумувати. Бот поверне короткий підсумок вашого тексту.

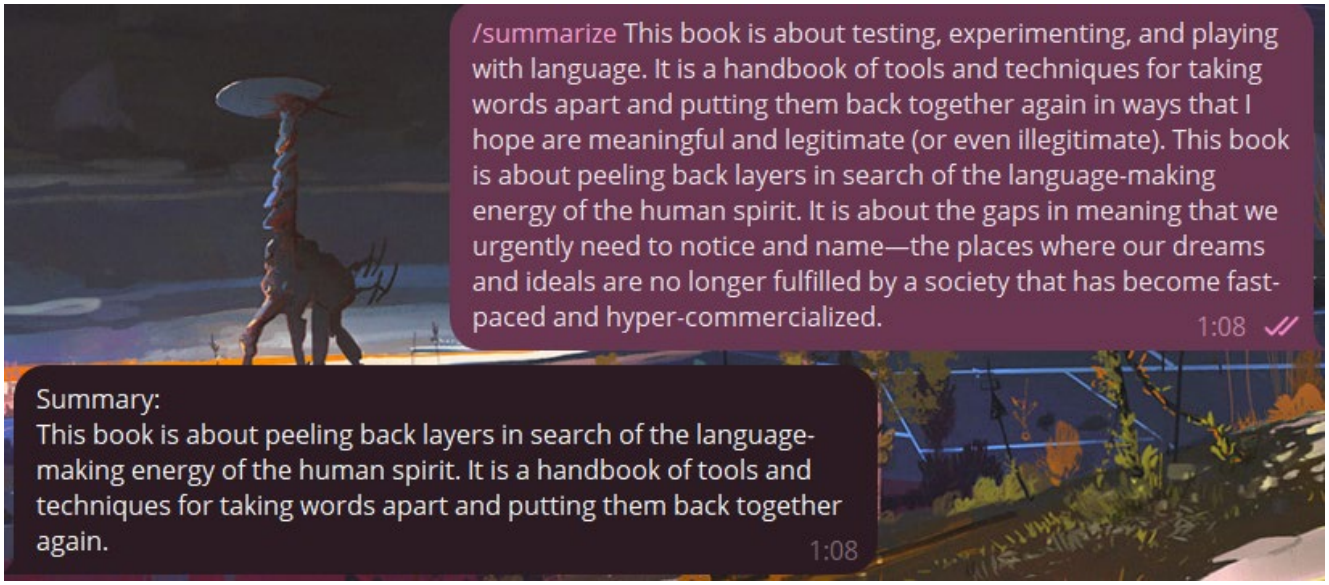

Рисунок 4.9 – Демонстрація підсумовування довгого тексту при виконання команди `/summarize`

`/transform <emotion>` змінює емоційний тон наданого тексту на вказаний. Як використовувати: введіть `/transform <emotion>` (наприклад, `/transform joy`) для вибору бажаного емоційного тону. Відправте повідомлення з текстом, який

потрібно трансформувати. Бот перепише текст із заданим емоційним тоном, зберігаючи оригінальний зміст.

Підтримувані емоції: happy, sad, angry, neutral, fearful, surprised, disgusted. Нижче наведено приклад використання цієї команди та результатів її роботи із довгим (рис. 4.10) та коротким (рис. 4.11) текстовими повідомленнями.

