AI ASSISTED CODING Hall

Name : G. Sanjay

Ticket No: 2303A510B9

Batch:13

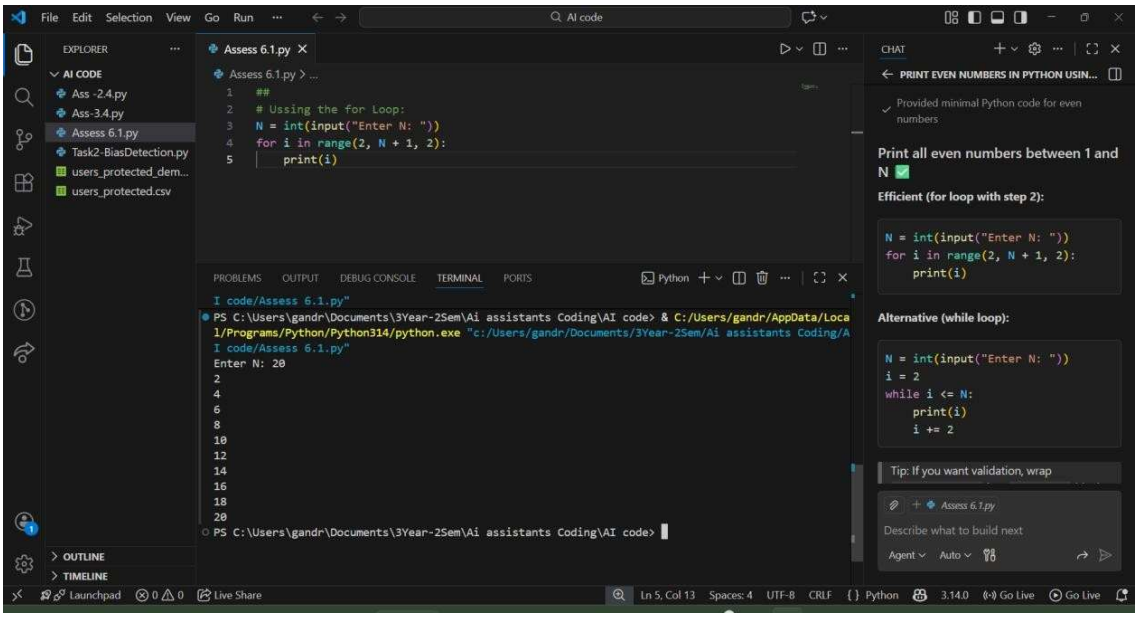
Assignment-6.1 Task-1:

Task Description #1 (AI-Based Code Completion for Loops) Task: Use an AI code completion tool to generate a loop-based Program

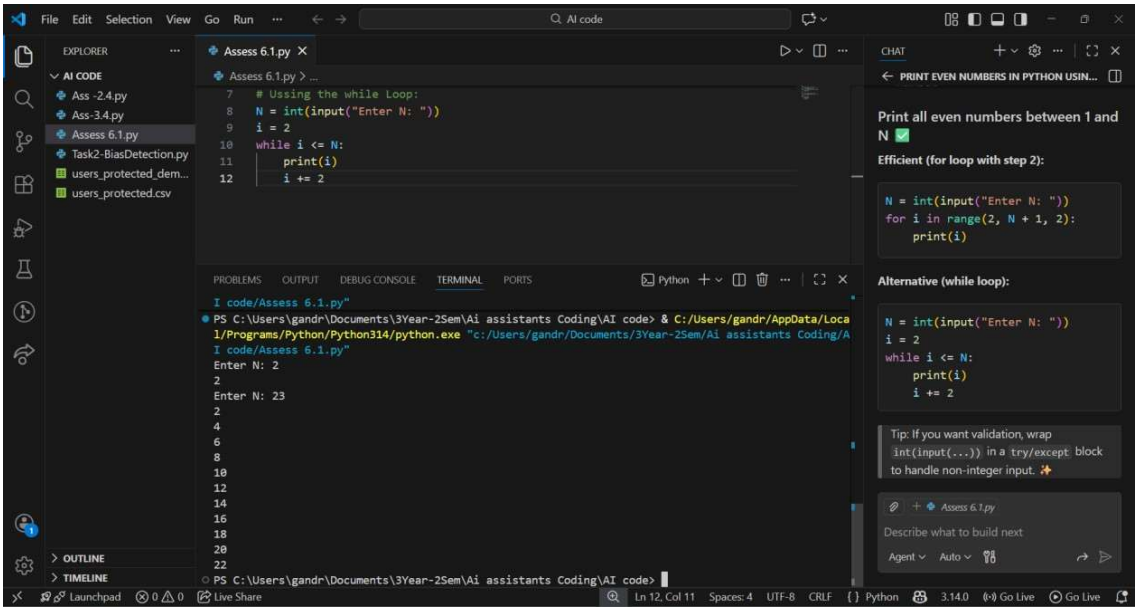
Prompt :

