ROLL NO:2303A51820 BATCH:26

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE	DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Program Name: B. Tech	Assignment Type: Lab	Academic Year:2025-2026
Course Coordinator Name	Dr. Rishabh Mittal	
Instructor(s) Name	Mr. S Naresh Kumar Ms. B. Swathi Dr. Sasanko Shekhar Gantayat Mr. Md Sallauddin Dr. Mathivanan Mr. Y Srikanth Ms. N Shilpa Dr. Rishabh Mittal (Coordinator) Dr. R. Prashant Kumar Mr. Ankushavali MD Mr. B Viswanath Ms. Sujitha Reddy Ms. A. Anitha Ms. M.Madhuri Ms. Katherashala Swetha Ms. Velpula sumalatha Mr. Bingi Raju	

Course Code	23CS002PC304	Course Title	AI Assisted Coding
Year/Sem	III/II	Regulation	R23
Date and Day of Assignment	Week3 – Wednesday	Time(s)	23CSBTB01 To 23CSBTB52
Duration	2 Hours	Applicable to Batches	All batches

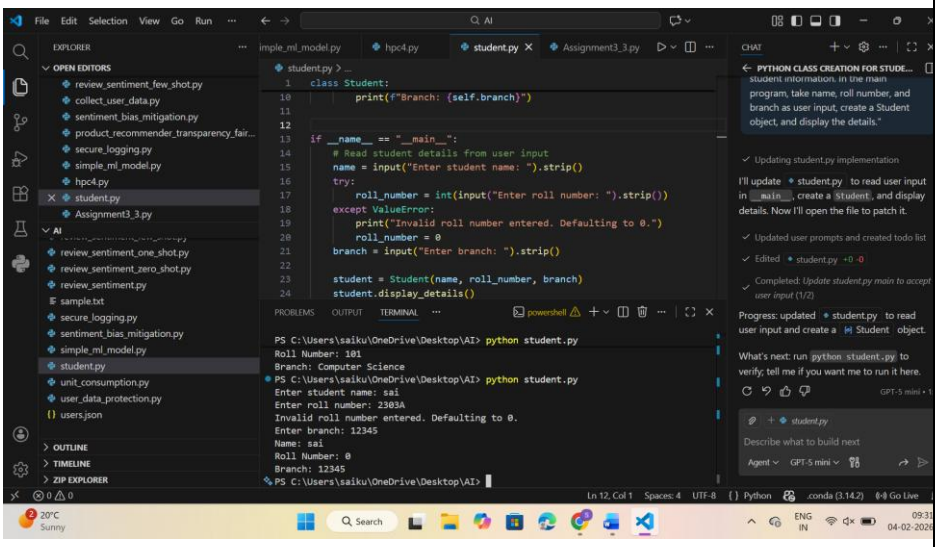
AssignmentNumber:6.3(Present assignment number)/**24**(Total number of assignments)

Q.No.	Question	*Expected Time to complete*
1	**Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals** **Lab Objectives** • To explore AI-powered auto-completion features for core Python constructs such as classes, loops, and conditional statements. • To analyze how AI tools suggest logic for object-oriented programming and control structures. • To evaluate the correctness, readability, and completeness of AI-generated Python code.	Week3 - Wednesday

Lab Outcomes (LOs)
After completing this lab, students will be able to:
• Use AI tools to generate and complete Python class definitions and methods.
• Understand and assess AI-suggested loop constructs for iterative tasks.
• Generate and evaluate conditional statements using AI-driven prompts.
• Critically analyze AI-assisted code for correctness, clarity, and efficiency.

Task Description #1: Classes (Student Class)

