



Generative AI & LLM Application Development

A hands-on, 2-day course taking developers from LLM fundamentals to building a production-ready RAG application — 16 hours of practical, code-first learning.

2 DAYS

16 HOURS

10 LEARNING OUTCOMES



PREREQUISITES

What You Should Know Before This Course

To get the most out of this hands-on course, participants should have the following baseline knowledge.



Basic Programming

Familiarity with Python programming (variables, functions, loops, conditionals).



APIs & Web Basics

Understanding of APIs, HTTP requests and responses (GET/POST).



Basic Data & File Handling

Comfort working with files (JSON, CSV, TXT) and basic data structures.



General Software Concepts

Basic understanding of applications, software components and logic.



Good to Have

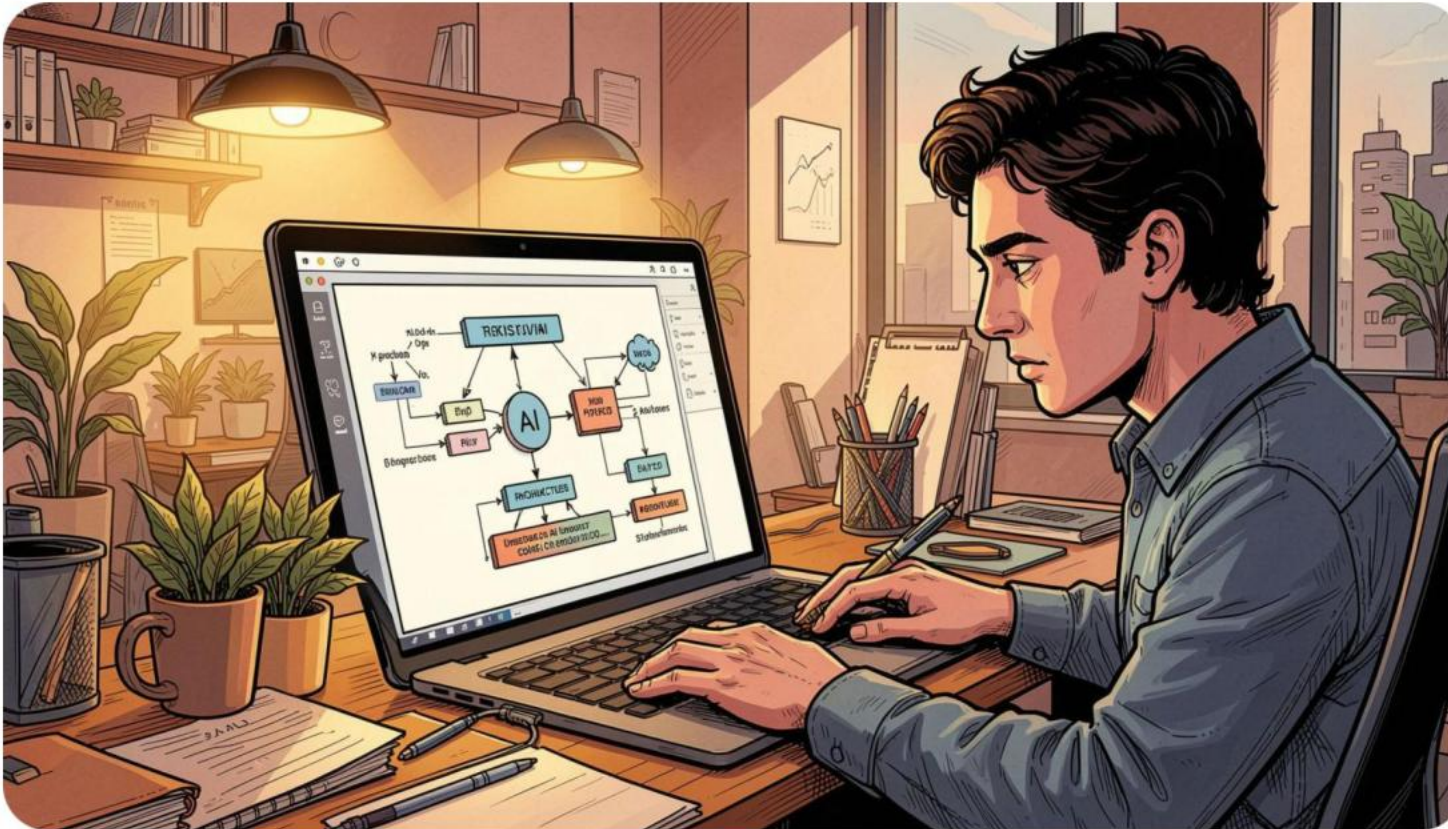
- Basic understanding of databases
- Familiarity with any IDE (VS Code, PyCharm, etc.)
- Curiosity to build and experiment!

