Retrieval-Augmented Generation (RAG) – Simple Guide

Retrieval-Augmented Generation (RAG) is a technique that improves Large Language Model (LLM) answers by providing external knowledge from documents like PDFs, text files, or databases.

Why RAG is Needed

1. LLMs have limited knowledge (training cutoff).
2. LLMs cannot remember your private documents.
3. RAG allows answering questions from custom data.

RAG Architecture (High Level)

1. 1. Load Documents (PDF, text, etc.)
2. 2. Split documents into chunks
3. 3. Convert chunks into embeddings
4. 4. Store embeddings in a Vector Database
5. 5. Retrieve relevant chunks for a user query
6. 6. Send context + query to the LLM

Simple RAG Flow

User Question → Vector Search → Relevant Documents → Prompt + Context → LLM → Final Answer

Basic Tools Used

1. LangChain – RAG framework
2. Embedding Model – Converts text to vectors
3. Vector Store (FAISS) – Stores embeddings
4. LLM (OpenAI / Cerebras / etc.) – Generates answers

Advantages of RAG

1. More accurate answers
2. Uses latest or private data
3. Reduces hallucinations
4. No need to retrain the model

Real-World Use Cases

1. Chat with PDFs
2. College notes chatbot
3. Company document assistant
4. Customer support bot

This PDF gives a beginner-friendly understanding of RAG. You can extend this by adding code examples and advanced retrievers.