Introduction

The AI Chat Assistant is a web-based application developed using **Python**, **Streamlit**, and **LangChain**, integrated with **Azure OpenAI** services. This project demonstrates the ability to build an interactive chatbot powered by a GPT model (GPT-4) deployed on Azure. The chatbot allows users to input natural language queries and receive intelligent, context-aware responses in real-time.

The primary purpose of this project is to create a user-friendly interface for conversational AI, showcasing capabilities such as question answering, assistance, and general conversational interactions. It is ideal for beginners to understand AI integration with web applications and also demonstrates practical usage of cloud-based AI APIs.

Requirement Determination & Analysis

Hardware Requirements:

- A computer with at least 8GB RAM and modern CPU.
- Stable internet connection for API calls to Azure OpenAI.

Software Requirements:

- **Python**: Programming language for the application.
- **Streamlit**: For building a simple and interactive web interface.
- **LangChain**: Provides a framework to integrate LLMs like GPT.
- **Azure OpenAI API**: Provides access to GPT models for generating responses.
- **Python-dotenv**: For secure storage of API keys and environment variables.

Functional Requirements:

1. Users should be able to type queries in natural language.
2. The AI should respond intelligently using GPT-4 deployed on Azure.
3. Chat history should persist during a session.
4. The interface should be interactive and user-friendly.

Non-functional Requirements:

- Secure storage of API keys using .env file.
- Efficient response time with minimal lag.
- Clean UI with custom styling for chat messages.

System Design

Architecture Overview:
The system uses a client-server model where:

- **Frontend (Streamlit)**: Handles user input, displays chat history, and manages the UI.
- **Backend (LangChain + Azure OpenAI)**: Handles AI processing and generates responses using the GPT-4 model.

Flow of the System:

1. User enters a message in the chat input field.
2. The message is appended to the chat history stored in st.session_state.
3. The message is sent to Azure OpenAI via LangChain's AzureChatOpenAI.
4. GPT-4 processes the input and returns a response.
5. The response is appended to the chat history and displayed in the Streamlit interface.

Folder Structure:

```
AI_Chat_Assistant/
│
├── .env                 # Stores API keys securely
├── requirements.txt     # Lists Python dependencies
└── app.py               # Main Streamlit application
```

4) Development

Step 1: Environment Setup

- Install Python and required libraries using pip install -r requirements.txt.
- Create .env file with Azure OpenAI API credentials.

Step 2: Streamlit UI

- Configure page layout, title, and input fields using st.set_page_config and st.chat_input.
- Add custom CSS for chat message styling to differentiate user and AI responses.

Step 3: Integration with Azure OpenAI

- Initialize AzureChatOpenAI class from langchain_community.chat_models.
- Load environment variables using dotenv.

- Define a function get_llm() to cache the LLM instance for efficiency.

Step 4: Chat Logic

- Maintain chat history in st.session_state.messages.
- When the user inputs a message, call the LLM to generate a response.
- Append AI response to chat history and rerun Streamlit to refresh the interface.

