

Name:-U.Siddhartha
Hall No:-2403A54122

Task Description #1 (AI-Assisted Bug Detection)

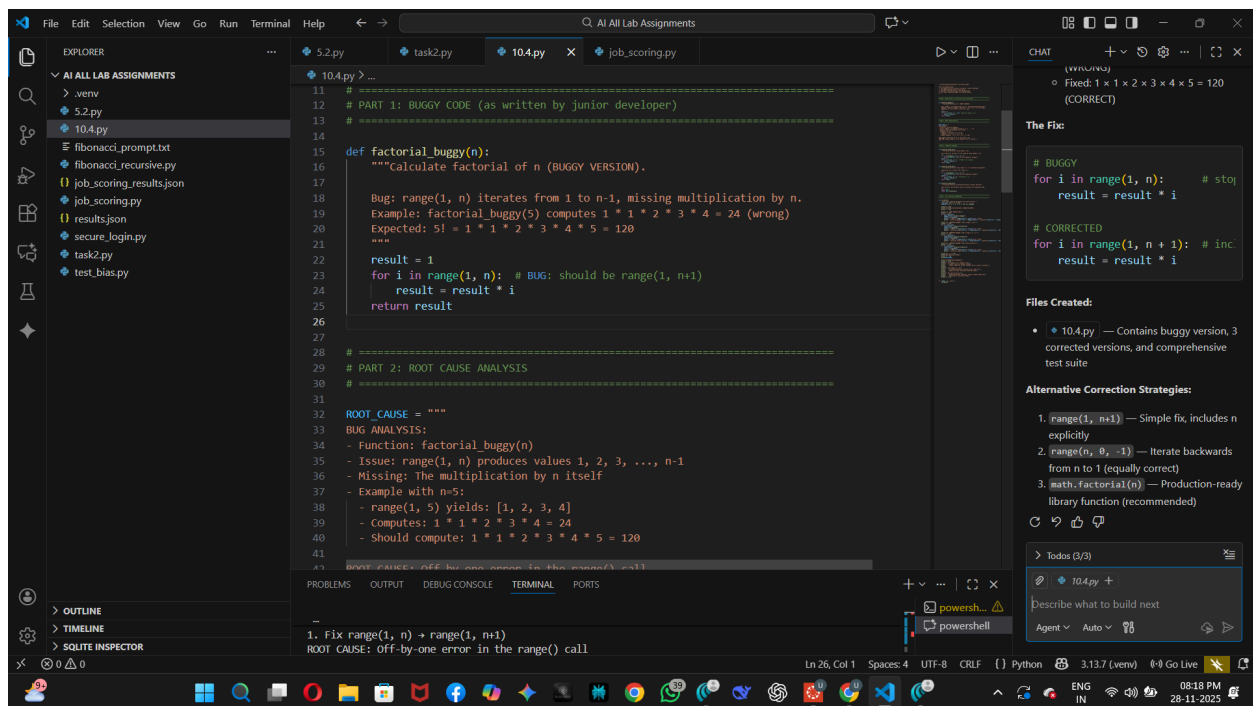
Scenario:

A junior developer wrote the following Python function to calculate factorials:

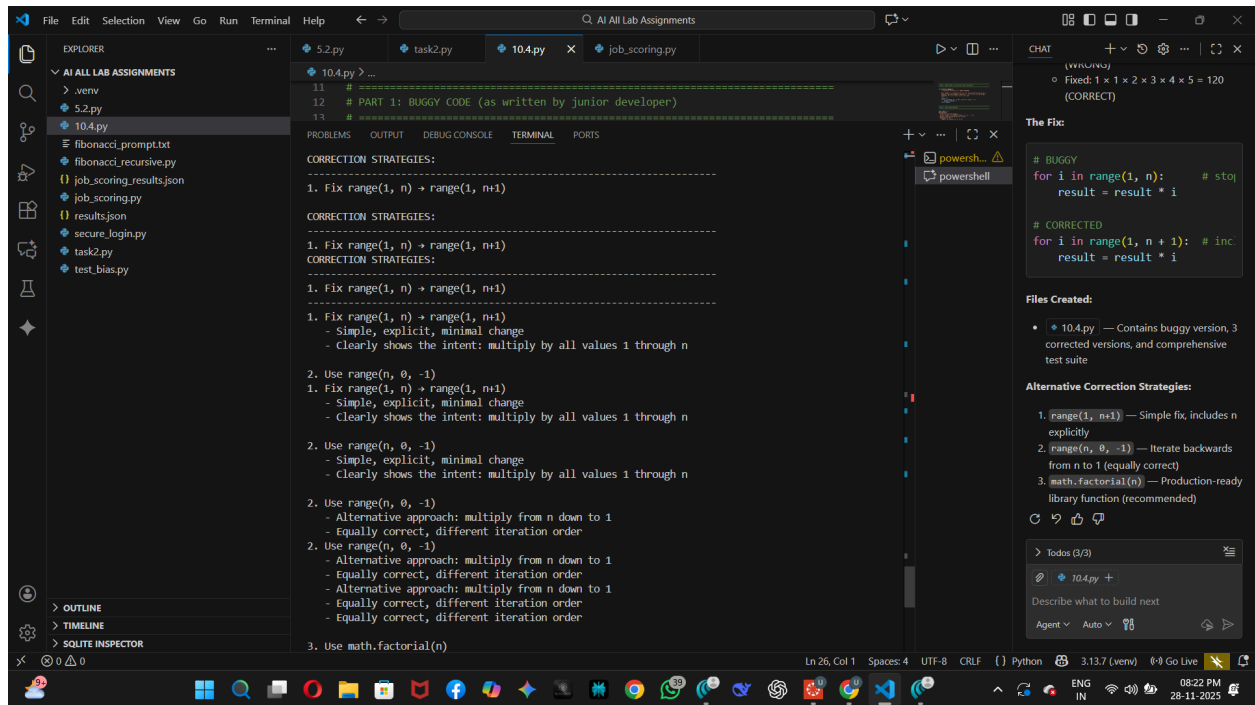
```
def factorial(n):  
    result = 1  
    for i in range(1, n):  
        result = result * i  
    return result
```

- Run the code and test it with `factorial(5)` (expected output = 120).
- Use AI (prompting) to review this code and identify the bug.
- Ask AI to suggest corrections and rewrite the code.
- Compare AI's corrected code with your own fix.

Input:-



Output:-



Task Description #2 (Improving Readability & Documentation)

Scenario:

The following code works but is poorly written:

```
def calc(a,b,c):
```

```
if c=="add":
```

```
    return a+b
```

```
elif c=="sub":
```

```
    return a-b
```

```
elif c=="mul":
```

```
    return a*b
```

```
elif c=="div":
```

```
    return a/b
```

•Use AI to review this code

For readability, naming, and documentation issues.

- Prompt AI to rewrite the code with:

- Clear function & variable names.

- Proper docstrings.

- Exception handling for division by zero.

- Compare the before-and-after versions to evaluate AI's contribution.

Input:-

```
12 # PART 1: BEFORE - POORLY WRITTEN CODE
13 #
14
15 def calc(a, b, c):
16     """Poor readability version."""
17     if c == "add":
18         return a + b
19     elif c == "sub":
20         return a - b
21     elif c == "mul":
22         return a * b
23     elif c == "div":
24         return a / b
25
26
27 #
28 # PART 2: AFTER - IMPROVED VERSION (AI REFACTORED)
29 #
30
31 def perform_arithmetic_operation(operand1: float, operand2: float, operation: str) -> float:
32     """Perform a basic arithmetic operation on two operands.
33
34     This function executes one of four arithmetic operations (addition, subtraction,
35     multiplication, or division) based on the specified operation string.
36
37     Args:
38         operand1 (float): The first operand (left-hand side).
39         operand2 (float): The second operand (right-hand side).
40         operation (str): The operation to perform. Valid values are:
41             - 'add': Addition (operand1 + operand2)
42             - 'sub': Subtraction (operand1 - operand2)
43             - 'mul': Multiplication (operand1 * operand2)
44             - 'div': Division (operand1 / operand2)
45
46     Returns:
47         float: The result of the arithmetic operation.
48     """
49     if operation == "add":
50         return operand1 + operand2
51     elif operation == "sub":
52         return operand1 - operand2
53     elif operation == "mul":
54         return operand1 * operand2
55     elif operation == "div":
56         if operand2 == 0:
57             raise ZeroDivisionError("Division by zero is not allowed.")
58         return operand1 / operand2
59     else:
60         raise ValueError(f"Invalid operation: {operation}")
```