GenAI Foundations, Prompt Engineering & LLM Apps

From understanding LLMs to building your first LLM-powered application.

Time	Module	Hands-On Activity
09:00–09:30	Opening & Orientation	Environment setup
09:30–10:45	1. GenAI & LLM Foundations	Deconstruct an AI application
11:00–12:30	2. Prompt Engineering — Part 1	Prompt engineering lab
13:15–14:30	2. Prompt Engineering — Part 2	Build structured extraction prompt
14:45–16:15	3. LLM App Dev — Part 1	First LLM API call in Python
16:15–17:15	3. LLM App Dev — Part 2	Build a Python LLM app
17:15–17:30	Day 1 Wrap-Up	Mini challenge

GenAI & LLM Foundations



Key Concepts

- GenAI vs. traditional AI/ML/DL
- Transformer architecture, tokens & context windows
- Pre-training, instruction tuning & foundation models
- LLM capabilities, limitations & hallucination
- Enterprise use cases

Hands-On: Deconstruct an AI Application

Map the architecture: **User** → **App** → **Prompt** → **LLM** → **Tools** → **Response**

Prompt & Context Engineering

Prompt Fundamentals

- System & user instructions, role prompting
- Zero-shot, few-shot & chain-of-thought
- Role + Task + Context + Constraints pattern

