**Brance AI Applied Researcher Intern Task**

Name: Tanish Khandelwal

LinkedIn Profile: https://www.linkedin.com/in/tknishh/

Date Challenge Received: 7th July, 2023

Date Solution Delivered: 9th July, 2023

1. Problem Statement

The Task assigned to me was to create a chatbot using RAG model that can retrieve the knowledge from the input document and answer questions.

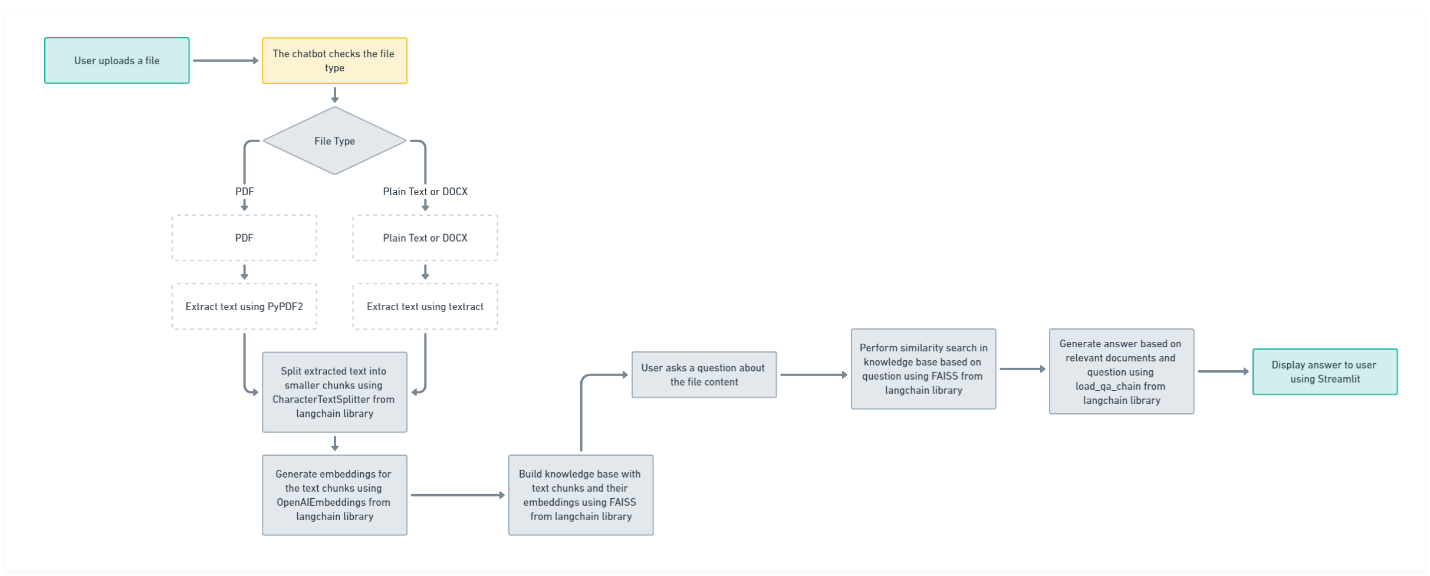
2. Approach

The approach I made for this task consisted of steps in following manner,   
  
User -> File Upload -> Text Extraction -> Chunking -> Embeddings -> Knowledge Base -> User Query -> Similarity Search -> Question-Answering -> Answer Display

First of Extracting the text out of given document and then dividing them into various chunks of chunk size 1000, then exploiting the OpenAIEmbeddings we embed our extracted text, then we create a knowledge base using FAISS. The process up until now roughly resemble Retrieval Phase in a RAG model. After getting the User query or question from User we use it for similarity search within knowledge base and generating appropriate answer for the user query.

Github: <https://github.com/tknishh/FileWise>

3. Solution



\*\*Please increase size of diagram according to your convenience or refer to the one in github repo.

The diagram illustrates the workflow of a chatbot for file-based question answering. Users upload files, which are processed based on their type. PDF files are extracted using PyPDF2, while plain text and DOCX files are extracted using textract. The text is then split into smaller chunks and converted into embeddings. A knowledge base is built using FAISS. When users ask questions, similarity search is performed to retrieve relevant documents, and load\_qa\_chain generates answers. The answers are displayed to users using Streamlit.

4. Future Scope

* The presented solution could’ve been improved by using sample questions as an input to further improve answers provided by the system.
* It can also be improved by using more advance models like DPR for passage re-ranking which was introduced by ofc meta, that can help with retrieval process.
* We can look into more generation techniques to that are more domain specific and provides better results.

**Note:** In the test folder of the repository there are two on-going files, 1. App2.py which adds bonus feature 2. rag.ipynb which contains haystack’s RAG model implementation.