Retrieval-Augmented Generation (RAG)

What is RAG?

Retrieval-Augmented Generation (RAG) is an architecture that combines a retriever and a generative large language model (LLM). Instead of answering solely from the model’s trained parameters, RAG fetches up-to-date or domain-specific documents in real time and uses them as additional context for the LLM.

How RAG Works (4 steps)

1. User asks a question
2. Retriever (dense vector search) finds the top-k most relevant text chunks from an external knowledge base (using embeddings)
3. Retrieved chunks are injected into the LLM prompt as context
4. LLM generates a grounded, accurate answer and (optionally) cites the sources

Key Advantages

- Dramatically reduces hallucinations
- Works with private, proprietary, or constantly changing data
- No need to retrain or fine-tune the LLM — just update the vector database
- Cost-efficient and scalable
- Provides traceability and citations

Common Use Cases

- Enterprise search & chatbots
- Legal, medical, and financial Q&A
- Customer support with internal docs
- Research assistants over latest papers or reports
- Personal assistants using your own files

Popular Tools & Stack (2025)

- Frameworks: LangChain, LlamaIndex, Haystack, Flowise
- Vector databases: Pinecone, Weaviate, Chroma, Qdrant, Milvus, PGVector
- Embedding models: text-embedding-3-large (OpenAI), voyage-law-2, BGE-large, Cohere embed-v3
- Evaluation: Ragas, DeepEval, TruLens

RAG has become the default method for building reliable, knowledge-intensive AI applications that stay accurate and current without constant model retraining.