Agents:

Agents

Agents in LangChain are powerful components that enable language models to make decisions dynamically. They choose which tools to use and in what order — based on user input — allowing for more flexible and intelligent workflows beyond static chains.

What are Agents in LangChain?

Definition

An **Agent** in LangChain is an intelligent, goal-oriented system that can use a **language model**, **tools**, **memory**, and **decision-making logic** to dynamically decide **what action to take next** based on user input.

Unlike static chains, **agents** reason step-by-step, choose tools, and adapt to context — similar to how a human would approach a task.

Why Use Agents?

- ❑ Allow **dynamic tool use** — the LLM chooses *what* to use and *when*
- ❑ Ideal for **multi-step**, **open-ended**, and **interactive** tasks
- ❑ Capable of

- Web search
- Math/calculations
- Code execution
- Database querying
- API integration

Key Components of an Agent

Component	Description
LLM	Core reasoning engine (e.g., GPT-4, Claude, Mistral)
Prompt Template	Tells the LLM how to think, reason, and respond
Tools	External functions the agent can call (e.g., calculator, search, API calls)
Memory	(Optional) Keeps history of conversation or past actions
Agent Executor	Executes the decision-making loop (LangChain's AgentExecutor)

Agent Workflow

User Input
↓
LLM (reasoning: "What should I do?")
↓
Select Tool (e.g., Calculator, Web Search)
↓
Use Tool → Get Result
↓
LLM Reflects on Output
↓
Final Answer or Next Step

Commonly Used Tools with Agents

Tool Name	Purpose	Example
LLMMathTool	For math operations	"What's 23.5% of 690?"
SerpAPI Tool	Search engine access	"What's the weather in Bangalore?"
Python REPL Tool	Execute Python code	"Plot a sine wave from 0 to 2π"
RequestsTool	Fetch data from APIs	"Get exchange rate from an API"
SQLDatabaseTool	Query databases	"What's the highest revenue city?"

Agent Types in LangChain

Agent Type	Description
Zero-shot ReAct	Uses reasoning + tools with no examples
Conversational Agent	Supports memory and multi-turn dialogs
Tool-Calling Agent	Automatically uses tools based on function schemas
Plan-and-Execute Agent	Breaks task into subgoals, then executes
LangGraph Agent	Visual, state-based agent execution (like a flowchart)