Context-Aware Study Assistant

Project Title: Context-Aware Study Assistant
Name: Asit Ghosh
Date: 17-07-2025

---

1. Challenge Description

Students often struggle to revise large volumes of notes before exams, and existing study tools lack personalisation or contextual understanding. The challenge is to build an intelligent assistant that can:

- Read and understand personal study notes (PDFs)
- Answer student queries using only those notes
- Provide reliable, cited responses

---

2. Proposed Solution

The proposed solution is an **AI-powered Context-Aware Study Assistant** that uses Retrieval-Augmented Generation (RAG). It retrieves the most relevant information from the user's uploaded PDFs using FAISS and uses GPT-4 to answer questions contextually.

Key Components:

- PDF ingestion and text parsing (PyMuPDF)
- Text chunking and embedding (OpenAI Embeddings)
- FAISS
- LLM-based question answering (GPT-4)
- Metadata tagging for source tracking

---

3. Implementation Details

Tools & Frameworks:

- **Python 3.10+**
- **OpenAI GPT-4** (via API)
- **text-embedding-3-small** model
- **FAISS** (IndexFlatL2)
- **PyMuPDF** (PDF text extraction)

- **LangChain (optional)** for chunking
- **dotenv** for API key management

Workflow Steps:

1. User places one or more PDFs in the data/ folder.
2. Text is extracted, chunked, and embedded.
3. Embeddings are stored in a FAISS vector index.
4. When a question is asked, the system retrieves top-k relevant chunks.
5. A prompt is constructed and sent to GPT-4.
6. GPT-4 returns an answer grounded in the provided context.

---

4. Results & Challenges

Results:

- Successfully answered questions from multi-document PDF collections.
- Delivered fast and relevant answers using local vector search.
- Maintained traceability by showing document and chunk metadata.

Challenges:

- Chunking needed tuning to avoid broken sentences.
- FAISS required consistent dimensionality in embeddings.
- Trimming context for GPT-4 input limits without losing meaning.
- API key security during development.

---

5. Recommendations

- Add Streamlit web interface for non-technical users
- Add caching to reuse embeddings instead of recalculating
- Implement local LLM fallback using Mistral or LLaMA
- Support .txt, .docx, .md files along with PDFs
- Store user sessions and Q&A logs for future revision
- Highlight referenced sections in retrieved documents

Outcome & Learnings

Through this project, I developed a deeper understanding of:

- Retrieval-Augmented Generation using FAISS and GPT-4
- Embedding workflows and metadata tracking
- Optimising prompt engineering for educational use cases