Assignment 3 – Retrieval-Augmented Generation (RAG) using LangChain

# Task1: Conceptual Understanding

Ques 1. What is the motivation behind Retrieval-Augmented Generation (RAG)?

RAG is useful when the model doesn't have all the info it needs. It lets the model search in documents first and then give better answers using both the document and its own knowledge.

Ques 2. Explain the difference between RAG and standard LLM-based QA.

In normal LLM-based QA, the model answers from what it was trained on. In RAG, it first retrieves relevant data from an external source like a PDF, then answers using that specific data.

Ques 3. What is the role of a vector store in a RAG pipeline?

The vector store saves the document chunks as embeddings (numbers). When a user asks a question, it helps find the most related chunks to use in the answer.

Ques 4. Compare “stuff”, “map\_reduce”, and “refine” document chain types in LangChain.

Stuff: puts all text together at once. Map\_reduce: processes chunks separately, then combines. Refine: gives an initial answer and improves it step by step with more info.

Ques 5. What are the main components of a basic LangChain RAG pipeline?

Main parts are: loader (to load PDF), splitter (to break into chunks), embedder (to make vectors), vector store (to save them), retriever (to fetch), and LLM (to answer).

# Task2: RAG Pipeline Flow (Written Description)

1. The user asks a question (Query).  
2. The retriever searches in the document's vector store.  
3. It finds the most relevant chunks.  
4. These chunks are sent to the LLM.  
5. The LLM reads those and generates the final answer.