

## Course Scope & Learning Outcome

### Objective:

By the end of this crash course, learners will:

- Understand the correct mental model of Agentic AI
- Clearly differentiate between Generative AI, RAG, AI Assistants, AI Agents, Multi-Agent Systems, and Deep Agents
- Understand the core building blocks of agentic systems
- Know which frameworks exist and why they are used
- Be ready to implement a real-world agentic AI project

### Pre-Requisites

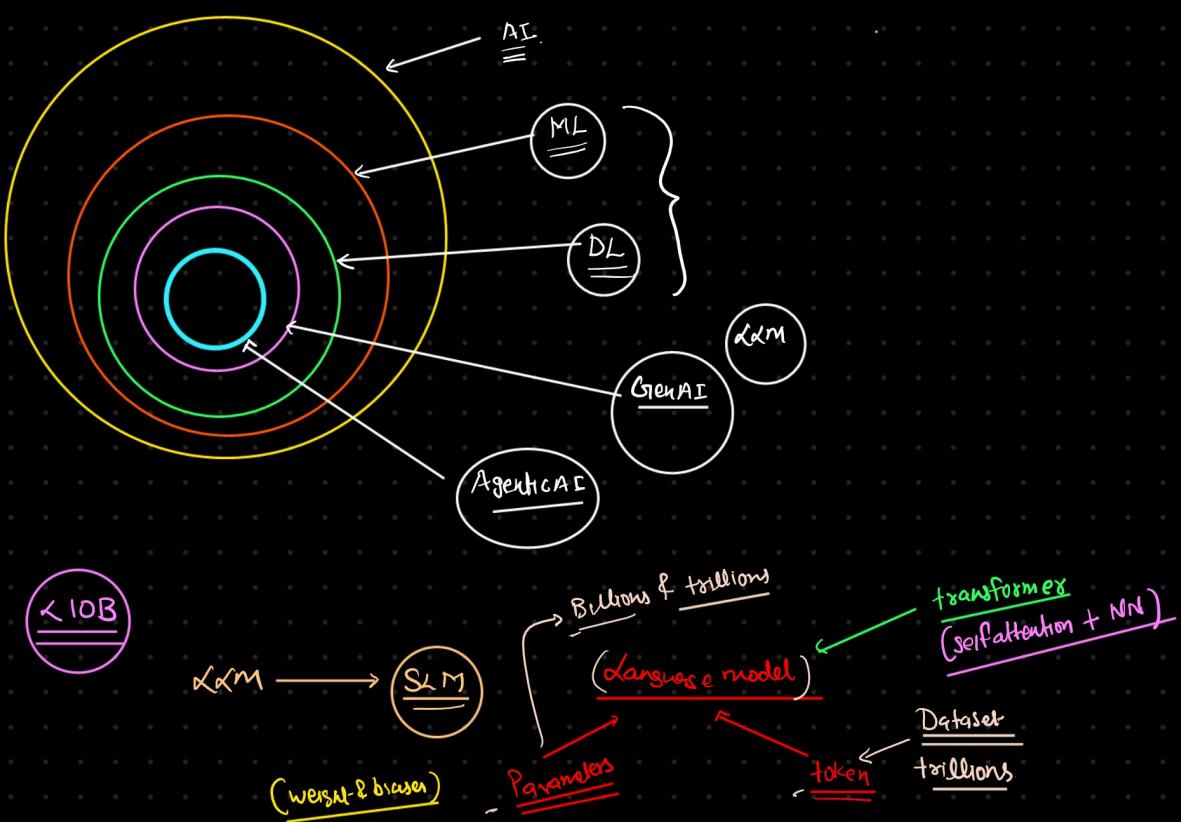
#### Required Knowledge

- Basic understanding of Python (functions, classes, APIs)  
project will be developed in Python.

#### Conceptual Knowledge

- Basic understanding of:
  - Large Language Models (LLMs)
  - LLM APIs (OpenAI / Gemini / Groq – conceptual)
  - LangChain (what problem it solves, not internals)

No deep ML, training, or math background required.



## Generative AI

AI systems that generate content such as text, images, or code.

Focus: content creation

## Fine-Tuning

The process of adapting a pre-trained model to a specific task, domain, or behavior using custom data.

Focus: permanently modifying model behavior

## AI Assistant



A conversational AI that responds to user queries, mostly reactive.

Example: chatbots, FAQ bots

## RAG (Retrieval-Augmented Generation)

A technique where AI retrieves external knowledge before generating a response.

Example: chat with PDFs or internal documents

## Agentic AI

A design philosophy where AI systems show autonomy, decision-making, and adaptive behavior.

Focus: how agents are designed

### AI Agent (Core Definition: Single Agent)

An AI system that can:

- take input
- decide what to do next
- execute actions (tool call, API call, response, or delegation)

Decision-making is the defining property.

### Deep Agent

A highly autonomous agent with:

- planning
- memory
- reflection
- looping
- often human-in-the-loop

Focus: long-running, self-improving autonomy

### Multi-Agent AI

A system where multiple AI agents collaborate, each with a defined role, to solve a problem.

Focus: coordination and teamwork

# Agentic AI Ecosystem – Core Concepts & Building Blocks

## Intelligence Layer

- LLM ← backbone
  - Prompt ← ↘
  - Thinking / Reasoning ↗
- (Chain-of-Thought, Tree-of-Thoughts – conceptual)

## Decision & Control Layer

- Planning (step breakdown, task ordering) fm inst
- Output Check / Evaluation  
(quality checks, constraint validation, rule-based or LLM-based)
- Reflection (self-critique, improvement)
- Iteration / Looping (retry, self-correction)

## Action Layer

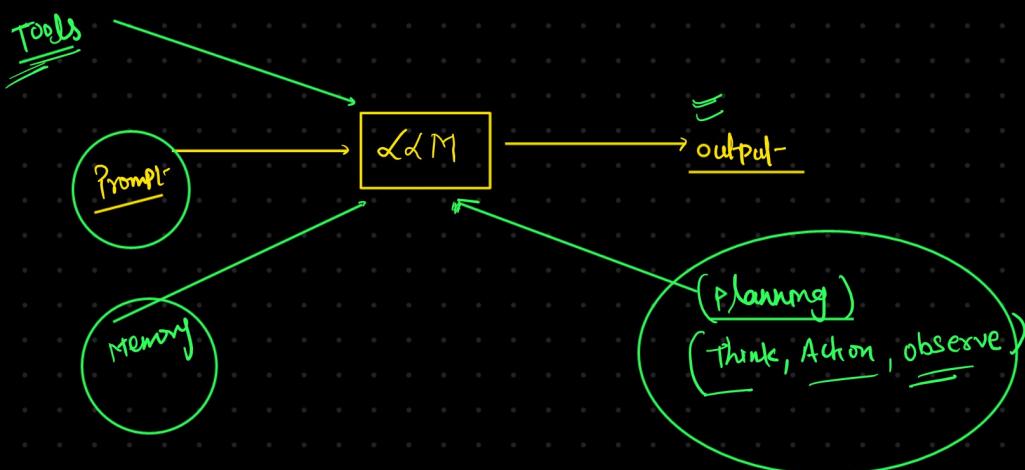
- Tool calls (API, database, code execution)
- Calling another agent
- Producing the final output

## State & Memory Layer

- State
  - what the agent knows right now
  - conversation state + task state
- Memory
  - short-term
  - long-term
  - persistent (vector databases, files, logs)

## Safety, Reliability & Infrastructure Layer

- Human-in-the-loop (approval, feedback, override)
- Guardrails (responsible AI, constraints)
- Context engineering
- Checkpointing (pause, resume, recovery for long-running agents)

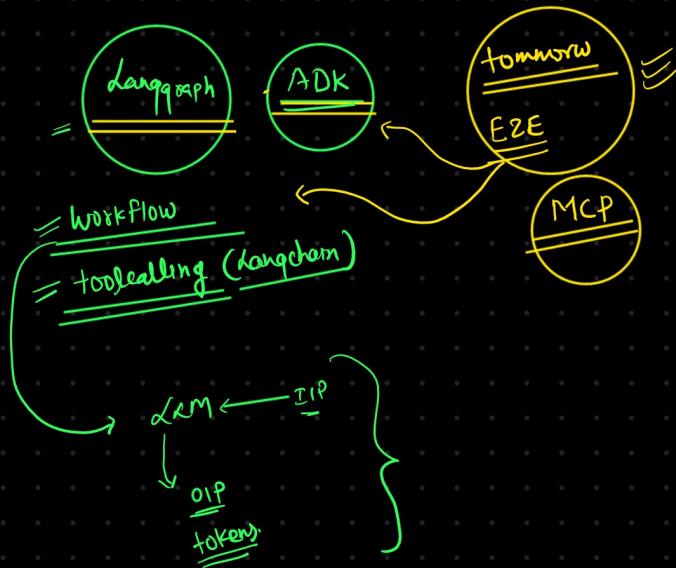


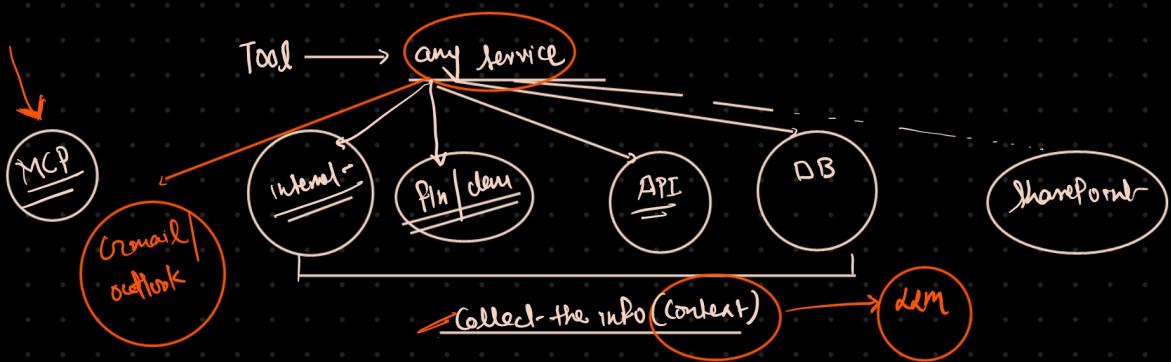
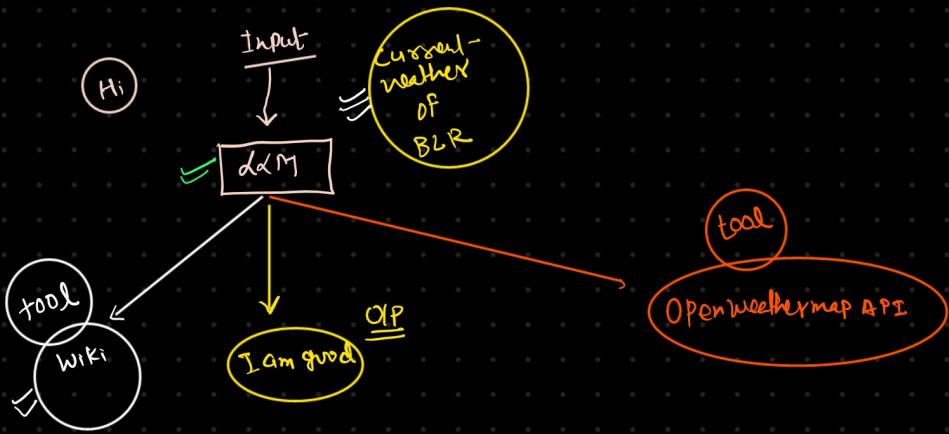
## Framework Awareness (Conceptual, Not Tool Training)

Focus is why and when, not syntax.

LangChain – foundational building blocks

- LangGraph – stateful, agentic workflows, multi-agent system
- CrewAI / AutoGen – agentic workflows, multi-agent coordination
- n8n / LangFlow – orchestration and automation without coding





Agentic Flow → **dLM + [tool calling] + Memory + Planning**

