Prompts:

Prompts

Prompts are the core instructions that guide how language models behave and respond. LangChain helps structure, template, and dynamically format prompts to ensure consistent and context-aware outputs.

What is a Prompt?

In simple terms:

A prompt is like asking a smart assistant a question — the better you ask, the better the answer you get.

Definition:

A **prompt** is the input or instruction given to a Large Language Model (LLM) to guide its response. It acts as a *conversation starter* or *task command* that tells the model what you want it to do.

Types of Prompts

- ❑ Static vs Dynamic Prompts
- ❑ Zero-shot, One-shot, Few-shot Prompting
- ❑ Role-Based Prompts
- ❑ Instruction vs Conversational Prompts
- ❑ Template-based Prompts (used in LangChain)

Types of Prompts

Prompt Type	Description	Example	Use Case
Static vs Dynamic Prompts	- **Static**: Hardcoded prompt with no variables. - **Dynamic**: Uses variables/placeholders, can be reused with different inputs.	Static: "Summarize this article about AI." Dynamic: "Summarize this article: {text}"	Content generation, chatbots, automation
Zero-shot / One-shot / Few-shot	- **Zero-shot**: No examples, just instructions. - **One-shot**: One input-output example provided. - **Few-shot**: Few examples provided.	Zero-shot: "Translate to French: Good night" Few-shot: with 2–3 translation examples given	Language translation, Q&A, text classification
Role-Based Prompts	Assigns a persona or role to the LLM to control tone, depth, or format of output.	"You are a helpful medical expert. Explain diabetes in simple terms."	Agents, teaching bots, support chatbots
Instruction vs Conversational	- **Instruction**: Direct command-style prompts. - **Conversational**: Back-and-forth, human-style dialogue prompts.	Instruction: "Summarize the text." Conversational: "Hi, can you help summarize this?"	Task-specific tools vs chat interfaces
Template-based Prompts	Predefined templates using placeholders ({}) for reusability. Often used with LangChain or programmatic APIs.	Template: "Generate pros and cons of using {technology} in business"	Dynamic chains, scalable prompt-based systems

Dynamic and Reusable Prompts

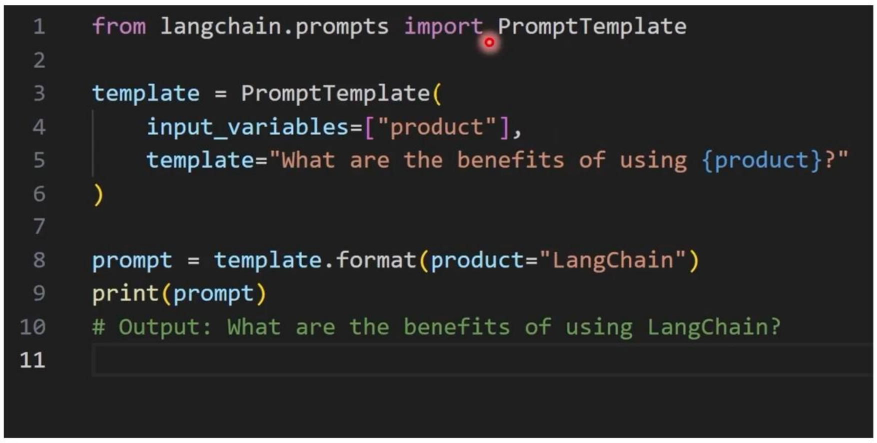
Definition

Dynamic and reusable prompts are prompt templates that can accept inputs at runtime, allowing them to be reused across multiple contexts or queries.

Explanation

- ❑ Instead of hardcoding the entire prompt each time, you define a **template** with variables (placeholders) that are dynamically filled during execution.
- ❑ This is useful in real-world applications like chatbots, Q&A systems, or summarization tools where only part of the prompt changes.

Dynamic and Reusable Prompts



```python
from langchain.prompts import PromptTemplate

template = PromptTemplate(
    input_variables=["product"],
    template="What are the benefits of using {product}?"
)

prompt = template.format(product="LangChain")
print(prompt)
# Output: What are the benefits of using LangChain?

