DAY 04 MODULE

The Memory:

Relational databases, Embeddings & VectorDBs

Grounding your Agent in Reality with RAG.

By,

Shikha Tyagi

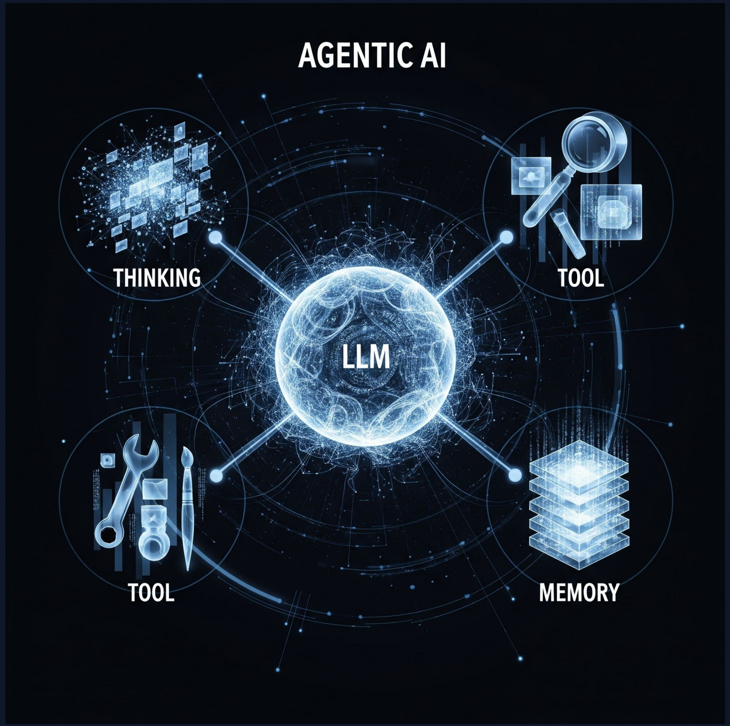
Founder - AI JAMIC (AI Research and Consulting)

Education: IIT Delhi (M.Tech.)

Guidelines

- Attendance is mandatory for all 5 sessions
- Hands on activity is mandatory
- 15 min break at 10:30PM
- QnA session at the end (10-15 min)
- Feel free to drop your questions in chat
- There will be quizzes in-between, drop your answers in chat

5 day roadmap



1
Shift
Agentic Thinking vs. Chatbots

2
Brain
LLM Types & Prompting

3
Hands
Function Calling & Tools

4
Memory
RAG & Vectors

Build
End to end pipeline & Capstone

Today's Agenda

01 **The Problem**
Why LLMs need memory.

02 **Connecting to relational databases.**

03 **Embedding & Vector Databases**
ChromaDB, FAISS, and Semantic Search.

04 **RAG Pipeline**

05 **Hands on**

Quiz 1

Which API we used for getting real time weather information?

- Open meteo
- get_weather

Quiz 2

Tool usage is not needed for

- Getting historical facts
- Getting latest facts

Quiz 3

Currency conversion api takes

- 6 arguments
- 2 arguments
- 10 arguments

Quiz 4

An agentic ai application can handle exactly 1 tool

Yes/No?

The Problem
Why LLMs Need Help

Hallucinations and Knowledge Cutoffs.

Limitation 1: Frozen in Time

Training Data Cutoff

LLMs are trained on data up to a specific date (e.g., 2023).

They do not know about:

- Yesterday's news.
- Your private company documents.
- The email you just received.

Asking about these topics leads to **Hallucinations**.

https://platform.openai.com/docs/models

https://ai.google.dev/gemini-api/docs/models

"I'm sorry, I don't know about events after 2023."

Quick Check

Let us say you want to summarize a 500 page document containing 512k words using GPT-4o

What can be potential issue?

•

Limitation 2: The Context Window

Let us say you want to summarize a 500 page document

Why not upload entire document?

You can paste text into ChatGPT, but there is a limit (Context Window).

- **Cost:** Paying for 1M tokens per query is expensive.
- **Accuracy:** "Lost in the Middle" phenomenon (LLMs forget details in the middle of massive prompts).
- **Latency:** Processing huge prompts takes time.

128k

Common Token Limit (GPT-4)

(Approx 300 pages of text)

Quick Exercise

Upload a document

Limitation 3: Domain Specific Knowledge Understanding

LLMs can only answer using publicly available data

Why it matters : They don’t have access to private company/institute datta

What is EOS

Multiple answers, none is correct

Solution

1. Integrate relational database to LLMs

2. Store unstructured data into vector databases using embedding. It is also the foundation of RAG (end to end hands on will be done on Day5)

Solution

1. Integrate relational database to LLMs

Hands On

Solution

1. Integrate relational database to LLMs

2. Store unstructured data into vector databases using embedding. It is also the foundation of RAG (end to end hands on will be done on Day5)

RAG

Retrieval-Augmented Generation

Instead of memorizing everything, give the LLM an **Open Book Exam**.

1. User Query

"What is our refund policy?"

2. Retrieve

Search database for "refund" documents.

3. Augment

Paste found text into Prompt.

4. Generate

LLM answers using ONLY that text.

Embeddings
The Core Concept

Turning Words into Numbers.

Analogy: The Supermarket

Keyword Search

Like searching for "Fruit". You only find items explicitly labeled "Fruit". You miss "Apples" if they aren't labeled.

Semantic Search (Embeddings)

Like walking to the produce aisle. Apples are near Oranges. Bread is far away. Items are organized by **Concept**, not just name.

Visualizing Vector Space

An embedding model converts text into a list of numbers (Vector).

Similar concepts end up **close together** in mathematical space.

```
"King" -> [0.9, 0.1, 0.5] "Queen" -> [0.9, 0.2, 0.5]
"Apple" -> [0.1, 0.9, 0.1]
```

What do the numbers mean?

Each number in the vector represents a "feature" of meaning, though often abstract to humans.

Word	Dimension 1 (e.g., Royalty)	Dimension 2 (e.g., Gender)	Dimension 3 (e.g., Age)
King	0.99 (High)	0.99 (Male)	0.7 (Adult)
Queen	0.99 (High)	0.01 (Female)	0.7 (Adult)
Boy	0.01 (Low)	0.99 (Male)	0.1 (Child)

*Simplified conceptual example. Real vectors have 1536+ dimensions.

Semantic Similarity

The Magic

Because "King" and "Queen" have similar numbers in most dimensions, their **Cosine Similarity** score is high.

This allows us to search for "Monarch" and find "King", even if the word "Monarch" isn't in the text.

Cosine Similarity

We measure the **Angle** between vectors.

- **1.0:** Identical meaning.
- **0.0:** Unrelated.
- **-1.0:** Opposite meaning.

Quiz: Concept Check

Question 1:

Which pair of words would likely have the closest vector distance?

A. "Cat" and "Car"

B. "Doctor" and "Surgeon"

C. "Red" and "Apple"

Quiz: Concept Check

What does -1 cosine similarity means?

Vector Databases
The Storage Engine

Where we keep the numbers.

Traditional DB vs. Vector DB

Feature	SQL (PostgreSQL, MySQL)	Vector DB (Chroma, Pinecone)
Data Type	Rows, Columns, Relations	Vectors (Arrays of floats)
Search	Exact Match (WHERE id=1)	Approximate Nearest Neighbor (ANN)
Goal	Transaction integrity	Semantic Similarity

The Vector DB Landscape

ChromaDB

Open Source & Local. Runs in-memory or as a simple file. Perfect for development and this workshop.

FAISS

Meta (Facebook). A library for efficient similarity search. Not a full database, but the engine behind many.

Pinecone

Managed Service. Fully hosted, scalable, production-ready. Paid tier.

Spotlight: ChromaDB

https://docs.trychroma.com/guides/build/intro-to-retrieval

Why we use Chroma

Chroma is "AI-native". It handles the tokenization and embedding for you automatically by default.

It creates a **Collection** (like a table) where you store documents, metadata, and embeddings.

```
import chromadb client = chromadb.Client() collection
= client.create_collection("my_docs") collection.add(
documents=["This is a document", "This is another"],
metadatas=[{"source": "doc1"}, {"source": "doc2"}],
ids=["id1", "id2"] )
```

Spotlight: FAISS

Facebook AI Similarity Search

- Focuses purely on the algorithms to find neighbors fast in high-dimensional space.
- Can search billions of vectors in milliseconds.
- Often used *inside* other databases or LangChain.

Key Algo: HNSW

(Hierarchical Navigable Small World). A graph-based algorithm that acts like a "highway system" for finding nearest neighbors quickly.

Match the Use Case

Scenario A

You are building a prototype on your laptop for a class project.

Use ChromaDB (Local)

Scenario B

You are building Netflix's recommendation engine with 100M users.

Use Pinecone / Milvus (Distributed)

Hands on
The RAG Pipeline
Step-by-Step (Focus on embedding and storage)

Ingest $\to$ Chunk $\to$ Embed $\to$ Retrieve.

Embed & Store

Pass the chunks through an Embedding Model (e.g., OpenAI `text-embedding-3-small`).

Save the resulting vectors into ChromaDB.

```
from langchain_openai import OpenAIEmbeddings from
langchain_chroma import Chroma db =
Chroma.from_documents( chunks, OpenAIEmbeddings() )
```

How do we measure ”Close”?

Cosine Similarity

Measures the **angle**. Best for text similarity (most common). Unaffected by vector magnitude.

Euclidean Distance (L2)

Measures the **straight line** distance between points. Good for clustering.

Dot Product

Projection of one vector onto another. Faster, but requires normalized vectors.

Step 5: Augment & Generate

The final prompt looks like this:

```
Answer the question based only on the following context: {context} <-- Inserted Retrieved Chunks Question: {question}
```

The LLM reads the context and synthesizes the answer.

Quiz: Pipeline Check

Question 2:

Why do we split text into "Chunks" before embedding?

- **A** To save money on storage.
- **B** To capture specific meanings and fit within embedding model limits.
- **C** Because Vector DBs cannot store strings.

Hands-on Lab
Build a RAG Pipeline

Chat with a College Policy Document.

Day5 guideline

Apply what you've learned.

Choose Your Track

1. E-Commerce Agent

RAG over product catalog

2. Academic Assistant

RAG over textbooks

3. Legal Analyzer

RAG over contracts

Capstone Kickoff
The Challenge

Apply what you've learned.

Choose Your Track

1. E-Commerce Agent

RAG over product catalog + Tool to check stock status.

2. Academic Assistant

RAG over textbooks + Tool to create quiz questions.

3. Legal Analyzer

RAG over contracts + Tool to summarize risks.

Day 4 Summary

You have now unlocked the third pillar of Agentic AI:

Brain	Hands	Memory
LLM	Tools	Vector DB

Q & A