Step 5: Testing & Debugging

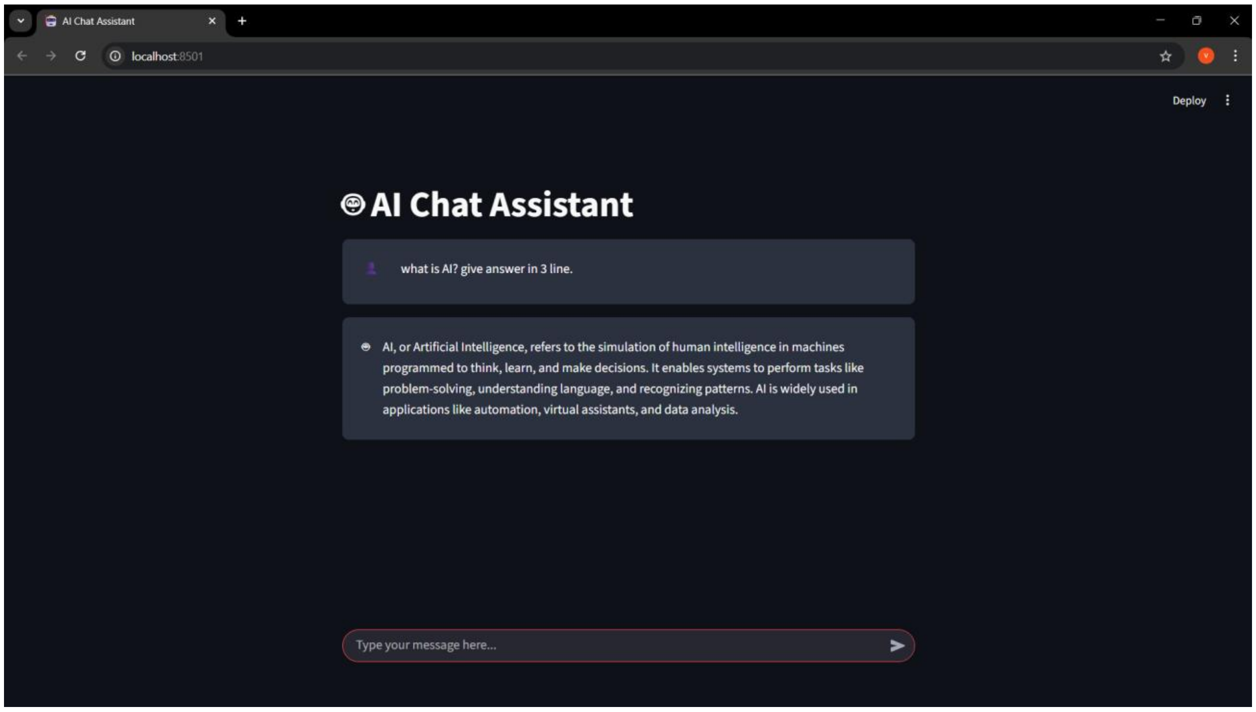
- Verify that the chat history persists across interactions.
- Ensure correct responses from the GPT-4 model.
- Confirm API keys are loaded correctly from .env.

Proposed Enhancements

1. **Context Awareness:** Maintain longer conversation context for multi-turn dialogues.
2. **File Upload & Processing:** Allow users to upload documents or PDFs for Q&A.
3. **Voice Interaction:** Enable speech-to-text input and text-to-speech output.
4. **Multiple LLM Models:** Provide users with options to switch between GPT-3.5, GPT-4, and other specialized models.
5. **Enhanced UI/UX:** Add chat bubbles, timestamps, and user avatars for better user experience.
6. **Database Integration:** Store past chat sessions for analysis or learning.

Output Photos

Chatbot interface on Streamlit.

