Retrieval-Augmented Generation (RAG) Overview

1. Introduction to RAG
   - RAG combines information retrieval with text generation.
   - Helps large language models (LLMs) generate more accurate responses.
   - Uses vector databases to store and retrieve relevant document chunks.

2. Data Ingestion
   - Load unstructured data from PDFs, text files, or web pages.
   - Pre-process text: remove stopwords, tokenize, and clean.
   - Chunk data for efficient retrieval.

3. Embedding & Storage
   - Convert text into vector embeddings using models like OpenAI or Mistral.
   - Store embeddings in a vector database (FAISS, Chroma, Weaviate).

4. Retrieval Mechanisms
   - Keyword-based retrieval (BM25).
   - Semantic search using embeddings.
   - Hybrid retrieval (combining keyword and semantic search).

5. Response Generation
   - Retrieve the most relevant chunks based on query similarity.
   - Use an LLM (e.g., LLaMA, Mistral, or Gemma) to generate a response.
   - Ensure context is injected properly to avoid hallucinations.

6. Use Cases of RAG
   - Chatbots and virtual assistants.
   - Enterprise search solutions.
   - Research and knowledge retrieval systems.

This document provides a high-level overview of RAG concepts and implementation strategies.