Chatbot with Streamlit and RAG

This project is a Streamlit-based chatbot that answers Apache Spark-related queries using Retrieval-Augmented Generation (RAG). It processes a PDF (spark_definitive_guide_notes.pdf) to provide context-aware answers, leveraging LangChain, ChatGroq (xAI's LLM), and a ChromaDB vector store with HuggingFace embeddings.

Features

Interactive UI: Built with Streamlit, displaying chat history and user/assistant messages.

Grok Integration: Uses xAI's llama3-8b-8192 model via ChatGroq for precise answers.

RAG Pipeline: Retrieves relevant PDF content using a ChromaDB vector store and all-MiniLM-L6-v2 embeddings.

Persistent Storage: Saves the vector store to disk for faster subsequent loads.

Error Handling: Robust checks for API keys, PDF existence, and processing errors.

Prerequisites

Python: 3.8 or higher.

GROQ API Key: Obtain from xAI's API portal.

PDF File: spark_definitive_guide_notes.pdf (notes from The Definitive Guide to Apache Spark).

Hardware: At least 8GB RAM; GPU optional for faster embeddings.

Setu

Create a Virtual Environment:

python -m venv venv

source venv/bin/activate # On Windows: venv\Scripts\activate

install Dependencies: Save the following to requirements.txt:

streamlit==1.31.0

langchain==0.2.0

langchain-groq==0.1.0

langchain-community==0.2.0

sentence-transformers==2.7.0

PyPDF2==3.0.1

chromadb==0.5.0

Install:

pip install -r requirements.txt

Set GROQ API Key:

export GROQ_API_KEY="your-api-key-here" # On Windows: set GROQ_API_KEY=your-api-key-here

Add PDF: Place spark_definitive_guide_notes.pdf in the project root directory.

Usage

Run the App:

streamlit run phase_3.py

The app opens at http://localhost:8501 in your browser.

The first run may take time to build the vector store (saved to ./chroma_db for reuse).

Interact with the Chatbot:

Enter a prompt (e.g., "explain client mode") in the chat input.

The chatbot retrieves relevant PDF content and generates a response using ChatGroq.

File Structure

your-repo-name/

├── phase_3.py # Main Streamlit app with RAG pipeline

├── spark_definitive_guide_notes.pdf # PDF for RAG (not included in repo)

├── requirements.txt # Python dependencies

├── chroma_db/ # Persistent vector store (auto-generated)

└── README.md # This documentation

Example

Prompt: "explain client mode" Response (approximate):

In Apache Spark, client mode is a deployment mode where the Spark driver runs on the client machine that submits the application, while executors run on the cluster. The driver manages task scheduling and coordination locally, making it ideal for interactive or debugging scenarios but less suitable for production.

Troubleshooting

Slow Startup: The vector store takes time to build initially. Subsequent runs are faster due to persistence in ./chroma_db.

PDF Not Found: Ensure spark_definitive_guide_notes.pdf is in the root directory.

API Key Error: Verify GROQ_API_KEY is set (echo $GROQ_API_KEY).

Port Conflict: If localhost:8501 is busy, use:

streamlit run phase_3.py --server.port 8502

Memory Issues: Close other applications or use a smaller chunk_size in phase_3.py.

License

Contributing

Feel free to open issues or submit pull requests on GitHub!