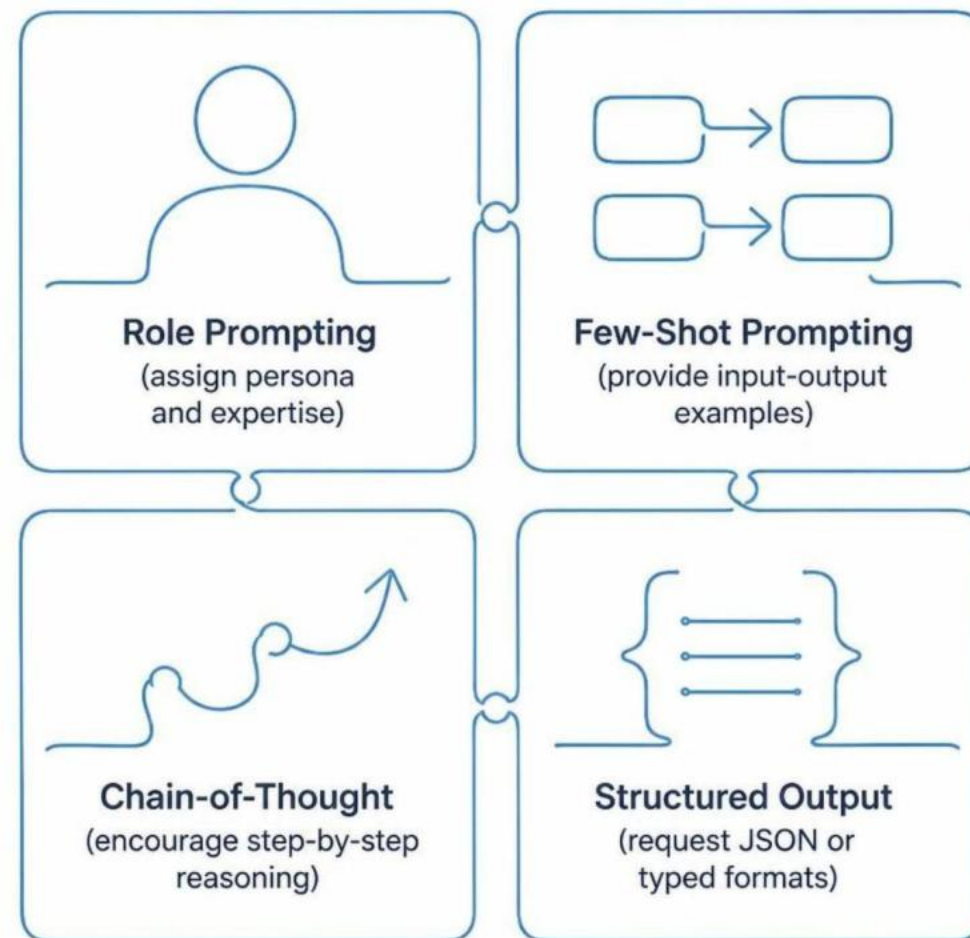
Structured Outputs

Request JSON, lists, tables, and typed responses from the r

Example: Customer feedback → structured JSON with senti

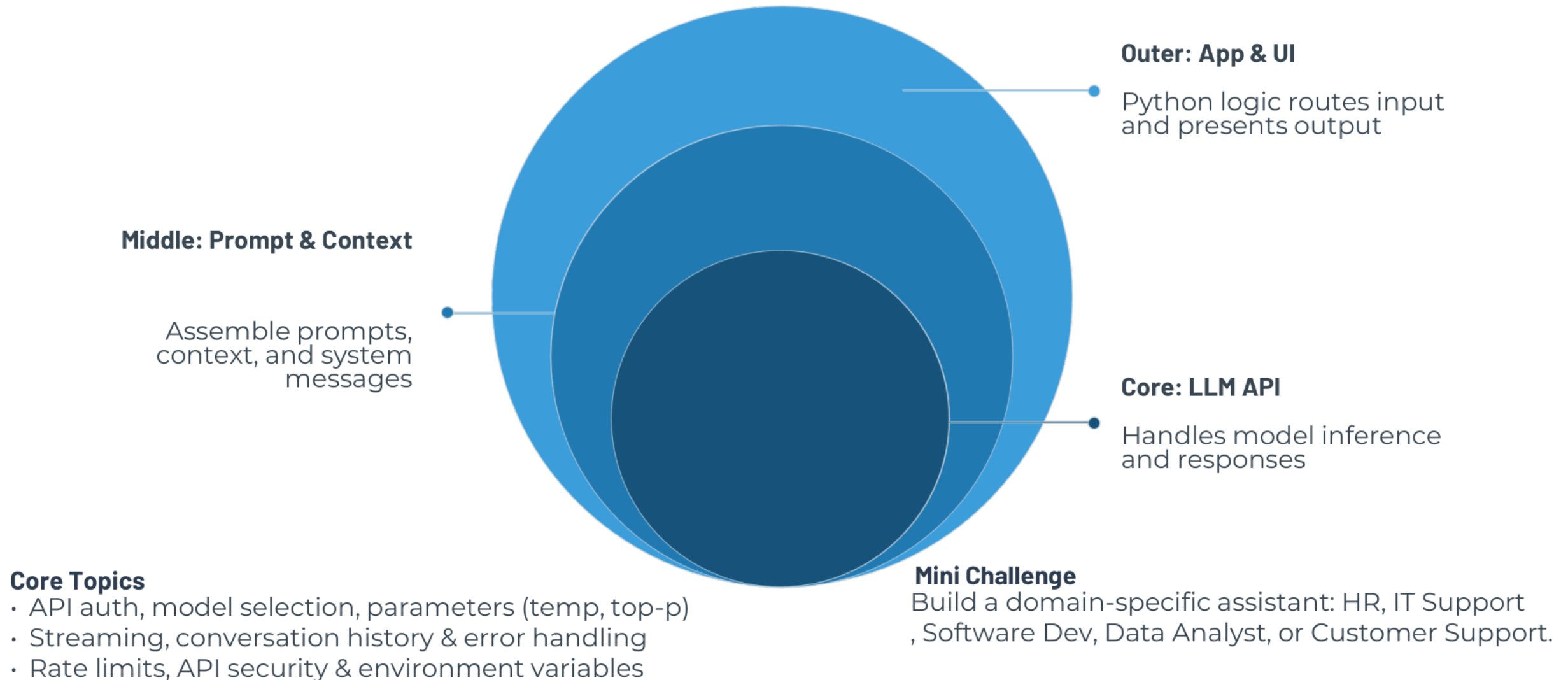
Context Engineering

- What to include vs. exclude
- Context compression & grounding



LLM Application Development

From **using ChatGPT** to **building an application** — API integration, architecture, and error handling in Python.



Embeddings, RAG, Tool Calling & Capstone Project

Give the LLM access to enterprise knowledge and external capabilities — then build and demo a full integrated application.

Time	Module	Hands-On Activity
09:20–10:45	4. Embeddings & Knowledge	Generate embeddings, semantic search
11:00–12:45	5. RAG — Part 1	Build document ingestion pipeline
13:30–14:45	5. RAG — Part 2	Build a basic RAG pipeline
15:00–16:00	6. Tools & Function Calling	Build LLM tool-calling app
16:00–16:45	7. Evaluation & Responsible AI	Evaluate responses, identify risks
16:45–17:30	8. Capstone Project	Team LLM + RAG implementation
17:30–18:00	Demos & Wrap-Up	Project presentations

