Lecture 05: Planning and Decision-Making in Agents

🎯 Learning Objectives

By the end of this lecture, you should be able to:

- Understand how agents plan and make decisions based on goals and context.
- Learn about the ReAct and Tree-of-Thought (ToT) prompting strategies.
- Implement basic planning loops using LLMs.
- Differentiate reactive responses from deliberative planning.

🧩 Key Concepts

Agent Planning

- **Planning**: The ability to decompose a goal into a sequence of intermediate actions or steps.
- **Decision-making**: Choosing the next best action based on the agent's internal state and external environment.

Prompt-Based Planning

- Use prompting strategies to guide LLMs in reasoning:
  - **ReAct (Reason + Act)**: Combines step-by-step thinking with actions.
  - **Tree-of-Thought (ToT)**: Explores multiple reasoning paths before making a decision.
  - **Chain-of-Thought (CoT)**: Focuses on coherent linear reasoning.

Example: ReAct Prompt Structure

```
Thought: I need to look up the weather before recommending clothes.
Action: Search["weather in Paris"]
Observation: It's 12°C and raining.
Thought: Since it's cold and wet, I'll suggest a jacket and umbrella.
Final Answer: Wear a jacket and take an umbrella.
```

🛠️ Required Tools/Libraries

- OpenAI API or Hugging Face models
- LangChain (optional for tool orchestration)
- Python

🔬 Hands-on Exercise: Implement a ReAct Agent

Goal: Create a simple agent that can solve a trivia task using reasoning and simulated tools.

Step 1: Define a prompt template

```
You are a smart agent that can think and act. Use Thought and Action steps to
solve the problem.

Question: What U.S. state is known as the Sunshine State?

Thought: I should recall U.S. state nicknames.
Action: Lookup["state nicknames"]
Observation: Florida is the Sunshine State.
Thought: Now I know the answer.
Final Answer: Florida
```

Step 2: Implement the logic in Python (pseudo-interpreter)

```
1. Feed a ReAct-style prompt to the LLM.
2. Simulate tool use with a lookup dictionary or stub function.
3. Loop until the agent outputs `Final Answer`.
```

Step 3: Analyze

- What if the agent makes a wrong assumption?
- How can you improve it with multiple thoughts or fallback strategies?

Bonus:

- Try implementing a Tree-of-Thought style: generate multiple reasoning paths and compare them before acting.
- Compare outputs with and without structured reasoning prompts.