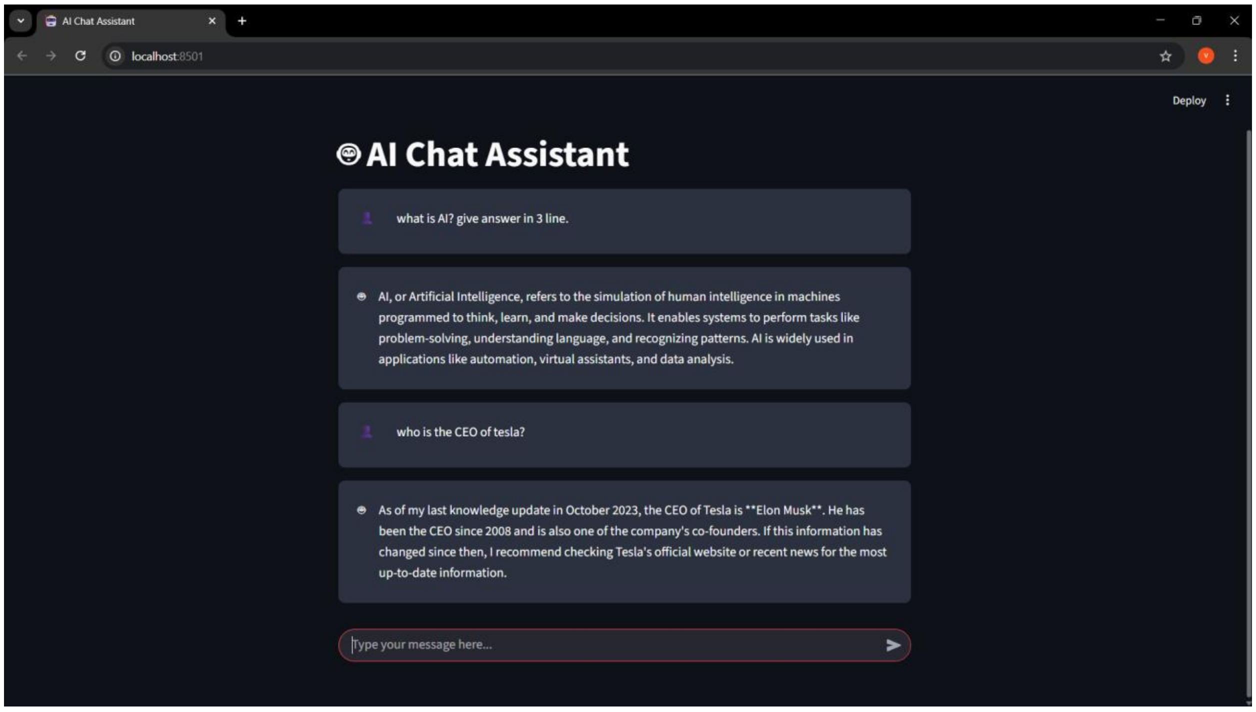


AI Chat Assistant

what is AI? give answer in 3 line.

AI, or Artificial Intelligence, refers to the simulation of human intelligence in machines programmed to think, learn, and make decisions. It enables systems to perform tasks like problem-solving, understanding language, and recognizing patterns. AI is widely used in applications like automation, virtual assistants, and data analysis.

Type your message here...



AI Chat Assistant

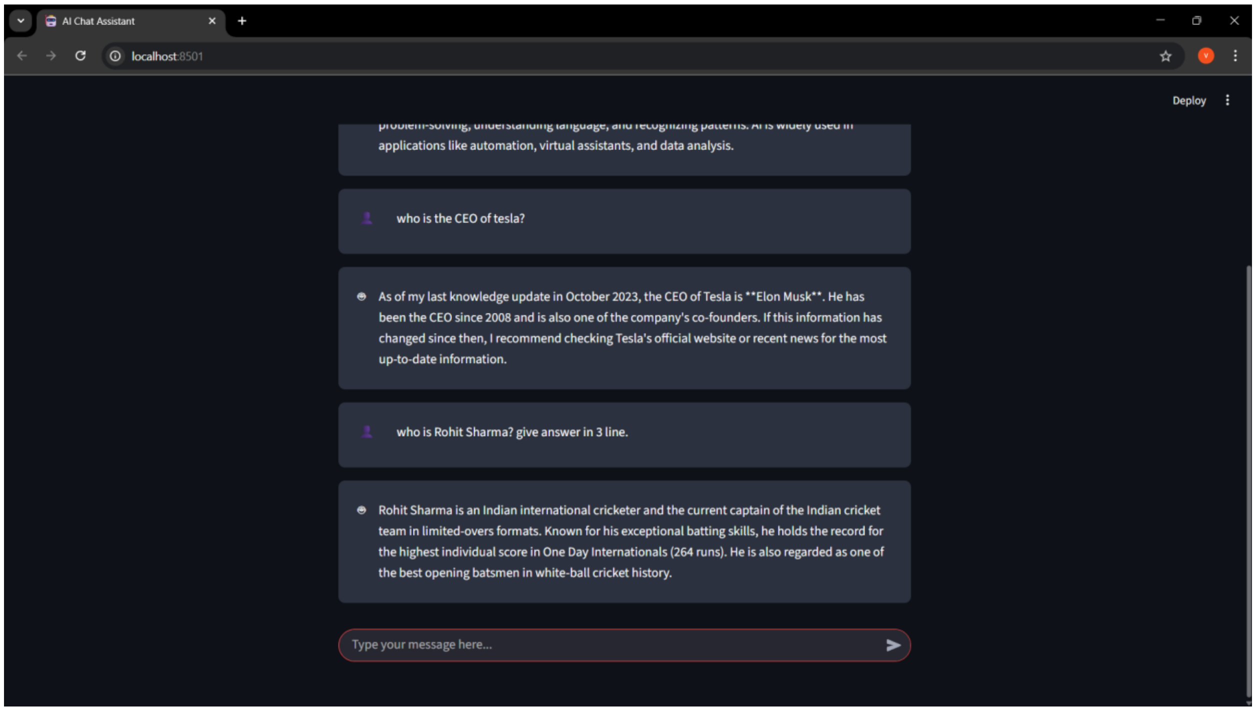
what is AI? give answer in 3 line.

AI, or Artificial Intelligence, refers to the simulation of human intelligence in machines programmed to think, learn, and make decisions. It enables systems to perform tasks like problem-solving, understanding language, and recognizing patterns. AI is widely used in applications like automation, virtual assistants, and data analysis.

who is the CEO of tesla?

As of my last knowledge update in October 2023, the CEO of Tesla is **Elon Musk**. He has been the CEO since 2008 and is also one of the company's co-founders. If this information has changed since then, I recommend checking Tesla's official website or recent news for the most up-to-date information.

Type your message here...



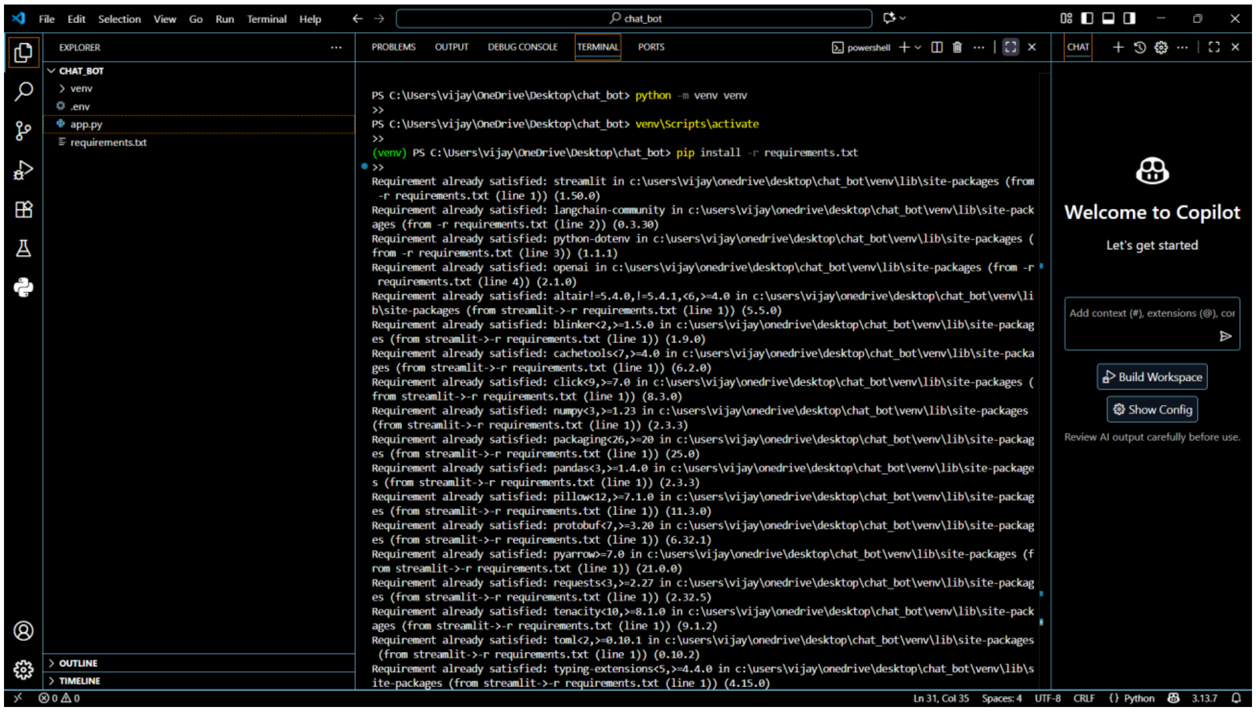
problem-solving, understanding language, and recognizing patterns. AI is widely used in applications like automation, virtual assistants, and data analysis.

who is the CEO of tesla?

