RAG - Retrieval-Augmented Generation

- What is RAG pipeline? How does it work?
- What is the context in RAG pipeline?
- What is AI Hallucination?
- RAG = Data Indexing + Data Retrieval + Data Generation

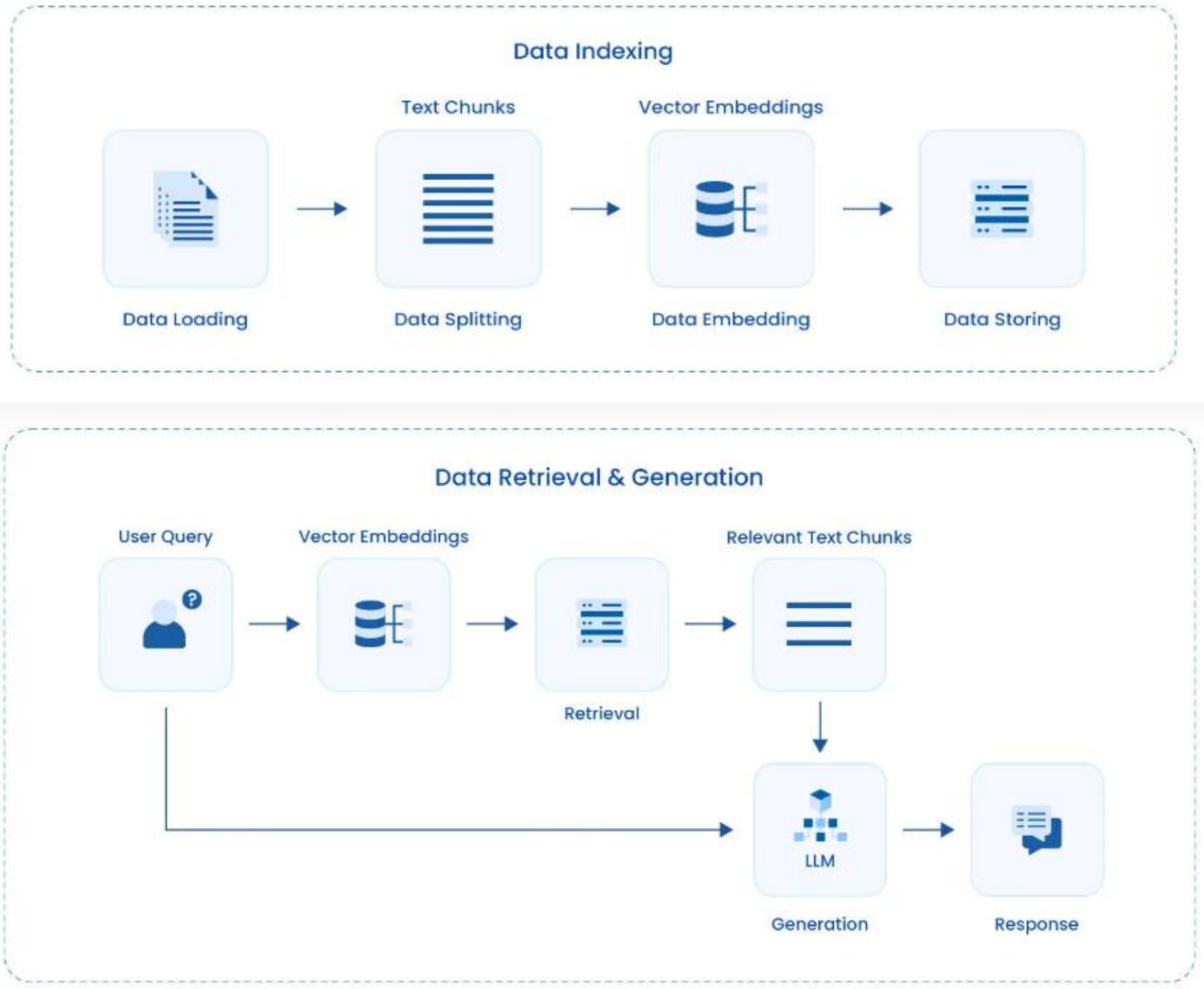
RAG Pipeline operates in two main phases:

I. Data Indexing (Preparatory stage)

II. Data Retrieval & Generation (Real-time stage that occurs when a user asks a question

Let’s see How each module works & why it is used …..!!!

