Retrieval-Augmented Generation (RAG)

Part 1: Foundations

Understanding how RAG improves LLM accuracy and reduces hallucination by combining retrieval with generation

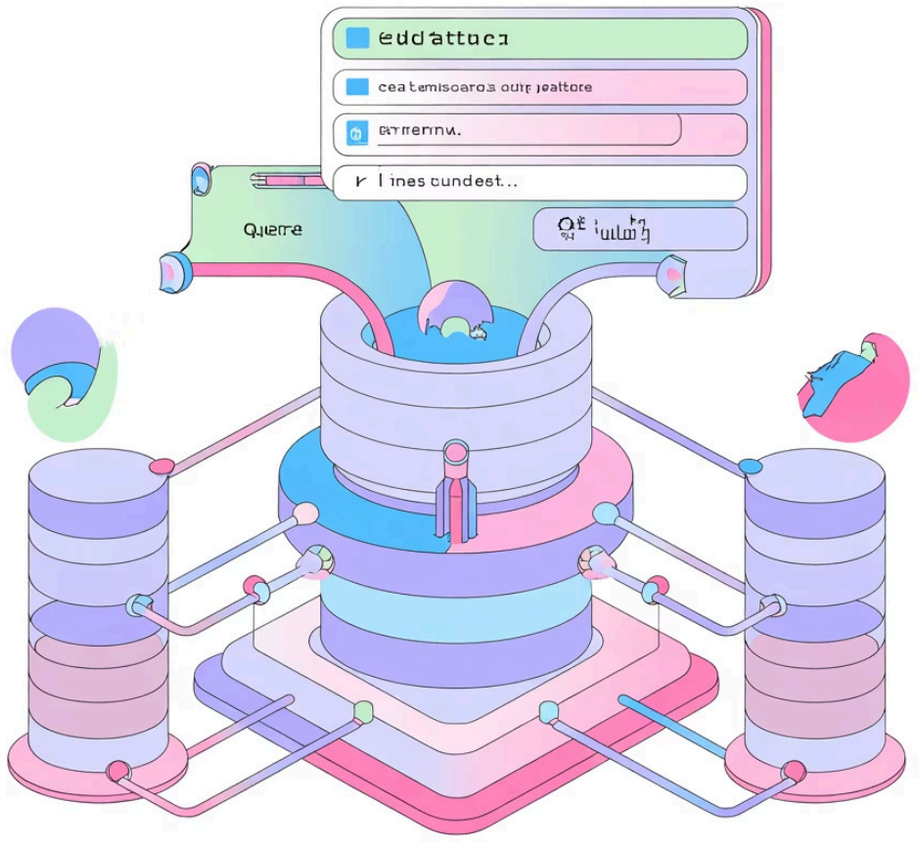
What is RAG?

Retrieval-Augmented Generation (RAG) combines information retrieval and text generation capabilities. Before generating an answer, the model retrieves relevant external knowledge to ground its response in verified facts.

Core Formula

RAG = Retrieval + Generation

This powerful approach ensures that AI-generated responses are anchored in actual data rather than purely relying on the model's parametric memory.



Why RAG? Solving Critical LLM Challenges

Hallucinations

Traditional LLMs generate made-up facts

RAG Solution: Retrieves real documents before answering, grounding responses in verified information

Outdated Knowledge

Models become stale over time

RAG Solution: Uses external, continuously updated sources without retraining

Expensive Fine-Tuning

Retraining models is costly and slow

RAG Solution: Plug-and-play approach—just retrieve new data as needed

Limited Context

Models have fixed context windows

RAG Solution: Fetches only the most relevant text chunks dynamically

✅ **Goal:** Generate accurate, context-aware, domain-specific responses that users can trust.

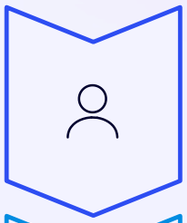
Traditional LLM vs. RAG-Based LLM

Feature	Traditional LLM	RAG-Based LLM
Data Source	Model training only	External knowledge base
Context Awareness	Limited	High
Accuracy	May hallucinate	Grounded in facts
Adaptability	Needs retraining	Plug-and-play updates
Knowledge Freshness	Frozen at training time	Always current

💡 **In short:** Traditional LLM = memory-based | RAG = retrieval + reasoning

RAG Workflow Overview

Understanding the step-by-step process of how RAG transforms queries into accurate, grounded responses