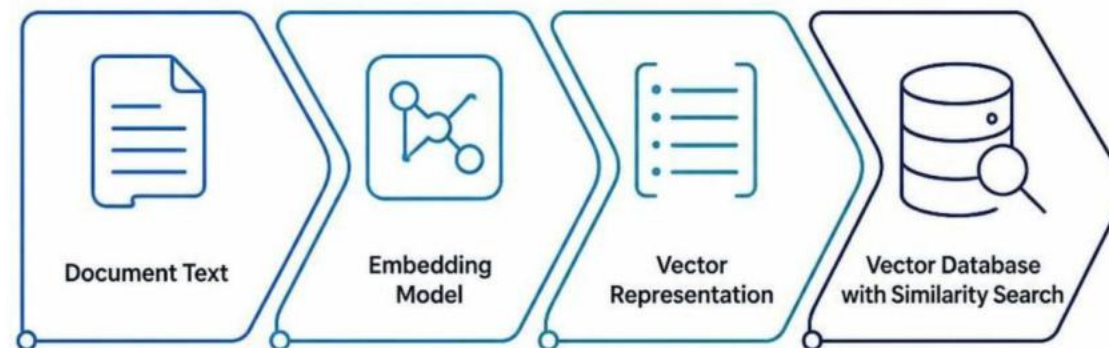
Embeddings & Semantic Knowledge

What Is an Embedding?

An embedding converts text into a high-dimensional vector, capturing semantic meaning so similar texts land close together in vector space.

Key Concepts

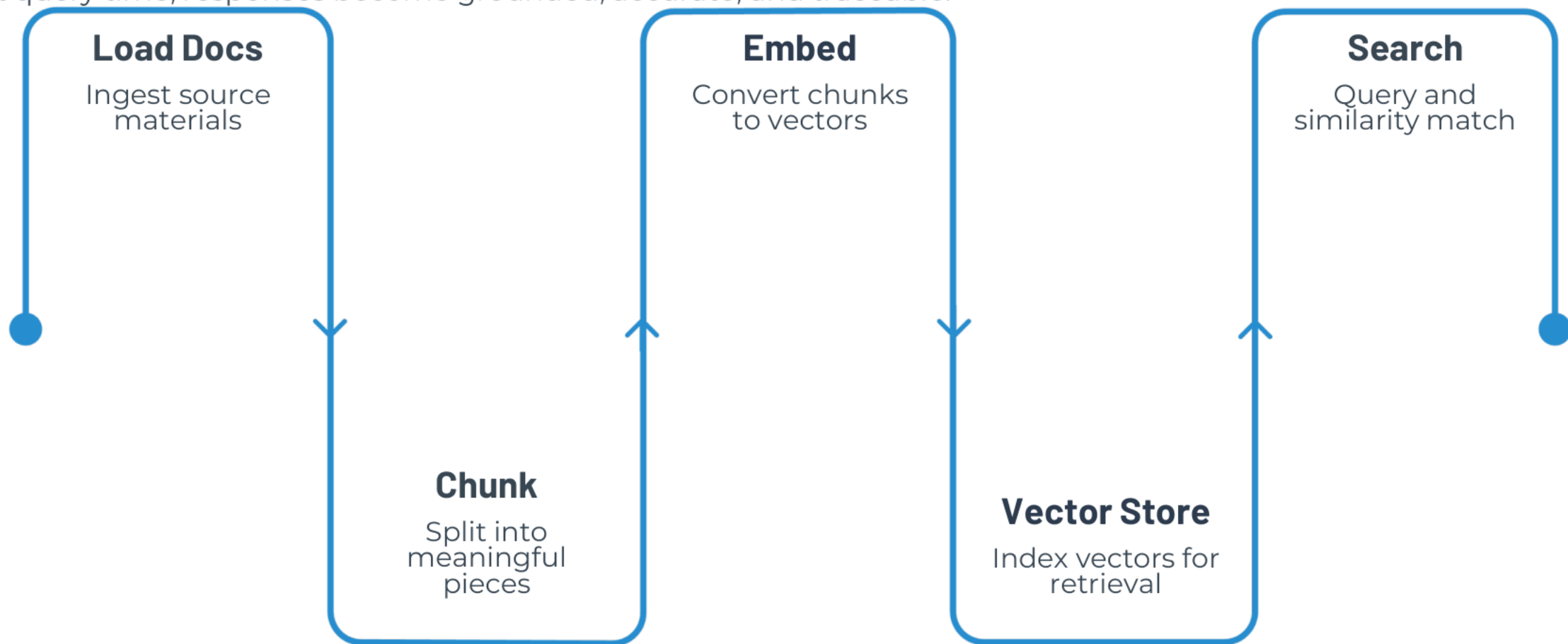
- Text → vector representation
- Cosine similarity & semantic search
- Vector databases (FAISS, Chroma)
- Metadata filtering



Lab: Create documents → generate embeddings → store → (

Introduction to Retrieval-Augmented Generation (RAG)

RAG solves a core problem: **LLMs lack enterprise knowledge**. By retrieving relevant documents at query time, responses become grounded, accurate, and traceable.



Each stage transforms raw documents into retrievable knowledge, then injects the most relevant context into the LLM prompt at query time.

Hands-On: Enterprise Document Assistant

Build a RAG pipeline over company policies, HR handbooks, or product docs — answering questions grounded in real sources.

Out of Scope (Course 2)

Advanced chunking, hybrid search, reranking, agentic RAG, and production-scale optimization are reserved for the advanced course.

AI Tools & Function Calling

From Text Generation to Action

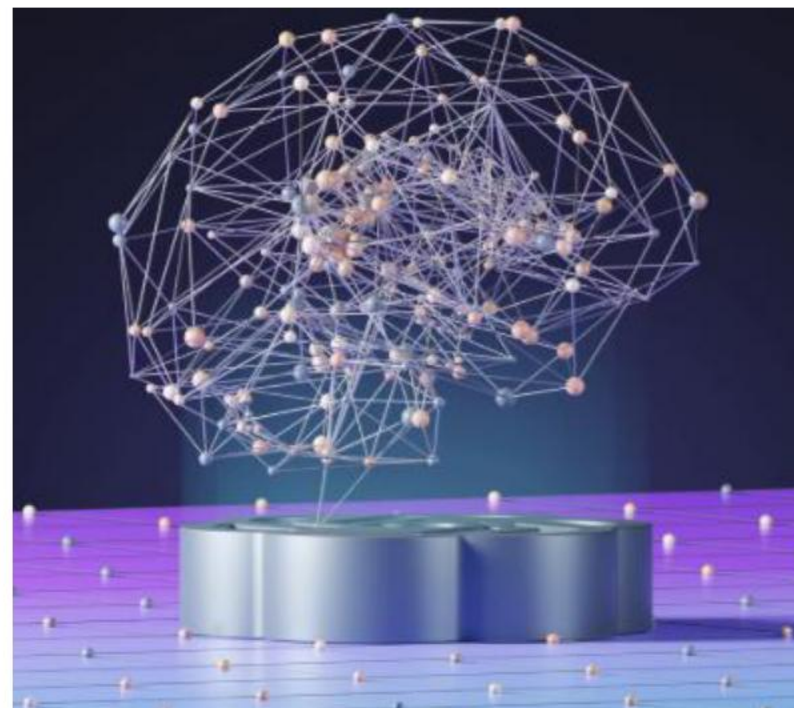
The LLM **decides** which tool to call — the application **executes** it and returns the result.

Topics

- Function schemas & tool selection
- API integration & tool execution
- Returning results to the LLM

Examples

- Weather, search, CRM, database queries
- Calendar, ticket creation, order status



Accuracy, relevance, groundedness, hallucination detection, and RAG evaluation criteria.

API key protection, PII handling, prompt injection, data leakage, and access control.

Privacy, transparency, bias, human oversight, explainability, and accountability.

Build and demo a full LLM + RAG application with tool calling, then discuss next steps.





COURSE OUTCOMES

By the End of This Course, You Will Be Able To

By completing this 2-day, 16-hour hands-on program, you will be able to:



Understand LLM Fundamentals

Explain how LLMs work, their capabilities, limitations, and real-world use cases.



Engineer Effective Prompts

Design prompts and contexts that produce accurate, reliable, and structured outputs.



Build LLM Applications

Integrate LLM APIs in Python to build robust, production-ready applications.



Use Embeddings & Semantic Search

Create embeddings, store them in vector databases, and perform semantic search.



Build Introductory RAG Applications

Implement a basic Retrieval-Augmented Generation pipeline over your own documents.



Add Tools & Function Calling

Enable LLMs to call external tools and APIs to extend their capabilities.



Evaluate & Improve LLM Outputs

Evaluate responses, identify risks like bias and hallucination, and improve reliability.



Apply Responsible AI Principles

Implement security best practices and build AI applications responsibly.



Deliver an End-to-End Project

Build and demo a complete LLM + RAG application solving a real-world problem.



CERTIFICATE

Earn a Recognized Credential



Awarded Upon Successful Completion Of:

- ✓ All hands-on labs
- ✓ Practical assessment
- ✓ End-to-end LLM + RAG capstone project
- ✓ Project demonstration & evaluation



You don't just learn GenAI.
You build, apply, and earn a certification that validates your skills.



Earn the
Panaashe Certified
GenAI Application Developer
credential.