# 

## **Your Own Chatbot with Streamlit and LangChain**

This project demonstrates how to build a chatbot application using **Streamlit**, **LangChain**, and **Ollama**'s LLM (Llama). The application supports chat history persistence and interactive user sessions.

## **Prerequisites**

### **1. Install Required Tools**

* **Python (>= 3.9)**: Ensure Python is installed on your system.
* **Virtual Environment**: Create an isolated Python environment using venv or any other tool of your choice.

### **2. Install Dependencies**

Install the following packages:

pip install streamlit langchain-core langchain-ollama langchain-community

### **3. Set Up Ollama**

* **Install Ollama**: Download and install Ollama from [Ollama's official website](https://ollama.ai/download).

**Run Ollama Server**:  
  
 Start the local Ollama server:  
  
 ollama serve

* By default, the server runs on http://localhost:11434.

**Pull the Required Model**:  
  
 ollama pull llama3.2:3b

* Replace llama3.2:3b with the desired model name if using a different one.

### **4. Database Setup**

This chatbot uses an SQLite database to store chat history. Ensure sqlite3 is installed (comes preinstalled with most Python distributions).

## **Setting Up the Project**

### **1. Clone or Download the Repository**

Download the project files to your local machine.

### **2. Create a Virtual Environment**

Create a virtual environment and activate it:

python -m venv chatbot\_env

source chatbot\_env/bin/activate # On Windows: chatbot\_env\Scripts\activate

### **3. Install Project Dependencies**

Run the following command to install all required Python packages:

pip install -r requirements.txt

### **4. Launch the Chatbot Application**

Run the Streamlit application using:

streamlit run chatbot.py

## **Project Files**

* **chatbot.py**: Main application file containing the Streamlit chatbot implementation.
* **requirements.txt**: Lists all dependencies required for the project.
* **chat\_history.db**: SQLite database file for storing chat history (auto-created when running the app).

## **Key Features**

1. **Interactive Chat Interface**:  
   * Users can interact with the chatbot through a Streamlit-powered web interface.
2. **Chat History Management**:  
   * Stores chat history in a persistent SQLite database.
   * Allows clearing chat history for new conversations.
3. **Integration with Llama LLM**:  
   * Uses Ollama's local LLM for natural language processing.
4. **Customizable Prompt Templates**:  
   * System and user prompts are defined using LangChain's prompt templates.

## **Code Walkthrough**

### **Main Components**

1. **Streamlit Interface**:  
   * st.title() and st.write() create the UI.
   * st.text\_input() and st.chat\_input() accept user inputs.
   * st.chat\_message() displays chat messages.
2. **Chat History**:  
   * Handled using st.session\_state and SQLChatMessageHistory from langchain\_community.
3. **LangChain Integration**:  
   * Prompts and message handling are defined using ChatPromptTemplate and RunnableWithMessageHistory.
4. **Ollama Integration**:  
   * Configured with ChatOllama to connect to the local Llama model.

## **Troubleshooting**

### **Common Errors**

1. **Ollama server not running**:  
   * Ensure the Ollama server is started using ollama serve.
2. **Model not found**:  
   * Pull the required model using ollama pull llama3.2:3b.
3. **Database file missing**:  
   * Ensure the SQLite database file path is correct or allow the application to auto-create it.
4. **Dependency Issues**:  
   * Check the requirements.txt file and ensure all dependencies are installed.

## **Future Enhancements**

* Add user authentication for personalized chat histories.
* Support for additional LLMs or remote API integration.
* Improved UI with additional features like file uploads or custom theme support.