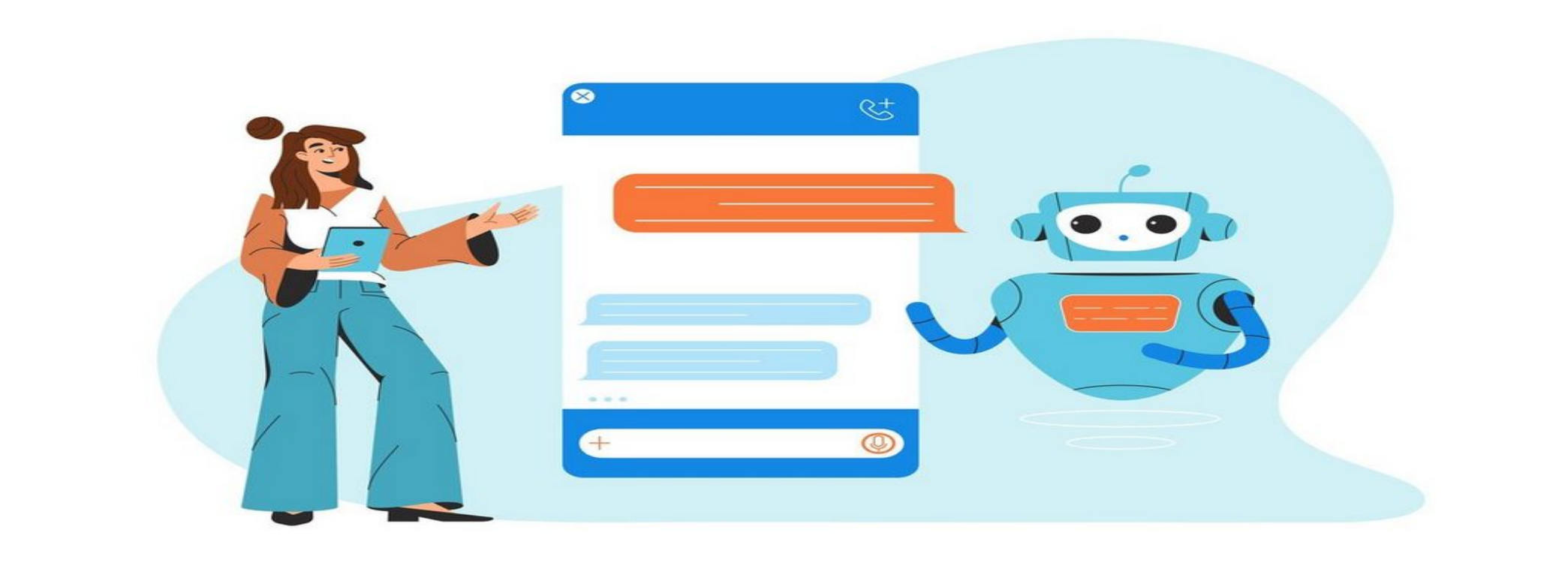
STREAMLIT-POWERED CHATBOT WITH GPT-4 INTEGRATION



Prof: Dr. Chang, Henry

Project by:
Vaishnavi Patil
ID: 20133
Course: Generative AI-Driven Intelligent Apps Development
Date: October 2024

TABLE OF CONTENTS

INTRODUCTION

- **Project Overview**:
  - Develop a web-based chatbot using Streamlit and GPT-4 API.
  - Implement a visually appealing UI with a shaded blue gradient.
- **Purpose**:
  - Create an interactive chatbot to answer user queries.
- **Technologies Used**:
  - Streamlit, Python, OpenAI GPT-4 API, CSS for styling.

DESIGN - PROBLEM IDENTIFICATION

Why This Approach?

- **Problem Identified**:

  - ✔ Need for Efficient Human-Computer Interaction
  - ✔ Demand for Scalable and User-Friendly Interfaces
  - ✔ Maintaining Conversational Context
  - ✔ Integrating Real-Time Response Generation
  - ✔ Customizing User Experience with Visual Appeal
  - ✔ Accessibility and Deployment Challenges
  - ✔ Future Proofing and Scalability

- **Solution Requirements**:

  - ✔ A simple, scalable, web-based chatbot.
  - ✔ Integration with GPT-4 for dynamic responses.
  - ✔ Custom background for enhanced UX.

DESIGN - INVESTIGATING SOLUTIONS

Possible Solutions:

- ❑ **Option 1**: Building a custom backend API and web interface from scratch.
- ❑ **Option 2**: Using pre-built web frameworks like Streamlit or Flask.

Chosen Approach:

- ✔ **Streamlit** for faster prototyping and simplicity.
- ✔ **OpenAI GPT-4 API** for powerful, real-time response generation.
- ✔ **CSS Styling** for enhancing user experience.

IMPLEMENTATION - HOW IT WAS BUILT

Code Setup:

1. Using **Python** to manage OpenAI API integration.
2. **CSS**: Applied for custom gradient background and "Powered by Streamlit" footer.
3. **Session Management**: Streamlit’s session state to manage chat history.
4. **API Streaming**: Real-time response streaming with OpenAI's GPT-4.