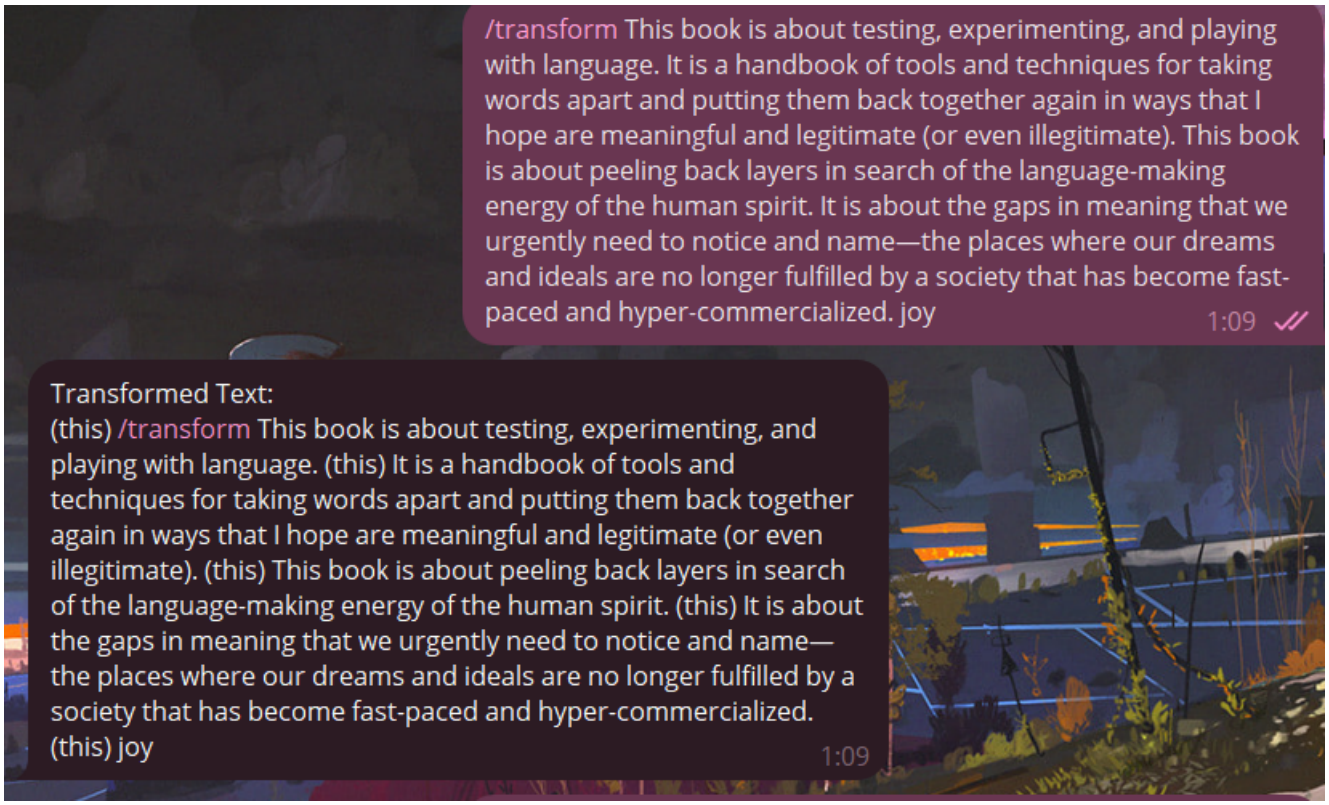

Рисунок 4.10 – Демонстрація трансформації довгого тексту на основі користувацького запиту та базового тексту командою `/transform`

