Local LLM Chat App

🪄 Project Overview

Local LLM Chat App is a lightweight AI chatbot interface that allows users to chat with locally hosted Large Language Models (LLMs) such as **LLaMA 3** and **Mistral** through **Ollama**.
It combines a **FastAPI backend** for API handling and a **Streamlit frontend** for an interactive web chat UI.

This setup is ideal for **offline**, **secure**, and **customizable** AI interactions without relying on cloud-based APIs.

⚙️ Tech Stack

Technology	Purpose
Python 3.10+	Core programming language
FastAPI	Serves as backend REST API handling chat requests
Streamlit	Frontend web interface for user interaction
Ollama	Local LLM runtime (runs models like LLaMA, Mistral locally)
Requests	Handles communication between Streamlit frontend and FastAPI backend
Pydantic	Defines data models for API input validation

📁 Project Structure

```
llm-chat-app/
│
├── backend/
│   ├── main.py                # FastAPI backend to route and process chat requests
│
├── frontend/
```

```
|    ├── app.py                # Streamlit frontend (chat interface)
|
├── requirements.txt           # Python dependencies
└── README.md                  # Documentation
```

🧩 requirements.txt

```
fastapi
uvicorn
requests
streamlit
pydantic
```

🛠️ Setup Instructions

1️⃣ Clone the repository

```
git clone https://github.com/yourusername/llm-chat-app.git
cd llm-chat-app
```

2️⃣ Create and activate virtual environment

```
python -m venv venv
venv\Scripts\activate   # On Windows
source venv/bin/activate  # On Mac/Linux
```

3️⃣ Install dependencies

```
pip install -r requirements.txt
```

4️⃣ Install and set up Ollama

- Download from: https://ollama.com/download

After installation, verify:

```
ollama --version
```

-

Pull a model (e.g., LLaMA 2):

```
ollama pull llama2
```

-

5️⃣ Run the backend (FastAPI)

```
uvicorn backend.main:app --reload
```

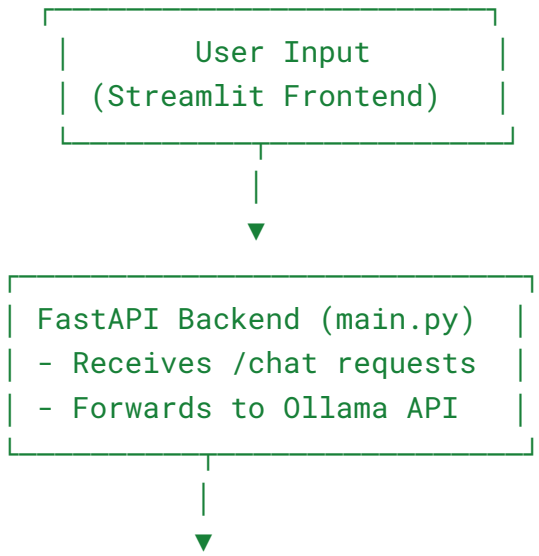
Backend runs at 👉 http://127.0.0.1:8000

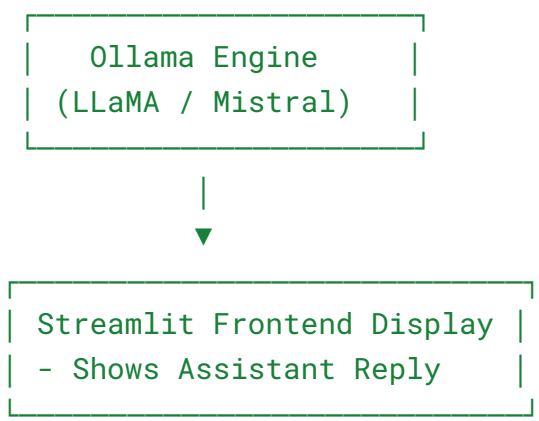
6️⃣ Run the frontend (Streamlit)

```
streamlit run app.py
```

Frontend runs at 👉 http://localhost:8501

🧱 Architecture





🔄 Flow of Execution

1. User types a message in **Streamlit chat input**.
2. Streamlit sends the message and chat history to **FastAPI backend** (`/chat` endpoint).
3. FastAPI forwards the request to **Ollama API** running locally.
4. Ollama processes the message using the selected LLM model (e.g., LLaMA3).
5. The generated response is sent back to **FastAPI**, then to **Streamlit**.
6. Streamlit updates the chat interface with both user and assistant messages.

🚧 Possible Issues and Fixes

Issue / Cause	Solution / Workaround
Ollama not running or not installed	Start Ollama service (`ollama serve`) or reinstall from ollama.com/download
Model not found (e.g., llama2 missing)	Run `ollama pull llama2` or any required model