Intro to Agentic AI

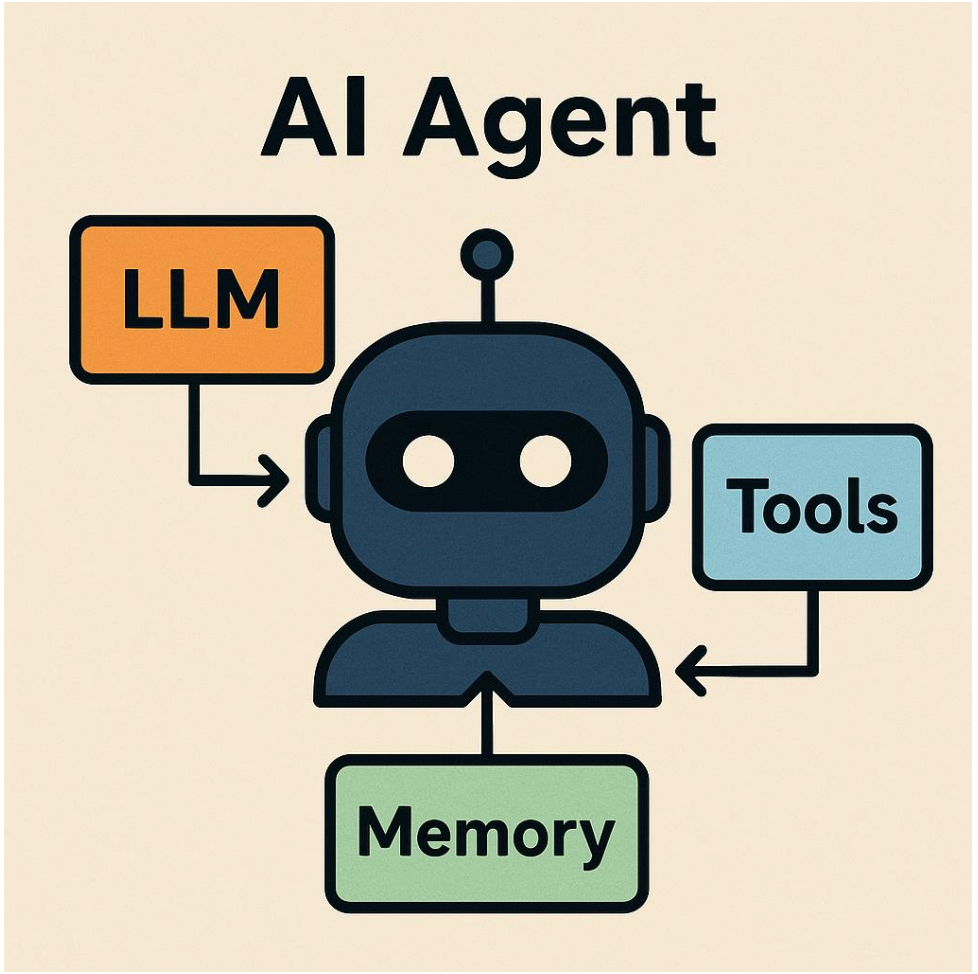
12/3/25• STAT-5350

What We Will Learn Today

- What is an AI agent?
- Why agents matter now
- Simple LangChain agents
- Tool calling & ReAct
- Hands-on mini-exercise

What Is an AI Agent?

- LLM that can plan, act, use tools
- Moves beyond static prompting
- Acts in multi-step loops



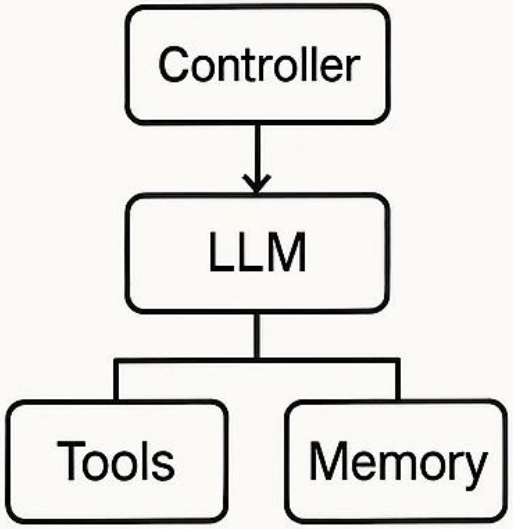
Why Agents Now?