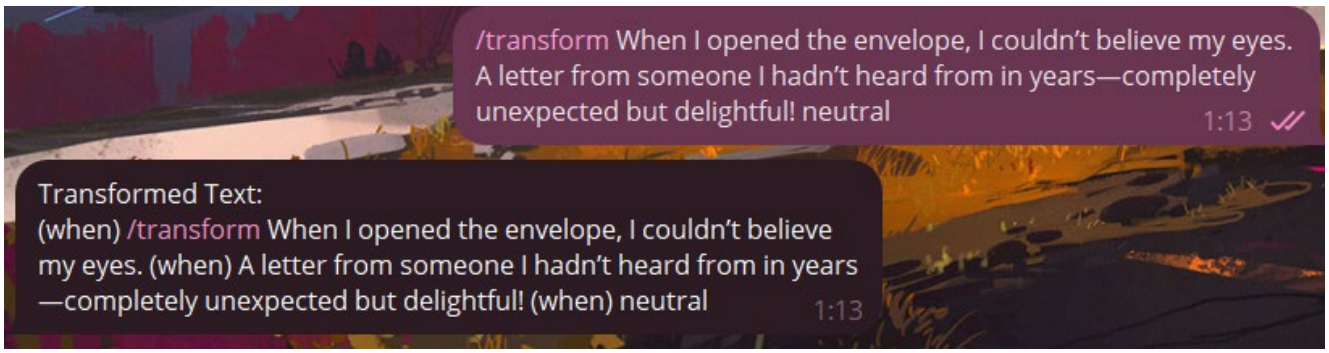

Рисунок 4.11 – Демонстрація трансформації короткого тексту на основі користувацького запиту та базового тексту командою `/transform`

4.5 Тестування програмного застосунку

Процес тестування бота для емоційного аналізу та трансформації спрямований на забезпечення коректної функціональності, стабільного користувацького досвіду та правильної обробки граничних випадків. Основним типом тестування було функціональне тестування – виконання всіх команд бота з різними вхідними параметрами для оцінки їх продуктивності.

Цілі тестування:

а) перевірити точність емоційного аналізу;

б) забезпечити надійність результатів підсумовування тексту;

в) перевірити коректність трансформації емоційного тону відповідно до зазначеного користувачем;

г) переконатися у стійкості бота до некоректних або граничних вхідних даних;

д) оцінити чіткість та зручність відповідей бота;

е) перевірити роботу меню та команд допомоги бота.

Етапи тестування

Тестування активації команд (табл. 4.1). Кожна команда була виконана з різними вхідними текстами для перевірки її функціональності. Перевірені команди: `/start`, `/help`, `/analyze`, `/summarize`, `/transform <emotion>`, `/guide`.

Таблиця 4.1 – Сценарії тестування активації команд

Команда	Тестове введення	Очікуваний результат	Результат
`/start`	None	Вітальне повідомлення зі списком доступних команд	Пройдено
`/help`	None	Список команд із їх описами	Пройдено
`/analyze`	«I feel great today!»	Виявлення емоції «щастя» із зазначенням впевненості	Пройдено

Кінець таблиці 4.1

`/analyze`	«I am so annoyed with this problem!»	Виявлення емоції «злість» із зазначенням впевненості	Пройдено
`/summarize`	Довгий абзац із кількома ідеями	Короткий підсумок, що охоплює основну ідею	Пройдено
`/transform happy`	«It’s been a tough day.»	Позитивна перефразована версія тексту	Пройдено
`/transform sad`	«I love spending time with friends.»	Негативна перефразована версія тексту	Пройдено
`/transform xyz`	«Text to transform.»	Повідомлення про помилку для непідтримуваної емоції	Пройдено
`/guide`	None	Детальна інструкція щодо використання бота	Пройдено

Тестування граничних випадків (табл. 4.2). Перевірка обробки ботом некоректних, неоднозначних або екстремальних варіантів введеного тексту.

Таблиця 4.2 – Сценарії тестування граничних випадків

Тип введення	**Приклад введення**	**Очікуваний результат**	**Результат**
Порожнє введення	«»	Повідомлення про помилку із запитом на введення тексту	Пройдено
Непідтримувана емоція	`/transform excited` «I’m so bored.»	Повідомлення про помилку: непідтримувана емоція	Пройдено

Кінець таблиці 4.2

Нетекстове введення	`/analyze` з емодзі	Повідомлення про помилку або нейтральна відповідь	Пройдено
Текст зі змішаними емоціями	«I’m happy but also a bit worried.»	Виявлення змішаних емоцій із відповідними оцінками	Пройдено
Текст зі сленгом або скороченнями	«OMG, today was lit!»	Коректна інтерпретація або нейтральна відповідь	Пройдено

Функціональність меню та команд допомоги (табл. 4.3). Перевірено зручність використання меню бота та команд допомоги.

