DOCUMENT QA SYSTEM

Description:

This project implements a document-based Question-Answering (QA) system that allows users to upload documents (PDF, TXT, or Excel) and ask questions about the content. The system processes the document, extracts relevant text, and enables users to query the information using an AI model. It also stores past queries in an SQLite database for persistent history.

The backend uses LangChain for document processing, Hugging Face transformers for embeddings and text generation, FAISS for vector search, and Streamlit for the frontend UI.

Data: Stock Data. Click here Link

Key Features:

Multi-Format Document Processing: Supports PDFs, TXT files, and Excel sheets, extracting textual content for analysis.

AI-Powered Q&A System: Utilizes FAISS vector search and Hugging Face transformers to efficiently retrieve and generate responses.

Persistent Query History: Stores user queries in an SQLite database, allowing users to view and revisit past questions.

Data Visualization for Excel Files: If structured data is uploaded, users can visualize it within the Streamlit interface.

User-Friendly Web Interface: A clean and intuitive UI built using Streamlit for easy document uploads and question interactions.

Scalability & Deployment: Designed for local execution and potential cloud deployment for broader accessibility.

Technical Stack:

Frontend: Streamlit

Backend Processing: LangChain for document loading and QA

Embedding & Retrieval: FAISS + Hugging Face's MiniLM embeddings

LLM Model: Transformer-based text generation (e.g., T5, OPT, or a locally hosted model)

Database: SQLite for storing user query history

Deliverables:

Web Application – A Streamlit-based UI to upload documents, process them, and allow interactive Q&A.

Document Processing Pipeline – Supports PDFs, text files, and Excel sheets, converting them into structured formats for retrieval.

QA System – Uses FAISS-based vector search with embeddings and a Hugging Face transformer model for text generation.

Persistent History Feature – Stores users' queries in an SQLite database for future reference.

Deployment Guide – Documentation for setting up and running the application locally or on a server.

Project Plan with Milestones:

Plan	**Task**	**Timeline**	**Status**
Research Setup & Document Processing & Query Storage	Identify necessary libraries and set up a basic Streamlit app. Read data from the API and display the data in the webpage and store the questions in DB	Week 3**(02/03 - 02/09)**	Completed
QA System Integration	Develop and integrate the QA system using FAISS and Hugging Face transformers.	Week 4,5**(02/10 - 02/23)**	In-Progress
UI Enhancement & Visualization	Improve UI and add optional data visualization for data	Week 6**(02/24 - 03/02)**	Not yet started
Testing & Optimization	Test performance, optimize retrieval accuracy, and handle edge cases.	Week 7,8**(03/03 - 03/16)**	Not yet started
ML- OPS Deployment	Deploying the pipeline in ML-Ops	Week 9,10**(03/31 – 04/14)**	Not yet started
ML-OPS evaluation	Evaluating the Pipeline	Week 11**(04/15 - 04/21)**	Not yet started
ML-OPS retraining	Retraining the pipeline	Week 12,13**(04/22 - 05/04)**	Not yet started
Final Report	Write user documentation.	Week 14**(05/05 – 05/08)**	Not yet started

Team Members:

Dinesh Kothandaraman

Sri Harshetha Amaravadi

Damodar Reddy Chirapureddy

Veda Samohitha Chaganti