# **Global News Topic Tracker**

**Documentation & Learning Log**

## **1. Project Overview**

**Objective**:  
Build a FastAPI backend that:

1. Scrapes global news headlines from Google News.
2. Uses Ollama (LLM) to summarize trends.

**Tech Stack**:

* Python (FastAPI, BeautifulSoup, Requests)
* Ollama (LLM locally via llama3/mistral)
* Uvicorn (ASGI server)

## **2. Learning Log**

### **Topics Learned**

| **Topic** | **Key Takeaways** |
| --- | --- |
| **FastAPI Basics** | Created REST endpoints, used pydantic for request validation. |
| **Web Scraping** | Used BeautifulSoup to parse HTML, learned CSS selectors (a.DY5T1d for Google News). |
| **Logging** | Configured Python’s logging module to debug issues (critical for async apps). |
| **Ollama Integration** | Sent prompts to a local LLM via API (http://localhost:11434/api/generate). |
| **Error Handling** | Added timeouts, headers, and exception handling for robustness. |

### **Issues Faced & Solutions**

| **Issue** | **Solution** |
| --- | --- |
| **Logs not appearing** | Moved logging.basicConfig() before FastAPI init. Used --log-level debug with Uvicorn. |
| **Scraping hung indefinitely** | Added timeout=10 and User-Agent headers to requests.get(). |
| **Ollama connection refused** | Verified Ollama was running (ollama serve). |
| **Duplicate logs** | Checked if not logger.handlers before adding handlers. |

### **Workflow**

1. **Streamlit frontend** sends request → FastAPI /scrape endpoint.
2. **FastAPI** scrapes Google News for trending headlines.
3. **FastAPI** sends news content → Ollama /api/generate for summarization.
4. **Ollama** processes and returns a concise summary.
5. **Frontend** displays topics and summaries to the user.