**TASK – 1**

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

The main motivation behind RAG is to enhance the factual accuracy and relevance of language model responses by grounding them in external, up-to-date knowledge. Instead of relying solely on the static training data of the model, RAG retrieves relevant documents from a knowledge base at runtime. This makes the model more adaptable and reduces hallucinations when answering domain-specific or recent queries.

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

Standard LLM-based QA relies entirely on the model’s internal knowledge, which is limited to its training data and cutoff date. RAG, on the other hand, introduces a retrieval step that pulls relevant information from an external source, like a document store, before generating a response. This integration allows RAG to provide more accurate and contextually rich answers, especially in scenarios requiring updated or specific information.

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

In a RAG pipeline, the vector store holds the embedded representations of documents or data chunks. When a query is made, it is also embedded and compared with the stored vectors to retrieve the most semantically relevant pieces of information. This retrieval process ensures that the language model has contextual support from relevant documents during response generation.

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

“Stuff” chains concatenate all retrieved documents into a single prompt for the LLM, which works well for small inputs but struggles with long contexts. “Map\_reduce” first generates summaries or answers from each document individually (map), then combines them into a final answer (reduce), which is more scalable. “Refine” creates an initial answer from the first document and then iteratively updates it with information from subsequent documents, improving coherence and detail with each step.

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

A basic LangChain RAG pipeline typically includes a document loader to bring in external data, a text splitter to break content into manageable chunks, and an embedding model to convert text into vector representations. These vectors are stored in a vector store, which is queried using a retriever to fetch relevant documents. Finally, a language model generates responses based on the retrieved content.

TASK – 2

Draw or describe the flow of a RAG system showing:

* User Query
* Retriever
* Vector Store
* LLM
* Final Answer Generation

1. **User Query**: The system starts with a user submitting a question or query. This query represents the information the user is seeking.
2. **Retriever**: The query is passed to a retriever module, which converts the query into a vector (embedding) and searches for semantically similar documents or text chunks.
3. **Vector Store**: The retriever interacts with a vector store, which holds precomputed embeddings of documents. The most relevant documents based on similarity to the query vector are retrieved from the store.
4. **LLM (Large Language Model)**: The retrieved documents, along with the original query, are provided as input to the language model. The LLM uses this external information to ground its response and ensure accuracy.
5. **Final Answer Generation**: The LLM processes the input and generates a coherent and contextually appropriate answer, which is then returned to the user as the final output.