Таблиця 4.3 – Сценарії тестування функціональності меню та команд допомоги

Тестовий сценарій	**Дія**	**Очікуваний результат**	**Результаті**
Відображення меню команд	Перевірити меню Telegram	Усі команди відображаються та доступні для кліку	Пройдено
Зрозумілість опису команд	Використати `/help`	Чіткий та зрозумілий заготовлений опис усіх команд	Пройдено
Функціональність меню	Натиснути `/analyze` у меню	Запит на введення тексту	Пройдено

Результати тестування:

а) точність: бот надавав надійний емоційний аналіз та точні результати підсумовування в більшості випадків;

б) зручність використання: відповіді були чіткими та зрозумілими, команди `/help` та `/guide` виявилися особливо корисними для ознайомлення нових користувачів;

в) обробка помилок: граничні випадки оброблялися коректно, а бот надавав відповідні повідомлення про помилки.

г) неоднозначність: тексти зі змішаними емоціями іноді призводили до менш точних результатів;

д) обробка емодзі та сленгу: надто стилістичні вирази класифікувалися некоректно;

е) обробка довгих текстів: варто покращити зрозумілість відповідей шляхом сповіщення користувачів про ліміти тексту.

У підсумку, бот успішно пройшов усі основні та крайові тестові випадки, продемонструвавши високий рівень функціональності та зручності для користувачів. Завдяки невеликим покращенням у обробці неоднозначностей у тексті та розумінні розмовних текстів, бот може забезпечити ще більш надійні та універсальні результати.

Висновки до розділу 4

В розділі 4 КМР проведено роботу із викладення програмної специфікації, кодування та підготовки користувацького середовища до запуску локальної версії програмного рішення КМР. Продемонстровано приклад виконання описаного функціоналу, наведено рисунки в якості підтвердження виконання поставлених вимог. Виконано та описано процес тестування ПЗ. У підсумку, було наглядно показано можливості програмного рішення.

ВИСНОВКИ

В ході виконання КМР досліджено та використано на практиці алгоритми обробки природньої мови для створення Telegram-боту. Створено рішення для аналізу та перетворення емоційного тону текстових повідомлень користувачів шляхом розробки програмного забезпечення розпізнавання їх емоцій.

Таким чином вдалося підвищити ясність, ефективність та задоволення від спілкування шляхом текстових повідомлень. Для досягнення поставленої мети виконано наступні завдання:

а) досліджено предметну область, проведено аналіз літератури за темою КМР;

б) розроблено специфікацію вимог до ПЗ;

в) проведено роботу із проєктування та моделювання ПЗ;

г) виконано виявлення емоційного тону шляхом впровадження NLP-моделі для точного виявлення та класифікації емоційного тону вхідного тексту;

д) виконано емоційну трансформацію шляхом створення алгоритму для генерації нової версії тексту зі зміненим емоційним тоном, зберігаючи оригінальний зміст;

е) розроблено користувацький інтерфейс;

ж) розроблено ПЗ бота, виконано тестування та розгортання бота в Telegram.

Практичне значення отриманих результатів полягає у створенні середовища, яке сприяє правильному тлумаченню емоційного тону тексту, що веде до кращого розуміння його емоційної та змістової складової. Це, в свою чергу, знижує ризик утворення конфліктів та сприяє утворенню емпатії під час текстової бесіди.

Розроблений застосунок можливо вдосконалити за рахунок використання більш точних алгоритмів обробки природньої мови та потужніших мовних моделей.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Mental health needs have multiplied. Support hasn't. Until now: вебсайт. URL: https://woebothealth.com/ (дата звернення: 25.11.2024).

