RAG-Powered PDF Q&A System

Overview

This Django project uses Retrieval-Augmented Generation (RAG) to answer user questions based on the content of a PDF. It uses FastAPI for the backend, PyMuPDF for PDF parsing, FAISS for vector search, and SentenceTransformers for embeddings. Answers are generated using the Groq API (LLaMA3).

Technologies Used

Django - Web framework

FastAPI - Lightweight API server

PyMuPDF - PDF reading

SentenceTransformers - Embedding model

FAISS - Vector index for similarity search

NumPy - Matrix ops

Requests - API calling

Groq API - LLM backend (LLaMA3)

Workflow

1. Load PDF using fitz.

2. Split text into overlapping chunks.

3. Embed chunks with SentenceTransformers.

4. Store in FAISS index.

5. Accept a question at /ask.

6. Search relevant chunks.

7. Combine context and query LLaMA3 via Groq API.

API Endpoint

POST /ask

Request:

{ "question": "Who is mentioned in the family tree?" }

Response:

{ "question": "Who is mentioned in the family tree?", "answer": "The family tree includes ..." }

Requirements.txt

fastapi

pydantic

PyMuPDF

numpy

sentence-transformers

faiss-cpu

requests

uvicorn

Run the Project

Run server:

uvicorn main:app --reload

API docs:

http://127.0.0.1:8000/docs