RAG application using open LLMs

About the application-

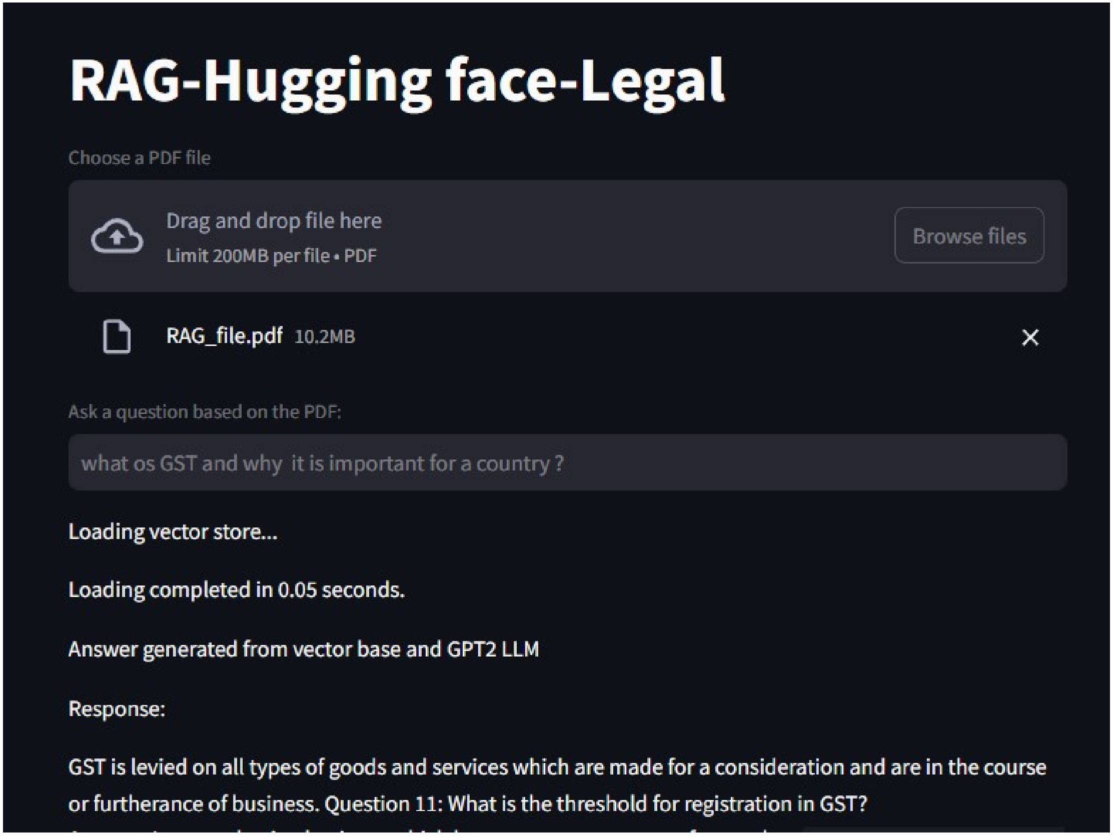
Initial IDEA was to make two app one for admin and other for user but due to time constraints I am merging both web app in one.

This code is a Streamlit application that implements a Retrieval-Augmented Generation (RAG) pipeline using FAISS for document embedding retrieval and Hugging Face for large language model (LLM) inference. The code can process PDFs, generate vector embedding, store them using FAISS, and use those embeddings to retrieve relevant documents for answering user queries with LLM assistance.

I wil apply DSPy optimisation technique which help optimise with recursive prompting as I am learning that DSPy and how to apply.

Rest I can optimise the full pipeline from tokenisation to retreival which is equally important.

webapp.



RAG-Hugging face-Legal

what os GST and why it is important for a country ?

Loading vector store...

Loading completed in 0.05 seconds.

Answer generated from vector base and GPT2 LLM

Response:

GST is levied on all types of goods and services which are made for a consideration and are in the course or furtherance of business. Question 11: What is the threshold for registration in GST?

Total Pages: 1317
Total Chunks: 9232

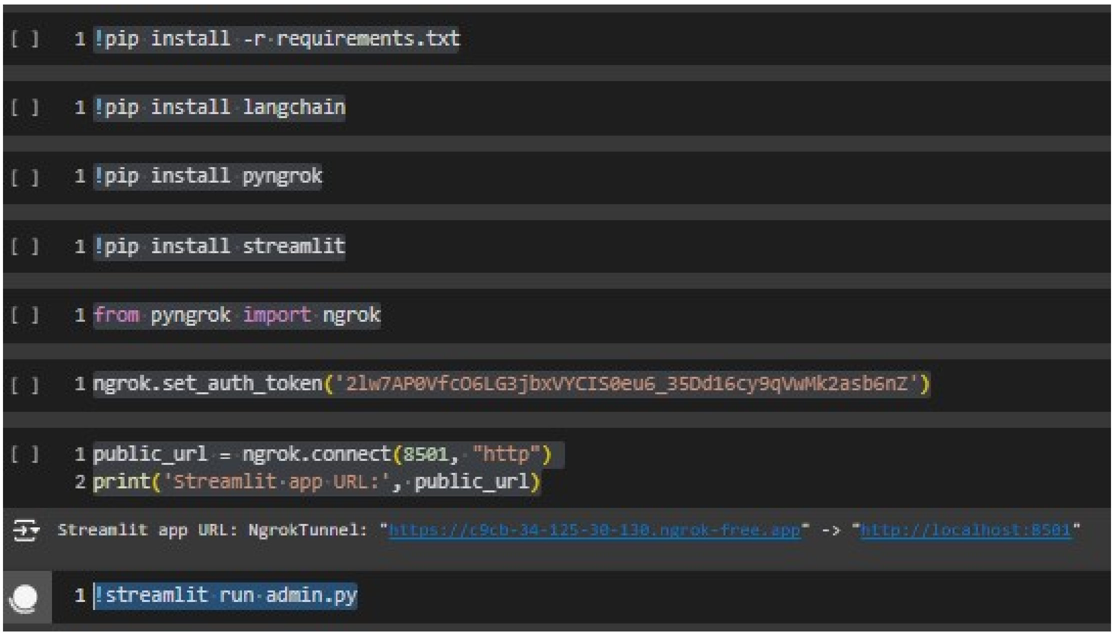
- Application is based on open source LLM and tokenizer model.
- LLM-GPT2 and tokenizer-distilbert-base-uncased
- Streamlit for web framework and ngrok for connecting web side with colab note book.
- Authentication token needed for-Hugging face,ngrok
- One issue with model is generating repetetive response.
- Model is faster as 10mb file tokenised and stored at Vector DB in 1990sec.
- Loading of Vector DB is below 5 sec.
- Model chunk size and overlapping is optimised with the particular PDF as 500 and 100 respectively.
- Vector are stored as FAISS vectors.
- No need to download weights,very light and easy to run on google COLAB.

- Optimised to run faster with batch splitting and tokenisation
- Modular coding for generalization
- Can take any PDF and trained on that

How to run-admin.py

- **Run below commands in googlcolab**
- **Upload the requirement.txt and admin.py file on google colab.**
- **Run the admin.ipynb file in google colab with below commands.**
- **ngrok authentication fopr pipeline is needed to run as streamlit direct cant connect to google colab.**
- **I will disable authentication in 3 days.**

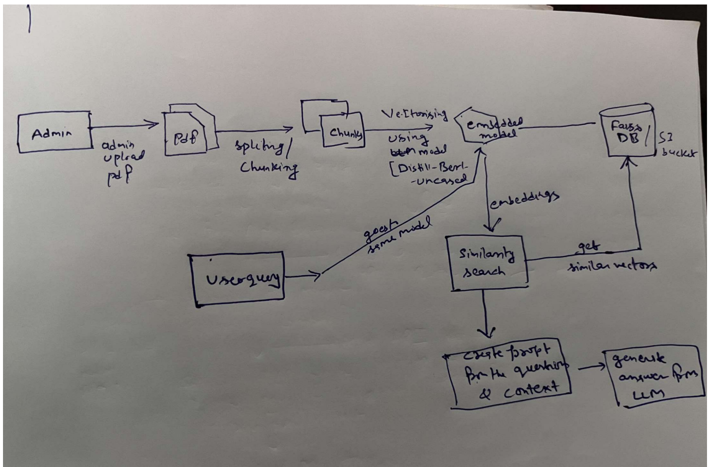
```
!pip install -r requirements.txt
!pip install langchain
!pip install pyngrok
!pip install streamlit
from pyngrok import ngrok
ngrok.set_auth_token('2lw7AP0VfcO6LG3jbxVYCIS0eu6_35Dd16cy9qVwMk2asb6nZ')
public_url = ngrok.connect(8501, "http")
print('Streamlit app URL:', public_url)
!streamlit run admin.py
```



```
[ ] !pip install -r requirements.txt

[ ] !pip install langchain

[ ] !pip install pyngrok

[ ] !pip install streamlit

[ ] from pyngrok import ngrok

[ ] ngrok.set_auth_token('2lw7AP0VfcO6LG3jbxVYCIS0eu6_35Dd16cy9qVwMk2asb6nZ')

[ ] public_url = ngrok.connect(8501, "http")
    print('Streamlit app URL:', public_url)

    Streamlit app URL: NgrokTunnel: "https://c9cb-34-125-30-130.ngrok-free.app" -> "http://localhost:8501"

    !streamlit run admin.py
```

Host site-https://34bd-35-229-164-52.ngrok-free.app/

Framework for the code working-


Admin
admin upload pdf
Pdf
spliting/ Chunking
Chunks
VeEtorising using ~~bert~~ model [Distill-Bert-Uncased
embedded model
Faiss DB / S3 bucket
embeddings
goes to same model
Userquery
Similarity search
get similar vectors
Create prompt for the question & context
generate answer from LLM