

ASSIGNMENT – 13

NAME: G.Jeevan

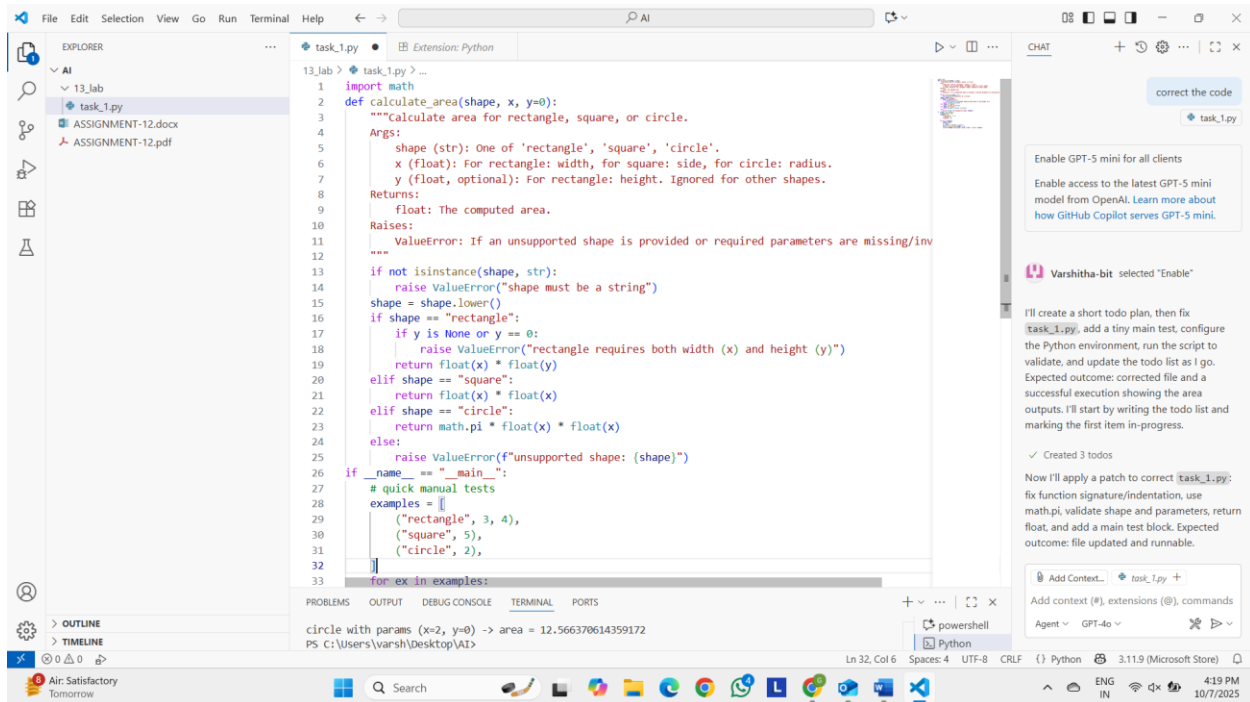
HT.NO: 2403A52055

BATCH: AIB03

Task-1:

Prompt: correct the code.

Code:



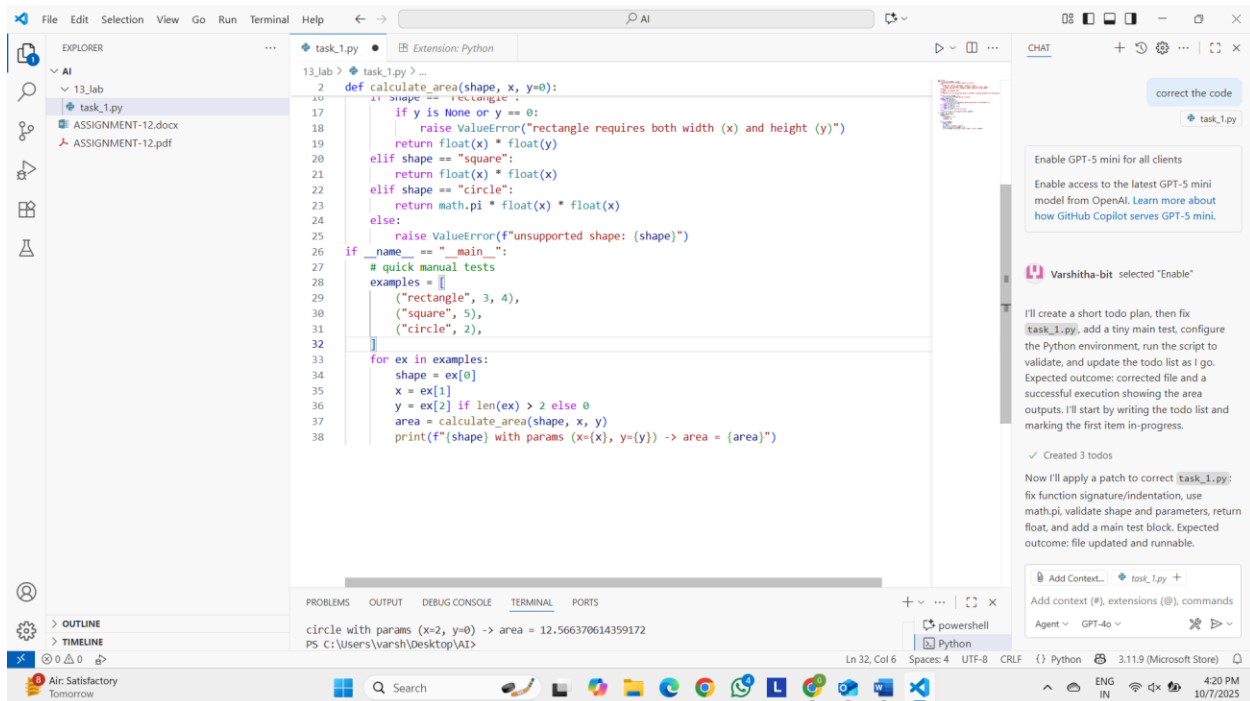
The screenshot shows a Visual Studio Code editor with a Python file named `task_1.py`. The code defines a function `calculate_area` that takes `shape`, `x`, and `y` as arguments. It calculates the area for rectangles, squares, and circles. The code includes docstrings, type hints, and error handling. A chat window on the right, titled "CHAT", shows a prompt "correct the code" and a response from "Varshitha-bit" that says "selected 'Enable'". The chat window also contains a list of todos and a patch to correct the code. The terminal at the bottom shows the output of the script: "circle with params (x=2, y=0) -> area = 12.566370614359172".

```
13_lab > task_1.py > ...
1 import math
2 def calculate_area(shape, x, y=0):
3     """Calculate area for rectangle, square, or circle.
4     Args:
5         shape (str): One of 'rectangle', 'square', 'circle'.
6         x (float): For rectangle: width, for square: side, for circle: radius.
7         y (float, optional): For rectangle: height. Ignored for other shapes.
8     Returns:
9         float: The computed area.
10    Raises:
11        ValueError: If an unsupported shape is provided or required parameters are missing/inv
12    """
13    if not isinstance(shape, str):
14        raise ValueError("shape must be a string")
15    shape = shape.lower()
16    if shape == "rectangle":
17        if y is None or y == 0:
18            raise ValueError("rectangle requires both width (x) and height (y)")
19        return float(x) * float(y)
20    elif shape == "square":
21        return float(x) * float(x)
22    elif shape == "circle":
23        return math.pi * float(x) * float(x)
24    else:
25        raise ValueError(f"unsupported shape: {shape}")
26 if __name__ == "__main__":
27     # quick manual tests
28     examples = [
29         ("rectangle", 3, 4),
30         ("square", 5),
31         ("circle", 2),
32     ]
33     for ex in examples:
```

