AI ChatBot Pro: Intelligent Conversational and File Analysis Assistant

Date: July 13, 2025
Submitted By: Anbhi Thakur

1. Introduction

The **AI ChatBot Pro** is an advanced, interactive chatbot application powered by OpenAI's GPT-4o model. It supports multi-modal file analysis (text, image, PDF) and natural conversation through a sleek **Streamlit** interface. Designed for users needing intelligent dialogue and file insights, the system merges **LLM (Large Language Model) reasoning** with **real-time file analysis** and **dynamic task handling**.

---

2. Key Components

Component	Description
`app.py`	Main Streamlit UI for chatbot, file upload, and communication flow.
`ai_client.py`	Handles interaction with OpenAI's GPT-4o via the API.
`file_processor.py`	Processes and analyzes uploaded files (image, PDF, text).
`run.py`	Launcher script to start the application with API key and port validation.
`requirements.txt`	Lists all Python dependencies.

---

3. Technologies Used

- **Streamlit**: Interactive web UI for the chatbot.
- **OpenAI GPT-4o**: Text + vision LLM used for processing user input and uploaded files.
- **Pillow (PIL)**: For image file processing.
- **PyPDF2**: PDF content extraction and analysis.
- **python-magic + chardet**: MIME-type detection for uploaded files.
- **Base64**: Used to encode images for vision-based LLM input.
- **Docker / Render** (optional): Deployment.

---

4. Implementation Steps

1. **Frontend Setup**:
    - Created UI using Streamlit with theme-based CSS.
    - UI includes chat interface, file uploader, and response area.
2. **Backend Integration**:
    - `AIClient` class manages interactions with OpenAI.
    - Selects either GPT-4o text or vision model based on file content.
3. **File Processing**:
    - Files uploaded are analyzed using `FileProcessor`.
    - Extracts metadata, text content (PDF), and encodes images.
4. **Chat Handling**:
    - Chat history stored and sent along with the latest query.
    - GPT-4o model generates responses, including file-based insights.
5. **Execution & API Key Handling**:
    - `run.py` ensures API key setup and dependency check before launching.

---

5. Project Folder Structure

```
bash
CopyEdit
ai-chatbot-pro/
├── app.py                 # Main Streamlit application
├── utils/
│   ├── ai_client.py       # OpenAI API client
│   └── file_processor.py  # File analysis engine
├── .streamlit/
│   └── config.toml        # Streamlit configuration
├── pyproject.toml         # Project metadata (optional)
├── run.py                 # Launcher script
├── README.md              # Project documentation
└── requirements.txt       # Python dependencies
```

---

6. Deployment Steps

A. Local Deployment:

1. Clone the repository.
2. Install dependencies:
   `pip install -r requirements.txt`
3. Set the OpenAI API key:
   `export OPENAI_API_KEY=your-key-here`
4. Run the app:
   `python run.py`

B. Render Deployment:

1. Push code to GitHub.
2. Create a new **Render Web Service**.
3. Add environment variable `OPENAI_API_KEY`.
4. Set start command:
   `python run.py`
5. Deploy and access via the given Render URL.

---

7. Output

Upon successful deployment, users can:

- **Chat with the bot**: Ask questions, solve problems, or converse naturally.
- **Upload files**: Upload PDFs, images, or text files for AI-based content analysis.
- **Receive tailored responses**: The chatbot integrates file insights into its replies.
- **View enhanced UI**: Aesthetic and user-friendly interface with responsive CSS.

---

8. Conclusion

AI ChatBot Pro blends the power of **GPT-4o's multimodal capabilities** with a clean, interactive UI built on Streamlit. Its support for **file analysis**, **math/problem-solving**, and **contextual conversation** makes it a versatile assistant for educational, technical, and general use cases. The modular structure allows easy customization and future scalability.