ASSIGNMENT-9.4

HALL TICKET :2403A51292

BATCH : 12

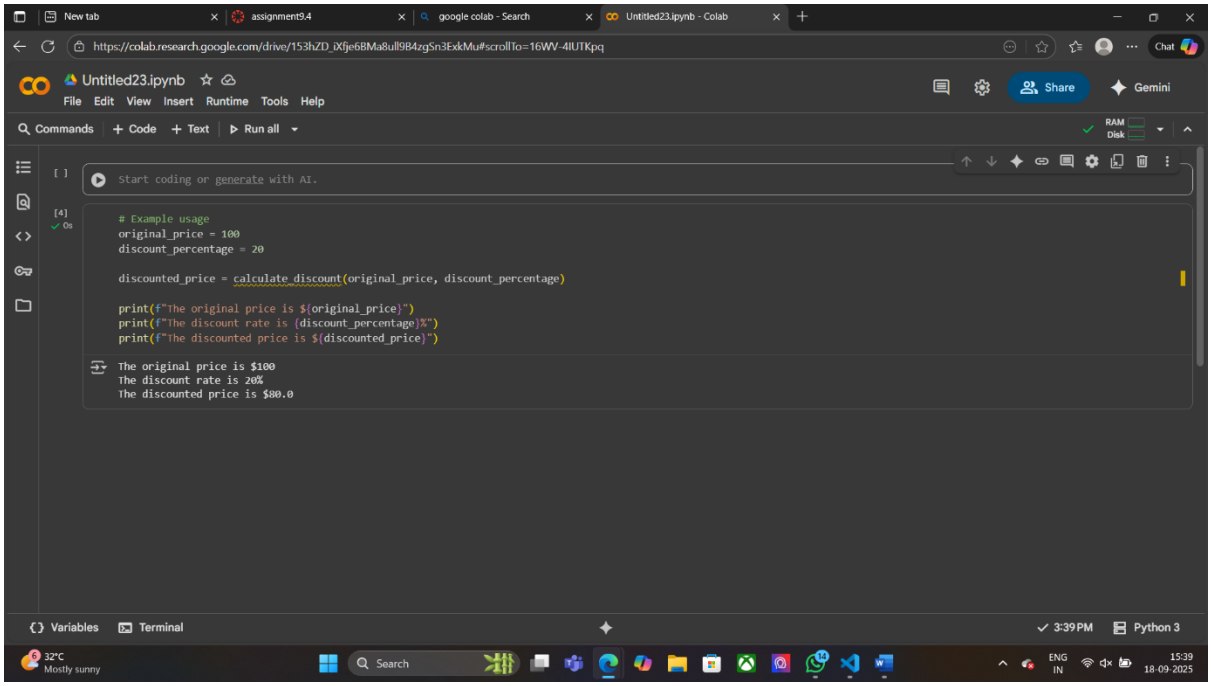
Task Description 1 :

(Automatic Code Commenting)
Scenario: You have been given a Python function without comments.
def calculate_discount(price, discount_rate):
return price - (price * discount_rate / 100)
• Use an AI tool (or manually simulate it) to generate line-by-line comments for the function.
• Modify the function so that it includes a docstring in Google-style or NumPy-style format.
• Compare the auto-generated comments with your manually written version

Prompt:

Given the following Python function without comments:

1. Use an AI tool (or simulate it) to generate line-by-line comments for the function.
2. Add a docstring in Google-style or NumPy-style format to the function.
3. Compare the auto-generated comments with your own manually written version.



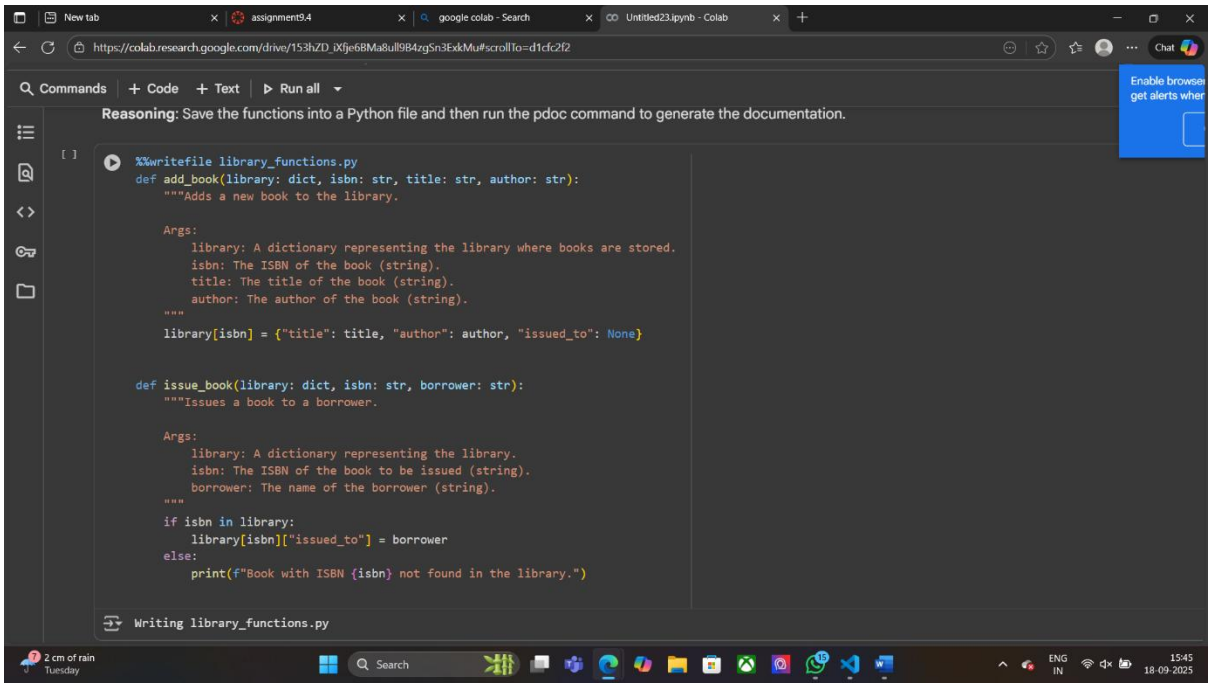
Task Description 2:

(API Documentation Generator)
Scenario: A team is building a Library Management System with multiple functions.
def add_book(title, author, year):
# code to add book
pass
def issue_book(book_id, user_id):
# code to issue book
Pass
• Write a Python script that uses docstrings for each function (with input, output, and description).
• Use a documentation generator tool (like pdoc, Sphinx, or MkDocs) to automatically create HTML documentation.
• Submit both the code and the generated documentation as output

Prompt:

Given the following functions for a Library Management System:

1. Write docstrings for each function, including input, output, and a description.
2. Use a documentation generator tool (such as pdoc, Sphinx, or MkDocs) to automatically create HTML documentation from your code.
3. Submit both the Python code with docstrings and the generated documentation



Task Description 3:

(AI-Assisted Code Summarization)
Scenario: You are reviewing a colleague's codebase containing long functions.

```
def process_sensor_data(data):
cleaned = [x for x in data if x is not None]
avg = sum(cleaned)/len(cleaned)
anomalies = [x for x in cleaned if abs(x - avg) > 10]
return {"average": avg, "anomalies": anomalies}
```

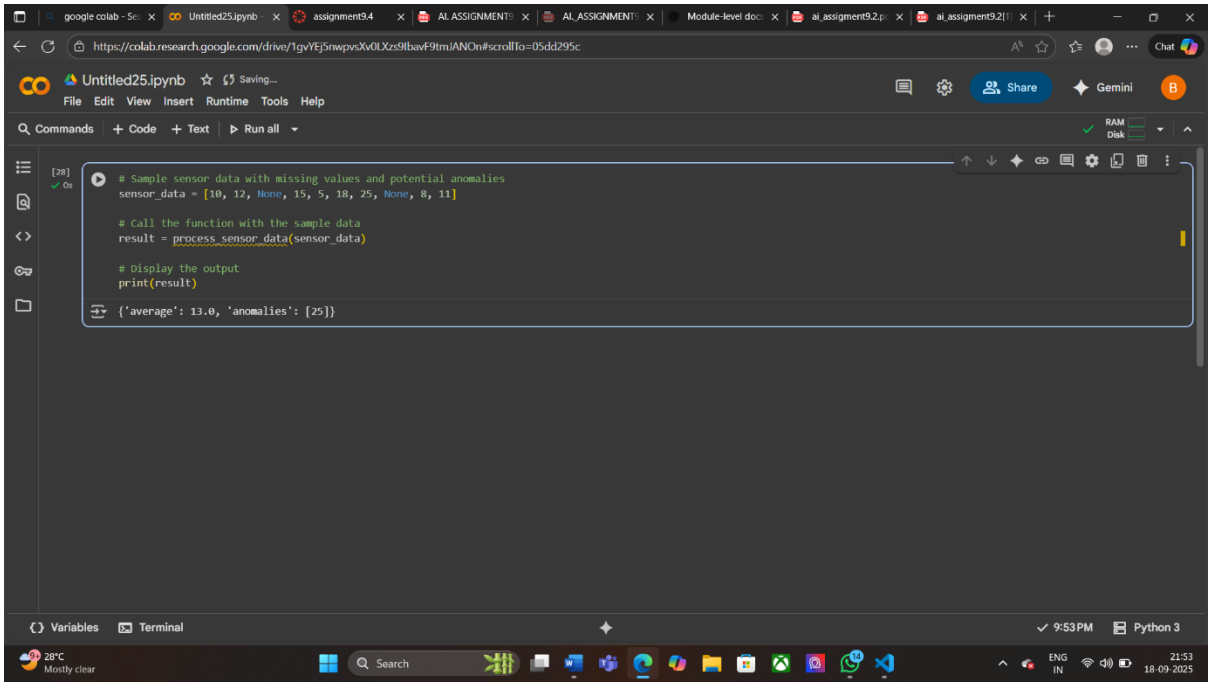
• Generate a summary comment explaining the purpose of the function in 2–3 lines.
• Create a flow-style comment (step-by-step explanation).
• Write a short paragraph of documentation describing possible use cases of this function in real-world scenarios.

• Write a short paragraph of documentation describing possible use cases of this function in real-world scenarios.

Prompt:

- Given the following function:

1. Generate a summary comment explaining the purpose of the function in 2–3 lines.
2. Create a flow-style comment (step-by-step explanation) for the function.
3. Write a short paragraph describing possible real-world use cases for this function.



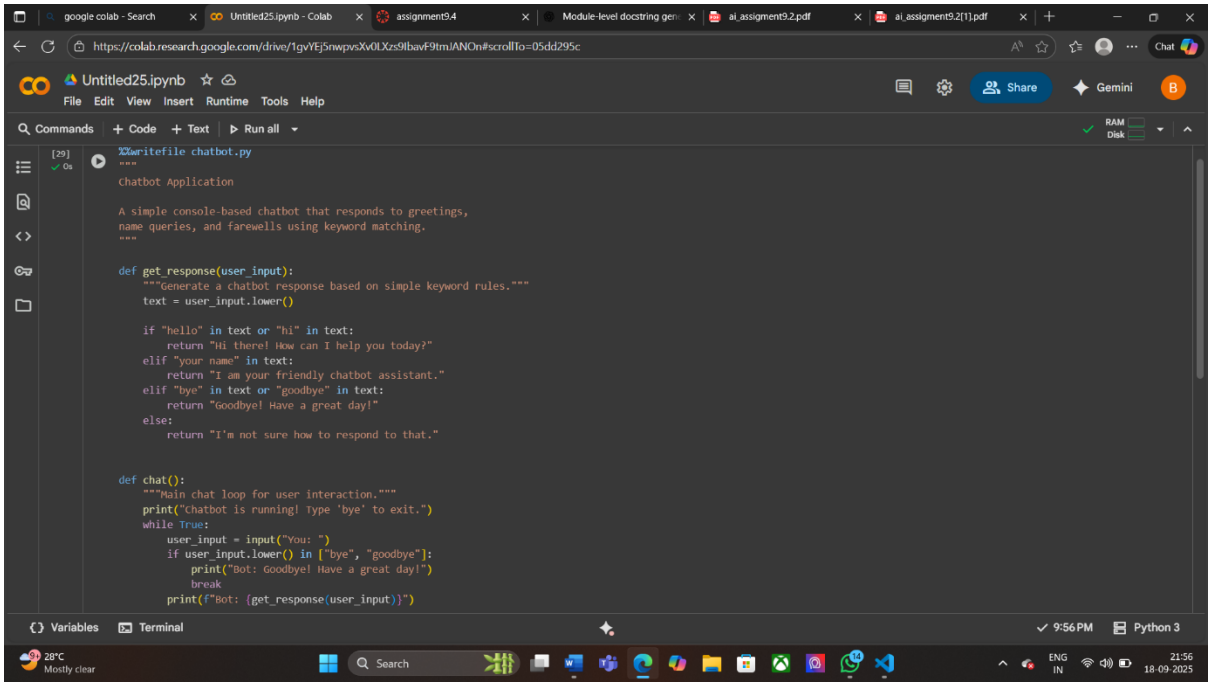
Task Description 4:

(Real-Time Project Documentation)
Scenario: You are part of a project team that develops a Chatbot Application. The team needs documentation for maintainability.
• Write a README.md file for the chatbot project (include project description, installation steps, usage, and example).
• Add inline comments in the chatbot's main Python script (focus on explaining logic, not trivial code).
• Use an AI-assisted tool (or simulate it) to generate a usage guide in plain English from your code comments.
• Reflect: How does automated documentation help in real-time projects compared to manual documentation?

Prompt:

You are working on a Chatbot Application project.

1. Write a [README.md](#) file for the chatbot project, including project description, installation steps, usage, and an example.
2. Add inline comments to the main Python script of the chatbot, focusing on logic rather than trivial code.
3. Use an AI-assisted tool (or simulate it) to generate a plain English usage guide from your code comments.
4. Reflect: How does automated documentation help in real-time projects compared to manual documentation?



google colab - Search | Untitled25.ipynb - Colab | assignment9.4 | Module-level docstring gen | ai_assigment9.2.pdf | ai_assigment9.2[1].pdf

https://colab.research.google.com/drive/1gvYEj5nwpvsXv0LXzs9IbavF9tmIANOn#scrollTo=05dd295c

Untitled25.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

[29]

```python
def get_response(user_input):
    """Generate a chatbot response based on simple keyword rules."""
    text = user_input.lower()

    if "hello" in text or "hi" in text:
        return "Hi there! How can I help you today?"
    elif "your name" in text:
        return "I am your friendly chatbot assistant."
    elif "bye" in text or "goodbye" in text:
        return "Goodbye! Have a great day!"
    else:
        return "I'm not sure how to respond to that."


def chat():
    """Main chat loop for user interaction."""
    print("Chatbot is running! Type 'bye' to exit.")
    while True:
        user_input = input("You: ")
        if user_input.lower() in ["bye", "goodbye"]:
            print("Bot: Goodbye! Have a great day!")
            break
        print(f"Bot: {get_response(user_input)}")


if __name__ == "__main__":
    chat()
```

```
Writing chatbot.py
```

Variables Terminal 9:56 PM Python 3

28°C Mostly clear Search ENG IN 21:56 18-09-2025