Summary: Improving Readability & Documentation

File: Task2_in_10.2.py

Test Results:

```
1. NORMAL OPERATIONS:
add: calc(10, 5, 'add') = 15
sub: calc(10, 5, 'sub') = 5
mul: calc(10, 5, 'mul') = 50
div: calc(10, 5, 'div') = 2

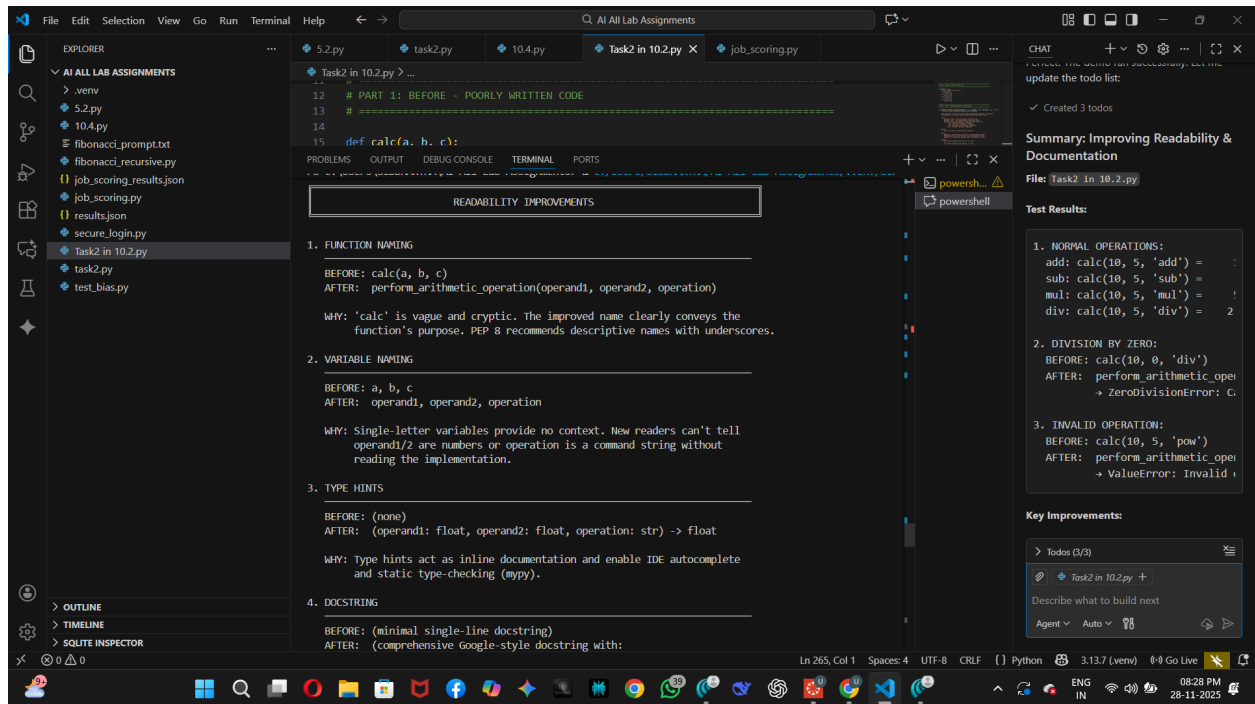
2. DIVISION BY ZERO:
BEFORE: calc(10, 0, 'div')
AFTER: perform_arithmetic_operation(10, 0, 'div')
       → ZeroDivisionError: Division by zero is not allowed.

3. INVALID OPERATION:
BEFORE: calc(10, 5, 'pow')
AFTER: perform_arithmetic_operation(10, 5, 'pow')
       → ValueError: Invalid operation: pow
```

Key Improvements:

- > Todos (3/3)
- Task2_in_10.2.py +
- Describe what to build next
- Agent Auto

Output:-



Task Description #3 (Enforcing Coding Standards)

