Voice Assistant Application using KivyMD

Introduction

The Voice Assistant Application is a smart AI-driven assistant built using KivyMD, Speech Recognition, and OpenAI's GPT model. This application allows users to interact via voice commands, get responses, fetch weather updates, retrieve news headlines, tell jokes, open/close applications, and more.

Features

- **Speech Recognition**: Uses speech_recognition library to capture voice commands.
- **Text-to-Speech**: Uses pyttsx3 for speaking responses.
- **ChatGPT Integration**: Fetches AI-generated responses for user queries.
- **Weather Updates**: Retrieves real-time weather data using OpenWeather API.
- **News Headlines**: Fetches top news headlines from News API.
- **Joke Generator**: Provides random jokes using an API.
- **Application Control**: Opens and closes applications like Notepad, Chrome, and Calculator.
- **Interactive UI**: Built using KivyMD with a chat-style display.

Technologies Used

- **Python**
- **KivyMD** (for UI components)
- **SpeechRecognition** (for voice input processing)
- **Pyttsx3** (for text-to-speech conversion)
- **OpenAI API** (for AI-driven responses)
- **OpenWeather API** (for weather updates)
- **NewsAPI** (for fetching news headlines)
- **Psutil** (for process management)

Technical Architecture

1. **User Interaction**: Voice input is captured using speech_recognition.
2. **Processing**: The command is analyzed, and an appropriate response is generated.
3. **Response Handling**: The assistant either speaks the response using pyttsx3 or performs an action (e.g., opening an app).
4. **Integration**: The application interacts with external APIs (OpenAI, OpenWeather, NewsAPI) for dynamic data.
5. **User Interface**: Displays responses in a chat-style UI using KivyMD components.

Challenges Faced & Solutions

1. Speech Recognition Delays

- *Challenge*: Capturing voice accurately without delays.
- *Solution*: Used adjust_for_ambient_noise() to minimize background noise interference.

2. API Response Latency

- *Challenge*: Slow response times from APIs.
- *Solution*: Implemented threading to ensure UI responsiveness while fetching data.

3. Application Execution Issues

- *Challenge*: Not all applications open via os.system.
- *Solution*: Used psutil to dynamically manage processes.

Future Enhancements

- **Mobile Deployment**: Convert the app into a mobile version using Buildozer.
- **Voice Wake Word**: Implement a wake word to activate the assistant without a button press.
- **Customizable Commands**: Allow users to define their own voice commands.
- **UI Improvements**: Enhance the chat interface with avatars and animations.

Conclusion

The Voice Assistant Application provides an intelligent and interactive way for users to interact via voice commands. It integrates multiple APIs to deliver a smart, AI-powered experience. This project serves as a strong foundation for building more advanced AI assistants in the future.

About the Developer

- **Name**: Viki Kumar
- **Role**: Developer
- **Project**: Voice Assistant
- **Institution**: Lovely Professional University
- **Email**: vikikumar0505@gmail.com
- **Submission for**: Placement Exam