2. Mental health, redefined: вебсайт. URL: https://www.wysa.com (дата звернення: 25.11.2024).

3. The AI companion who cares: вебсайт. URL: https://replika.ai/ (дата звернення25.11.2024).

4. Empathetic, safe, and clinically validated chatbot for mental healthcare : вебсайт. URL: https://www.youper.ai/ (дата звернення: 25.11.2024).

5. A Complete Overview of The Telegram App - DXB Apps: вебсайт. URL: https://dxbapps.com/blog/a-complete-overview-of-the-telegram-app (дата звернення: 25.11.2024).

6. Telegram chat bot development: вебсайт. URL: https://gerabot.com/en/telegram-chatbot (дата звернення: 25.11.2024).

7. How Telegram Chatbots are changing communication with customers: вебсайт. URL: https://www.michele.com.ua/en/how-telegram-chatbots-are-changing-communication-with-customers/ (дата звернення: 25.11.2024).

8. Telegram APIs: вебсайт. URL: https://core.telegram.org/ (дата звернення: 25.11.2024).

9. David B. Olawade. Enhancing mental health with Artificial Intelligence: Current trends and future prospects [Електронний ресурс] / David B. Olawade, Ojima Z. Wada, Aderonke Odetayo, Aanuoluwapo Clement David-Olawade, Fiyinfoluwa Asaolu, Judith Eberhardt // Journal of Medicine, Surgery, and Public Health. – 2024/ - Vol. 3. – Mode of access: https://www.sciencedirect.com/science/article/pii/S2949916X24000525 (date of access: 01.10.2024). — Title from screen.

10. Python Libraries for NLP: NLTK, spaCy, and Transformers: вебсайт. URL: https://www.analyticsinsight.net/python-2/python-libraries-for-nlp-nltk-spacy-and-transformers (дата звернення: 25.11.2024).

11. draw.io: вебсайт. URL: https://app.diagrams.net/ (дата звернення: 25.11.2024).

12. What is Python? Executive Summary: вебсайт. URL: https://www.python.org/doc/essays/blurb/ (дата звернення: 25.11.2024).

13. What is Python Language? Overview of the Python Language: вебсайт. URL: https://www.scholarhat.com/tutorial/python/overview-of-the-python-language (дата звернення: 25.11.2024).

14. python-telegram-bot 13.14: вебсайт. URL: https://pypi.org/project/python-telegram-bot/13.14/ (дата звернення: 25.11.2024).

15. python-telegram-bot v21.9: вебсайт. URL: https://docs.python-telegram-bot.org/en/v21.9/ (дата звернення: 25.11.2024).

16. Hugging Face Transformers: вебсайт. URL: https://docs.determined.ai/0.28.0/model-hub-library/transformers/_index.html (дата звернення: 25.11.2024).

17. Hugging Face Transformers: Leverage Open-Source AI in Python: вебсайт. URL: https://realpython.com/huggingface-transformers/ (дата звернення: 25.11.2024).

18. How to Use the Hugging Face Transformer Library: вебсайт. URL: https://www.freecodecamp.org/news/hugging-face-transformer-library-overview/ (дата звернення: 25.11.2024).

19. Documentation. Natural Language Toolkit: вебсайт. URL: https://www.nltk.org/ (дата звернення: 25.11.2024).

20. Natural Language Toolkit (NLTK): вебсайт. URL: https://deepgram.com/ai-glossary/natural-language-toolkit-nltk (дата звернення: 25.11.2024).

21. Natural Language Processing With Python's NLTK Package: вебсайт. URL: https://realpython.com/nltk-nlp-python/ (дата звернення: 25.11.2024).

22. Python 3.12.8: вебсайт. URL: https://www.python.org/downloads/release/python-3128/ (дата звернення: 25.11.2024).
23. Rustup: вебсайт. URL: https://rustup.rs/ (дата звернення: 25.11.2024).