- Better reasoning models
- Tool calling standardization
- Library support (LangChain)

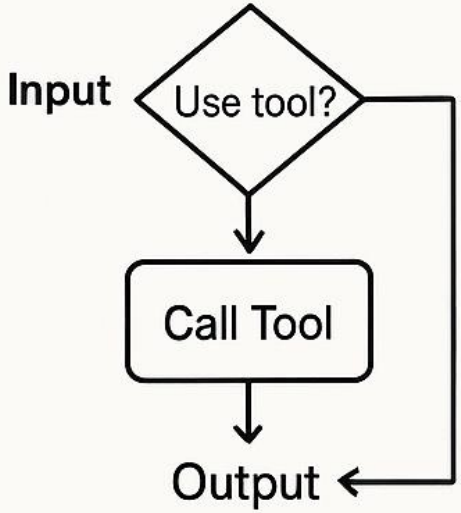
Agent Architecture (Simple)

- LLM core
- Tools
- Memory (optional)
- Controller / Executor

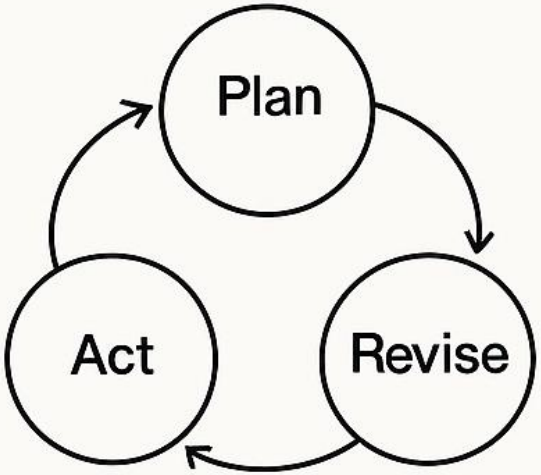
Agent Architecture



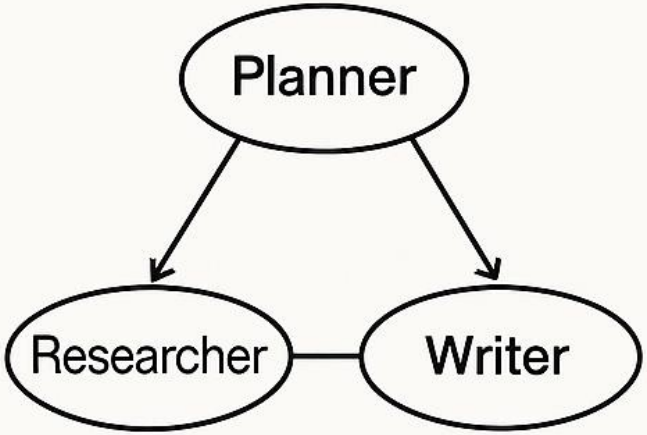
Tool Calling Flow



Planning Loop

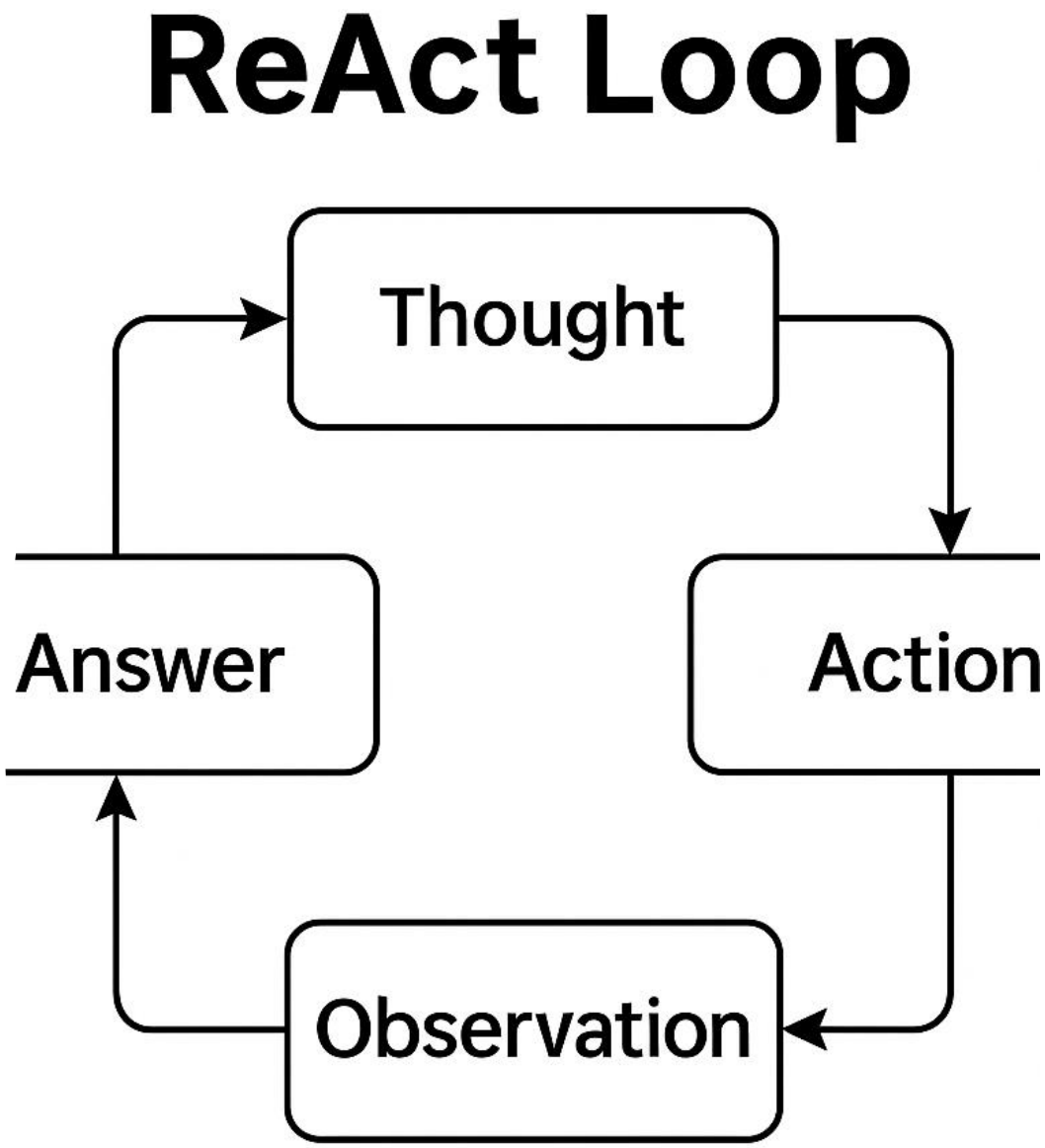


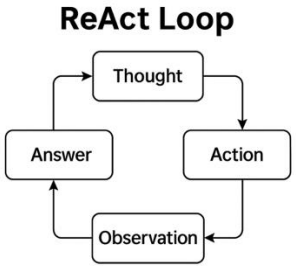
Multi-Agent System



LLM Core

- Thinks
- Plans
- Decides next action





ReACT Loop Example

Best coffee shop nearby?

Thought: I should search the user's query.
Action: search["best coffee shops in Denver"]
Observation: 1) Huckleberry Roasters ... 2) Corvus Coffee ...
Thought: Now I can answer the user's question.
Final Answer: Here are the best coffee shops in Denver...