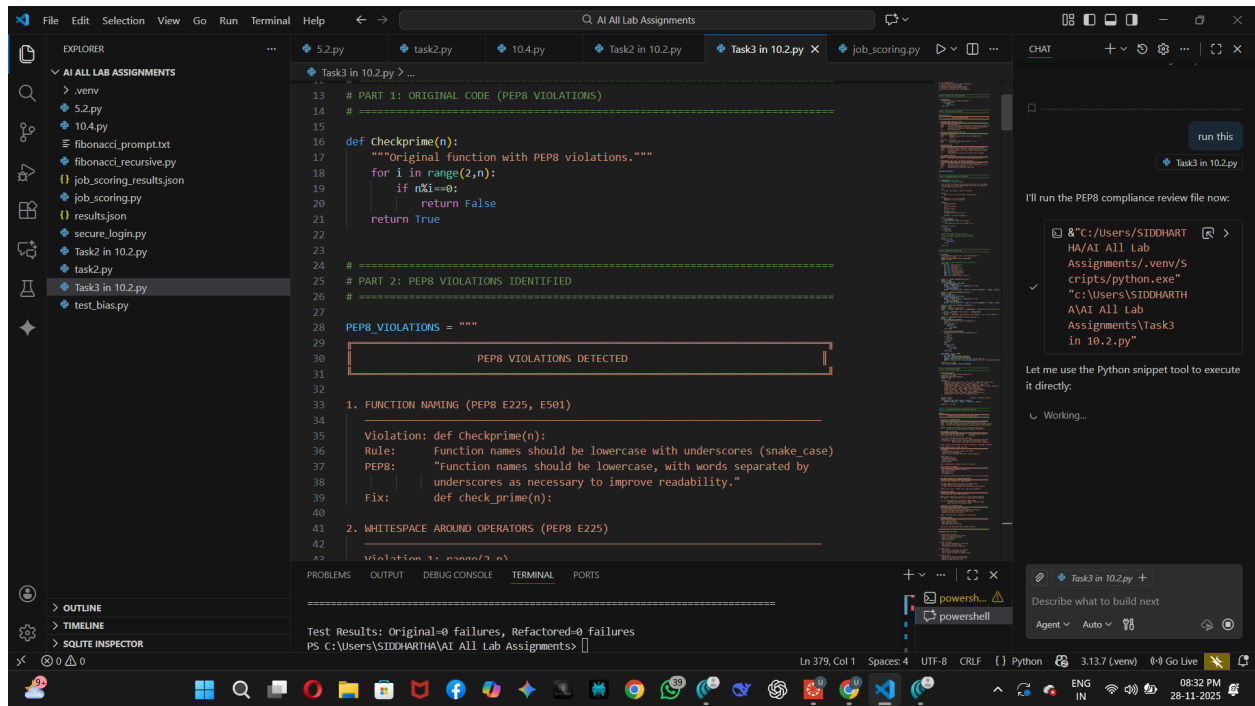
Scenario:

A team project requires following PEP8 style guide. Onedevloper submits:

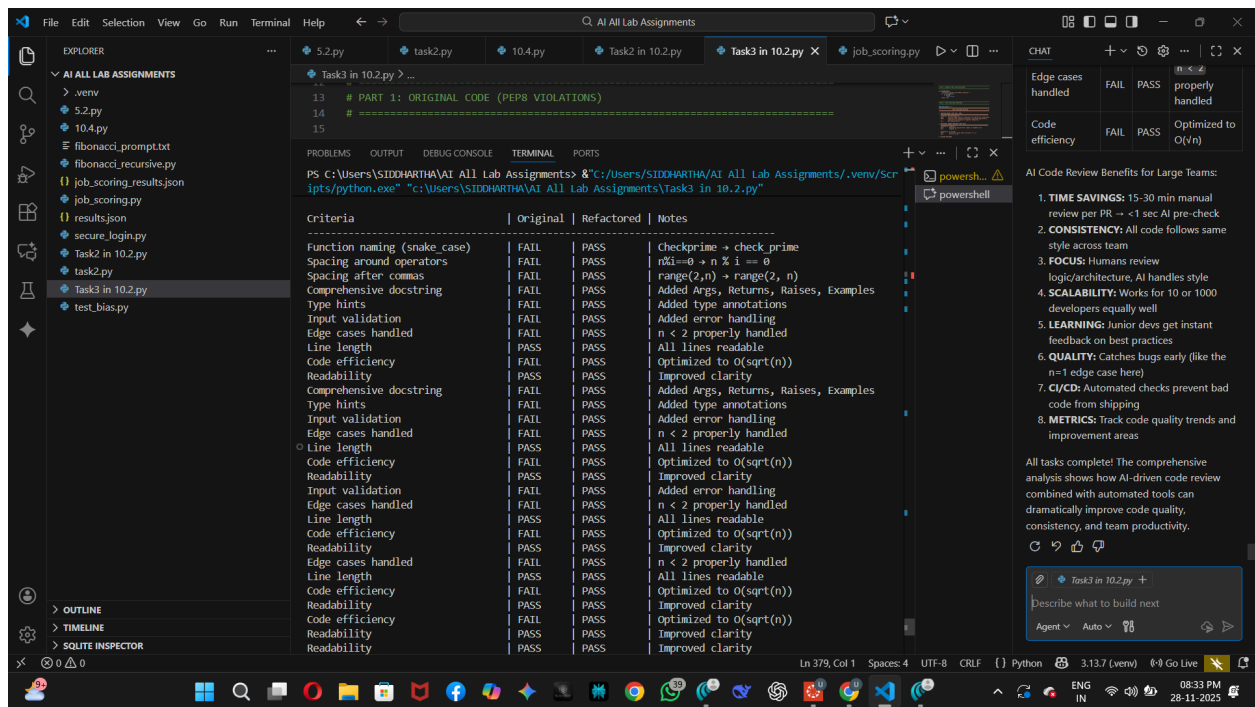
```
def Checkprime(n):
    for i in range(2,n):
        if n%i==0:
            return False
        return True
```

