Project Motivation & Overview

Why I Made This

In today's information-rich world, valuable knowledge is often locked away in diverse document formats—PDFs, Word files, images, spreadsheets, and databases. Manually searching and extracting insights from these sources is time-consuming and inefficient. I wanted to create a solution that empowers anyone to instantly ask questions and get answers from their own documents, regardless of file type, using the latest advances in AI.

Motivation

I am passionate about making AI accessible and useful for real-world problems. This project demonstrates how modern AI can bridge the gap between unstructured data and actionable knowledge. My goal is to help students, professionals, and organizations unlock the value hidden in their documents—quickly, securely, and intuitively. I hope this inspires others to explore the intersection of AI, automation, and information retrieval to solve meaningful challenges.

Personal Reflection

I am currently on a learning journey and am always eager to expand my knowledge. While working on this project, I encountered many technologies and concepts that were completely new to me. I didn't know a lot about the tools and frameworks I used, but I was determined to learn, and I implemented them step by step as I progressed through the task.

Throughout this project, I realized the value of hands-on learning and how much can be achieved with curiosity and persistence. However, I also discovered that having a mentor would make a huge difference. I genuinely feel the need for guidance and support from someone experienced, and I am actively seeking a mentor to help me grow further in this field.

How I Made This

I built a Retrieval-Augmented Generation (RAG) system using Python, FastAPI, and Streamlit. The backend leverages FastAPI for robust, asynchronous APIs, while the frontend uses Streamlit for a simple, interactive user experience. I implemented document ingestion pipelines that extract and preprocess content from PDFs, DOCX, TXT, images (with OCR), CSV, and SQLite databases. All extracted content is embedded using Google Gemini and stored in a FAISS vector database for efficient similarity search. When a user asks a question (even with an image), the system retrieves relevant context and uses a powerful LLM to generate accurate, context-aware answers. The entire solution is containerized with Docker for easy deployment

Technologies Used

- Python: The main programming language for backend and scripting.
- FastAPI: For building asynchronous, high-performance APIs.
- Streamlit: For creating a simple and interactive web frontend.
- FAISS: For efficient vector search and similarity retrieval.
- LangChain: For chaining and orchestrating LLM-based workflows.
- Google Gemini LLM: For advanced question answering and embeddings.
- pytesseract: For OCR and extracting text from images.
- pdfplumber, python-docx, pandas, sqlite3: For extracting and processing content from various document types.
- Docker: For containerizing the application and simplifying deployment

This project has been a great learning experience, and I am excited to continue exploring and building with new technologies!