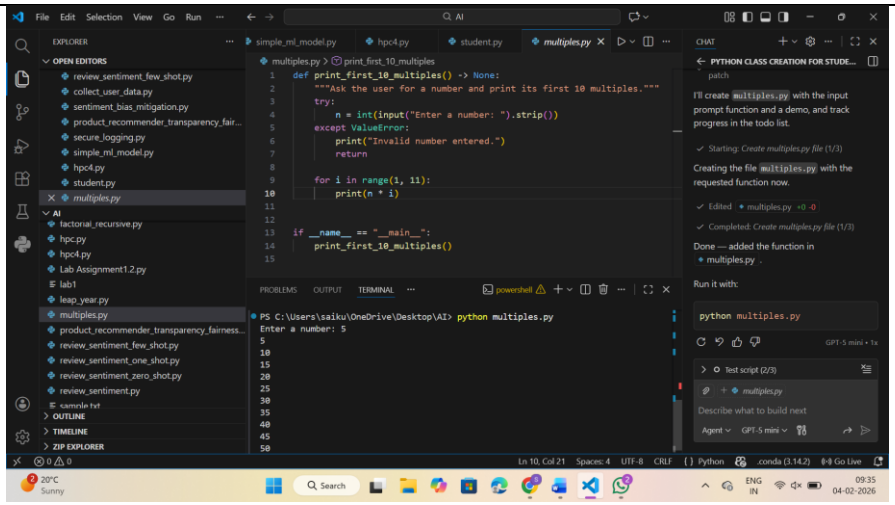
Scenario
You are developing a simple student information management module.
Prompt: Write a Python program that defines a class named Student with attributes name, roll number, and branch. Use a constructor to initialize these values. Add a method display_details() to print student information. In the main program, take name, roll number, and branch as user input, create a Student object, and display the details.

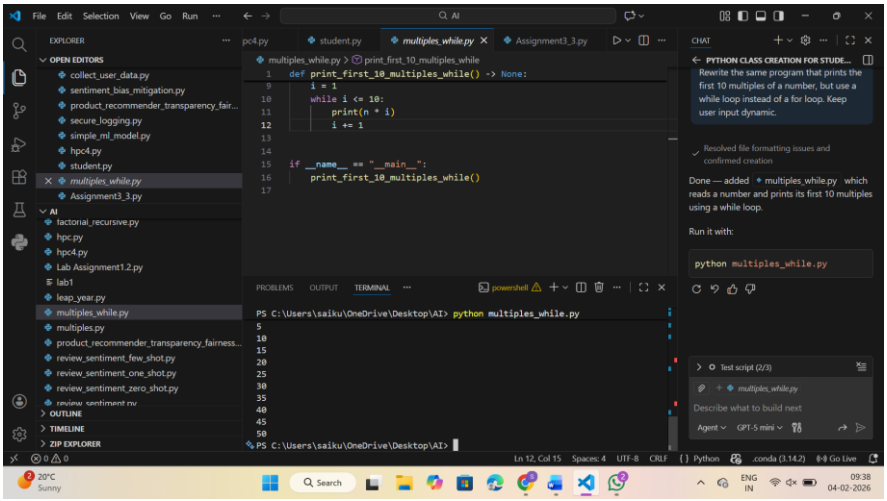


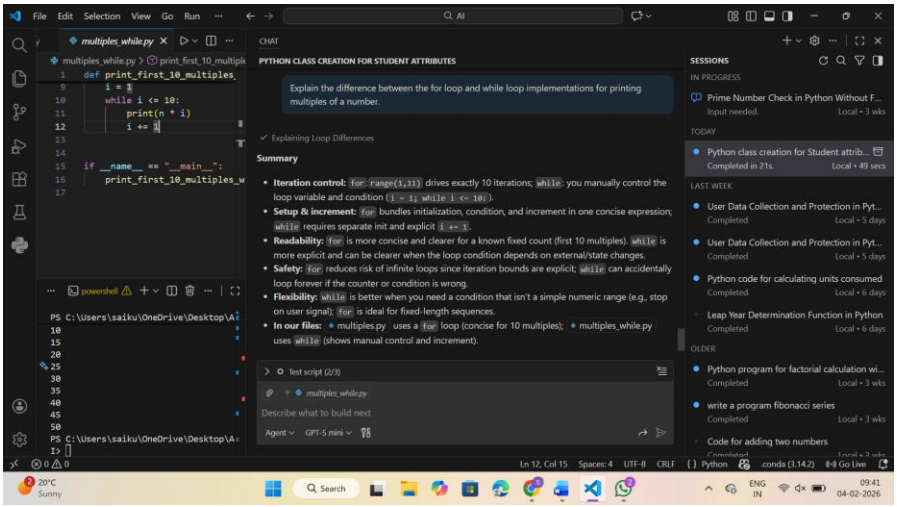
Task Description #2: Loops (Multiples of a Number)

Scenario
You are writing a utility function to display multiples of a given number.
Prompt:
1 – For Loop
Write a Python function that asks the user to enter a number and prints the first 10 multiples of that number using a for loop.
2-While Loop
Rewrite the same program that prints the first 10 multiples of a number, but use a while loop instead of a for loop. Keep user input dynamic.
3- **Analysis**
Explain the difference between the for loop and while loop implementations for printing multiples of a number.

