AI Voice Receptionist: Project Documentation

Overview

This project implements an AI-powered voice receptionist using:

- **Google Gemini API** for AI conversation.
- **SpeechRecognition** for real-time voice transcription.
- **gTTS (Google Text-to-Speech)** for generating speech.
- **playsound** to play audio responses.

It simulates a dental clinic receptionist that greets and assists users in a conversation-like interface.

Features and Components

1. Environment Setup

```
from dotenv import load_dotenv

load_dotenv()
```

Purpose: Loads environment variables (like API keys) securely from a .env file.

2. Google Gemini API Integration

```
import google.generativeai as genai

genai.configure(api_key=GOOGLE_API_KEY)

model = genai.GenerativeModel("gemini-1.5-flash")
```

Purpose:

- Authenticates and sets up the Gemini AI model.
- gemini-1.5-flash is used for fast and responsive conversations.

3. Speech Recognition

```
import speech_recognition as sr

self.recognizer = sr.Recognizer()

self.microphone = sr.Microphone()
```

Purpose:

- Captures microphone input and converts spoken language into text using Google Speech Recognition.

4. Conversation History

```
self.full_transcript = [

    {"role": "system", "content": "You are a receptionist..."},
]
```

Purpose:

- Stores dialogue history between the user and the AI for context-aware responses.

5. Text-to-Speech with gTTS

```
from gtts import gTTS

tts = gTTS(text=text, lang='en')

tts.save(filename)
```

Purpose: Converts text responses from the AI into spoken audio.

6. Audio Playback

```
import playsound

playsound.playsound(filename)
```

Purpose: Plays the generated .mp3 audio file to simulate a receptionist speaking.

7. Main Functionality

- start_transcription(): Listens to user speech and converts to text.
- generate_ai_response(transcript): Sends user input and full conversation history to Gemini and handles the response.
- generate_audio(text): Converts AI text to speech and plays it.

8. Looped Interaction

```
while True:

  ai_assistant.start_transcription()
```

Purpose: Allows continuous, back-and-forth interaction.

File Structure

```
AI voice/

|

 ├── app.py              # Main application file

 ├── .env                # Contains GOOGLE_API_KEY

└── requirements.txt      # Python dependencies
```

Requirements

Add the following to your requirements.txt:

```
SpeechRecognition
playsound==1.2.2
gtts
google-generativeai
dotenv
```

Notes

- You must have a working microphone for this app.
- Ensure your .env file includes:

```
GOOGLE_API_KEY=your_google_api_key_here
```

- The Gemini API used here is **free to use** within reasonable quotas.

Future Enhancements

- GUI using Streamlit or Tkinter.
- Add multiple language support.
- Store conversation logs.
- Integrate appointment booking backend.

Summary

This project showcases how to build a conversational AI receptionist using only **free tools**, voice input, and speech output, making it suitable for low-resource environments such as clinics or small businesses.