Building an AI-Powered Conversational Assistant with Advanced Prompting Techniques

1. Objective

- Practice constructing conversation messages for Azure OpenAI chat completions using roles (system, user, assistant).
- Implement few-shot prompting for tasks like sentiment analysis within conversations.
- Use chain-of-thought prompting to encourage reasoning in AI responses.
- Demonstrate how to provide context through ongoing conversation messages and system prompts.
- Analyze and output results in a clear, structured format.

2. Problem Statement

- Building conversational AI requires effective prompt design and context management.
- By practicing multi-turn conversations and applying advanced prompting strategies, you can improve the quality and relevance of AI responses.
- This exercise focuses on simulating an event management assistant who can perform tasks such as providing conversation starters, analyzing sentiment, and reasoning through problems with context awareness.

3. Inputs / Shared Artifacts

- Azure OpenAI Resource: API key, endpoint URL, and deployment name (to be set as environment variables)
- Python environment with Azure OpenAI Python client installed.
- Example conversation message templates.
- Sample few-shot examples of sentiment analysis and chain-of-thought reasoning. (3 samples)

4. Expected Outcomes

- A Python script that:
  - Constructs conversation messages with properly assigned roles (system, user, assistant).
  - Implements few-shot examples within messages to guide AI behavior.
  - Uses chain-of-thought style prompts to generate reasoning.

    - Maintains context via conversation history and system prompts.
    - Produces sentiment analysis output with explanation.
- Clear, formatted output showing AI responses and analysis.

5. Concepts Covered

- Azure OpenAI chat completions API with conversation messages.
- Role assignment: system, user, assistant.
- Few-shot prompting to demonstrate expected outputs.
- Chain-of-thought prompting to encourage detailed reasoning.
- Context management via message history and system prompt content.
- Sentiment analysis within conversational context.

6. Example: Step-by-Step Instructions with Code

```
from openai import AzureOpenAI
import os

client = AzureOpenAI(
    api_version="2024-07-01-preview",
    azure_endpoint=os.getenv("AZURE_OPENAI_ENDPOINT"),
    api_key=os.getenv("AZURE_OPENAI_API_KEY"),
)


# Few-shot examples for sentiment analysis embedded in conversation
few_shot_examples = [
    {"role": "user", "content": "Analyze the sentiment of this text: 'I
love attending networking events!'"},
    {"role": "assistant", "content": "Sentiment: Positive. The text
expresses enthusiasm and enjoyment."},
    {"role": "user", "content": "Analyze the sentiment of this text:
'Networking can be really stressful sometimes.'"},
    {"role": "assistant", "content": "Sentiment: Negative. The text
shows discomfort and stress related to networking."},
]

# Conversation messages setup with system prompt and user question
conversation_messages = [
    {"role": "system", "content": "You are a helpful event management
assistant."},
]

# Append few-shot examples to conversation for context
conversation_messages.extend(few_shot_examples)

# Add user question with chain-of-thought prompt to encourage reasoning
conversation_messages.append(
    {
        "role": "user",
        "content": (
            "What are some good conversation starters at networking
events? "
            "Explain your reasoning step-by-step."
        ),
    }
)

# Call Azure OpenAI chat completion with conversation messages
response = client.chat.completions.create(
    model="gpt-4o-mini",
    messages=conversation_messages,
    temperature=0.7,
)

# Extract and print assistant reply
assistant_reply = response.choices[0].message.content
print("Assistant Response:\n")
print(assistant_reply)
```

7. Final Submission Checklist

- Python script demonstrating:
  - Proper conversation message construction with roles.
  - Few-shot prompting examples included in the conversation.
  - Chain-of-thought prompting applied.
  - Context preserved through message history and system prompt.
- Sample output of assistant responses printed and well-formatted. (3 samples)
- (Optional) Brief documentation explaining:
  - How few-shot prompting influences output.
  - Benefits of chain-of-thought prompting for reasoning tasks.
  - Importance of context via system prompts and conversation history.