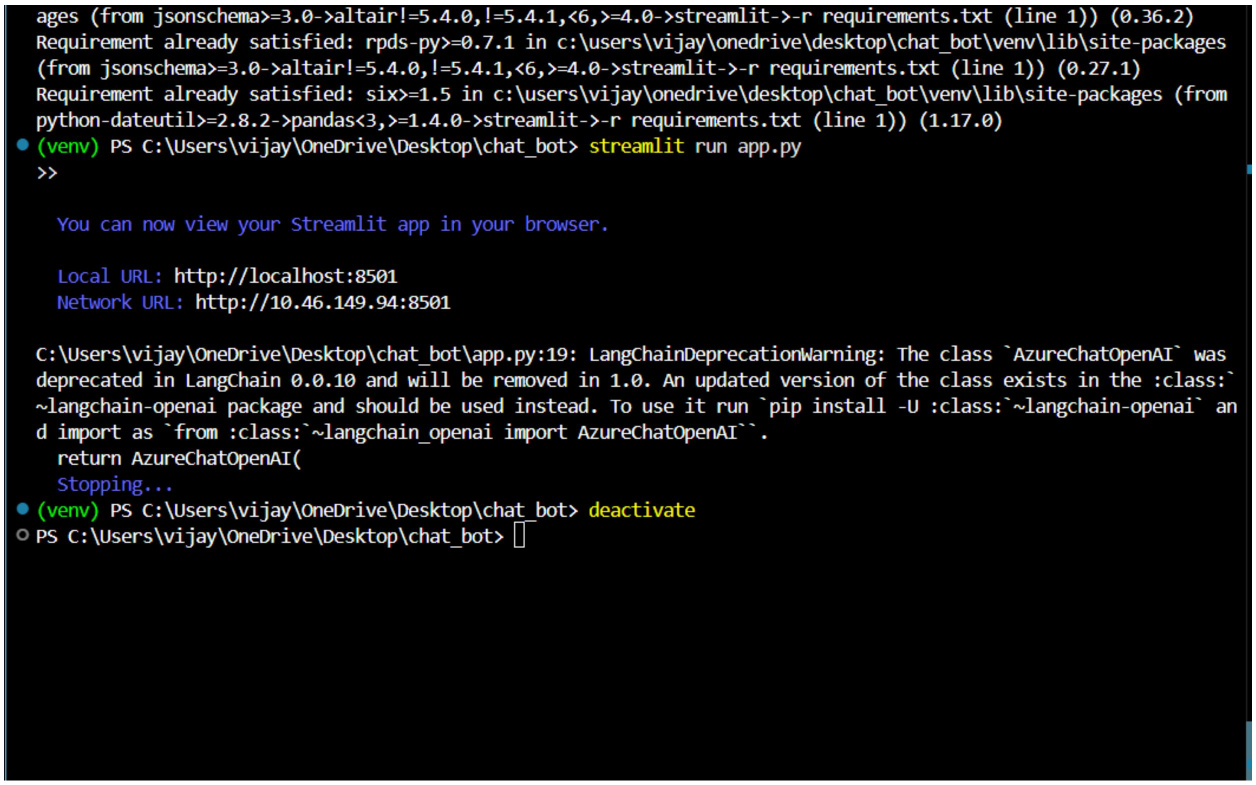
As of my last knowledge update in October 2023, the CEO of Tesla is **Elon Musk**. He has been the CEO since 2008 and is also one of the company's co-founders. If this information has changed since then, I recommend checking Tesla's official website or recent news for the most up-to-date information.

who is Rohit Sharma? give answer in 3 line.