ROLL NO:2303A51820 BATCH:26







Task Description #3: Conditional Statements (Age Classification)

Scenario

You are building a basic classification system based on age.

Prompt

1 – if-elif-else

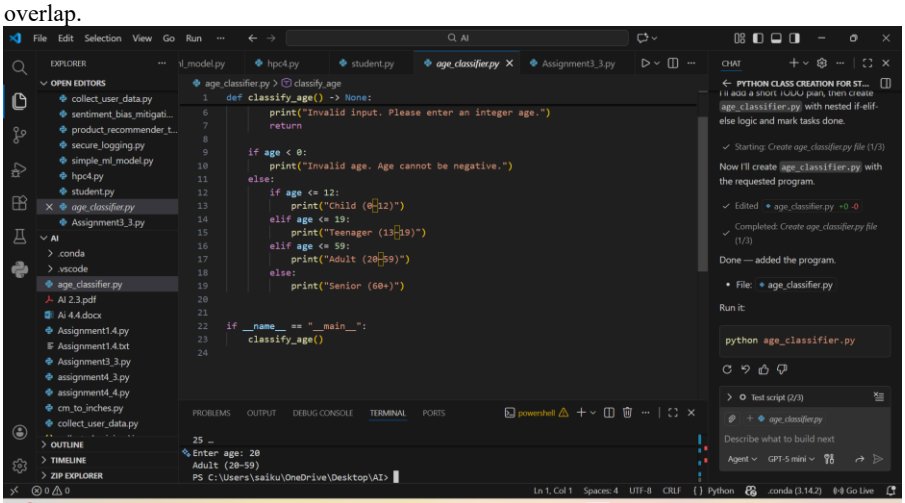
Write a Python program that takes age as user input and classifies the person into Child (0–12), Teenager (13–19), Adult (20–59), or Senior (60+) using nested if-elif-else statements.