- Run this code and verify correctness.
- Use AI to perform a code quality review for PEP8 compliance.
- Prompt AI to return a refactored version with proper indentation,spacing, and naming conv
- Discuss how automated AI review can save time in large-scaleprojects

Input:-



Output:-



Task Description #4

(AI as a Code Reviewer in Real Projects)

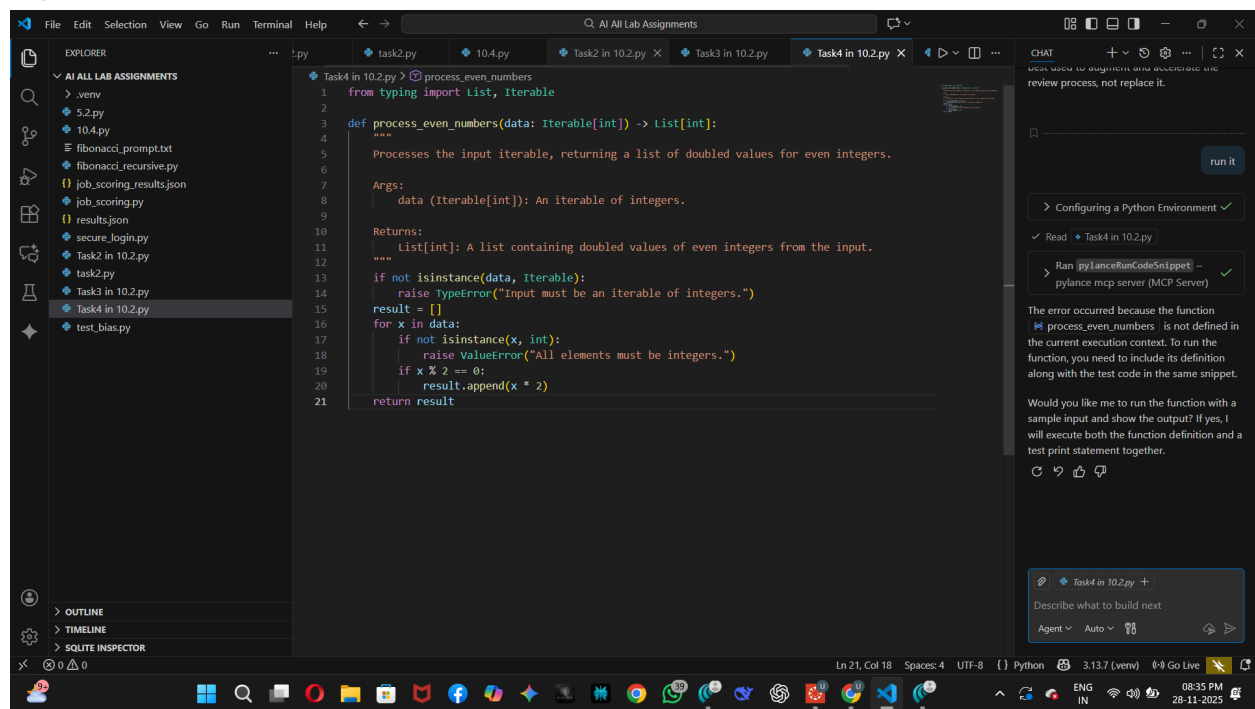
Scenario:

You are part of a GitHub project. A teammate submits this pull request:

```
def processData(d):  
    return [x*2 for x in d if x%2==0]
```

- Review this function manually for readability, reusability, and edge cases.
- Use AI to generate a code review comment, focusing on:
 - Naming conventions.
 - Input validation (e.g., what if d is not a list?).
 - Adding type hints.
- Modify the function based on AI's suggestions.
- Write a short reflection: Would you trust AI as a standalone reviewer, or only as a support tool?

Input:-



Output:-