**Vector database**

A vector database is a database in which we store data in the form of vectors. A vector is a list of numbers that represents data like texts or images. These vectors are usually generated by AI models called embeddings. For example, a sentence can be turned into a 300- number vector. Vector databases let you search by meaning, not just keywords. They are great for similarity search which means finding things that are close in meaning. Imagine searching for apple and getting results for both the fruit and brand. This works because vector search captures context and semantics. Traditional databases can’t easily compare unstructured data this way. Vector databases solve that by using math and geometry in high dimensions. Each vector is a point in space, and the database finds the nearest points. This is called a nearest neighbor search. It is very useful in AI applications like chatbots, recommendation systems and image search.*shis* Tools like Pinecone, FAISS, Weaviate and Milvus are popular vector databases. ChromaDB is also a vector database which stores and retrieves embeddings (vectors) generated from data like text, images or audio. We can store documents and search by meaning, not just exact words. Chroma has simple API and is easy to use in Python projects. It supports filtering, metadata storage and real-time querying. Vector databases work well with large language models and generative AI. We can build smart apps that understands meaning, not just exact words.

**Agentic RAG**

Agentic RAG is an advanced AI system where a reasoning agent uses retrieval-augmented generation to plan search and generate more accurate, goal-driven responses through multiple steps. In regular RAG, an LLM pulls relevant info from a knowledge base to answer questions. Agentic RAG adds an “agent” layer for making the system more autonomous and smart. The agent decides how to search, what to search and how to use the results. It’s like giving the LLM a brain that can plan and reason step by step. Instead of just one query and one answer, it can run multiple steps. The agent might rephrase the question, search multiple times and combine answers. It can even reflect, check its own results and improve them. This makes answers more accurate, complete and human-like. Agentic RAG often uses tools like vector databases or APIs. It can plan, call tools, store memory and refine its response. Think of it as RAG + AI agent= smarter reasoning. Tools like LangChain, AutoGen and CrewAI help build Agentic RAG systems. This is useful in complex tasks, like research bots, coding assistants or advisors. Agentic RAG makes AU more interactive, flexible and goal-driven. It’s a big step toward autonomous AI systems. In simple terms Agentic RAG is RAG with a smart AI agent that can think and act. It’s the future of AI-powered search and reasoning.