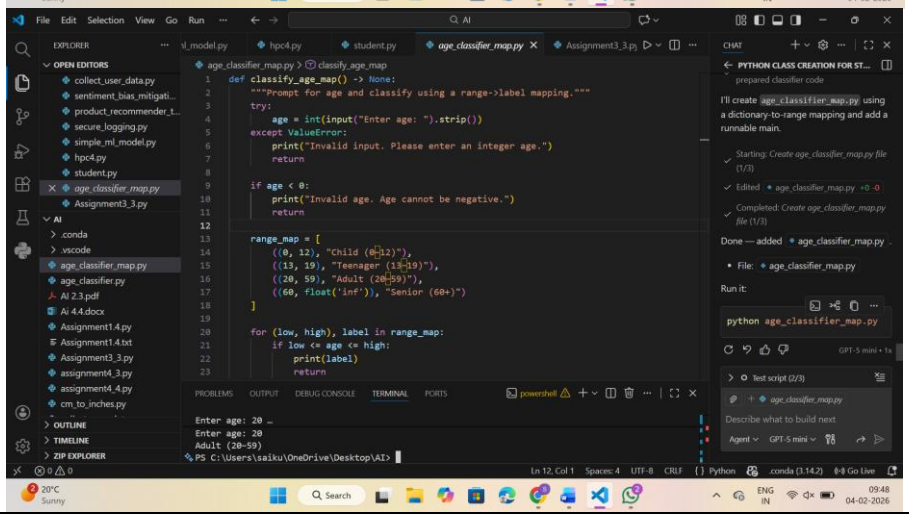
2 – Alternative Approach

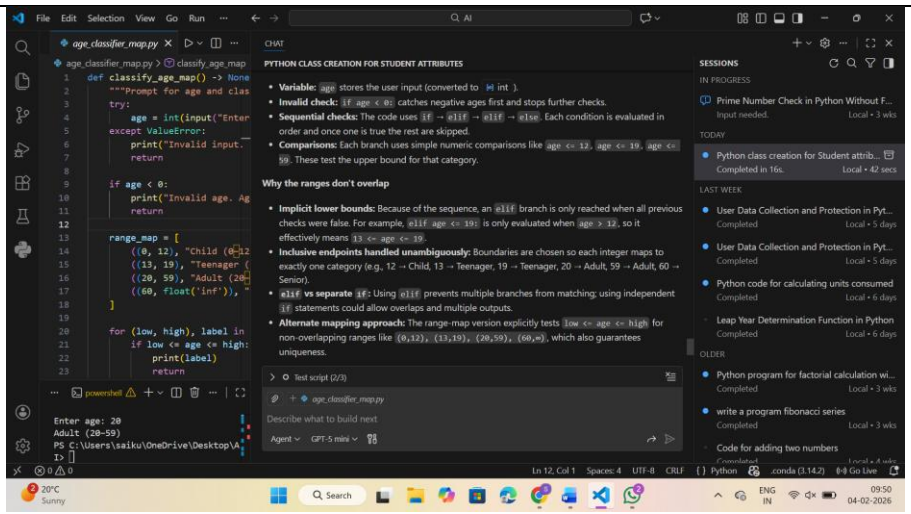
Rewrite the age classification program using a different approach such as dictionary-based range mapping or simplified logical conditions. Take age as user input.

3 – Explanation

Explain how the conditions in the age classification program work and why the ranges do not overlap.



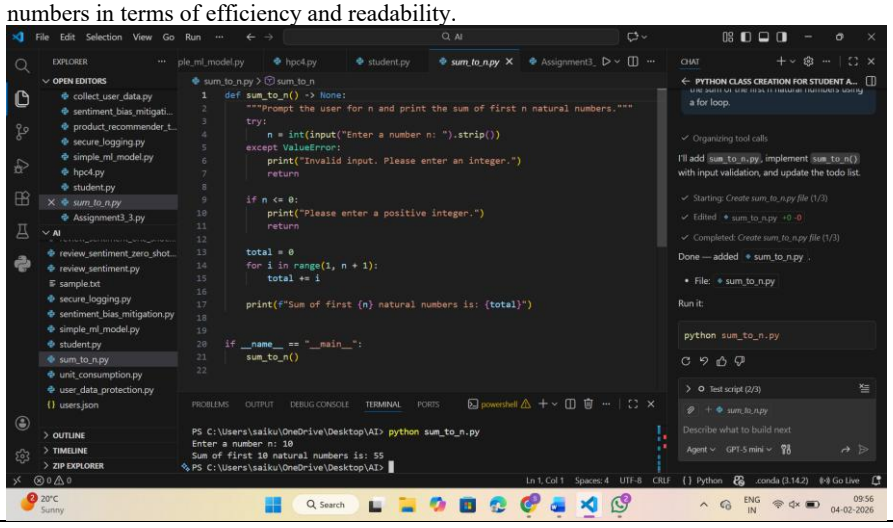




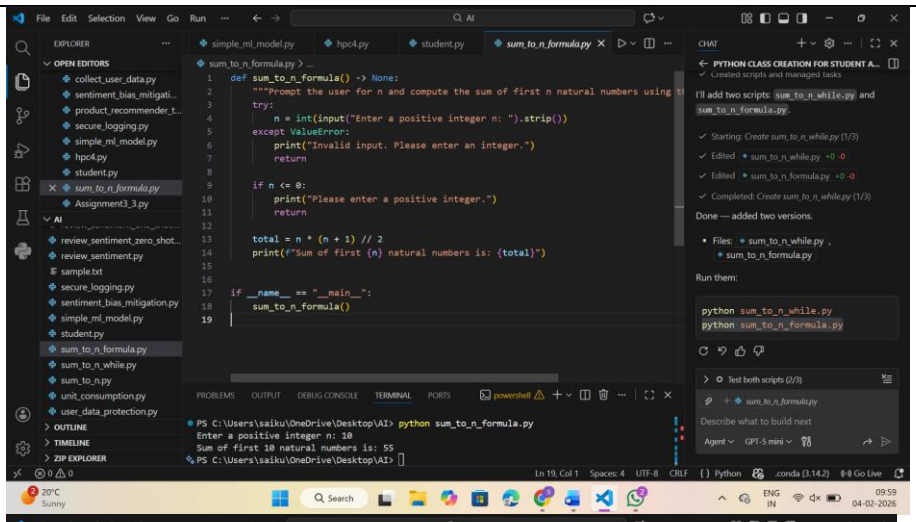
Task Description #4: For and While Loops (Sum of First n Numbers)

Scenario
You need to calculate the sum of the first n natural numbers.

Prompt
1 – For Loop
Write a Python function sum_to_n() that asks the user to enter a number n and calculates the sum of the first n natural numbers using a for loop.
Prompt 2 – Alternatives
Provide two more versions of the sum_to_n() program: one using a while loop and another using the mathematical formula n(n+1)//2. Use user input.
Prompt 3 – Comparison
Compare the loop-based and formula-based approaches for calculating the sum of n natural numbers in terms of efficiency and readability.

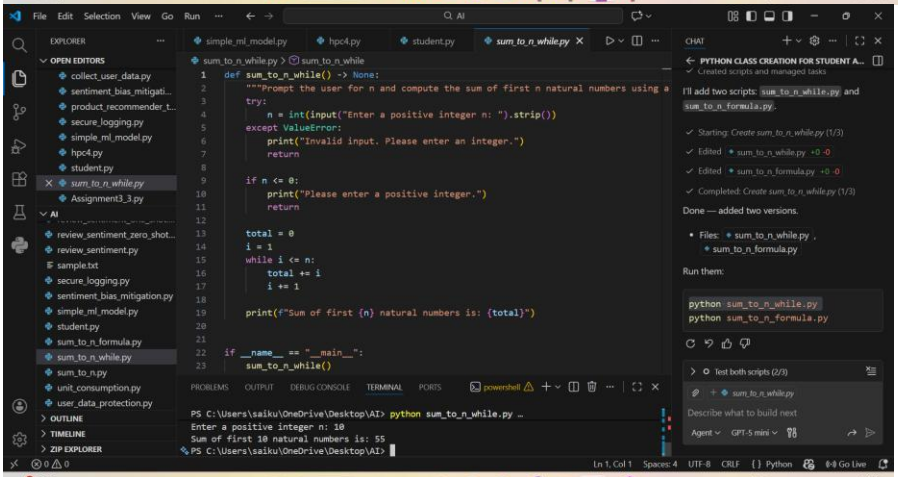


ROLL NO:2303A51820 BATCH:26



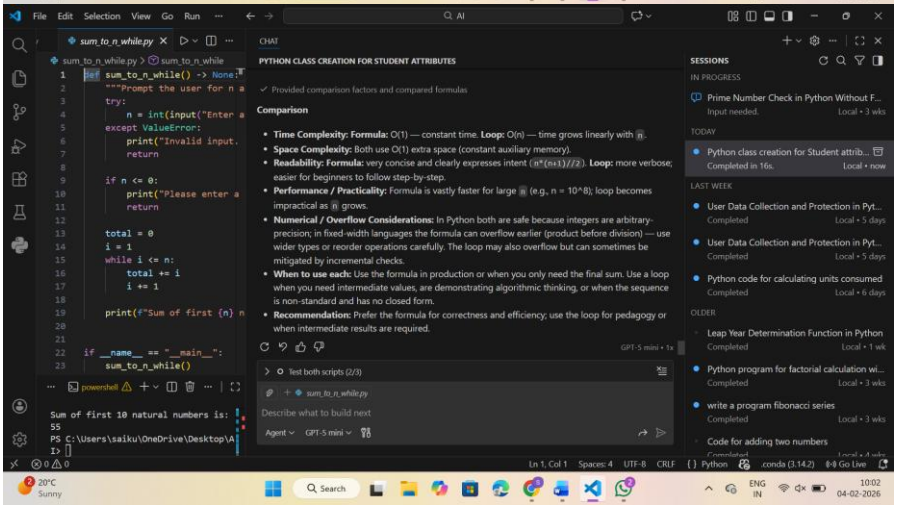
```
def sum_to_n_formula() -> None:
    """Prompt the user for n and compute the sum of first n natural numbers using t
    try:
        n = int(input("Enter a positive integer n: ").strip())
    except ValueError:
        print("Invalid input. Please enter an integer.")
        return

    if n <= 0:
        print("Please enter a positive integer.")
        return

    total = n * (n + 1) // 2
    print(f"Sum of first {n} natural numbers is: {total}")


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

```
PS C:\Users\saiku\OneDrive\Desktop\AI> python sum_to_n_formula.py
Enter a positive integer n: 10
Sum of first 10 natural numbers is: 55
PS C:\Users\saiku\OneDrive\Desktop\AI>
```



```
def sum_to_n_while() -> None:
    """Prompt the user for n and compute the sum of first n natural numbers using a
    try:
        n = int(input("Enter a positive integer n: ").strip())
    except ValueError:
        print("Invalid input. Please enter an integer.")
        return

    if n <= 0:
        print("Please enter a positive integer.")
        return

    total = 0
    i = 1
    while i <= n:
        total += i
        i += 1

    print(f"Sum of first {n} natural numbers is: {total}")