```

Benefits:

- ❑ Encourages modular and DRY (Don't Repeat Yourself) coding
- ❑ Easy to maintain and scale
- ❑ Works seamlessly with chains and agents in LangChain

Role-Based Prompts

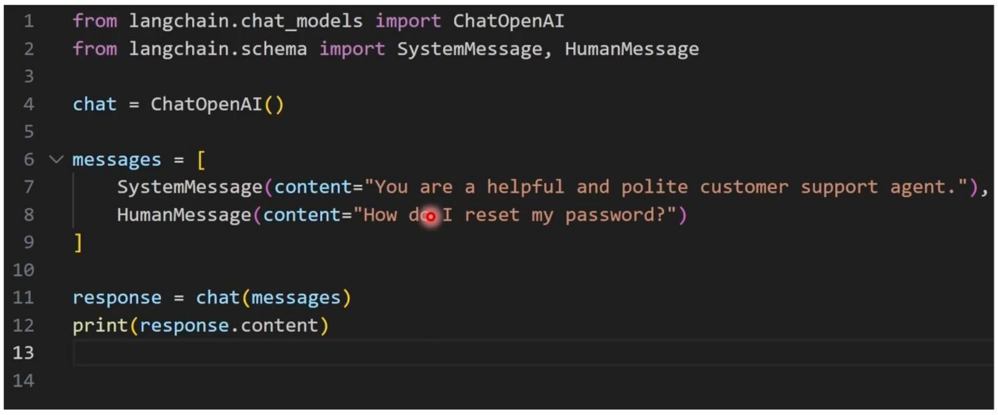
Definition

Role-based prompts instruct the LLM to assume a specific identity or role before responding. This provides **contextual behavior control** and improves response relevance and tone.

Explanation

- ❑ Roles can be anything: a teacher, doctor, interviewer, mentor, assistant, etc.
- ❑ Helps guide the model's tone, format, and depth of knowledge.

Role-Based Prompts



```python
from langchain.chat_models import ChatOpenAI
from langchain.schema import SystemMessage, HumanMessage

chat = ChatOpenAI()

messages = [
    SystemMessage(content="You are a helpful and polite customer support agent."),
    HumanMessage(content="How do I reset my password?")
]

response = chat(messages)
print(response.content)
```

Benefits:

- ❑ Improves reliability of tone and persona
- ❑ Ensures consistency across chatbot/agent interactions
- ❑ Enhances user trust and clarity

Few-Shot Prompting

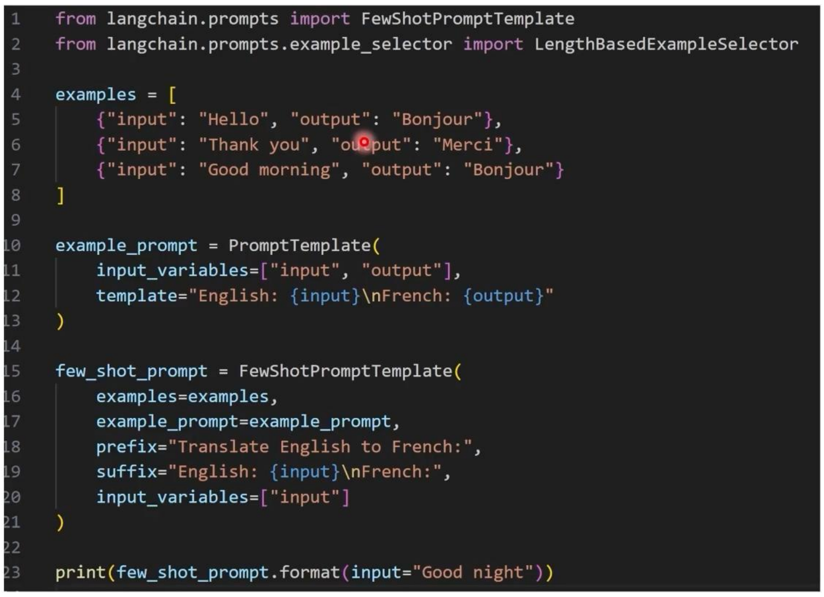
Definition

Few-shot prompting involves **giving the LLM a few examples** of input-output pairs to guide its behavior for the current task.

Explanation

- ❑ The idea is to show the model *how* to respond by providing demonstrations.
- ❑ This technique is useful for classification, summarization, translation, and formatting tasks

Few-Shot Prompting



```python
from langchain.prompts import FewShotPromptTemplate
from langchain.prompts.example_selector import LengthBasedExampleSelector

examples = [
    {"input": "Hello", "output": "Bonjour"},
    {"input": "Thank you", "output": "Merci"},
    {"input": "Good morning", "output": "Bonjour"}
]

example_prompt = PromptTemplate(
    input_variables=["input", "output"],
    template="English: {input}\nFrench: {output}"
)

few_shot_prompt = FewShotPromptTemplate(
    examples=examples,
    example_prompt=example_prompt,
    prefix="Translate English to French:",
    suffix="English: {input}\nFrench:",
    input_variables=["input"]
)

print(few_shot_prompt.format(input="Good night"))
```