#Ok Generate Python code to print all even numbers between 1 and N using a loop.

Code and Out put: Using For loop



Code : Using While Loop/



Task-2:

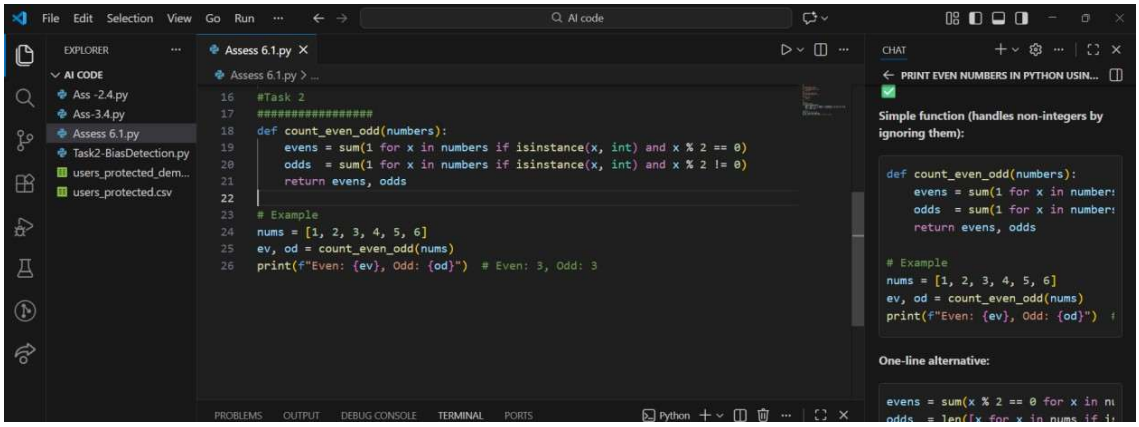
Task Description #2 (AI-Based Code Completion for Loop with Conditionals)

Task: Use an AI code completion tool to combine loops and conditionals program.

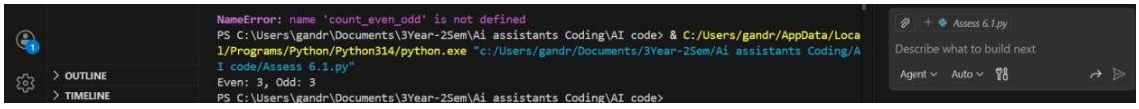
Prompt:

 # ok Generate Python code to count how many numbers in a list are even and odd.

Code:



Output:



Task-3:

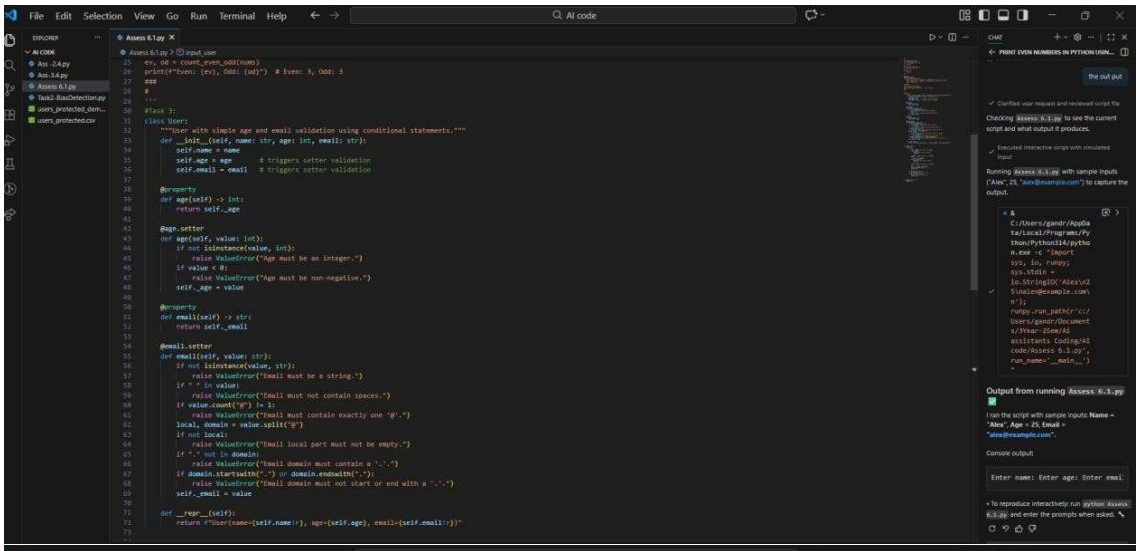
Task Description #3 (AI-Based Code Completion for Class Attributes Validation)

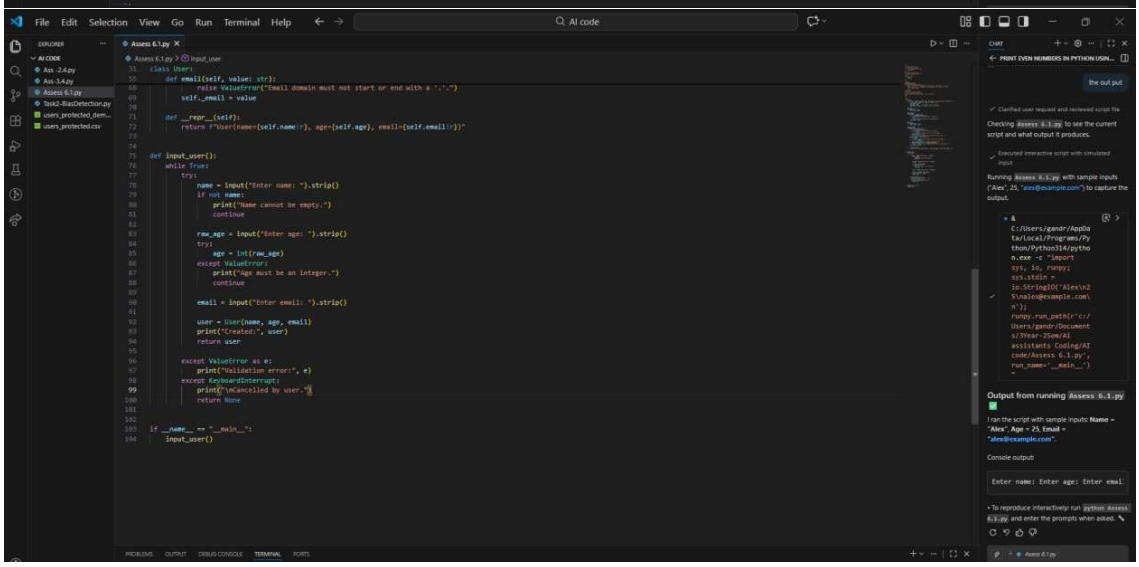
Task: Use an AI tool to complete a Python class that validates user Input

