KAREN-THE CHATBOT

<u>INTRODUCTION</u>

Background and Context:

In recent years, advancements in artificial intelligence and natural language processing have revolutionized the way we interact with computers. Chatbots have gained popularity for their ability to simulate human-like conversations and provide assistance in various domains.

The goal of this project is to develop an AI chatbot named Karen that can engage in conversations with users in a natural and human-like manner. Karen leverages the power of the ChatGPT API, which is based on the GPT-3.5 architecture developed by OpenAI.

Problem Statement and objectives:

Traditional chatbot systems often require users to sign up or create accounts, adding unnecessary steps and barriers to accessing the service. This can be inconvenient and time-consuming for users who simply want to engage in a conversation or obtain information quickly. The need for sign-in procedures limits the accessibility and ease of use of traditional chatbots.

Moreover, traditional chatbots may rely on predefined conversational scripts or rules, leading to limited conversational capabilities and rigid interactions. Users may find it challenging to have natural and contextually relevant conversations with these chatbots, resulting in a less engaging and satisfactory user experience.

Significance and motivation:

The significance and motivation of this project lie in providing users with an interactive and personalized virtual assistant. By developing an AI chatbot like Karen that can engage in human-like conversations, open popular websites, and respond in a girl's voice, we aim to enhance user experience and bridge the gap between traditional chatbots and natural human interactions.

PROJECT OVERVIEW

Project goals and scope:

The project goals are to develop an AI chatbot named Karen that engages in natural conversations, opens popular websites, and responds in a girl's voice. The scope of the project includes integrating the ChatGPT API for generating contextually relevant responses, incorporating web browsing capabilities for accessing online content, and implementing voice synthesis technology.

Project timeline:

Inception - Development - Testing - Refinement - Completion. (50+ hours)

Project repository:

https://github.com/parvatkhattak/Karen

Team members and contributions:

Parvat (102201028)

1. Project planning
2. Importing modules
3. Creating and importing API key
4. Creating a function to return written response

Shubham Yadav (112201032)

1. Deciding necessary modules
2. Setting functions to get suitable voice
3. Given some necessary tasks to perform for some specific query
4. Testing and analyzing and debugging

Sai Teja (142201027)

1. Deciding project name
2. Defining functions to open websites
3. Debugging errors
4. Function to open calculator

Methodology

Approach and methodology employed:

The approach employed involves integrating the ChatGPT API for natural language processing and response generation. The methodology includes developing conversational logic, incorporating web browsing capabilities, implementing voice synthesis, and conducting iterative testing and refinement to ensure a seamless and engaging user experience.

Tools, technologies and framework used:

1. Python 3.10
2. ChatGPT API
3. Speech Recognition
4. Win32com.client
5. Web browser
6. OpenAI
7. Subprocess

Conclusion and future work

Summary of outcomes and contributions:

The project has resulted in the successful development of Karen, an AI chatbot capable of engaging in natural conversations, opening popular websites, and responding in a girl's voice. The integration of the ChatGPT API, along with web browsing capabilities and voice synthesis technology, has contributed to an immersive and interactive user experience. The project's outcome includes a comprehensive chatbot solution that bridges the gap between traditional chatbots and human-like interactions, enhancing the overall user satisfaction and accessibility.

Assessment of project success:

The success of the project is assessed based on the achievement of its objectives, including the development of a conversational AI chatbot, integration of the ChatGPT API, seamless web browsing capabilities, and implementation of a girl's voice. Additionally, user satisfaction, accuracy of responses, seamless integration, accessibility, and performance contribute to the assessment of success. Future work involves refining the chatbot's conversational abilities, enhancing web browsing capabilities, expanding the range of supported websites, and conducting further user studies to continually improve and optimize Karen's performance and user experience.

Recommendations for future improvements:

For future improvements, it is recommended to enhance Karen's natural language understanding for improved conversation quality. Expanding the knowledge base will ensure access to up-to-date information. Adding multimodal capabilities, such as image and video processing, can enhance interactivity. Personalization and user profiling can offer tailored responses and recommendations. Establishing a continuous evaluation and feedback process with users will provide valuable insights for iterative enhancements. By implementing these recommendations, Karen can evolve into a more sophisticated and user-centric AI chatbot.

Team members GitHub accounts:

Parvat:[https://github.com/parvatkhattak]

Shubham Yadav: [https://github.com/Shubham04567]

Sai Teja: [https://github.com/Saiteja1016M]

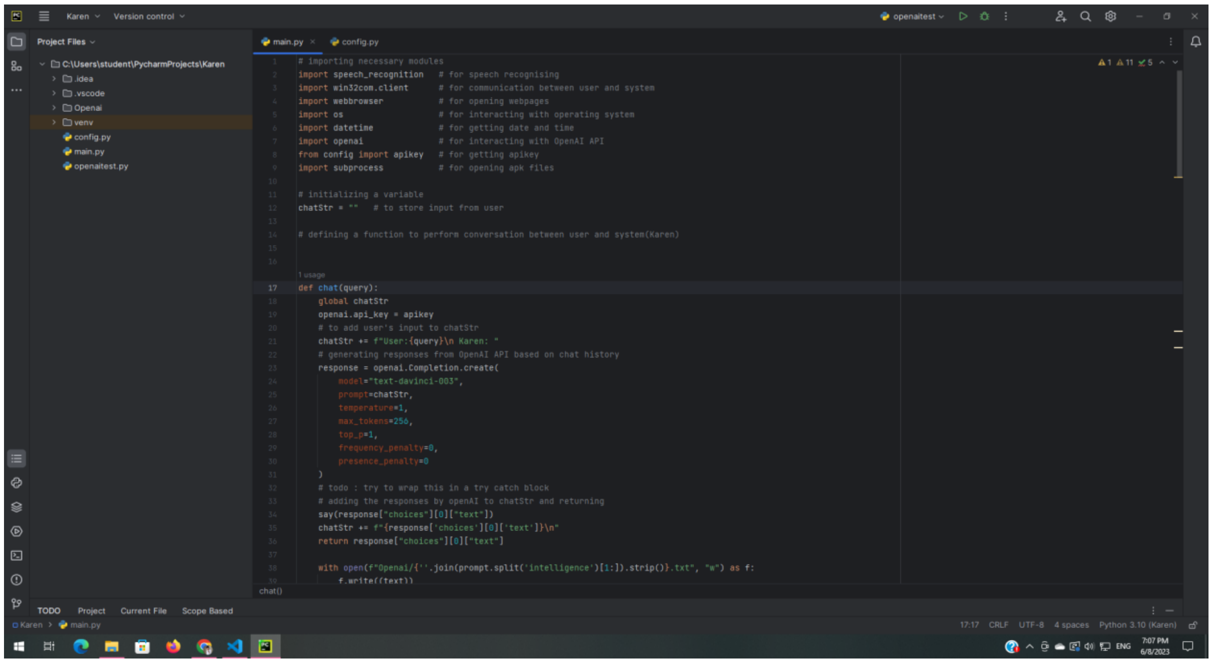
References

1. https://youtu.be/s_8b5iq4Rvk
2. https://youtu.be/DFmmiYlbgX0
3. https://youtu.be/EIYapGbNRwk
4. https://stackoverflow.com/search?q=jarvis+project&s=05be39d7-6b31-497c-bb9d-471b5fef425c

5. https://pythontutor.com/python-debugger.html#mode=edit

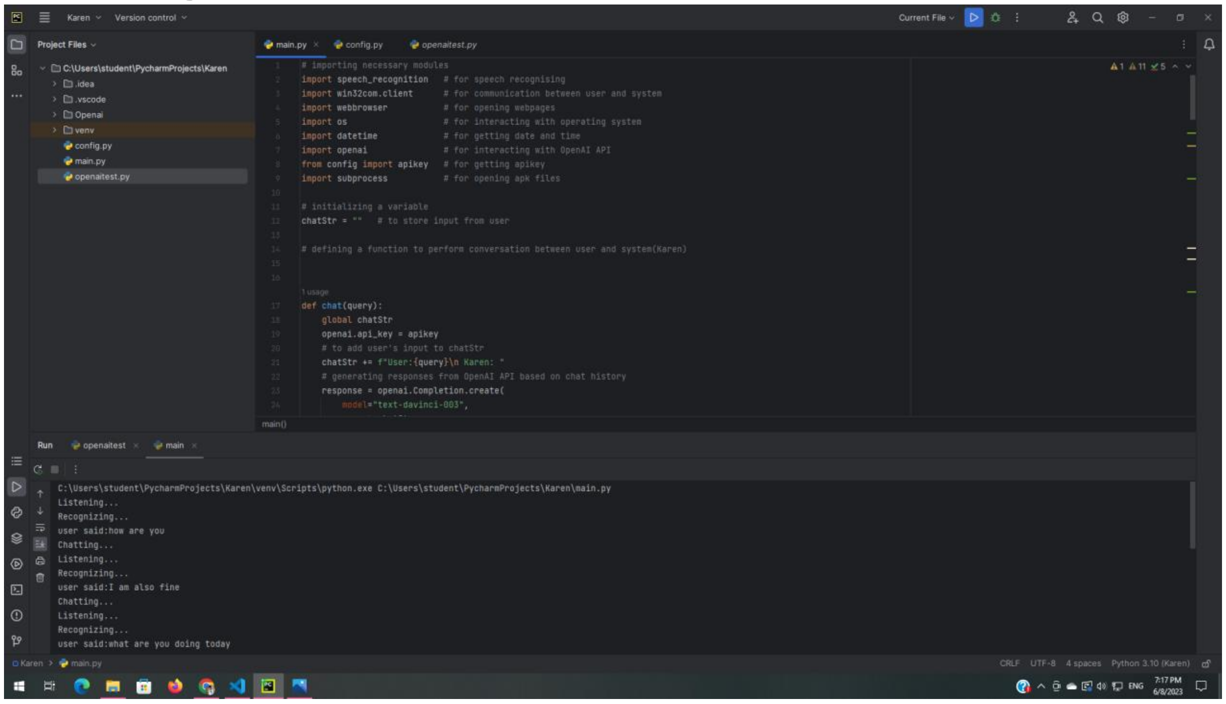
Appendices

Normal interface:



```python
# importing necessary modules
import speech_recognition   # for speech recognising
import win32com.client      # for communication between user and system
import webbrowser           # for opening webpages
import os                   # for interacting with operating system
import datetime             # for getting date and time
import openai               # for interacting with OpenAI API
from config import apikey   # for getting apikey
import subprocess           # for opening apk files

# initializing a variable
chatStr = ""   # to store input from user

# defining a function to perform conversation between user and system(Karen)


1 usage
def chat(query):
    global chatStr
    openai.api_key = apikey
    # to add user's input to chatStr
    chatStr += f"User:{query}\n Karen: "
    # generating responses from OpenAI API based on chat history
    response = openai.Completion.create(
        model="text-davinci-003",
        prompt=chatStr,
        temperature=1,
        max_tokens=256,
        top_p=1,
        frequency_penalty=0,
        presence_penalty=0
    )
    # todo : try to wrap this in a try catch block
    # adding the responses by openAI to chatStr and returning
    say(response["choices"][0]["text"])
    chatStr += f"{response['choices'][0]['text']}\n"
    return response["choices"][0]["text"]

    with open(f"Openai/{''.join(prompt.split('intelligence')[1:]).strip()}.txt", "w") as f:
        f.write((text))
```

Working interface:



```python
# importing necessary modules
import speech_recognition   # for speech recognising
import win32com.client      # for communication between user and system
import webbrowser           # for opening webpages
import os                   # for interacting with operating system
import datetime             # for getting date and time
import openai               # for interacting with OpenAI API
from config import apikey   # for getting apikey
import subprocess           # for opening apk files

# initializing a variable
chatStr = ""   # to store input from user

# defining a function to perform conversation between user and system(Karen)


1 usage
def chat(query):
    global chatStr
    openai.api_key = apikey
    # to add user's input to chatStr
    chatStr += f"User:{query}\n Karen: "
    # generating responses from OpenAI API based on chat history
    response = openai.Completion.create(
        model="text-davinci-003",
```

```
C:\Users\student\PycharmProjects\Karen\venv\Scripts\python.exe C:\Users\student\PycharmProjects\Karen\main.py
Listening...
Recognizing...
user said:how are you
Chatting...
Listening...
Recognizing...
user said:I am also fine
Chatting...
Listening...
Recognizing...
user said:what are you doing today
```