Rohit Sharma is an Indian international cricketer and the current captain of the Indian cricket team in limited-overs formats. Known for his exceptional batting skills, he holds the record for the highest individual score in One Day Internationals (264 runs). He is also regarded as one of the best opening batsmen in white-ball cricket history.



```
PS C:\Users\vijay\OneDrive\Desktop\chat_bot> python -m venv venv
>>
PS C:\Users\vijay\OneDrive\Desktop\chat_bot> venv\Scripts\activate
>>
(venv) PS C:\Users\vijay\OneDrive\Desktop\chat_bot> pip install -r requirements.txt
>>
Requirement already satisfied: streamlit in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages (from
 -r requirements.txt (line 1)) (1.50.0)
Requirement already satisfied: langchain-community in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-pack
ages (from -r requirements.txt (line 2)) (0.3.30)
Requirement already satisfied: python-dotenv in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages (
from -r requirements.txt (line 3)) (1.1.1)
Requirement already satisfied: openai in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages (from -r
 requirements.txt (line 4)) (2.1.0)
Requirement already satisfied: altair!=5.4.0,!=5.4.1,<6,>=4.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\li
b\site-packages (from streamlit->-r requirements.txt (line 1)) (5.5.0)
Requirement already satisfied: blinker<2,>=1.5.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packag
es (from streamlit->-r requirements.txt (line 1)) (1.9.0)
Requirement already satisfied: cachetools<7,>=4.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packa
ges (from streamlit->-r requirements.txt (line 1)) (6.2.0)
Requirement already satisfied: click<9,>=7.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages (
from streamlit->-r requirements.txt (line 1)) (8.3.0)
Requirement already satisfied: numpy<3,>=1.23 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages
(from streamlit->-r requirements.txt (line 1)) (2.3.3)
Requirement already satisfied: packaging<26,>=20 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packag
es (from streamlit->-r requirements.txt (line 1)) (25.0)
Requirement already satisfied: pandas<3,>=1.4.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-package
s (from streamlit->-r requirements.txt (line 1)) (2.3.3)
Requirement already satisfied: pillow<12,>=7.1.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packag
es (from streamlit->-r requirements.txt (line 1)) (11.3.0)
Requirement already satisfied: protobuf<7,>=3.20 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packag
es (from streamlit->-r requirements.txt (line 1)) (6.32.1)
Requirement already satisfied: pyarrow>=7.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages (f
rom streamlit->-r requirements.txt (line 1)) (21.0.0)
Requirement already satisfied: requests<3,>=2.27 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packag
es (from streamlit->-r requirements.txt (line 1)) (2.32.5)
Requirement already satisfied: tenacity<10,>=8.1.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-pack
ages (from streamlit->-r requirements.txt (line 1)) (9.1.2)
Requirement already satisfied: toml<2,>=0.10.1 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages
 (from streamlit->-r requirements.txt (line 1)) (0.10.2)
Requirement already satisfied: typing-extensions<5,>=4.4.0 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\s
ite-packages (from streamlit->-r requirements.txt (line 1)) (4.15.0)
```



```
ages (from jsonschema>=3.0->altair!=5.4.0,!=5.4.1,<6,>=4.0->streamlit->-r requirements.txt (line 1)) (0.36.2)
Requirement already satisfied: rpds-py>=0.7.1 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages
(from jsonschema>=3.0->altair!=5.4.0,!=5.4.1,<6,>=4.0->streamlit->-r requirements.txt (line 1)) (0.27.1)
Requirement already satisfied: six>=1.5 in c:\users\vijay\onedrive\desktop\chat_bot\venv\lib\site-packages (from
python-dateutil>=2.8.2->pandas<3,>=1.4.0->streamlit->-r requirements.txt (line 1)) (1.17.0)
(venv) PS C:\Users\vijay\OneDrive\Desktop\chat_bot> streamlit run app.py
>>

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://10.46.149.94:8501

C:\Users\vijay\OneDrive\Desktop\chat_bot\app.py:19: LangChainDeprecationWarning: The class `AzureChatOpenAI` was
deprecated in LangChain 0.0.10 and will be removed in 1.0. An updated version of the class exists in the :class:`
~langchain-openai package and should be used instead. To use it run `pip install -U :class:`~langchain-openai` an
d import as `from :class:`~langchain_openai import AzureChatOpenAI``.
  return AzureChatOpenAI(
  Stopping...
(venv) PS C:\Users\vijay\OneDrive\Desktop\chat_bot> deactivate
PS C:\Users\vijay\OneDrive\Desktop\chat_bot>
```

Conclusion

The AI Chat Assistant project demonstrates a successful integration of **Streamlit**, **LangChain**, and **Azure OpenAI** services to create an interactive conversational AI system. The project highlights key aspects of AI deployment, user interface design, and real-time interaction.

Through this project, one can understand:

- How to securely manage API credentials using .env files.
- How to build responsive web interfaces using Streamlit.
- How to utilize GPT models via cloud-based APIs.

The project serves as a foundation for more advanced AI chatbot applications and provides ample scope for enhancements such as context retention, voice interaction, and database connectivity.

Author’s Note

Dear Kunal Kapur,

I have prepared the PDF for the AI Chat Assistant project and included all explanations, steps, and output screenshots. For security reasons, I have not shared my Azure API key in the document or code files; in the code, the API key fields are masked with asterisks (*****). I hope you understand this precaution to keep my credentials secure. All other code and project details are provided for review.

Thank you for your understanding.

Best regards,
Vijay Makwana