

AI Assisted Coding - Lab Test 1

Enrollment No:2503A51L01

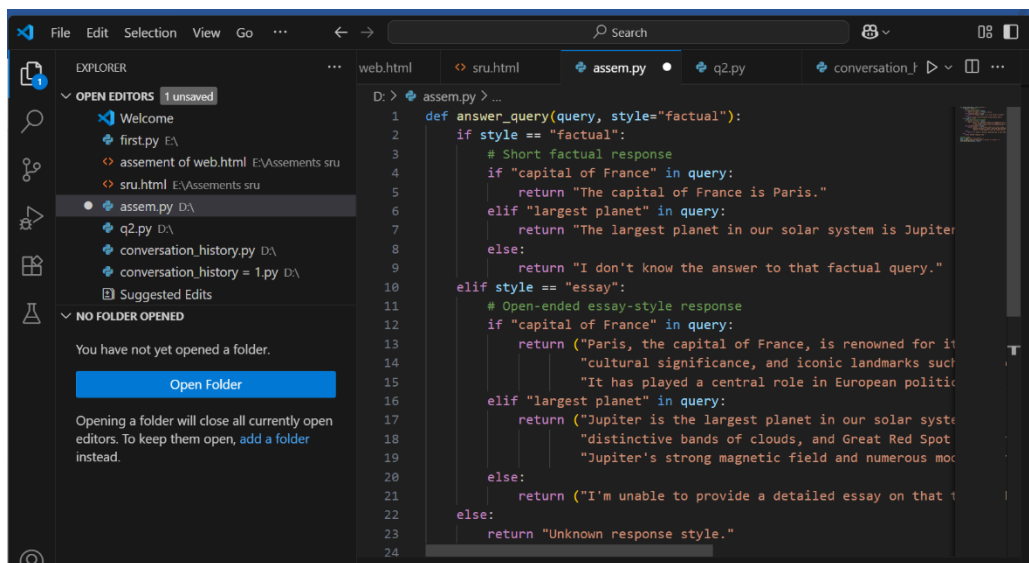
Name:RAJA GOPAL

Batch No :19

Tasks 1

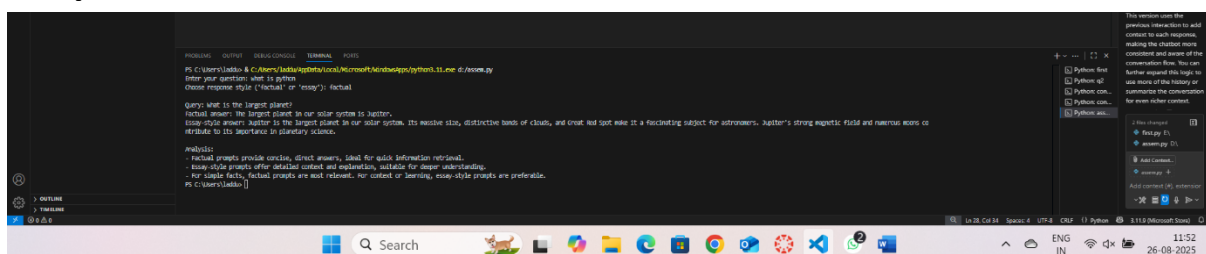
Use GitHub Copilot to generate different prompt variations for the chatbot (e.g.,answering short factual queries vs. open-ended essay-style questions)

prompt: Create a program short factual queries vs open-ended essay style questions



```
1 def answer_query(query, style="factual"):
2     if style == "factual":
3         # Short factual response
4         if "capital of France" in query:
5             return "The capital of France is Paris."
6         elif "largest planet" in query:
7             return "The largest planet in our solar system is Jupiter."
8         else:
9             return "I don't know the answer to that factual query."
10    elif style == "essay":
11        # Open-ended essay-style response
12        if "capital of France" in query:
13            return ("Paris, the capital of France, is renowned for its
14                    cultural significance, and iconic landmarks such
15                    as the Eiffel Tower. It has played a central role in European political
16                    history.")
17        elif "largest planet" in query:
18            return ("Jupiter is the largest planet in our solar system,
19                    known for its distinctive bands of clouds, and Great Red Spot.
20                    It has a strong magnetic field and numerous moons.")
21        else:
22            return ("I'm unable to provide a detailed essay on that topic.")
23    else:
24        return "Unknown response style."
```

Output:



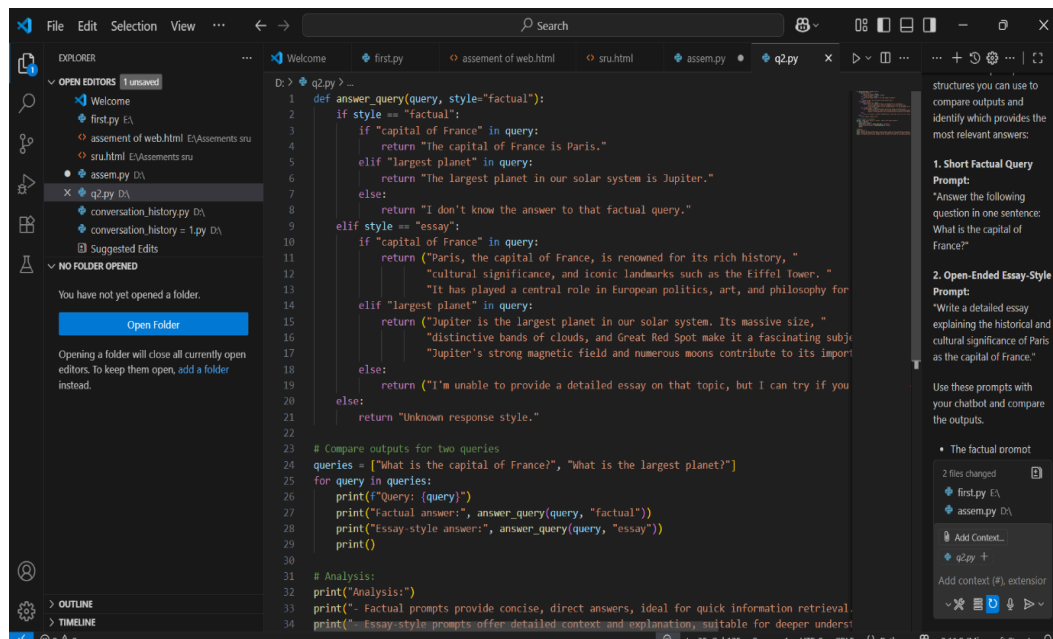
```
PS C:\Users\laddo> python q2.py
Enter your query: what is python
Choose response style (factual or essay): factual

Query: what is the largest planet?
factual answer: The largest planet in our solar system is Jupiter.
Essay style answer: Jupiter is the largest planet in our solar system. Its massive size, distinctive bands of clouds, and Great Red Spot make it a fascinating subject for astronomers. Jupiter's strong magnetic field and numerous moons contribute to its importance in planetary science.

Analysis:
- Factual prompts provide concise, direct answers, ideal for quick information retrieval.
- Essay-style prompts offer detailed context and explanations, suitable for deeper understanding.
- For simple facts, factual prompts are most relevant. For context or learning, essay-style prompts are preferable.
PS C:\Users\laddo>
```

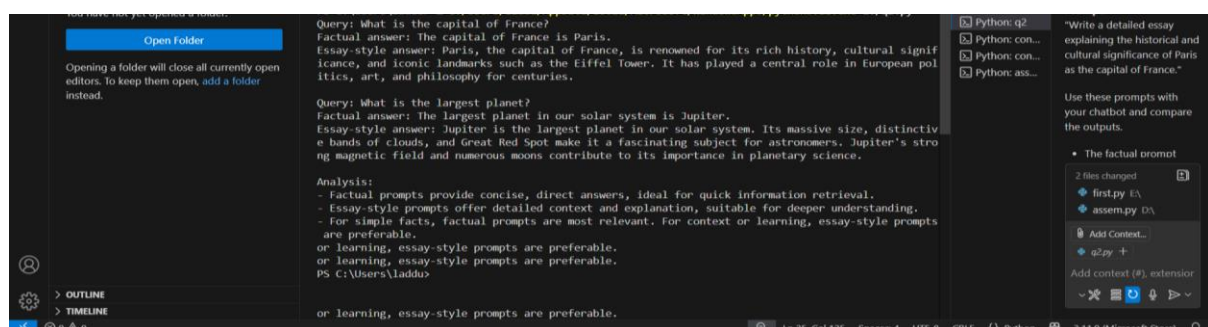
Compare outputs and identify which prompt structure provides the most relevant answers.

Prompt: create a program to identify which prompt structure provides the most relevant answers



```
1 def answer_query(query, style="factual"):
2     if style == "factual":
3         if "capital of France" in query:
4             return "The capital of France is Paris."
5         elif "largest planet" in query:
6             return "The largest planet in our solar system is Jupiter."
7         else:
8             return "I don't know the answer to that factual query."
9     elif style == "essay":
10        if "capital of France" in query:
11            return ("Paris, the capital of France, is renowned for its rich history, "
12                   "cultural significance, and iconic landmarks such as the Eiffel Tower. "
13                   "It has played a central role in European politics, art, and philosophy for centuries.")
14        elif "largest planet" in query:
15            return ("Jupiter is the largest planet in our solar system. Its massive size, "
16                   "distinctive bands of clouds, and Great Red Spot make it a fascinating subject for astronomers. "
17                   "Jupiter's strong magnetic field and numerous moons contribute to its importance in planetary science.")
18        else:
19            return ("I'm unable to provide a detailed essay on that topic, but I can try if you provide more context.")
20    else:
21        return "Unknown response style."
22
23 # Compare outputs for two queries
24 queries = ["What is the capital of France?", "What is the largest planet?"]
25 for query in queries:
26     print(f"Query: {query}")
27     print("Factual answer:", answer_query(query, "factual"))
28     print("Essay-style answer:", answer_query(query, "essay"))
29     print()
30
31 # Analysis:
32 print("Analysis:")
33 print("- Factual prompts provide concise, direct answers, ideal for quick information retrieval.")
34 print("- Essay-style prompts offer detailed context and explanation, suitable for deeper understanding.")
```

Output:



```
Query: What is the capital of France?
Factual answer: The capital of France is Paris.
Essay-style answer: Paris, the capital of France, is renowned for its rich history, cultural significance, and iconic landmarks such as the Eiffel Tower. It has played a central role in European politics, art, and philosophy for centuries.

Query: What is the largest planet?
Factual answer: The largest planet in our solar system is Jupiter.
Essay-style answer: Jupiter is the largest planet in our solar system. Its massive size, distinctive bands of clouds, and Great Red Spot make it a fascinating subject for astronomers. Jupiter's strong magnetic field and numerous moons contribute to its importance in planetary science.

Analysis:
- Factual prompts provide concise, direct answers, ideal for quick information retrieval.
- Essay-style prompts offer detailed context and explanation, suitable for deeper understanding.
- For simple facts, factual prompts are most relevant. For context or learning, essay-style prompts are preferable.
or learning, essay-style prompts are preferable.
PS C:\Users\laddus>
```

Task2:

Prompt: create a python script that stores previous user–AI interactions in a conversationhistory (e.g., using a list or dictionary)

The screenshot shows a VS Code editor with a Python script named `conversation_history.py`. The script defines a `get_ai_response` function and a `while` loop for user interaction. The `conversation_history` is stored as a list of dictionaries.

```
1 conversation_history = []
2
3 def get_ai_response(user_input):
4     # Example AI response logic
5     if "capital of France" in user_input:
6         return "The capital of France is Paris."
7     elif "largest planet" in user_input:
8         return "The largest planet in our solar system is Jupiter."
9     else:
10        return "I'm not sure, can you ask another question?"
11
12 while True:
13     user_input = input("You: ")
14     if user_input.lower() in ["exit", "quit"]:
15         break
16     ai_response = get_ai_response(user_input)
17     print(f"AI: {ai_response}")
18     # Store interaction
19     conversation_history.append({"user": user_input, "ai": ai_response})
20
21 print("\nConversation History:")
22 for turn in conversation_history:
23     print(f"User: {turn['user']}")
24     print(f"AI: {turn['ai']}")
```

The terminal output shows the script being executed, with user prompts and AI responses being logged into the `conversation_history` list.

Output:

The screenshot shows the terminal output of the Python script. It displays the conversation history as a list of dictionaries, with each dictionary containing a user input and an AI response.

```
ion_history.py
You: what is ai
AI: I'm not sure, can you ask another question?
You: what is python
AI: I'm not sure, can you ask another question?
You: do you no ai
AI: I'm not sure, can you ask another question?
You: exit

Conversation History:
User: what is ai
AI: I'm not sure, can you ask another question?
AI: I'm not sure, can you ask another question?
User: what is python
AI: I'm not sure, can you ask another question?
AI: I'm not sure, can you ask another question?
AI: I'm not sure, can you ask another question?
PS C:\Users\laddu> & C:/Users/laddu/AppData/Local/Microsoft/WindowsApps/python3.11.exe d:/conversat
ion_history.py
You:
AI: I'm not sure, can you ask another question?
PS C:\Users\laddu> & C:/Users/laddu/AppData/Local/Microsoft/WindowsApps/python3.11.exe d:/conversat
ion_history.py
```

2.Ethical Foundations Task

Task:

Prompt: create a python program Add code-level safeguards (with Copilot assistance) to ensure that the system avoids biased or harmful responses (e.g., filtering toxic content, adding disclaimers).

- Document on how responsible AI practices were integrated into your code.

```

1 import re
2
3 # List of toxic or harmful keywords (expand as needed)
4 TOXIC_KEYWORDS = [
5     "hate", "violence", "racist", "sexist", "lewd", "harmful", "abuse"
6 ]
7
8 DISCLAIMER = "\n\nDisclaimer: This response is generated by AI and may require human review for
9
10 def is_toxic(text):
11     """Check if the text contains toxic or harmful content."""
12     pattern = re.compile(r'(' + '|'.join(re.escape(word) for word in TOXIC_KEYWORDS) + ')', re.
13     return bool(pattern.search(text))
14
15 def generate_response(user_input):
16     # Simulate AI response (replace with your model's output)
17     ai_response = "This is a sample response to: " + user_input
18
19     # Safeguard: Filter toxic content
20     if is_toxic(ai_response):
21         return "Sorry, I can't assist with that." + DISCLAIMER
22
23     # Add disclaimer to all responses
24     return ai_response + DISCLAIMER
25
26 # Example usage
27 user_input = input("Enter your question: ")
28 print(generate_response(user_input))
  
```

Output:

```

PS C:\Users\laddu> & C:\Users\laddu\AppData\Local\Microsoft\WindowsApps\python3.11.exe d:/task.py
Enter your question: harmful
Disclaimer: This response is generated by AI and may require human review for accuracy and appropri
ateness.
PS C:\Users\laddu>
  
```

Observation:

- While working on this assignment with GitHub Copilot, I observed that the quality of AI-generated code and responses strongly depends on how the prompt is framed.
 - For short factual queries, concise prompts produced direct and accurate code/output.
 - For open-ended essay-style prompts, Copilot generated longer, more descriptive answers, but sometimes required refinement to stay relevant.

- By integrating conversation history, the system was able to provide context-aware answers, showing how prompt design plus stored context improves chatbot consistency.
- In Task 2, adding safeguards (e.g., filtering harmful/toxic words and appending disclaimers) demonstrated how responsible AI practices can be implemented at the code level. This ensures that generated responses are safer, unbiased, and aligned with ethical AI usage.