DRAFT PROJECT REPORT

Name : *Husnain Ali*

Roll.Num: TN/IN02/AIML/016

Date: *[20/8/2025]*

1.Chat with Multiple PDFs – Project Report

---

2. Introduction

This project is an AI-powered Streamlit web application that allows users to upload and interact with multiple PDF documents. It leverages **LangChain** for natural language processing, **Google Generative AI** for embeddings and conversational responses, and **FAISS** as a vector database for semantic search.

The main objective of this project is to provide an easy-to-use interface for document-based Question Answering (QA), where users can query uploaded PDFs and get accurate answers in natural language.

---

3. Features

- 📌 Upload and process multiple PDF files at once.
- 📌 Extract and split text into manageable chunks.
- 📌 Convert text into embeddings using Google Generative AI.
- 📌 Store and search embeddings efficiently with FAISS.
- 📌 Ask natural language questions and get AI-powered answers.
- 📌 Maintain a conversation history.
- 📌 Download the chat history as a CSV file.

---

4. System Requirements

- **Python Version:** 3.9+
- **Frameworks & Libraries:**
    - Streamlit (for web UI)
    - LangChain (LLM integration)

- Google Generative AI (embeddings & chat model)
- FAISS (vector database for semantic search)
- PyPDF2 (PDF text extraction)
- Pandas (data handling)

---

5. Installation

Follow these steps to set up the project locally:

```
# 1. Clone the repository
git clone https://github.com/yourusername/chat-with-multiple-pdfs.git
cd chat-with-multiple-pdfs

# 2. Create virtual environment (optional but recommended)
python -m venv venv
source venv/bin/activate   # On Linux/Mac
venv\Scripts\activate      # On Windows

# 3. Install dependencies
pip install -r requirements.txt

# 4. Run the Streamlit app
streamlit run app.py
```

---

6. Usage

1. Open the app in the browser (default: `http://localhost:8501`).
2. Enter your **Google API key** in the sidebar.
3. Upload one or more PDF documents.
4. Type your question in the input box.
5. The AI will search and generate an answer based on the uploaded PDFs.
6. Download the chat history as a **CSV file** if needed.

---

7. Architecture

- **Frontend:** Streamlit (for interactive web UI)
- **Backend Processing:**
  - PyPDF2 → Extract text from PDFs
  - LangChain → Chunking & processing
  - Google Generative AI → Embeddings + Q/A
  - FAISS → Vector storage & similarity search
- **Output:** Conversational answers + downloadable chat history

DRAFT PROJECT REPORT

---

8. Future Improvements

- Add support for other document types (Word, Excel, TXT).
- Enable multi-language question answering.
- Deploy the app on cloud platforms (e.g., Streamlit Cloud, HuggingFace Spaces).
- Add authentication for secure usage.

---

9. Conclusion

This project demonstrates how to build an AI-powered, document-based Q&A system using **Streamlit, LangChain, Google Generative AI, and FAISS**. It provides a practical way to query PDF documents interactively and can be extended for enterprise use cases like research paper analysis, legal document review, and business report summarization.