Build A Chatbot Interface with Streamlit

Let's build a chatbot interface with just Python using the Streamlit library. You can find more info here in below in the official Streamlit docs and check out the repository with all the code.

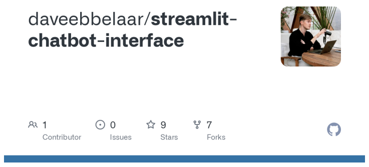
Build a basic LLM chat app - Streamlit Docs

https://docs.streamlit.io/knowledge-base/tutorials/build-conversational-apps

GitHub - daveebbelaar/streamlit-chatbot-interface

Contribute to daveebbelaar/streamlit-chatbot-interface development by creating an account on GitHub.

https://github.com/daveebbelaar/streamlit-chatbot-interface?tab=readme-o...



Steps

1. Create a new venv or conda environment

3. Install Streamlit

```
pip install streamlit openai python-dotenv
```

4. Test installation

```
streamlit hello
```

5. Create `.env` file with OPENAI_API_KEY

```
OPENAI_API_KEY="your-opnai-api-key"
```

6. Create your `app.py` file

7. Add imports

```
from openai import OpenAI
import streamlit as st
from dotenv import load_dotenv
import os
import shelve
```

8. Add setup

```
load_dotenv()
st.title("ChatGPT-like Chatbot Demo")

USER_AVATAR = "⊠"
BOT_AVATAR = "🤖"
client = OpenAI(api_key=os.getenv("OPENAI_API_KEY"))
```

9. Configure model and messages

```
# Ensure openai_model is initialized in session state
if "openai_model" not in st.session_state:
    st.session_state["openai_model"] = "gpt-3.5-turbo"


# Load chat history from shelve file
def load_chat_history():
    with shelve.open("chat_history") as db:
        return db.get("messages", [])


# Save chat history to shelve file
def save_chat_history(messages):
    with shelve.open("chat_history") as db:
        db["messages"] = messages


# Initialize or load chat history
if "messages" not in st.session_state:
    st.session_state.messages = load_chat_history()

# Sidebar with a button to delete chat history
with st.sidebar:
    if st.button("Delete Chat History"):
        st.session_state.messages = []
        save_chat_history([])
```

10. Configure main chat interface

```
# Display chat messages
for message in st.session_state.messages:
    avatar = USER_AVATAR if message["role"] == "user" else BOT_AVATAR
    with st.chat_message(message["role"], avatar=avatar):
        st.markdown(message["content"])

# Main chat interface
if prompt := st.chat_input("How can I help?"):
    st.session_state.messages.append({"role": "user", "content": prompt})
    with st.chat_message("user", avatar=USER_AVATAR):
        st.markdown(prompt)

    with st.chat_message("assistant", avatar=BOT_AVATAR):
        message_placeholder = st.empty()
        full_response = ""
        for response in client.chat.completions.create(
            model=st.session_state["openai_model"],
            messages=st.session_state["messages"],
            stream=True,
        ):
            full_response += response.choices[0].delta.content or ""
            message_placeholder.markdown(full_response + "|")
        message_placeholder.markdown(full_response)
    st.session_state.messages.append({"role": "assistant", "content": full_response})

# Save chat history after each interaction
save_chat_history(st.session_state.messages)
```

11. Run your Streamlit app with

```
streamlit run app.py
```

12. Learn how to integrate LangChain and deploy the app

Build an LLM app using LangChain - Streamlit Docs

https://docs.streamlit.io/knowledge-base/tutorials/llm-quickstart