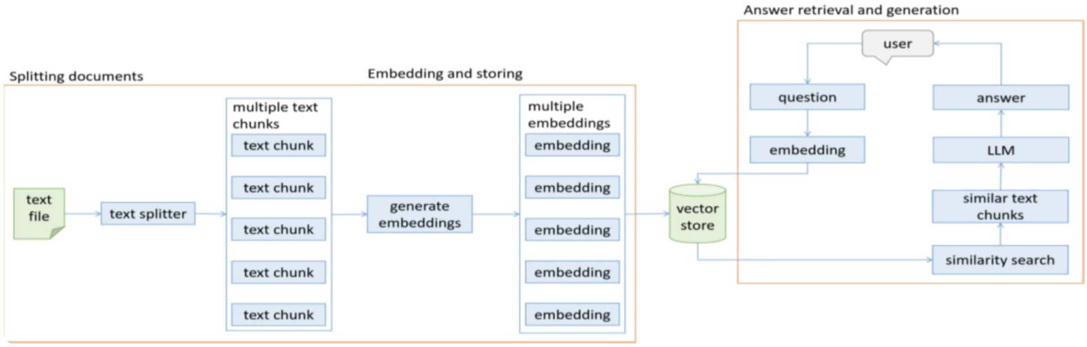
Summary of the overall project

a)Overall approach that you used

I basically used **RAG(Retrieval Augmentation Generation).** It is a method for augmenting large language models (LLMs) such as GPT-3 by feeding them more context while they are being generated.In this users can upload documents, and the chatbot can answer questions by referring to those documents.



The system converts the question text into embeddings and utilizes them to search and retrieve similar text chunks from the vector store. Subsequently, it sends these text chunks to the LLM to generate sentences for answering the user's question.

The embeddings of the text chunks that I used was **Hugging face** emebbings and I used **Pinecone** as the vector data base to store vector embeddings. Used **Mistral-7B-Instruct-v0.3** the LLM to generate answers related to the embeddings.

b)Frameworks/libraries/tools used along with where they were used

- Python
- LangChain
- Flask
- Mistral
- Pinecone

c)What are the problems that you faced and how did you overcome them?

I faced problem while creating the chat history of the chats so that it should give the answer related to the previous chat history.

I overcomed it by creating a another prompt for generating the question first then another prompt for answering the query , and then created whole in a chain with the help of **Langchain** library.To story it in a memory I used **ConversationBufferMemory.**

I resolved the whole problem by reading the Langchain's documentation.

d)**Future scope of this chatbot i.e. what more we can do to it, like adding features, etc.**

Yeah we can add interesting features in our chatbot like

- Making in multilangual so that user in any language can chat with it and find answers quickly.
- Improving the retrieving mechanism, by using semantic searching so that user can get answers faster
- Integrate image and video processing to allow the chatbot to understand and respond to visual inputs.
- Deploy the chatbot on scalable cloud infrastructure to handle increasing user loads.
- Implement self-learning capabilities so the chatbot can improve its responses over time based on user interactions.
- Can introduce chat in voice feature to find the answer to query.
- Can bring agents there for live chat when user’s query not resolved.