AI Agent Assignment – Final Submission

# SECTION 1: BASIC DETAILS

Name: Mirsha Kudukkan

AI Agent Title / Use Case: Python Interview Revision Agent

# SECTION 2: PROBLEM FRAMING

1.1 What problem does your AI Agent solve?  
This AI Agent helps users prepare for Python interview questions by providing structured, layered assistance through multiple intelligent steps.

1.2 Why is this agent useful?  
It gives clarity, saves time, and simulates how a mentor might help revise technical topics.

1.3 Who is the target user?  
Self-taught or entry-level data science/engineering candidates preparing for Python interviews.

1.4 What not to include?  
Does not cover full-stack development, deployment topics, or advanced system design.

# SECTION 3: 4-LAYER PROMPT DESIGN

## 3.1 INPUT UNDERSTANDING

Prompt: What Python topic would you like to revise today?

What is this prompt responsible for?  
It identifies and scopes the user’s query into a clear, revisable topic.

## 3.2 STATE TRACKER

Prompt: Keep track of previously asked topics and user preferences such as format (brief/long).

How does this help the agent remember?  
Simulates memory using system messages to avoid repetition and maintain session coherence.

## 3.3 TASK PLANNER

Prompt: Break the revision into layers: definition, example, application, and common mistakes.

Did you use chaining?  
Yes. One topic flows into multiple sub-tasks (definition → example → Q&A), simulating a guided study flow.

## 3.4 OUTPUT GENERATOR

Prompt: Summarize the entire topic in a revision-friendly format using headings and bullet points.

Special behavior?  
Used markdown-like formatting for clarity and examples to reinforce understanding.

# SECTION 4: CHATGPT EXPLORATION LOG

- Iteration 1: Agent understood topic input but didn't break it down.  
- Iteration 2: Added task planner for better structure.  
- Iteration 3: Finalized memory and output format to simulate tutor-like flow.

# SECTION 5: OUTPUT TESTS (Optional but Recommended)

Test 1: Normal input  
Input: 'Explain list comprehension in Python'  
Output: [Summary of definition, syntax, and examples]

Test 2: Vague input  
Input: 'Python stuff'  
Output: Prompted user to select specific subtopics (loops, strings, functions)

Test 3: Invalid input  
Input: ''  
Output: Asked the user to type a valid Python topic.

# SECTION 6: REFLECTION

6.1 Hardest part?  
Figuring out how to simulate memory and split prompts logically across components.

6.2 Most enjoyable part?  
Seeing how the layered prompt model made the agent behave like a smart tutor.

6.3 If given more time?  
Would add a testing dashboard and user session saving.

6.4 What did you learn?  
Prompt structure and chaining matter far more than expected.

6.5 Did you ever feel stuck?  
Yes, especially in the beginning. Solved it by iterating slowly and asking ChatGPT questions like a teammate.

# SECTION 7: HACK VALUE (Optional)

Simulated memory using prompt variables and system messages. Also added the ability to summarize at the end for real-world usability.