TESTING - CHATBOT FUNCTIONALITY

Testing Steps:

- Tested for different user inputs to ensure meaningful responses.
- Verified session persistence (conversation history stored).
- Ensured background and footer styling is consistent across all devices.

Test Cases:

- Short queries vs. complex queries.
- Checking response generation speed (around 2-3 seconds for complete response)
- Verifying correct CSS rendering.

ENHANCEMENT IDEAS

Improvements to Consider:

- **Voice Input**: Integrating speech-to-text for voice queries.
- **Multiple Models**: Allow users to choose between different OpenAI models.
- **Analytics Dashboard**: Add metrics to track user interaction with the chatbot.
- **Dark Mode**: Implement a toggle between light and dark themes.

CONCLUSION

- ✔ Successfully developed a chatbot using Streamlit and OpenAI’s GPT-4 API.
- ✔ Implemented session handling and styled the UI with a gradient background.
- ✔ Streamlit's ease of use facilitated rapid development.
- ✔ GPT-4 API provided strong, real-time conversational capabilities.

REFERENCES

Technologies Researched:

- **Streamlit Documentation**: Used for web app framework setup.
- **OpenAI API Documentation**: For integration of GPT-4 model.
- **CSS Gradients**: Researched gradient styling techniques.

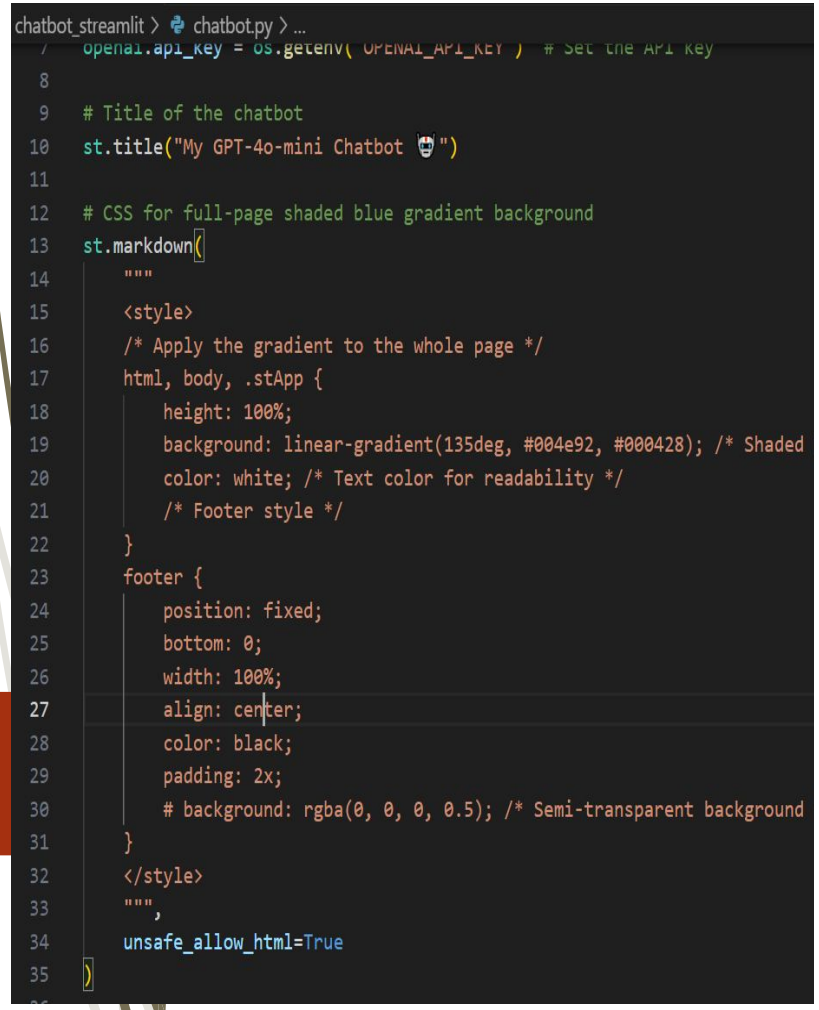
Information Literacy Search Technique:

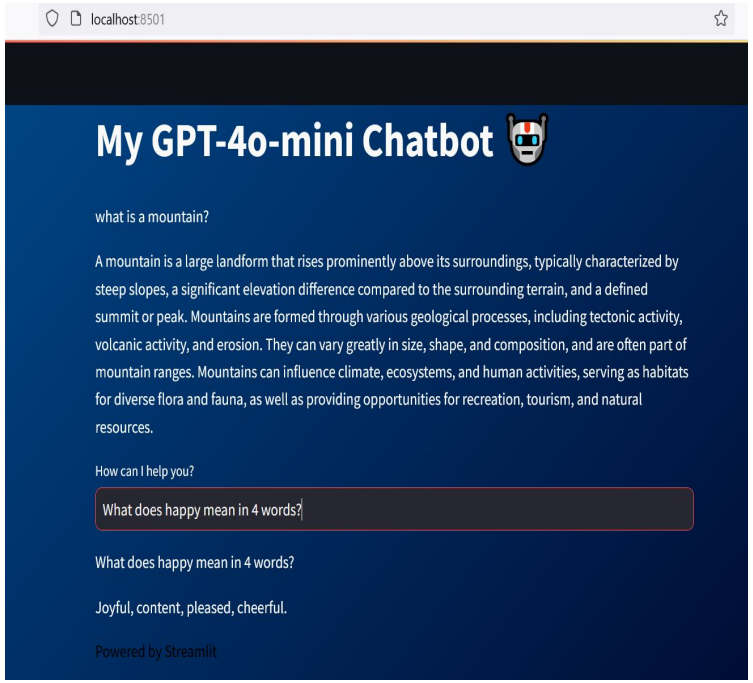
- Searched for chatbot technologies, real-time streaming, and web development frameworks.
- OpenAI_API_ChatBot_Streamlit

APPENDIX

chatbot_streamlit > chatbot.py > ...

```python

# Initialize messages in the session state
if "messages" not in st.session_state:
    st.session_state.messages = []

# Display messages
for message in st.session_state["messages"]:
    with st.text(message["role"]):
        st.markdown(message["content"])

# Handle user input and OpenAI response
if user_prompt := st.text_input("How can I help you?"):
    st.session_state.messages.append({"role": "user", "content": user_prompt})

    # Display user message
    with st.text("user"):
        st.markdown(user_prompt)

    # Assistant response
    with st.text("assistant"):
        chatbot_msg = st.empty()
        full_response = ""
        stream = openai.chat.completions.create(
            model="gpt-4o-mini",
            messages=[
                {"role": msg["role"], "content": msg["content"]}
                for msg in st.session_state["messages"]
            ],
            temperature=0,
            stream=True,
        )

        # Stream the response
        for chunk in stream:
            token = chunk.choices[0].delta.content
            if token is not None:
                full_response = full_response + token
                chatbot_msg.markdown(full_response)

        chatbot_msg.markdown(full_response)

    # Store assistant's response in session
    st.session_state.messages.append({"role": "assistant", "content": full_response})
```


chatbot_streamlit > chatbot.py > ...
openai.api_key = os.getenv(OPENAI_API_KEY) # Set the API key
# Title of the chatbot
st.title("My GPT-4o-mini Chatbot")
# CSS for full-page shaded blue gradient background
st.markdown(
"""
<style>
/* Apply the gradient to the whole page */
html, body, .stApp {
height: 100%;
background: linear-gradient(135deg, #004e92, #000428); /* Shaded
color: white; /* Text color for readability */
/* Footer style */
}
footer {
position: fixed;
bottom: 0;
width: 100%;
align: center;
color: black;
padding: 2x;
# background: rgba(0, 0, 0, 0.5); /* Semi-transparent background
}
</style>
""",
unsafe_allow_html=True


localhost:8501
My GPT-4o-mini Chatbot
what is a mountain?
A mountain is a large landform that rises prominently above its surroundings, typically characterized by steep slopes, a significant elevation difference compared to the surrounding terrain, and a defined summit or peak. Mountains are formed through various geological processes, including tectonic activity, volcanic activity, and erosion. They can vary greatly in size, shape, and composition, and are often part of mountain ranges. Mountains can influence climate, ecosystems, and human activities, serving as habitats for diverse flora and fauna, as well as providing opportunities for recreation, tourism, and natural resources.
How can I help you?
What does happy mean in 4 words?
What does happy mean in 4 words?
Joyful, content, pleased, cheerful.
Powered by Streamlit


My GPT-4o-mini Chatbot
How can I help you?
Give me a function to find a prime number
Give me a function to find a prime number
Certainly! Below is a simple Python function that checks if a number is prime. A prime number is a natural number greater than 1 that cannot be formed by multiplying two smaller natural numbers. In other words, a prime number has no positive divisors other than 1 and itself.
Here's the function:
def is_prime(n):
"""Check if a number is prime."""
if n <= 1:
return False
if n <= 3:
return True
if n % 2 == 0 or n % 3 == 0:
return False
i = 5
while i * i <= n:
if n % i == 0 or n % (i + 2) == 0:
return False
i += 6
return True
# Example usage:
number = 29
if is_prime(number):
print(f"{number} is a prime number.")
else:
print(f"{number} is not a prime number.")
Explanation:
1. The function first checks if n is less than or equal to 1, in which case it returns False since prime numbers are greater than 1.
2. It then checks if n is 2 or 3, which are both prime numbers.
3. The function eliminates even numbers greater than 2 and multiples of 3.
4. Finally, it checks for factors from 5 to the square root of n, incrementing by 6 (to check only numbers of the form 6k ± 1).

THANK YOU