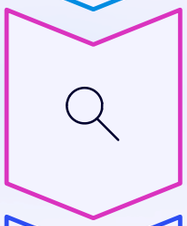


User Query

User sends a question or request

Embedding

Query is converted to a vector representation



Similarity Search

Vector DB performs semantic matching

Retrieval

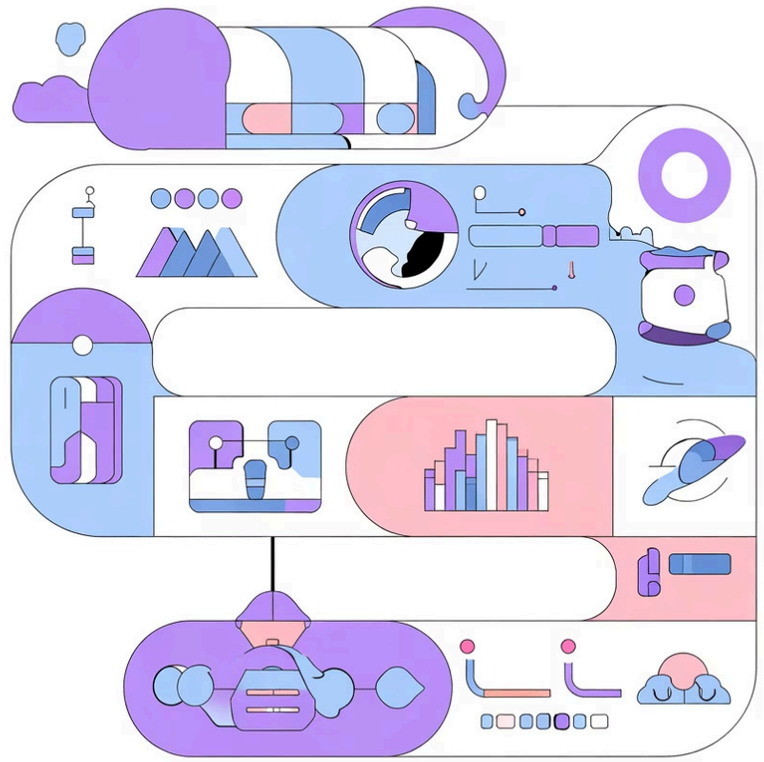
Relevant documents are fetched

Generation

LLM uses query + docs to create response

🔁 **Continuous feedback loop:** Each interaction makes the system smarter and more contextually aware

RAG Pipeline Formula



01

Retrieval

Find relevant chunks using vector similarity search across the knowledge base

02

Augmentation

Add retrieved text into the LLM prompt as contextual information

03

Generation

LLM produces the final factual answer grounded in retrieved data

✅ This three-step process keeps outputs accurate, traceable, and explainable

Key Components of RAG

Five essential building blocks that work together to power the RAG architecture

1

Retriever

The search engine that finds relevant documents from the knowledge base using semantic similarity

2

Vector Database

Specialised storage that stores and searches high-dimensional embeddings efficiently at scale

3

Embeddings

Mathematical representations that convert text into vectors, capturing semantic meaning

4

LLM / Chain

The language model that generates the final answer by synthesising query and retrieved context

5

Pipeline Orchestrator

Coordinates all components, managing data flow from query to final response generation

Example Scenario in Action

Query

> "Explain LangChain architecture."

Embed the Query

Convert question into vector representation

Vector DB Retrieval

Find and retrieve relevant LangChain documentation chunks

Context Assembly

Retrieved chunks + original query passed to LLM

Final Response

Result: Context-rich, accurate explanation grounded in actual documentation

Real-World Use Cases

RAG is transforming how organisations leverage AI across diverse domains and industries

Enterprise Chatbots

Company-specific customer support with access to internal knowledge bases and documentation

Medical Q&A Systems

Healthcare assistants that retrieve from medical journals and clinical guidelines

Legal Document Assistants

Contract analysis and legal research powered by case law and regulatory databases

Enterprise Knowledge Retrieval

Internal search engines that surface relevant information from company repositories

Research Support Tools

Academic assistants that search through papers and provide evidence-based insights

Key Takeaways

Bridges the Gap

RAG connects LLMs with real-world, up-to-date data sources, ensuring responses stay relevant

Solves Core Problems

Eliminates hallucination issues and knowledge freshness challenges inherent in traditional LLMs

Flexible Architecture

No retraining required—simply update your knowledge base to enhance the system

---

> "RAG makes your AI not just smart—but informed."