Tools

- Calculator
- Search
- Python
- APIs

Memory

- Optional
- Conversation history
- Vector memory

Controller

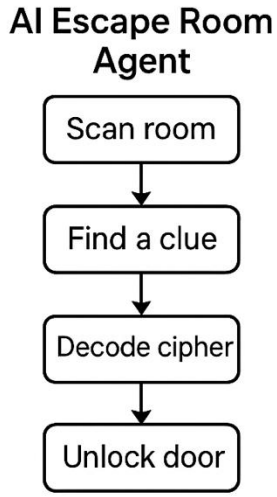
- Routes between model, tools, memory

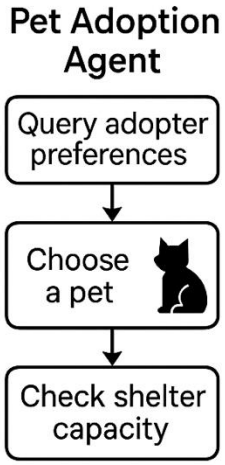
ReAct Loop (Thought→Action→Obs)

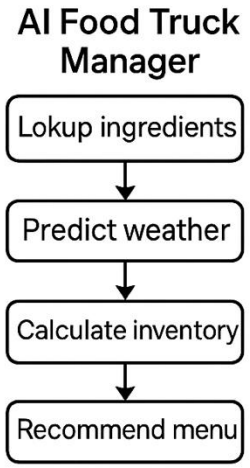
- Reason step
- Take action
- Observe result
- Repeat

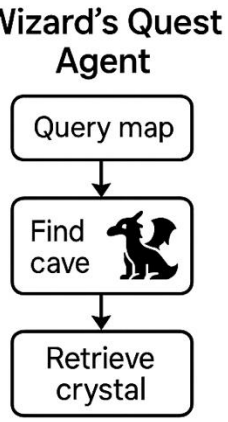
ReAct Loop (Lab)

- Reason step
- Take action
- Observe result
- Repeat









AI Escape Room Agent

Goal: Escape the locked room using clues and tools.
Tools:
- scan_room(clue)
- decode_cipher(text)
- check_inventory(item)
- unlock(door)

Instructions:
1. Identify visible clues.
2. Use scan_room to gather details.
3. Decode any symbols or text.
4. Check your inventory.
5. Determine the sequence to unlock the door.

AI Food Truck Manager

Goal: Plan the most profitable lunch service.
Tools:
- lookup_ingredients(item)
- predict_weather(location)
- calculate_inventory(levels)
- recommend_menu(day)

Instructions:
1. Check weather to predict customer turnout.
2. Review inventory levels.
3. Build a menu based on demand + supplies.
4. Use calculate_inventory to estimate costs.
5. Finalize menu and explain reasoning.

Pet Adoption Agent

Goal: Find the best match between adopter and pet.
Tools:
- lookup_pet_traits(pet)
- query_adopter_preferences(name)
- check_shelter_capacity()

Instructions:
1. Gather adopter preferences.
2. Compare preferences to available pets.
3. Evaluate traits and capacity constraints.
4. Recommend a pet and justify your match.

AI Travel Buddy Agent

Goal: Plan a weekend trip on budget.
Tools:
- search_flights(city)
- hotel_prices(date_range)
- weather(city)
- walking_distance(origin, destination)

Instructions:
1. Pick 2–3 destination candidates.
2. Find flight + hotel options.
3. Evaluate weather conditions.
4. Design a simple itinerary.
5. Stay within budget and explain trade-offs.

Wizard’s Quest Agent

Goal: Complete a magical quest safely.
Tools:
- spell_lookup(spell)
- creature_weakness(creature)
- map_query(location)
- inventory_check(item)

Instructions:
1. Map the route to your goal location.
2. Identify threats (creatures, barriers).
3. Look up spells or weaknesses.
4. Check inventory for magical items.
5. Produce a step-by-step quest plan.

Detective Agent

Goal: Solve the mystery using evidence.
Tools:
- fingerprint_scan(item)
- interrogate(character)
- analyze_bloodtype(sample)
- lookup_suspect_history(name)

Instructions:
1. Identify key evidence at the scene.
2. Run scientific analyses.
3. Interrogate suspects.
4. Cross-check histories + clues.
5. Present your suspect and reasoning.

Logistics Optimization Agent

Goal: Deliver all packages efficiently.
Tools:
- route_distance(start, end)
- truck_capacity_query(truck)
- package_priority(package_id)

Instructions:
1. Check package priority levels.
2. Assign packages to trucks based on capacity.
3. Map shortest delivery route.
4. Resolve conflicts (capacity, distance).
5. Deliver optimized plan + justification.

When to Use Agents

- Multi-step tasks
- Data access
- Planning workflows

When NOT to Use Agents

- Simple Q&A
- Single prompt tasks

LangChain: Minimal Agent

- LLM + Calculator tool

```
from langchain_openai import ChatOpenAI
from langchain.agents import Tool,
AgentExecutor, create_react_agent
from langchain.tools import Calculator

llm = ChatOpenAI(model="gpt-4o-mini")

tools = [Calculator()]

prompt = """
You are a math agent. Use tools when needed.
"""

agent = create_react_agent(llm, tools, prompt)
executor = AgentExecutor(agent=agent,
tools=tools)

executor.invoke({"input": "What is 22 * 47?"})
```

Code Demo: Math Agent

- LLM
- Calculator tool
- Agent executor

Understanding Tool Calls

- LLM generates JSON/tool schema
- Controller executes call

Exercise Setup

- Modify a simple agent
- Add or change one tool

Exercise Instructions

- 1. Run starter agent
- 2. Add tool
- 3. Test

Live Example: Adding a Search Tool

- Define fake search()
- Register tool
- Run agent

Plan → Act → Review Agents

- Planning first
- Executing steps
- Refining

Mini Workflow Agent

- Clarify
- Plan
- Execute

Multi-Agent Systems

- Planner
- Researcher
- Writer

Failure Cases

- Hallucinated tools
- Infinite loops
- Bad assumptions

Safety & Guardrails

- Stop conditions
- Tool whitelists
- Retries

Final Demo

- Agent solves a small problem
- Shows multi-step reasoning

Summary

- Agents = LLM + tools + loops
- Great for multi-step tasks

Assignment

- Build a two-tool agent
- Explain design choices