**LLM Chat App – Learning Log**

===========================

Platform: Local system with Ollama + Python (Streamlit + FastAPI)

Topics Learned

--------------------

1. Toolchain Setup

* Installed and configured:
* Python 3.13
* uvicorn + FastAPI
* streamlit
* Ollama with local models (mistral, llama2)
* Ran local Ollama server via: Ollama serve

2. Backend API Development (FastAPI)

Created a /chat endpoint to:

* Accept user prompt
* Optionally augment with time/headlines/weather (real-time context)
* Call local LLM using subprocess.run("ollama run ...")
* Return cleaned model output as JSON

Learned:

* How to safely run subprocesses
* Timeout handling
* Shell encoding issues and fix using text=True, encoding='utf-8'

3. Frontend Chat UI (Streamlit)

* Built a real-time chat interface:
* Persistent message history using st.session\_state
* st.text\_input for prompt submission
* POST to FastAPI and display response

Enhanced UI with:

* Color-coded chat bubbles
* Gradient headers and chat alignment
* Left/right message alignment using inline HTML + CSS

🛠 Issues Encountered & Solutions

-----------------------------------------------

| **Issue** | **Solution** |
| --- | --- |
| uvicorn not recognized | Installed via pip and used 'python -m uvicorn app:app --reload |
| Streamlit showed “file not found” | Ensured correct script path |
| Ollama response delay | Diagnosed prompt/model delays; added timeout logic |
| Bot replies not visible in Streamlit | Fixed rendering with inline HTML |
| Duplicate user messages | Moved append after backend response |
| Terminal encoding error | Used encoding='utf-8' |
| Model failed due to low RAM | Switched to lighter model (llama2) |
| st.experimental\_rerun() deprecated | Removed; used state to trigger rerun |

UI Enhancements Done

--------------------------------

* Bubble-style messages (user = blue, bot = purple)
* Bot and user messages right-aligned
* Consistent spacing and padding
* Clean message grouping
* Optional styling improvements

Improvements Considered

-----------------------------------

* History truncation (for speed)
* Scroll to bottom (implemented)
* Future ideas: avatars, timestamps, themes

Outcome

------------

We now have a fully functional, styled local chat app that:

* Runs offline with local LLMs (via Ollama)
* Uses FastAPI backend
* Live Streamlit UI with history
* Modern, responsive, extendable

Key Learnings Summary

---------------------------------

Streamlit - UI layout, markdown/HTML rendering, state handling

FastAPI - API routing, subprocess integration, JSON response

Ollama - Running LLMs via CLI, model selection

Debugging - Timeout, encoding, duplication fix

UI/UX - Clean alignment, styled chat bubbles