**# RAG(Retrieval Augmented Generation)**

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

RAG is designed to enhance the capabilities of language models by retrieving relevant external information instead of relying solely on their fixed training data. This approach allows LLMs to provide more accurate, up-to-date, and context-aware responses, especially for domain-specific or long-tail queries.

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

Standard LLM-based QA relies entirely on the pre-trained knowledge of the model, which may be outdated or limited. In contrast, RAG retrieves relevant documents from an external knowledge base and uses them to inform the answer generation, making it more grounded and contextually rich.

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

A vector store holds the embedded representations (vectors) of document chunks and enables efficient similarity search. It allows the system to retrieve the most relevant documents based on the user query’s embedding, which are then used to augment the generation proces

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

Stuff: Concatenates all retrieved documents into a single input and passes it to the LLM. It’s simple but limited by token constraints.

Map\_reduce: Processes each document independently (map) and combines results (reduce), suitable for large document sets.

Refine: Iteratively builds an answer by refining an initial response with additional documents, useful for deep synthesis across multiple documents.

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

The key components include:

1. Document Loader & Splitter (to chunk documents),

2. Embedding Model (to convert text to vectors),

3. Vector Store (to store and retrieve embeddings),

4. Retriever (to fetch relevant chunks), and

5. LLM (to generate the final response based on retrieved context).

Task 2: RAG Pipeline Diagram

Draw or describe the flow of a RAG system showing:

● User Query

● Retriever

● Vector Store

● LLM

● Final Answer Generatio

ANS..User Query

    │

    ▼

Query Embedding (using same embedding model as Vector Store)

    │

    ▼

Retriever

    │

    ▼

Vector Store ───► Retrieves Relevant Document Chunks

    │                            │

    ▼                            ▼

Retrieved Documents →→→→→→→→→→→→→

                                │

                                ▼

                          LLM (Prompt + Retrieved Context)

                                │

                                ▼

                      Final Answer Generation