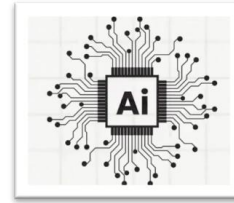


Agentic Design Patterns



In this session, we explore Agentic Design Patterns with a hands-on walkthrough of the ReAct Agent architecture. You'll learn how reasoning and acting are interleaved in modern AI agents, how LangChain simplifies agent setup, and how to integrate tools like web search APIs to build intelligent, tool-using agents.

01

What is ReACT Agent

Definition:

A ReAct Agent combines *Reasoning* and *Acting* in a loop to solve complex problems.

Inspired by humans:

It thinks, acts (like searching), observes results, and decides the next step — just like a human would.

How it works:

Thought → Action → Observation → Thought → ... until
Final Answer.

Framework Used:

Built using **LangChain** and **OpenAI**.

```
> Entering new AgentExecutor chain...
Thought : I need to get invoice amount of company C.
Action : GetInvoice[C]
Observation : 20000
Thought : I need to get invoice amount of company F.
Action : GetInvoice[F]
Observation : 4100
Thought : I need to get total amount of obtained amount C and F.
Action : Total[20000 4100]
Observation : 24100
Thought : I need to get invoice amount of company A.
Action : GetInvoice[A]
Observation : 2000
Thought : I need to get invoice amount of company E.
Action : GetInvoice[E]
Observation : 1000
Thought : I need to get total amount of obtained amount A and E.
Action : Total[2000 1000]
Observation : 3000
Thought : I need to get difference of obtained amount C, F and the total of A, E.
Action : Diff[24100 3000]
Observation : 21100
Thought : So the answer is 21100.
Action : Finish[21100]

> Finished chain.
Out[11]: '21100'
```

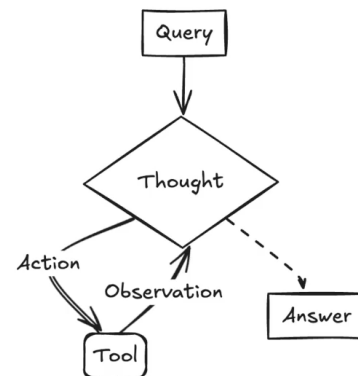
02

What Will We Build?

Project Goal

We'll build a real-world **ReAct Agent** that can:

- ❑ Think through problems step-by-step
- ❑ Take actions (like searching the web)
- ❑ Observe and adapt based on results
- ❑ Provide a final answer with full transparency



03

You Will Learn To:

- ❑ Implement the **ReAct Loop** using LangChain
- ❑ Build tools like a **custom DuckDuckGo search tool**
- ❑ Manage **conversation memory** and step tracking
- ❑ Design an **interactive UI** using Streamlit
- ❑ Debug and explain agent decisions with structured output

End Result:

A working, intelligent **AI agent** that mimics human problem-solving

Live UI + Reasoning Engine + Search Tool

04