Prompt:

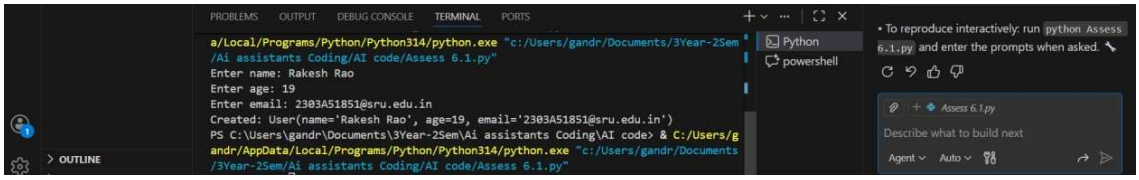
#Generate a Python class User that validates age and email using conditional statements

Code:





Output:



Task-4:

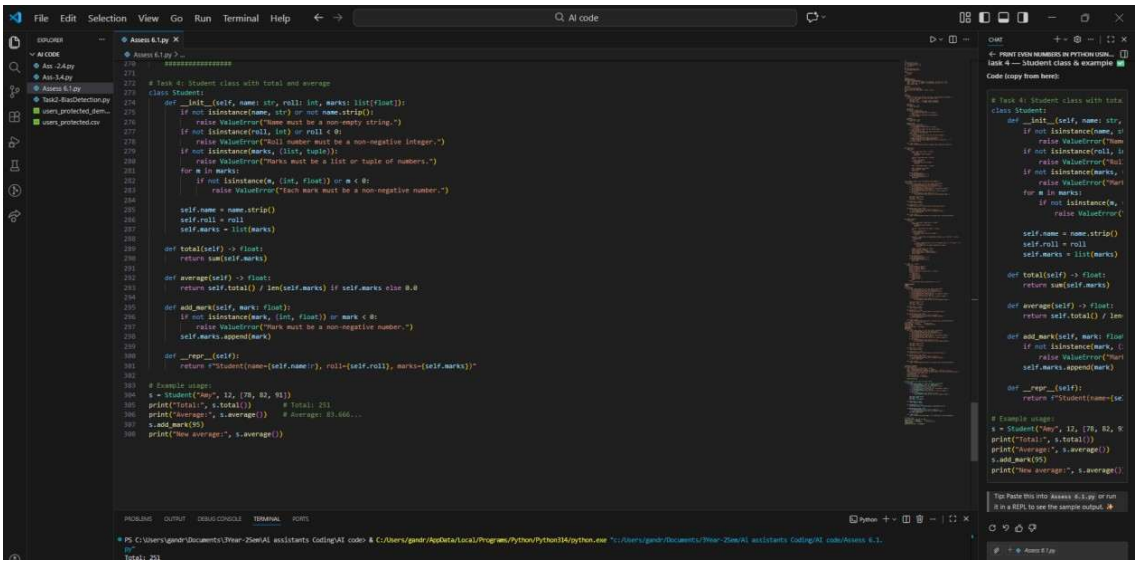
Task Description #4 (AI-Based Code Completion for Classes)

Task: Use an AI code completion tool to generate a Python class for managing student details.

Prompt:

 # Generate a Python class Student with attributes (name, roll number, marks) and methods to calculate total and average marks.