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

```
PS C:\Users\saiku\OneDrive\Desktop\AI> python sum_to_n_while.py ...
Enter a positive integer n: 10
Sum of first 10 natural numbers is: 55
PS C:\Users\saiku\OneDrive\Desktop\AI>
```



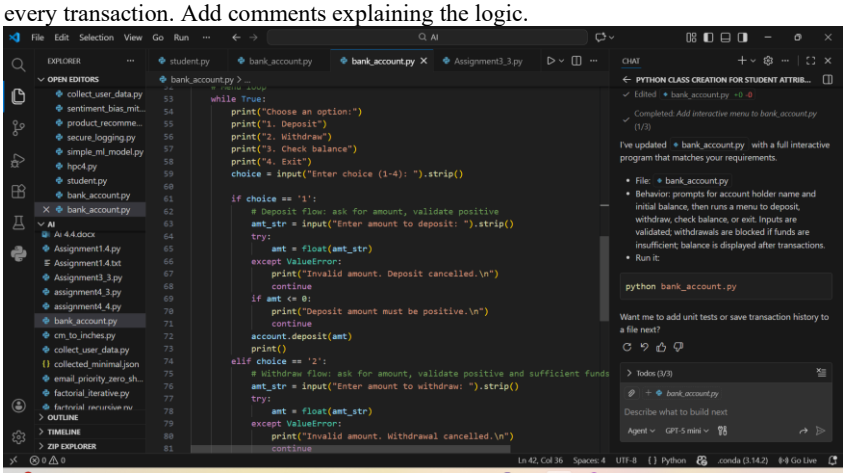
```
def sum_to_n_while() -> None:
    """Prompt the user for n a
    try:
        n = int(input("Enter a
    except ValueError:
        print("Invalid input.
        return

    if n <= 0:
        print("Please enter a
        return

    total = 0
    i = 1
    while i <= n:
        total += i
        i += 1

    print(f"Sum of first {n} n

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

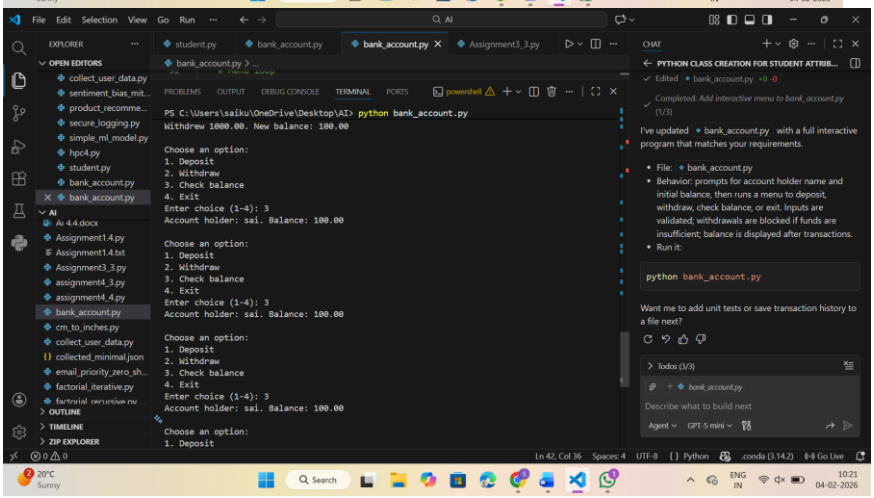
```
Sum of first 10 natural numbers is:
55
PS C:\Users\saiku\OneDrive\Desktop\A
I>
```

Task Description #5: Classes (Bank Account Class)

Scenario
You are designing a basic banking application.

Prompt : Write a complete Python program that defines a class BankAccount with attributes account_holder_name and balance. Use a constructor to initialize these values. Include methods deposit(amount), withdraw(amount), and check_balance(), ensuring withdrawal is not allowed if the balance is insufficient.
In the main program, take user input for account holder name and initial balance. Then run a looped menu system where the user can repeatedly choose to deposit money, withdraw money, check balance, or exit.
For deposit and withdrawal operations, ask the user to enter the amount dynamically each time. Validate inputs to prevent negative deposits and overdrawing. Display the updated balance after every transaction. Add comments explaining the logic.





Note: Report should be submitted as a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots.