Implementation of AI-Powered Document Query System
RickQuery – AI-Powered PDF Q&A

Project Overview

This was my first attempt at integrating AI into a project. The objective was to build a simple document query system that allows users to upload PDFs and ask AI-powered questions to retrieve relevant answers. The project was developed using Python and Streamlit with AI-backed natural language processing capabilities.

Getting Started

Initially, I had little idea about where to begin, so I referred to multiple YouTube tutorials on document query systems. The first step was setting up a suitable development environment using Anaconda or Miniconda. After that, I created a dedicated Python environment to manage dependencies efficiently.

Development Process

1. **Setting Up Dependencies**: I installed all necessary libraries, including LangChain, FAISS, Hugging Face, and Streamlit.
2. **Document Processing**: Implemented PDF extraction using PyPDFLoader and split text into smaller chunks for efficient retrieval.
3. **Embedding and Vector Storage**: Used HuggingFaceEmbeddings for text embeddings and FAISS for similarity search.
4. **AI Model Integration**: Initially, I attempted to use OpenAI's API but faced limitations, so I switched to Hugging Face's model.
5. **UI Implementation**: Built a simple UI using Streamlit, which turned out to be the easiest part of the process.
6. **Error Handling & Debugging**: Implemented exception handling to pinpoint errors efficiently, mostly related to LangChain library installations.

Challenges Faced & Solutions

- **Debugging Issues**: I struggled to identify errors in my code. To resolve this, I added exception handling in key sections to trace the source of issues.
- **OpenAI API Key Limitations**: Initially, I used OpenAI's API but encountered quota restrictions. I researched alternatives and integrated Hugging Face's models instead.
- **Library Installation Errors**: Many issues arose from improperly installed LangChain dependencies, which I resolved by carefully reinstalling and verifying the installations.

Tools & Technologies Used

Programming Language:

- Python

Web Application Framework:

- Streamlit (for UI)

Document Processing:

- PyPDFLoader (LangChain) for PDF text extraction
- CharacterTextSplitter for text segmentation

Embedding & Vector Search:

- HuggingFaceEmbeddings for creating embeddings
- FAISS for efficient similarity search

Machine Learning & NLP:

- Transformers (Hugging Face) for Question Answering models
- Pre-trained model: "deepset/roberta-base-squad2"
- PyTorch (Torch) for machine learning operations

Other Key Libraries & Features:

- Python's OS module for file handling
- Caching with @st.cache_resource
- Exception handling for debugging
- Pipeline-based machine learning inference