ДОДАТОК А

Програмний код

```
# Import necessary libraries

import logging

from telegram import Update, InlineKeyboardMarkup, InlineKeyboardButton

from telegram.ext import Application, CommandHandler, MessageHandler, ContextTypes, filters

from transformers import pipeline

import nltk

from nltk.tokenize import sent_tokenize

# Initialize the emotional analysis and summarization pipelines

emotion_analyzer = pipeline(

    "text-classification", model="j-hartmann/emotion-english-distilroberta-base", return_all_scores=True)

summarizer = pipeline("summarization", model="facebook/bart-large-cnn")

# Logging for debugging

logging.basicConfig(

        format="%(asctime)s - %(name)s - %(levelname)s - %(message)s", level=logging.INFO)

# Define commands

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

    """Handle /start command."""

    user = update.effective_user

    await update.message.reply_text(

         f"Hi {user.first_name}! I'm your Emotional Analysis and Transformation Bot.\n"

        "Use /help to see what I can do!"
```

```
    )
async def help_command(update: Update, context: ContextTypes.DEFAULT_TYPE):
    """Handle /help command."""
    await update.message.reply_text(
        "Available commands:\n"
        "/analyze - Analyze the emotional tone of the text.\n"
        "/summarize - Summarize the text content.\n"
        "/transform - Transform the text tone (choose a target emotion).\n"
        "/guide - Get a user guide for bot usage."
    )
async def guide(update: Update, context: ContextTypes.DEFAULT_TYPE):
    """Provide a user guide for bot usage."""
    await update.message.reply_text(
        "How to use this bot:\n"
        "1. Send a text to analyze its emotional tone with /analyze.\n"
        "2. Summarize the text content with /summarize.\n"
         "3. Transform the emotional tone with /transform followed by the desired
emotion.\n"
        "   Example: /transform happy"
    )
# Define analysis functions
async def analyze(update: Update, context: ContextTypes.DEFAULT_TYPE):
    """Perform emotional analysis."""
    text = update.message.text
    analysis = emotion_analyzer(text)
    scores = "\n".join(
```

```
        [f"{label['label']}: {label['score']:.2f}" for label in analysis[0]])
    await update.message.reply_text(f"Emotional Analysis:\n{scores}")
async def summarize(update: Update, context: ContextTypes.DEFAULT_TYPE):
    """Summarize text content."""
    text = update.message.text
    summary = summarizer(text, max_length=60, min_length=25, do_sample=False)
    await update.message.reply_text(f"Summary:\n{summary[0]['summary_text']}")
async def transform(update: Update, context: ContextTypes.DEFAULT_TYPE):
    """Transform emotional tone of text."""
    if len(context.args) < 1:
            await update.message.reply_text("Please specify the target emotion. Example: /transform happy")
        return
    target_emotion = context.args[0].lower()
    text = update.message.text
    sentences = sent_tokenize(text)
    transformed_text = []
    for sentence in sentences:
        analysis = emotion_analyzer(sentence)
        dominant_emotion = max(analysis[0], key=lambda x: x['score'])['label']
        if dominant_emotion != target_emotion:
            # Transform the text (simplified transformation for demonstration)
            transformed_text.append(f"({target_emotion}) {sentence}")
        else:
            transformed_text.append(sentence)
           await update.message.reply_text("Transformed Text:\n" + "
".join(transformed_text))
```

```
# Main bot application setup

def main():

    """Start the bot."""

    token = "USER_TELEGRAM_TOKEN"  # Replace with your bot token

    app = Application.builder().token(token).build()

    # Register handlers

    app.add_handler(CommandHandler("start", start))

    app.add_handler(CommandHandler("help", help_command))

    app.add_handler(CommandHandler("guide", guide))

    app.add_handler(MessageHandler(filters.TEXT & ~filters.COMMAND, analyze))

    app.add_handler(CommandHandler("analyze", analyze))

    app.add_handler(CommandHandler("summarize", summarize))

    app.add_handler(CommandHandler("transform", transform))

    # Run the bot

    app.run_polling()

if __name__ == "__main__":

    main()
```