```
!pip install -q --upgrade "langchain>=0.0.300" langchain-openai langchain_community langchain_huggingface langchain_google_genai sente
```

Show hidden output

Store API Keys for various models in Colab(Secrets) and fetch

```
from google.colab import userdata
import os
OpenAI_API_KEY = os.environ["OPENAI_API_KEY"] = userdata.get('OpenAI_API_KEY')
GEMINI_API_KEY = os.environ["GOOGLE_API_KEY"] = userdata.get('GEMINI_API_KEY')
```

First, let's upload your PDF file to the Colab environment. You can use the files.upload() function from google.colab to do this. After running the cell, a button will appear allowing you to select and upload your file.

```
from google.colab import files
uploaded = files.upload()
# Get the name of the uploaded file
for fn in uploaded.keys():
  print(f'User uploaded file "{fn}"')
  pdf_file_path = fn
```

Choose files TCS_interview.pdf

TCS_interview.pdf(application/pdf) - 626449 bytes, last modified: 12/11/2024 - 100% done

```
Saving TCS_interview.pdf to TCS_interview.pdf
User uploaded file "TCS_interview.pdf"
```

Document Loaders

Now that the PDF file is uploaded, we can use `PyPDFLoader` from `langchain_community.document_loaders` to load its content.

```
from langchain_community.document_loaders import PyPDFLoader
loader = PyPDFLoader(pdf_file_path)
documents = loader.load() # loads into memory

# print page_content
print(documents[2].page_content)
```

```
TCS Interview Questions
TCS Technical Interview Questions:
Freshers and Experienced (.....Continued)
19.   Enlist the differences between AWT and Swing in Java.
20.   Explain memory leak in C++. How can you avoid it?
21.   What is the meaning of a command line argument in C?
22.   What do you mean by concurrency control?
23.   What do you mean by SQL Correlated Subqueries?
24.   What is a checkpoint in a database management system, and when does it
eventuate?
25.   In a database management system, what are the two integrity rules?
26.   What exactly are macros? What are the benefits and drawbacks?
27.   What is the difference between the Java Development Kit (JDK), Java Runtime
Environment (JRE), and Java Virtual Machine (JVM)?
28.   How will you swap two numbers without the use of a third variable?
29.   Write the code to reverse a given number using Command Line Arguments.
30.   What is meant by Cycle Stealing?
31.   What are the two concepts of swapping in the context of OS? How does
swapping help in better memory management?
32.   What is the RR Scheduling Algorithm in OS?
33.   What are interrupts?
TCS Interview Preparation
34.   Interview Preparation Tips
Frequently Asked Questions
35.   Does TCS Nqt have coding questions?
Page 2 © Copyright by Interviewbit
```

Splitting the documents(Chunking)

chunks - small info having meaning

```
from langchain_text_splitters import RecursiveCharacterTextSplitter
# initialise the text splitter
text_splitter = RecursiveCharacterTextSplitter(
    chunk_size = 2000,
    chunk_overlap = 300   #smaller the splits, can understand context better
)
# split the documents(pages in pdf) using .split_documents
chunks = text_splitter.split_documents(documents)

# print([chunk.page_content for chunk in chunks])
print("Chunks created:", len(chunks))
print("Sample chunk metadata:",chunks[0].metadata)
print(chunks[5].page_content[:400])
```

```
Chunks created: 37
Sample chunk metadata: {'producer': 'Skia/PDF m85', 'creator': 'Chromium', 'creationdate': '2023-11-11T07:45:01+00:00', 'moddate': '202
TCS Interview Questions
TCS Recruitment Process
1.   Interview Process
TCS is an excellent location to begin your career as a new employee. It provides a
fantastic workplace as well as a welcoming setting with a good ambiance conducive
to individual and company progress. TCS holds a mass recruiting procedure every
year to find applicants for the position of  Soware Engineer. This article not only
```

Embedding Model - Convert chunks into Numerical Vectors

Embedding models aim to capture the "meaning" of text

```
from langchain_huggingface import HuggingFaceEmbeddings
```

```
emb_model_name = "all-MiniLM-L6-v2"  # free model
embeddings = HuggingFaceEmbeddings(model_name=emb_model_name)

# test embeddings
test_emb = embeddings.embed_documents([chunks[0].page_content[:200]])
print("Embedding vector length:",len(test_emb[0]))
```

Show hidden output

Store the embedding chunks in Vector DB as vectors

Define Chroma DB as Vector store

```
# using Chroma as vector db
from langchain.vectorstores import Chroma
db = Chroma.from_documents(
                documents = chunks,
                embedding = embeddings
    )
```

```
# can count the no.of splits embedded and stored in db
# print(db._collection.count())
```

```
# get the id's of that vectors in db
# print(db._collection.get())
```

```
# # we can see the documents and embeddings from particular id
# print(db._collection.get(ids = ['d6169d55-5fbf-4604-99e3-d9f55f557923'],include=['documents','embeddings']))
```

RAG Pipeline(Retrieval Chain)

1.Retrieval: My vectore db act like "Retriever"

2.Augementation (Query + Context)-> ChatPromptTemplate(consists of context & question)

3.Generation : LLM Creation

```
# 1.Vector store as retriever,creating a retriever and format the chunks
retriever = db.as_retriever(
    search_type="similarity",
    search_kwargs={"k": 6}     # k = results to return to chain;
)
```

```
# 2. getting default prompt from hub
# from langchain import hub
# prompt = hub.pull("rlm/rag-prompt")
# print(prompt.messages[0].prompt)

# OR see reference chat prompt templat
from langchain.prompts import ChatPromptTemplate
prompt = ChatPromptTemplate.from_template(
          """You are an assistant for question-answering tasks.
          Use the following pieces of retrieved context to answer
          the question. If you don't know the answer, just say
          that you don't know. Use three sentences maximum
          and keep the answer concise.

          Question: {question}
          Context: {context}
          Answer:
```

```
            """
        )
```

```
# # import OpenAI(if you've premium) or ChatGoogleGenerativeAI(for free)
# 3. LLM creation
from langchain_google_genai import ChatGoogleGenerativeAI
llm = ChatGoogleGenerativeAI(
    model="gemini-2.5-flash",
    google_api_key=GEMINI_API_KEY
)
```

```
# # 3. LLM Creation
# from langchain_openai import ChatOpenAI
# llm = ChatOpenAI(model = "gpt-4o-mini",api_key = OpenAI_API_KEY)
```

RAG Chain

```
# context -> the output (chunks or no.of splits) from the retriever should
# be formatted in "relevant" manner and Join them
def format_chunks(chunks):
  return "\n".join(chunk.page_content for chunk in chunks)
```

| (pipe)-> symbol is an operator in LCEL(LangChain Expression Language) used to connect different AI building blocks, such as prompts, models, and parsers, into a chain

```
from langchain_core.runnables import RunnablePassthrough # for sending question as it is to chain
from langchain_core.output_parsers import StrOutputParser

# 💥💥💥
rag_chain = (
    {
        # CONTEXT from retriever passed to -> format_docs function to fromat
        "context": retriever | format_chunks,
        "question": RunnablePassthrough()
    }
    | prompt # pass the question and context to prompt
    | llm    # pass prompt to LLM
    | StrOutputParser() # send the generated output in String format
```

```
)
```

Test the RAG Chain-> using .invoke()

```
rag_chain.invoke("What is Clustered Indexing in SQL?") # Question in PDF (What is the difference between a clustered index and non
# clustered index ?)
```

```
'In a clustered index, the data is physically stored on the disk in the same order as the index. This physical ordering allows for fas
ter read operations compared to non-clustered data. However, insert and update operations tend to be slower with a clustered index.'
```

```
rag_chain.invoke("What is salary for freshers in TCS for Ninja role?")
```

```
'TCS pays freshers a minimum salary of Rs. 3.30 to Rs. 3.60 lakhs per year, which is the initial pay for the TCS Ninja role. The provi
ded context does not mention salaries for Digital or Prime roles for freshers.'
```

```
rag_chain.invoke("What programming languages that you can utilize for TCS NQT exam?")
```

```
'For the TCS NQT exam, you can utilize programming languages such as C, C++, Python, Java, and Perl. The programming section assesses
knowledge and skill in areas including pseudo-code, algorithms, and programming structures. Candidates are expected to be able to code
in any of these listed languages.'
```

```
Start coding or generate with AI.
```