Code:



```
# Task 4: Student class with total and average
class Student:
    def __init__(self, name: str, roll: int, marks: list[float]):
        if not isinstance(name, str) or not name.strip():
            raise ValueError("Name must be a non-empty string.")
        if not isinstance(roll, int) or roll < 0:
            raise ValueError("Roll number must be a non-negative integer.")
        if not isinstance(marks, (list, tuple)):
            raise ValueError("Marks must be a list or tuple of numbers.")
        for m in marks:
            if not isinstance(m, (int, float)) or m < 0:
                raise ValueError("Each mark must be a non-negative number.")

        self.name = name.strip()
        self.roll = roll
        self.marks = list(marks)

    def total(self) -> float:
        return sum(self.marks)

    def average(self) -> float:
        return self.total() / len(self.marks) if self.marks else 0.0

    def add_mark(self, mark: float):
        if not isinstance(mark, (int, float)) or mark < 0:
            raise ValueError("Mark must be a non-negative number.")
        self.marks.append(mark)

    def __repr__(self):
        return f"Student(name={self.name!r}, roll={self.roll}, marks={self.marks})"

# Example usage:
s = Student("Amy", 12, [78, 82, 91])
print("Total:", s.total())      # Total: 251
print("Average:", s.average())  # Average: 83.666...
s.add_mark(95)
print("New average:", s.average())
```

Output:

```
PS C:\Users\gandr\Documents\3Year-2Sem\Ai assistants Coding\AI code> & C:/Users/gandr/AppData/Local/Programs/Python/Python314/python.exe "c:/Users/gandr/Documents/3Year-2Sem/Ai assistants Coding/AI code/Assess 6.1.py"
Total: 251
Average: 83.66666666666667
New average: 86.5
PS C:\Users\gandr\Documents\3Year-2Sem\Ai assistants Coding\AI code>
```

Task-5:

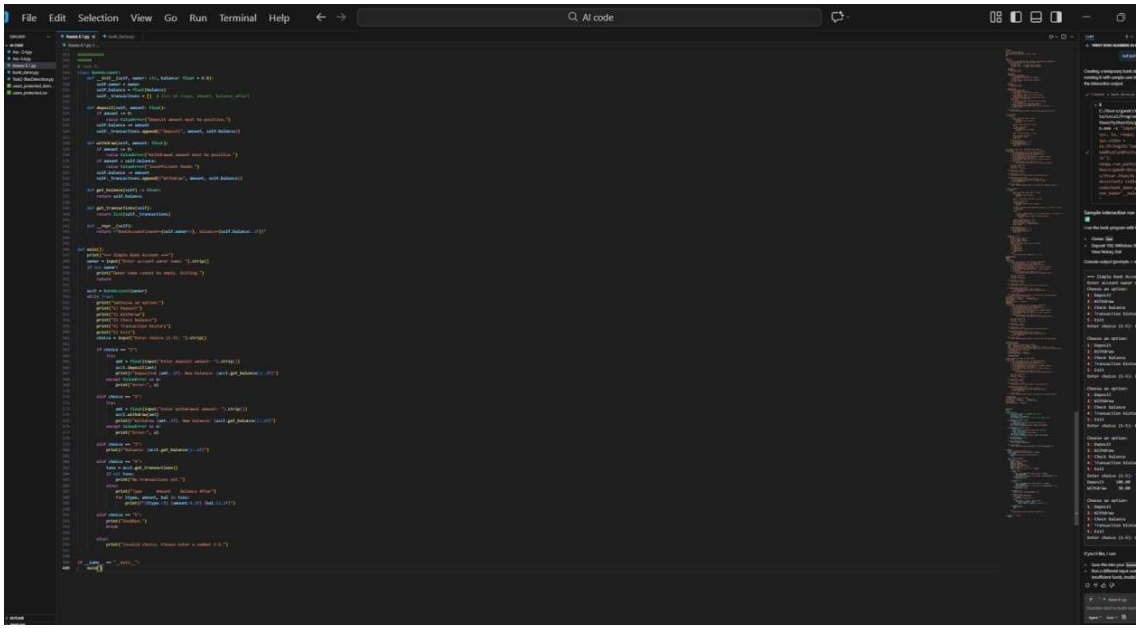
Task Description 5 (AI-Assisted Code Completion Review)

Task: Use an AI tool to generate a complete Python program using classes, loops, and conditionals together.

Prompt:

##Generate a Python program for a simple bank account system using class, loops, and conditional statements.

Code:



Out Put: