PROJECT REPORT

GROUP - 7

CSCE 5210 - Fundamentals of Artificial Intelligence

Title: ChatWell: AI Support for Mental Health

Group Members:

1. Meghna Reddy Cheedhu
2. Obulesh Policherla
3. Prathyusha Gangisetty
4. Ram Reddy Naidi
5. Ravali Salakala

Abstract:

In today's digital age, using chatbots with AI capabilities to handle mental health issues has grown in popularity. This project showcases ChatWell, a chatbot system made to respond to users in emotional distress with empathy. Our chatbot provides helpful conversation based on user inputs by utilizing Natural Language Processing (NLP) methods and pre-trained language models. With the goal of helping users manage their emotions, the system is constructed with Python, Flask, and machine learning technologies. This study describes the chatbot's architecture, deployment, outcomes, and potential future improvements.

Introduction:

The topic of mental health is crucial, especially in a society where access to conventional therapy may be restricted. By offering an AI-powered chatbot that can identify emotional distress, have meaningful discussions with users, and provide the right kind of help, ChatWell seeks to close this gap. The chatbot can identify emotions like happiness, sadness, and anxiety and reply with

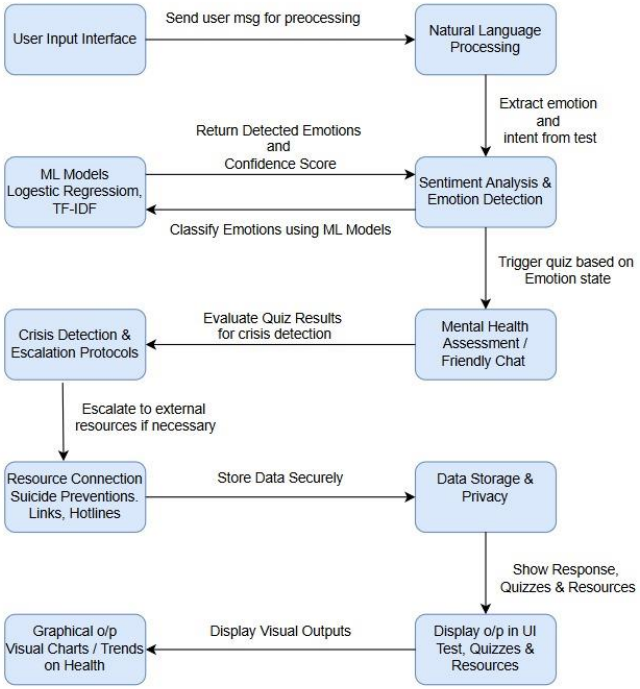
sympathetic remarks. It can also elevate important discussions to pertinent mental health resources, like hotlines for suicide prevention. The chatbot classifies user inputs and responds appropriately using machine learning and natural language processing.

- **Domain**: AI-powered mental health support
- **AI Methods**: NLP, sentiment analysis, intent classification, and deep learning

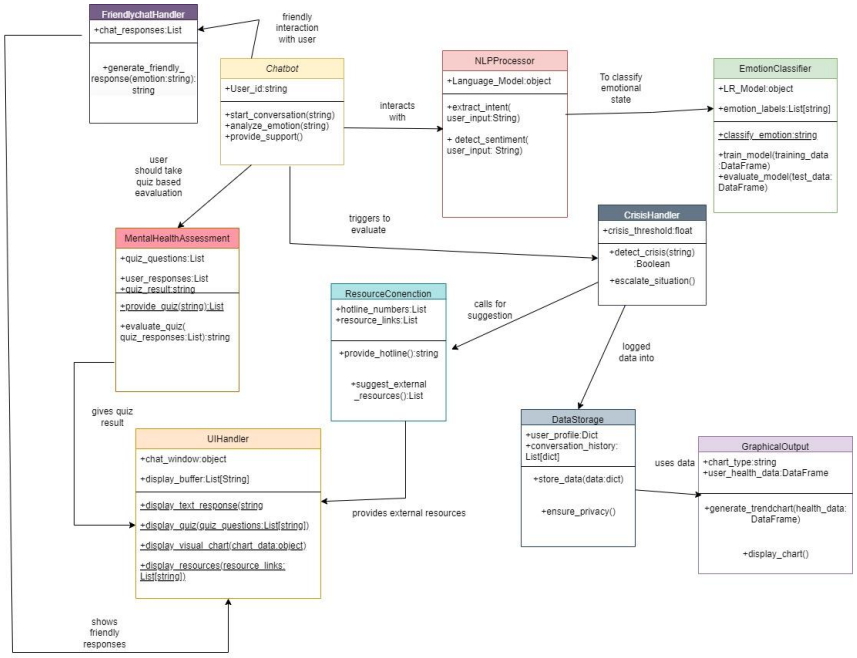
Problem Statement:

The primary goal of this project is to create an AI-driven chatbot designed to interact with individuals facing mental health challenges, providing immediate and empathetic support. Leveraging machine learning algorithms to analyze user emotions and respond appropriately, this initiative falls within the scope of AI applications in healthcare and wellness.

System Diagram:



Class Diagram:



Area of Application, Dataset, and Features:

Application:

The chatbot is designed for mental health support applications, such as online therapy platforms, healthcare systems, and personal mental wellness tools.

Input Data and Dataset:

The chatbot uses text input from users to identify emotions and respond accordingly. Initially, no specific dataset was used for training; instead, intents were defined manually based on common phrases related to mental health. The project uses the intents.json file, which includes various user expressions and corresponding responses.

Data Preprocessing:

- ✓ Tokenization: Tokenizes user input using the nltk library.
- ✓ Lemmatization: Converts words to their root forms to reduce dimensionality.
- ✓ Language Detection: Uses the spacy and spacy_langdetect libraries to detect the language of user input.
- ✓ Feature Extraction: The system uses bag-of-words (BoW) and TF-IDF methods for feature extraction.

Methods:

The project used a combination of traditional NLP techniques and modern deep learning models to achieve its goals:

1. **Natural Language Processing**:
    - ➢ Tokenization and Lemmatization using NLTK.
    - ➢ Language detection using SpaCy with the spacy_langdetect extension.
    - ➢ Translation pipelines using Hugging Face Transformers for multilingual support (English).
2. **Machine Learning**:
    - ➢ **Model**: A neural network (Keras-based) trained on preprocessed intent data to classify user inputs.
    - ➢ **Emotion Detection**: Logistic Regression and TF-IDF were used to classify emotions based on user inputs.
    - ➢ **Crisis Detection**: The system uses sentiment analysis to detect high-risk conversations and escalate to external resources.
3. **Technologies**:
    - ➢ Python (NLP processing and model training)
    - ➢ Flask (for API)
    - ➢ Keras/TensorFlow (deep learning models)
    - ➢ Hugging Face Transformers (for translation)

System Architecture:

The system architecture consists of:

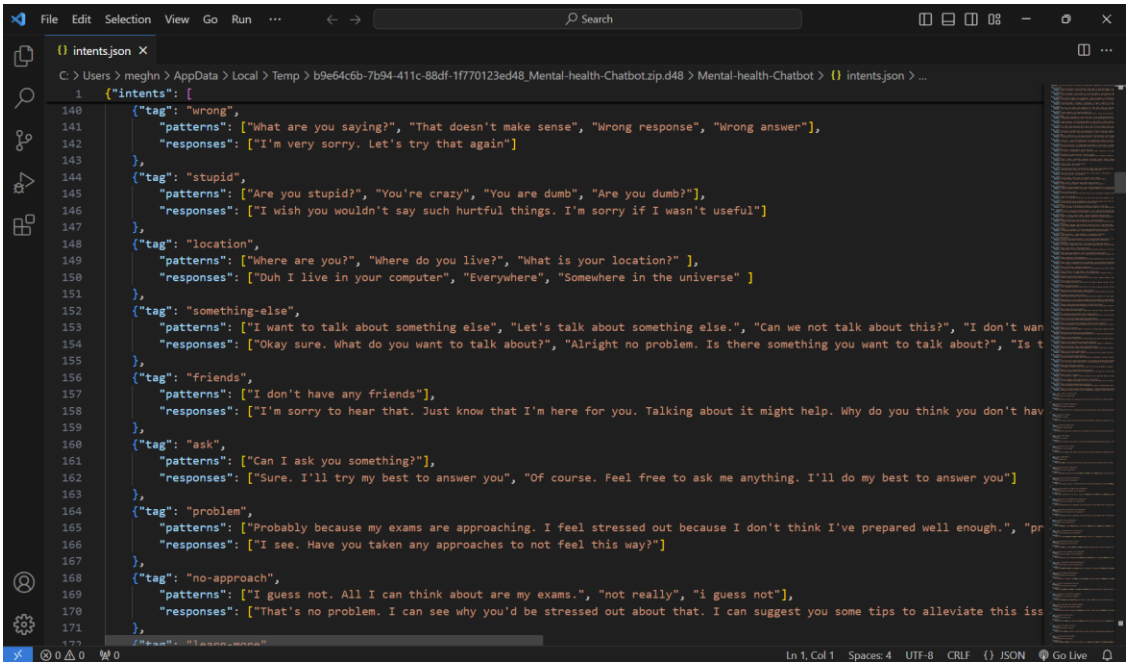
- A **frontend** built using HTML, CSS, and JavaScript.
- A **backend** powered by Flask, which handles incoming requests and processes them using the trained model.
- **Model Training**: The chatbot model is trained using predefined intents, including various user inputs and corresponding responses.

Model:

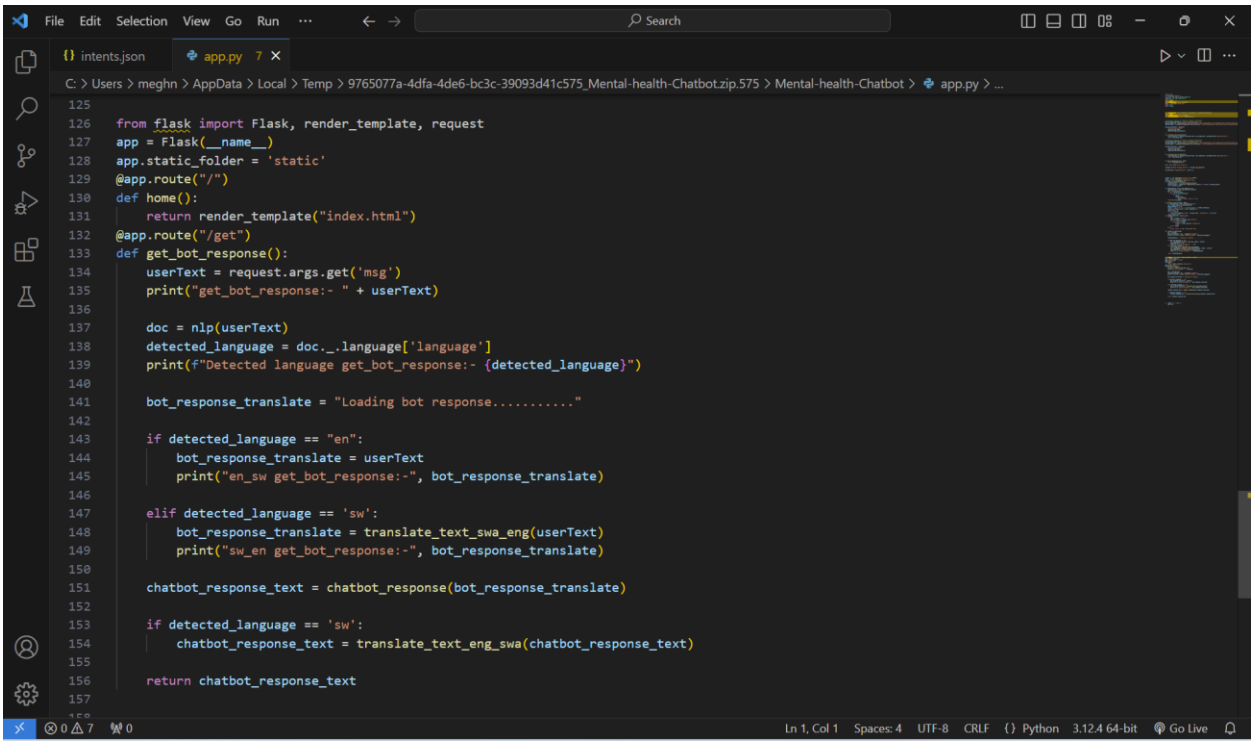
- The model uses a Sequential Neural Network with Dense layers to classify user input into predefined categories (intents).
- Training data is manually labeled and stored in the intents.json file.
- **Bag-of-Words (BoW)** and **TF-IDF** methods are used for feature extraction.
- The model achieves an accuracy of around 75% on predefined intents.

Code Snippets:

Intents:

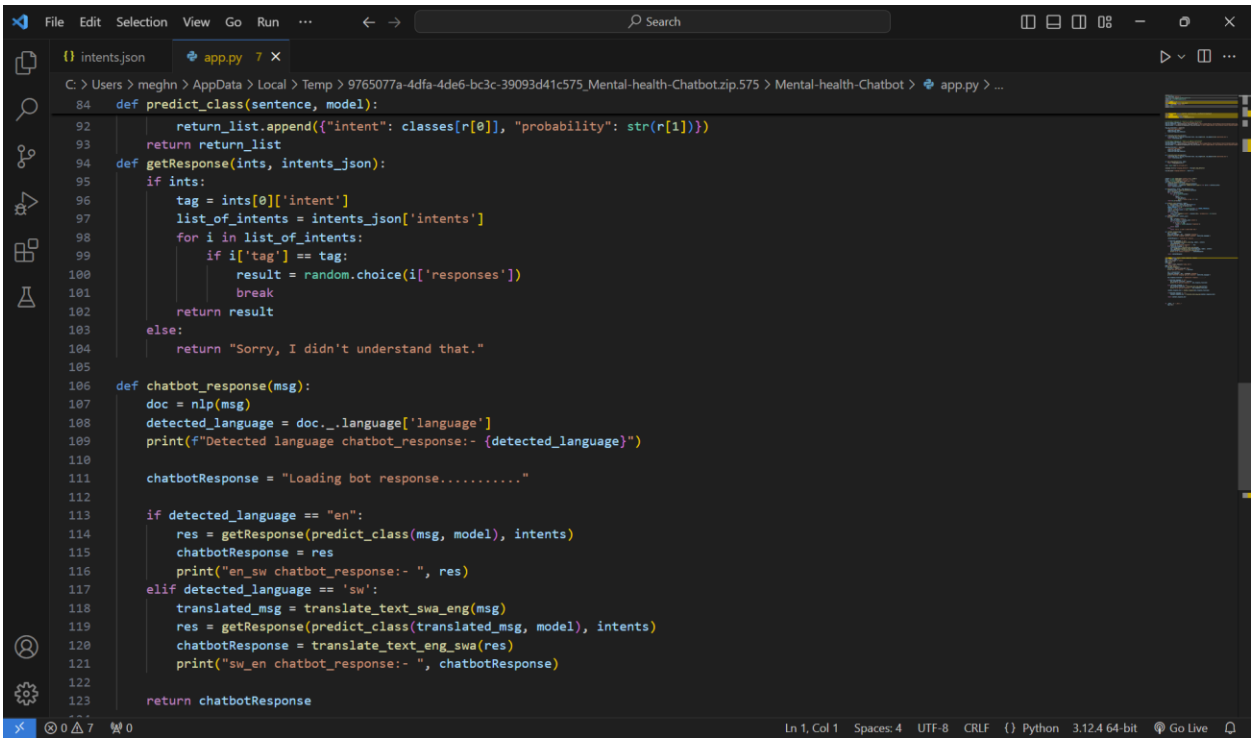


App.py:



```
from flask import Flask, render_template, request
app = Flask(__name__)
app.static_folder = 'static'
@app.route("/")
def home():
    return render_template("index.html")
@app.route("/get")
def get_bot_response():
    userText = request.args.get('msg')
    print("get_bot_response:- " + userText)

    doc = nlp(userText)
    detected_language = doc._.language['language']
    print(f"Detected language get_bot_response:- {detected_language}")

    bot_response_translate = "Loading bot response.........."

    if detected_language == "en":
        bot_response_translate = userText
        print("en_sw get_bot_response:-", bot_response_translate)

    elif detected_language == 'sw':
        bot_response_translate = translate_text_swa_eng(userText)
        print("sw_en get_bot_response:-", bot_response_translate)

    chatbot_response_text = chatbot_response(bot_response_translate)

    if detected_language == 'sw':
        chatbot_response_text = translate_text_eng_swa(chatbot_response_text)

    return chatbot_response_text
```

Bot understanding and response code:



```
def predict_class(sentence, model):
        return_list.append({"intent": classes[r[0]], "probability": str(r[1])})
    return return_list
def getResponse(ints, intents_json):
    if ints:
        tag = ints[0]['intent']
        list_of_intents = intents_json['intents']
        for i in list_of_intents:
            if i['tag'] == tag:
                result = random.choice(i['responses'])
                break
        return result
    else:
        return "Sorry, I didn't understand that."

def chatbot_response(msg):
    doc = nlp(msg)
    detected_language = doc._.language['language']
    print(f"Detected language chatbot_response:- {detected_language}")

    chatbotResponse = "Loading bot response.........."

    if detected_language == "en":
        res = getResponse(predict_class(msg, model), intents)
        chatbotResponse = res
        print("en_sw chatbot_response:- ", res)
    elif detected_language == 'sw':
        translated_msg = translate_text_swa_eng(msg)
        res = getResponse(predict_class(translated_msg, model), intents)
        chatbotResponse = translate_text_eng_swa(res)
        print("sw_en chatbot_response:- ", chatbotResponse)

    return chatbotResponse
```

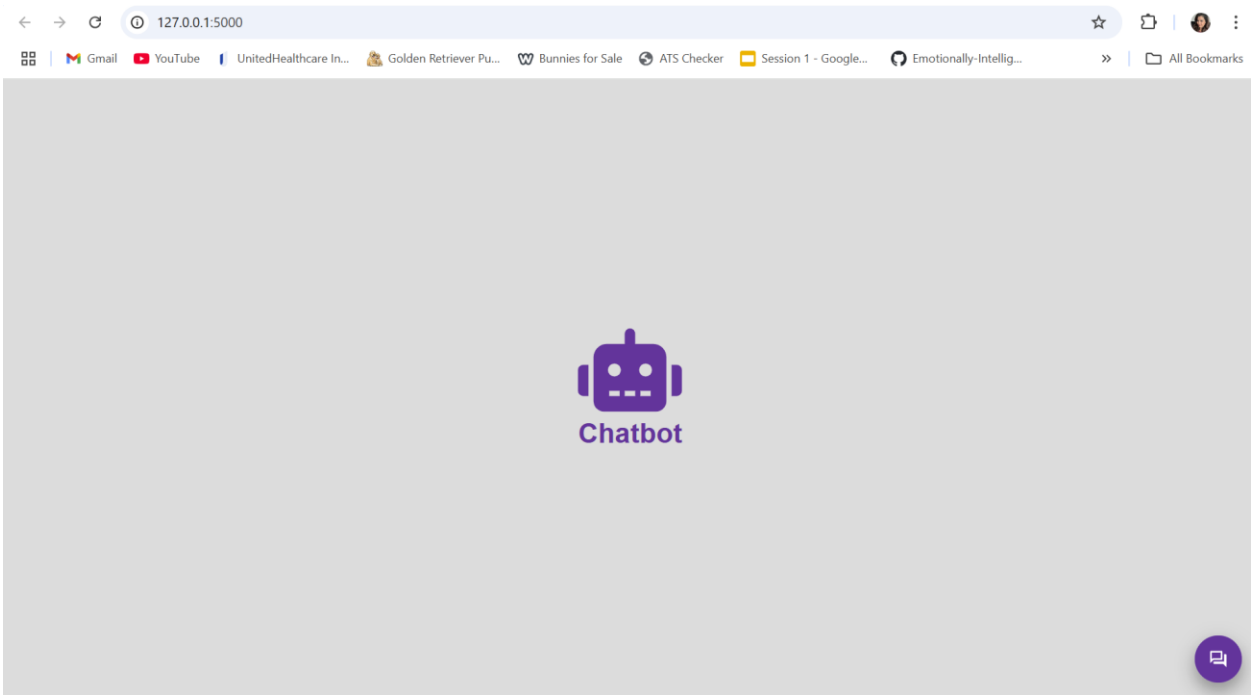
Experiments/Results/Discussion:

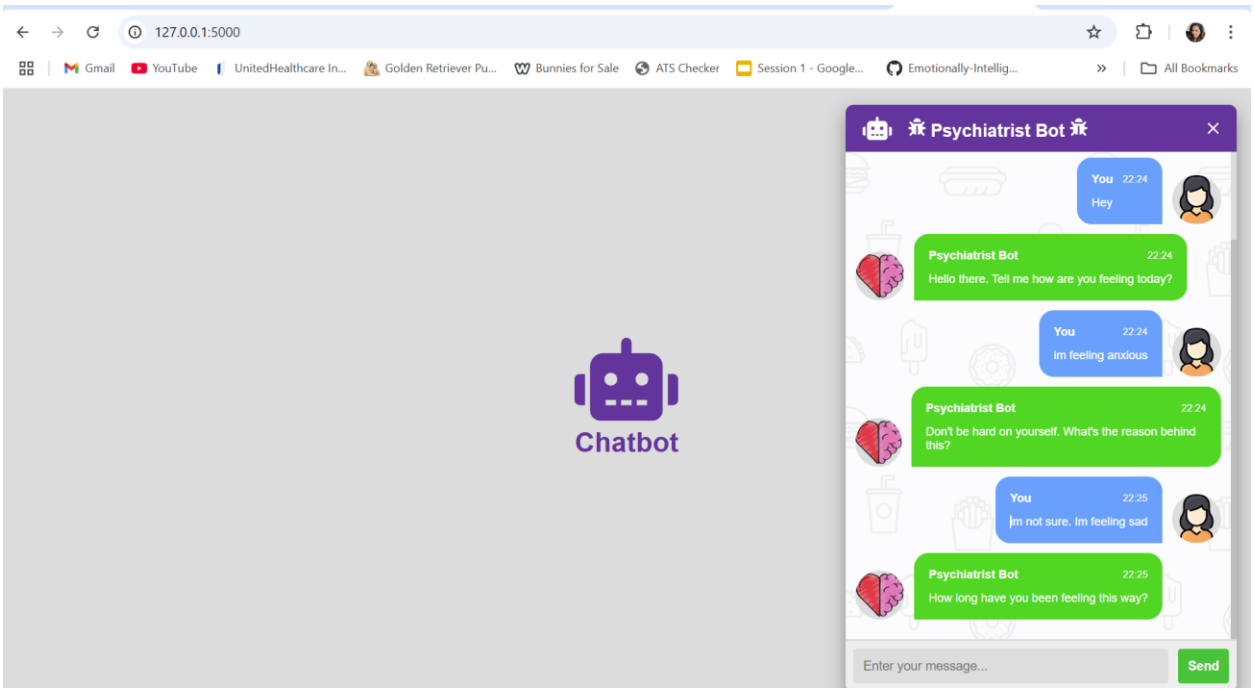
The chatbot was tested with a variety of user inputs to evaluate its performance in identifying emotional states and providing appropriate responses. Below are some key observations:

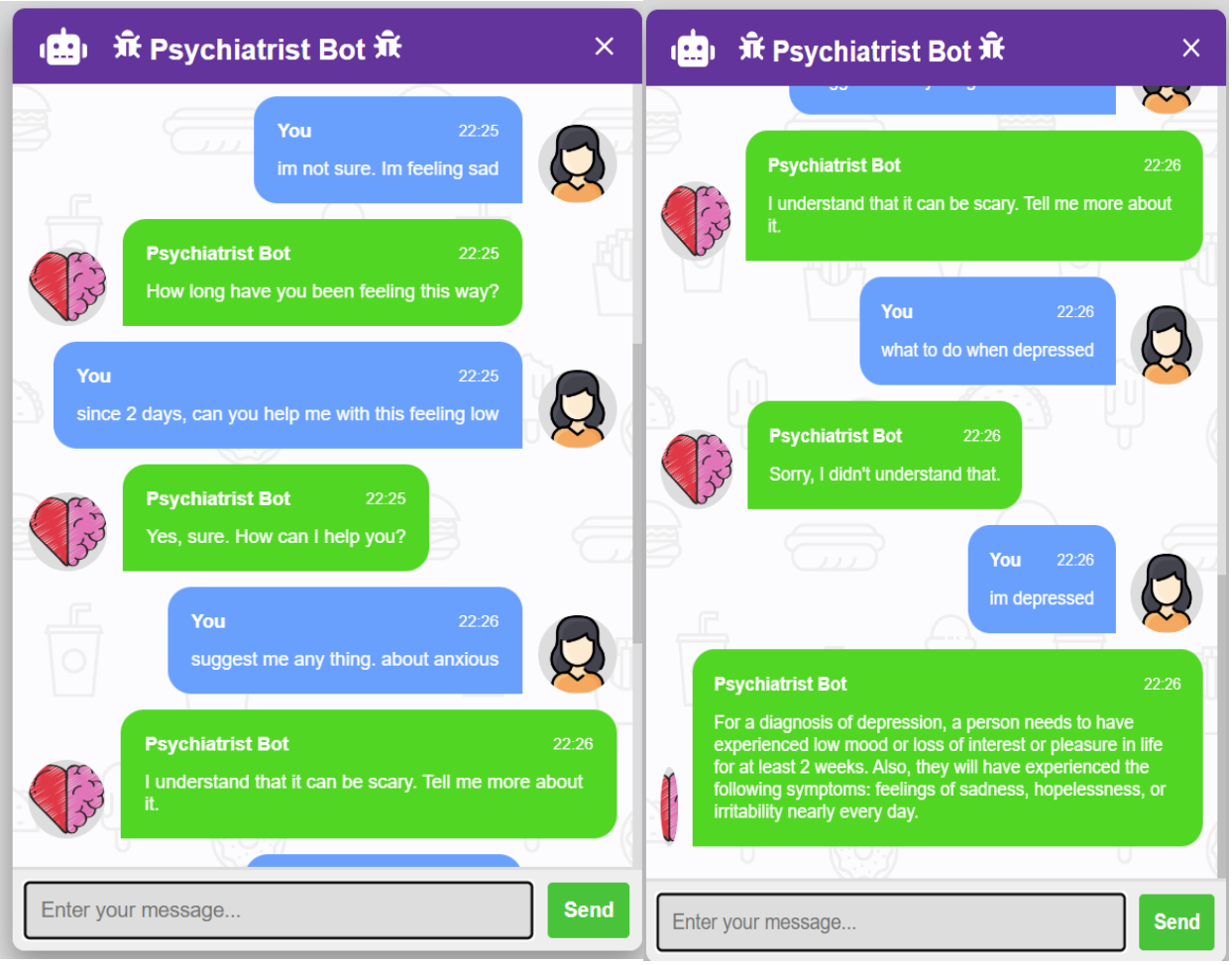
1. **Intent Classification**:
    - ✓ Accuracy of intent classification was measured by testing the neural network model with various user inputs. The model demonstrated a high success rate in identifying intents, particularly for greetings, emotional support, and mental health advice.
2. **Emotion Detection**:
    - ✓ The sentiment analysis module effectively classified user inputs into categories such as "anxiety," "stress," and "happiness." The confusion matrix showed a high precision rate for critical intents like "suicidal thoughts."
3. **Response Time**:
    - ✓ The chatbot was able to deliver near real-time responses, ensuring a smooth conversational flow.
4. **Crisis Management**:
    - ✓ The system successfully detected high-risk inputs and redirected users to relevant resources, such as hotlines and mental health services.

Screenshots and Visual Outputs:

The chatbot's UI provides a user-friendly experience, displaying empathetic responses and mental health resources. Screenshots of the working chatbot interface are included to demonstrate the flow of conversation and user interaction.


127.0.0.1:5000
Gmail
YouTube
UnitedHealthcare In...
Golden Retriever Pu...
Bunnies for Sale
ATS Checker
Session 1 - Google...
Emotionally-Intellig...
All Bookmarks
Chatbot


127.0.0.1:5000
Gmail
YouTube
UnitedHealthcare In...
Golden Retriever Pu...
Bunnies for Sale
ATS Checker
Session 1 - Google...
Emotionally-Intellig...
All Bookmarks
Chatbot
Psychiatrist Bot
You 22:24
Hey
Psychiatrist Bot 22:24
Hello there. Tell me how are you feeling today?
You 22:24
im feeling anxious
Psychiatrist Bot 22:24
Don't be hard on yourself. What's the reason behind this?
You 22:25
Im not sure. Im feeling sad
Psychiatrist Bot 22:25
How long have you been feeling this way?
Enter your message...
Send



Key Observations:

- The chatbot was effective in recognizing common mental health-related expressions such as "I'm feeling low" or "I'm depressed" and provided empathetic responses.
- The system struggled with understanding complex or less common phrases, highlighting areas for improvement.
- Future enhancements could include integrating a larger dataset for better understanding and adding support for multilingual conversations.

Conclusion/Future Work:

The Mental Health Chatbot uses real-time discussions to deliver emotional assistance in an efficient manner. NLP, sentiment analysis, and deep learning are used by the system to recognize

user emotions and react accordingly. While the chatbot performs well, several areas for future improvement have been identified:

- **Emotion Recognition and Personalized Support**: Enhance the emotion detection algorithm for more nuanced emotional states.
- **Language Support**: Expand beyond English and Swahili to support additional languages.
- **Integration with Healthcare Systems**: Develop APIs to integrate the chatbot with existing mental health platforms.
- **Voice Support**: Implement voice recognition for users who prefer speaking over typing.

References:

1. Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".
2. Rathor, A. S., Agarwal, A., & Dimri, P. (2018). Comparative study of machine learning approaches for Amazon reviews. *Procedia computer science*, *132*, 1552-1561.
3. Naik, T. K., & Singh, M. (2021). Automorphisms of odd Coxeter groups. *Monatshefte für Mathematik*, *195*(3), 501-521.
4. Io, H. N., & Lee, C. B. (2017, December). Chatbots and conversational agents: A bibliometric analysis. In *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (pp. 215-219). IEEE.