DAY 8 – Planning Agents (Task Decomposition + Reasoning)

What is Planning in Agentic AI?

Planning Agent takes a large user goal and intelligently breaks it into smaller, manageable steps before any action is taken.

Why break down goals?

If you simply tell a chatbot:
"Research RAG → summarize → save notes → create PDF."
It will likely fail because it tries to do everything at once without structure.

What does a planner agent do?

- Splits the goal into small tasks
- Understands the order and dependencies between tasks
- Creates a clear, step-by-step plan
- Returns output in a structured format (like JSON)
- Leaves execution to a separate agent (Executor)

Think of planning as the brain that thinks and plans ahead. Execution is like the hands that carry out the plan.

Why Planning is Needed?

Agents often fail because they:
- Skip important steps
- Hallucinate or invent wrong details
- Execute the wrong actions
- Call the wrong tools

Planning fixes these by:
- Creating a clear roadmap
- Avoiding errors through stepwise logic
- Making the agent's behavior consistent and structured

3 Major Planning Styles (Simplified)

Agents think in different ways depending on the task:

1. Chain of Thought (CoT)

Think step-by-step. The LLM writes down each reasoning step before answering.

Example task:
"How to study RAG?"
CoT output:
- Learn embeddings
- Learn vector databases
- Learn document splitting
- Learn retrieval
- Learn evaluation

2. ReAct (Reason + Act)

The LLM alternates between:
- Thinking
- Deciding an action
- Using tools
- Thinking again

This helps it handle
- Web search
- File reading
- Calculations
- API calls

3. Tree of Thoughts (ToT)

The LLM explores multiple reasoning paths in parallel, then chooses the best one.
Used in creative writing, coding, and complex planning.

How LangChain and LangGraph Do Planning

LangChain
- Uses carefully designed prompts
- Uses templates and tools integration
- Planning happens through LLM reasoning in prompts

LangGraph
- More advanced graph-based connections
- Connect nodes like planner → executor → evaluator
- Highly suitable for production use

Build Your Own Planner Agent (Python Code)

Create a simple, practical planning agent that takes a goal and returns steps in JSON.

Step 1 — Create `day6_planner.py`

Step 2 — Add Imports

```python
from langchain_openai import ChatOpenAI
from langchain.prompts import PromptTemplate
import json
```

Step 3 — Define Planner Prompt

```python
planner_prompt = """
```

You are a Planning Agent.

Your job:
- Understand the user goal.
- Break it into 3–7 clear steps.
- Do NOT execute the steps.
- Output in JSON only.

User Goal: {goal}

Return strictly this JSON:

```
{
  "goal": "...",
  "steps": ["step 1", "step 2", "..."]
}
```

Step 4 — Setup the LLM (OpenRouter Compatible)

```python
llm = ChatOpenAI(
    model="gpt-4o-mini",
    temperature=0  # deterministic output
)
```

Why temperature=0?
So the plan is consistent and reproducible.

Step 5 — Define Planner Function to Make Plans

```python
def make_plan(goal):
    prompt = planner_prompt.format(goal=goal)
    response = llm.invoke(prompt)
    return json.loads(response.content)
```

Test Your Planner

```python
plan = make_plan("Research RAG techniques and write a 200-word summary.")
print(plan)
```

Expected output:

```json
{
  "goal": "Research RAG techniques and write a 200-word summary",
  "steps": [
    "Identify high-quality sources on RAG",
    "Read 2–3 articles or documentation pages",
    "Extract important key points",
    "Write a rough summary",
    "Edit it to exactly 200 words"
  ]
}
```

This agent **does not execute any task**, it only *plans* the steps logically.

Practice Tasks

Try these user goals:

1. “Plan a 7-day weight loss diet.”
You should get 7 clearly defined steps.

2. “Search 5 AI tools, compare features, create a table.”
Expected steps: collect tools, search, extract features, build table, summarize.

3. “Learn RAG in 7 days.”
Expect daily topics like: Day 1 embeddings, Day 2 vector DB, etc.

Simple Explanation of Key Concepts

Task Decomposition

Breaking a big task into small, achievable sub-tasks.

Example: Make a YouTube video
- Script
- Voiceover
- Editing
- Thumbnail
- Upload

Reasoning
Agent’s ability to “think before acting”:
- Choose the right tool
- Decide the correct order
- Avoid mistakes

Why JSON Output?
- Clean and well-structured
- Easy to parse
- Can be used by executors later
- Ensures predictability