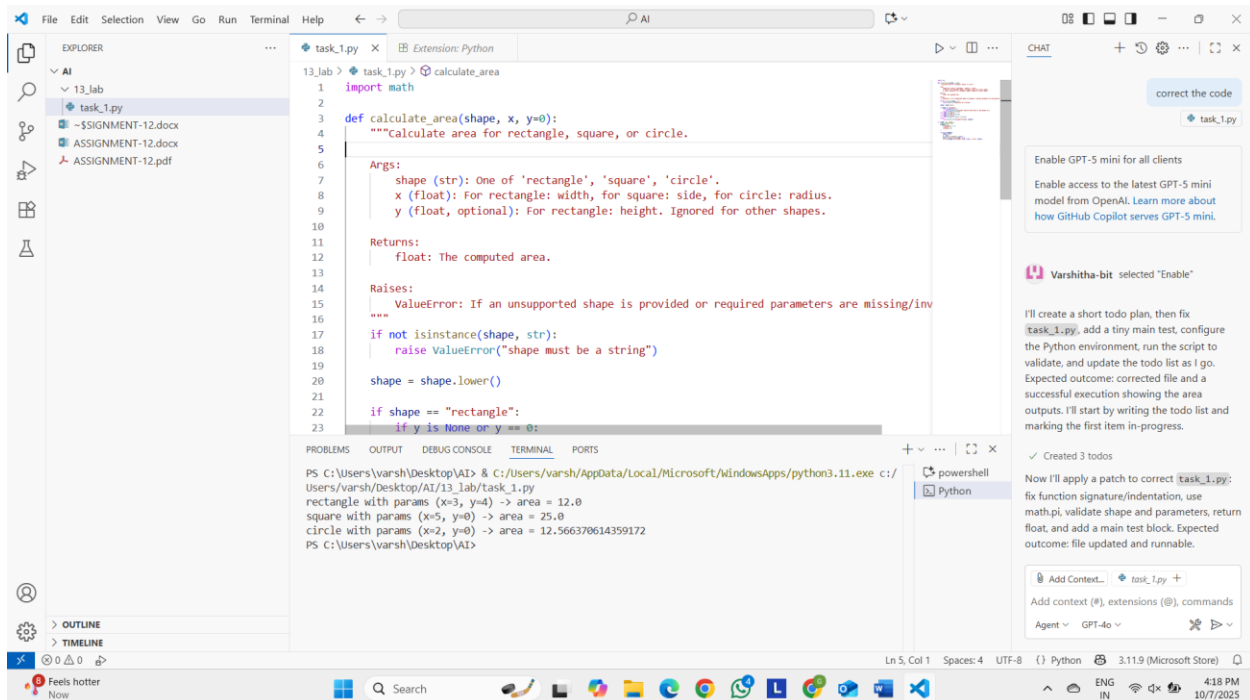
circle with params (x=2, y=0) -> area = 12.566370614359172
PS C:\Users\varsh\Desktop\AI>

Ln 32, Col 6 Spaces: 4 UTF-8 CRLF Python 3.11.9 (Microsoft Store)

419 PM 10/7/2025



Output:



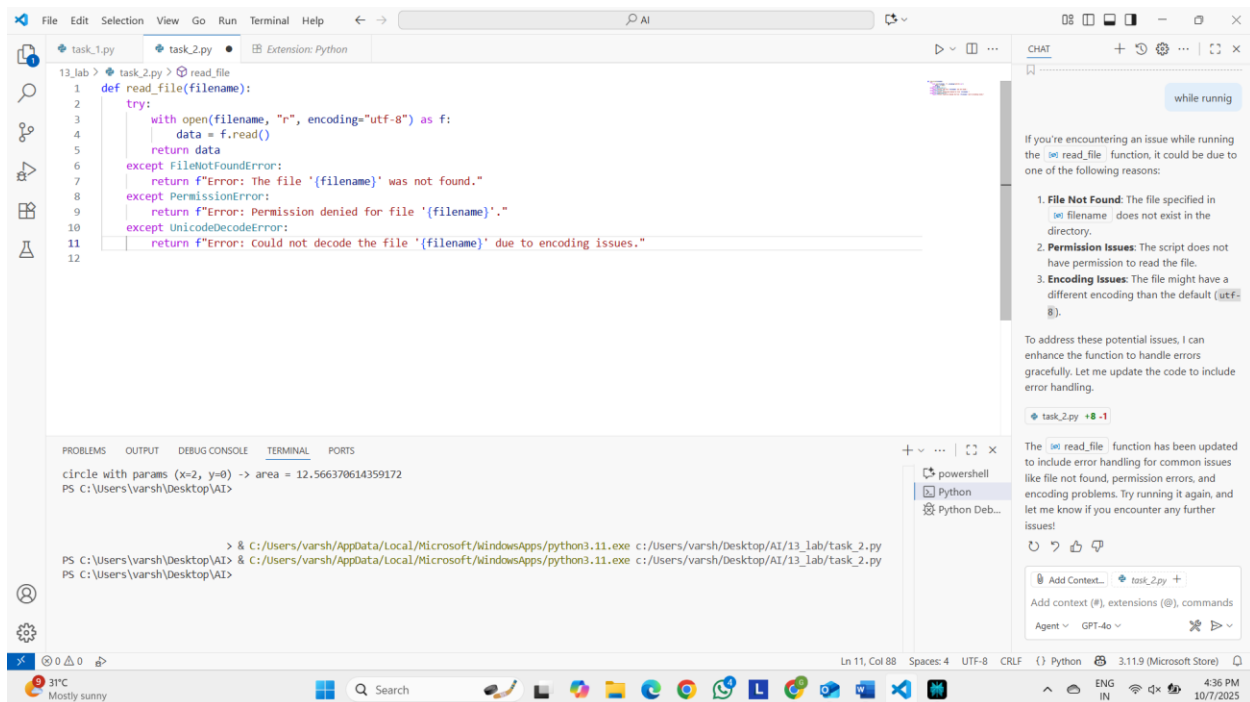
Observation:

