MedyBot Architecture & Workflow

Overview

MedyBot is a command-line medical assistant chatbot powered by Google's Gemini API. It interacts with users (patients), answers health-related questions, suggests over-the-counter medicines, and provides general advice, while always reminding users to consult a real doctor for serious issues. The assistant persona is Dr. Maya, the medical assistant at Wellness Medical Center, and never refers to itself as an AI.

Workflow

1. **User Input:**
   - The user starts the chatbot and enters their health-related question or concern via the command line.
2. **Conversation Context:**
   - The chatbot maintains a conversation history (context) as a list of message dictionaries, each with a `role` ("user" or "model") and a `parts` list (containing the message text).
3. **Instructions:**
   - The first message in the conversation history is a user message containing detailed instructions for the assistant's behavior, including available doctors and response style.
4. **Gemini API Call:**
   - The full conversation history is sent to the Gemini API (`gemini-2.5-flash` model) for processing.
5. **AI Response:**
   - The Gemini model generates a response, which is returned to the chatbot.
6. **Display to User:**
   - The chatbot prints the AI's response to the user in the command line.
7. **Loop:**
   - Steps 1–6 repeat until the user types an exit command (e.g., `exit`, `quit`, `bye`).

Architecture Diagram

```
User (Patient)


[Command Line Interface]
```

```
[Chatbot Script (chat_bot.py)]

      (Maintains conversation history as list of {role, parts})

[Google Gemini API]


[Gemini Model]


[Chatbot Script]


User (Patient)
```

Data Flow

1. **Input:**
   - User types a message/question in the command line.
   - Example: "I have a headache and mild fever. What should I do?"
2. **Processing:**
   - The script appends the user message to the conversation history as `{"role": "user", "parts": [user_input]}`.
   - The full conversation (instructions + history) is sent to the Gemini API.
3. **AI Generation:**
   - The Gemini model processes the input and generates a response based on the context and instructions.
4. **Output:**
   - The response is displayed to the user.
   - Example: "For mild headache and fever, you may consider taking paracetamol. However, if symptoms persist or worsen, please consult a healthcare professional."
5. **Repeat:**
   - The conversation continues, with each new user input and AI response appended to the context.

Key Components

- **Command Line Interface:** Handles user input and output.

- **Conversation Manager:** Maintains the conversation history for context-aware responses (list of message dicts).
- **Instructions Message:** Sets the assistant's behavior, available doctors, and boundaries.
- **Gemini API Client:** Sends requests and receives responses from the Gemini model.

Security & Privacy Notes

- No user data is stored permanently; all conversation context is kept in memory during the session only.
- The chatbot does not diagnose or prescribe; it only provides general advice and always recommends consulting a real doctor.

Extending the System

- **GUI Integration:** The chatbot can be extended to a web or mobile interface.
- **Medical Database Integration:** For more accurate medicine suggestions, integrate with a verified medical database.
- **Multilingual Support:** Add language translation for broader accessibility.

Disclaimer

This architecture is for a demo/educational chatbot and should not be used for real medical decision-making.