Basic RAG Pipeline



Data Indexing (Preparation)

- This phase starts with **Data Loading**, where raw documents are ingested.
- These documents are then subjected to **Data Splitting**, breaking them into smaller **Text Chunks**.
- These chunks are converted into numerical **Vector Embeddings** during **Data Embedding**.
- Finally, these embeddings are organized and saved in a searchable format during **Data Storing**, typically within a vector database, forming the **Knowledge Base**.

Data Indexing Phase:

- **Data Loading**:
  - **Why it's used**: This is the first step in building your knowledge base. It's used to ingest raw data from various sources (e.g., PDFs, web pages, databases, internal documents) into the RAG system.
  - **How it's used**: Software components, often called "loaders" or "connectors," read and extract content from your chosen data sources.
- **Data Splitting**:
  - **Why it's used**: Large documents are typically too long to fit into the context window of LLMs and can contain extraneous information. Splitting them into smaller, coherent "chunks" ensures that only the most relevant portions are retrieved, improving retrieval accuracy and reducing processing load for the LLM.
  - **How it's used**: Algorithms divide the loaded data into chunks based on predefined sizes, character counts, or semantic boundaries (e.g., paragraphs, sections, or even sentences).
- **Data Embedding**:
  - **Why it's used**: Computers cannot directly understand human language. Data embedding converts text (your chunks) into numerical representations called "vector embeddings." These vectors capture the semantic meaning of the text, allowing for mathematical comparisons of similarity.
  - **How it's used**: An "embedding model" (a type of AI model) processes each text chunk and outputs a dense vector of numbers. Texts with similar meanings will have vectors that are numerically "close" to each other in a multi-dimensional space.
- **Data Storing**:
  - **Why it's used**: Once text chunks are converted into vector embeddings, they need to be stored in a way that allows for extremely fast and efficient similarity searches.
  - **How it's used**: The vector embeddings (along with metadata and sometimes the original text chunks) are saved in a specialized database called a **vector database** (or vector store). This database is optimized for performing "nearest neighbour" searches, quickly finding vectors (and thus text chunks) that are most like a given query vector.

Data Retrieval & Generation (Execution)

- This phase begins when a **User Query** is submitted.
- The query is also converted into **Vector Embeddings**. This query is first fed into the **Retriever module**.
- The Retriever's role is to search a vast **Knowledge Base** to find relevant information or documents that could help answer the query.
- These **relevant chunks**, along with the original user query, are then fed into the LLM for Generation, producing the **final Response**.
- This entire process ensures that the LLM's answer is grounded in information from the Knowledge Base, **reducing** the likelihood of generating inaccurate or **hallucinated content**.

User Query

The **User Query** is the starting point of the Data Retrieval & Generation (Execution) phase. It's the question or prompt that a user submits to the system. This is the input that the RAG system aims to answer or address.

Retriever

- **Why it's used:** Its primary purpose is to identify and fetch the most relevant pieces of information from a large corpus of text (the Knowledge Base) that are pertinent to the user's query. This is crucial because LLMs, while powerful, have a limited context window and don't inherently know all external facts.
- **How it's used:** When a user query comes in, the Retriever analyzes it (often converting it into a numerical representation called an embedding) and then uses this representation to perform a similarity search against the embedded documents in the Knowledge Base. It then returns the top 'N' most similar or relevant documents.

Knowledge Base

- **Why it's used:** It serves as the factual foundation for the LLM's responses. By providing external, up-to-date, and domain-specific information, it allows the LLM to answer questions that it wasn't explicitly trained on or whose information might be outdated in its training data.
- **How it's used:** The Knowledge Base typically consists of a collection of documents, articles, web pages, databases, or any other textual information. Before retrieval, these documents are usually pre-processed and converted into numerical embeddings, making them searchable by the Retriever.

Large Language Model (LLM)

- **Why it's used:** The LLM's role is to synthesize information and generate a human-like response. By receiving both the original query and the relevant context from the Retriever, it can formulate an answer that is factually accurate and query specific, leveraging the provided external knowledge.
- **How it's used:** The LLM takes the user query and the retrieved documents as input. It then processes this combined input, using its vast pre-trained knowledge and the newly provided context to understand the question and generate a well-formed, informative, and relevant text response.

Response

- **Why it's used:** It's the answer or generated text that is presented back to the user, directly addressing their initial query.
- **How it's used:** This is the culmination of the entire process, demonstrating the RAG system's ability to provide accurate and contextually rich information by combining the strengths of information retrieval with large language model generation.