First when we write the code in the work space, then when we give the prompt to correct the code it immediately changed the whole code. The original code was hardly 6 lines but the generated code is more lines. It even included comments so that we can understand the code easily. Now the code is in a correct structure.

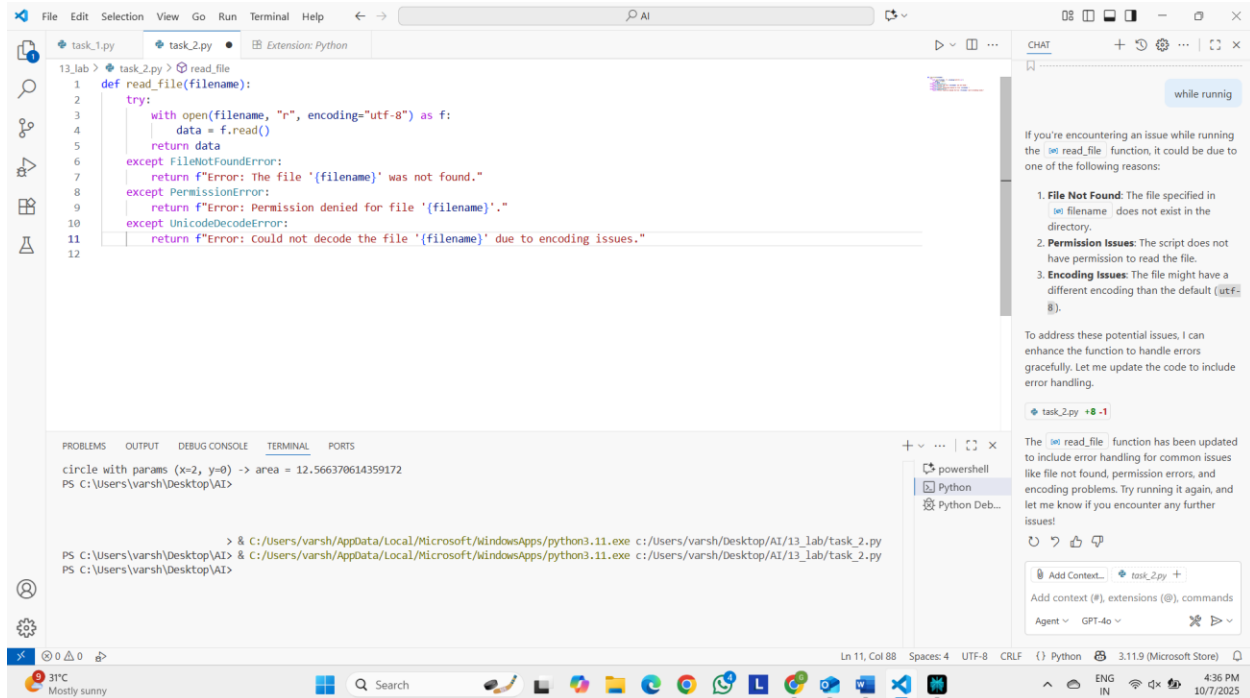
Task-2:

Prompt: Correct my mistake

Code:



Output:



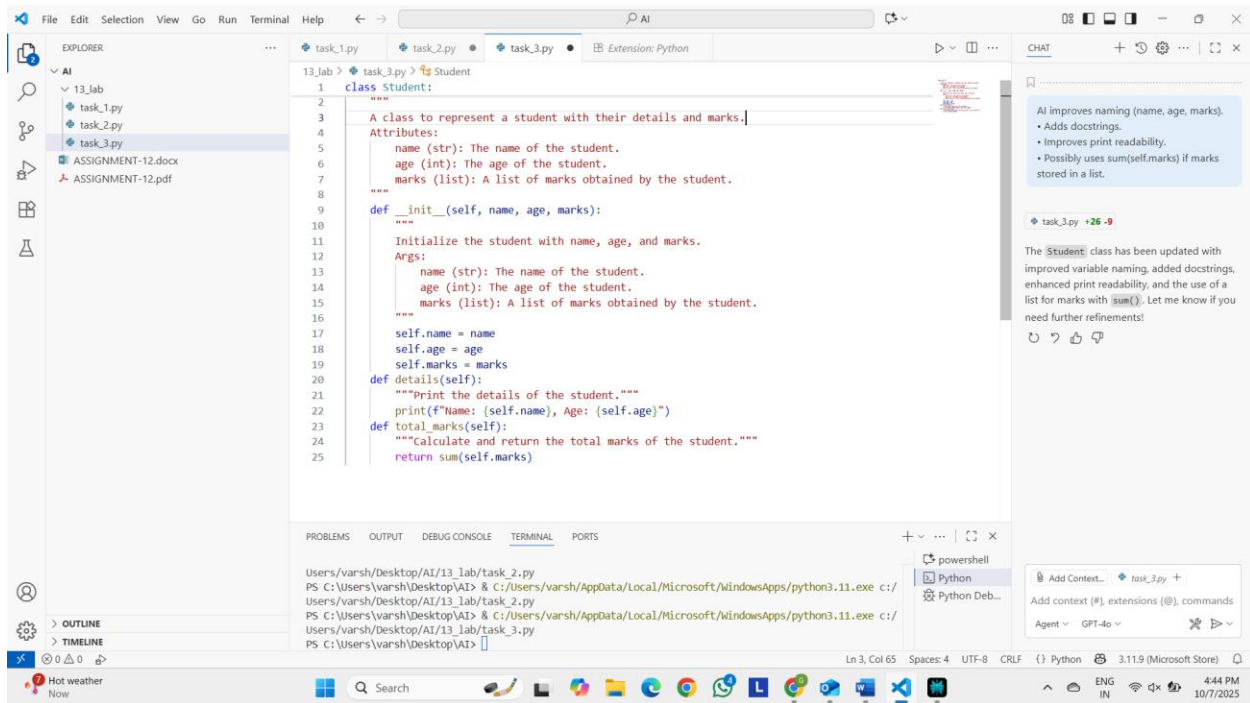
Observation:

In the given code we did not give a code with proper indentation. And file handling was also not perfect. But as we gave the prompt to correct it, it handled all those issues.

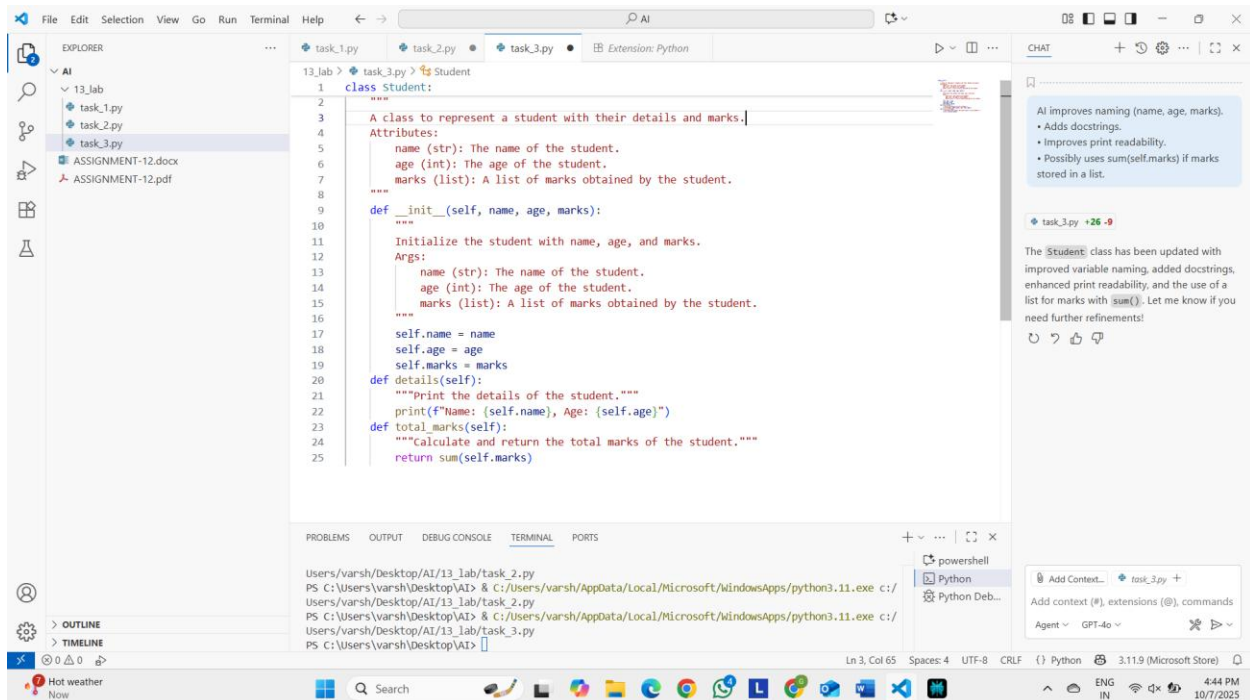
Task-3:

Prompt: correct my code. In this the attributes are name, age and marks.

Code:



Output:



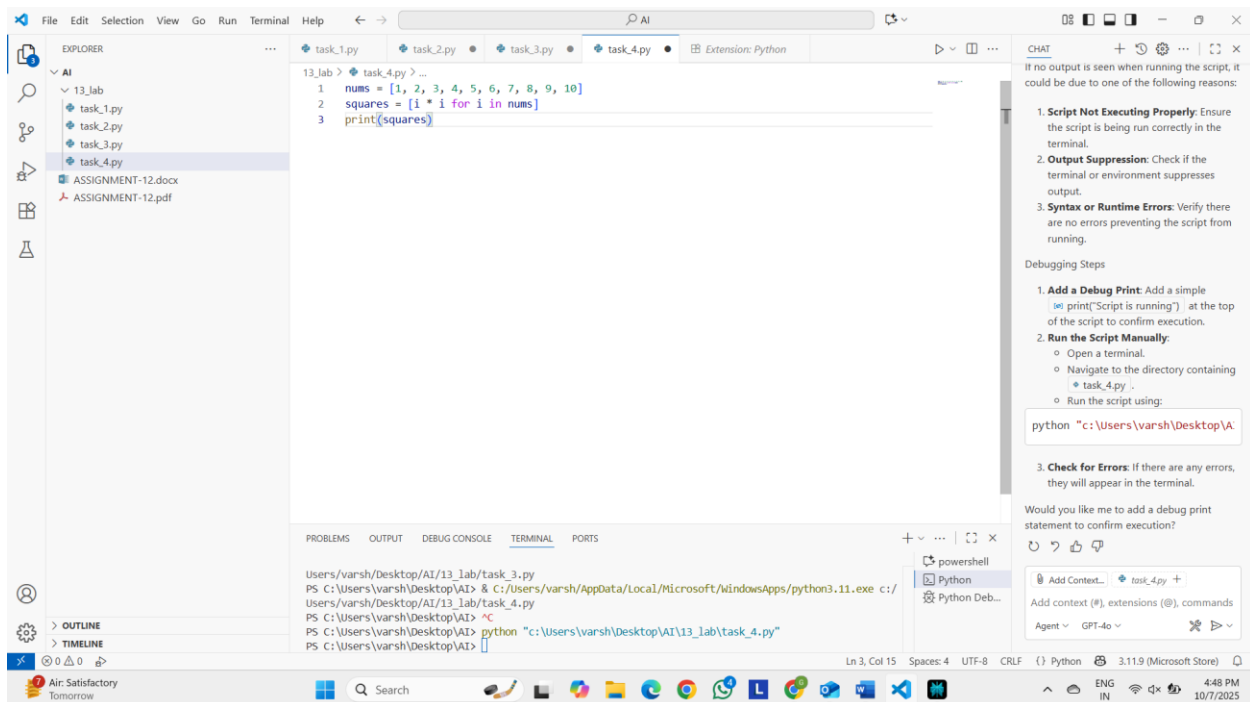
Observation:

First when we write the code in the work space, then when we give the prompt to correct the code it immediately changed the whole code. The original code was hardly 6 lines but the generated code is more lines. It even included comments so that we can understand the code easily. Now the code is in a correct structure

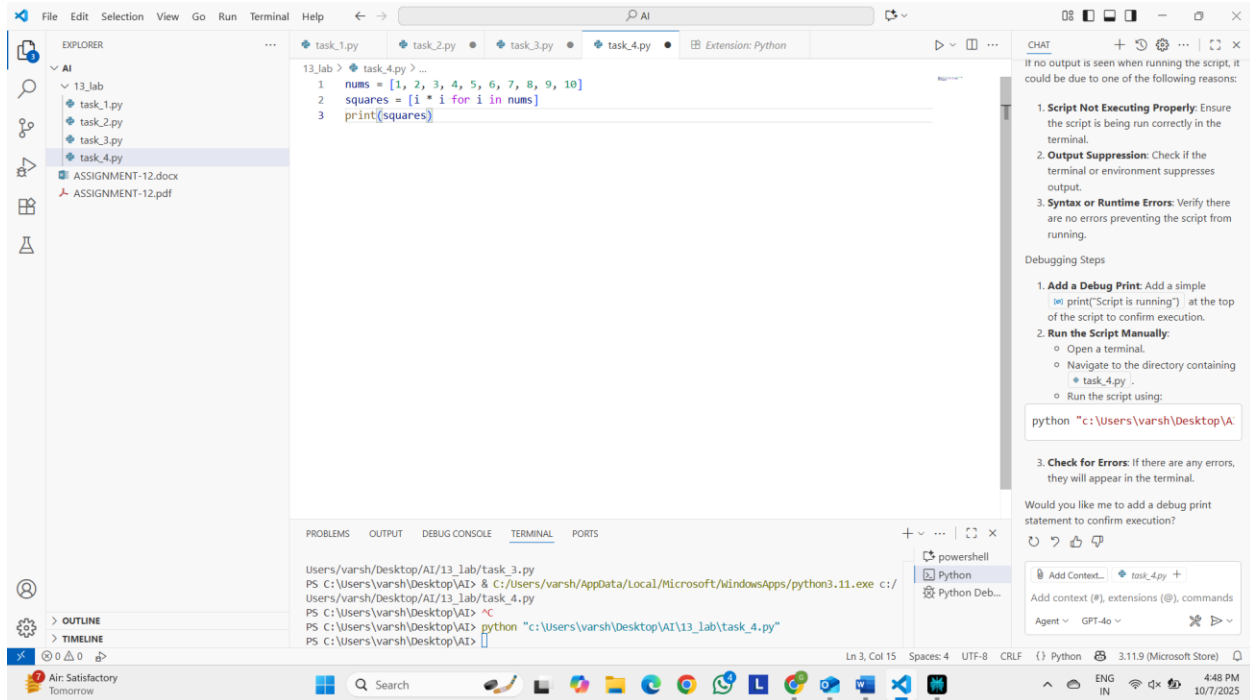
Task-4:

Prompt: Correct my mistake

Code:



Output:



Observation:

First when we write the code in the work space, then when we give the prompt to correct the code it immediately changed the whole code. The original code was hardly 6 lines but the generated code is more lines. It even included comments so that we can understand the code easily. Now the code is in a correct structure