ASSIGNMENT-2.5

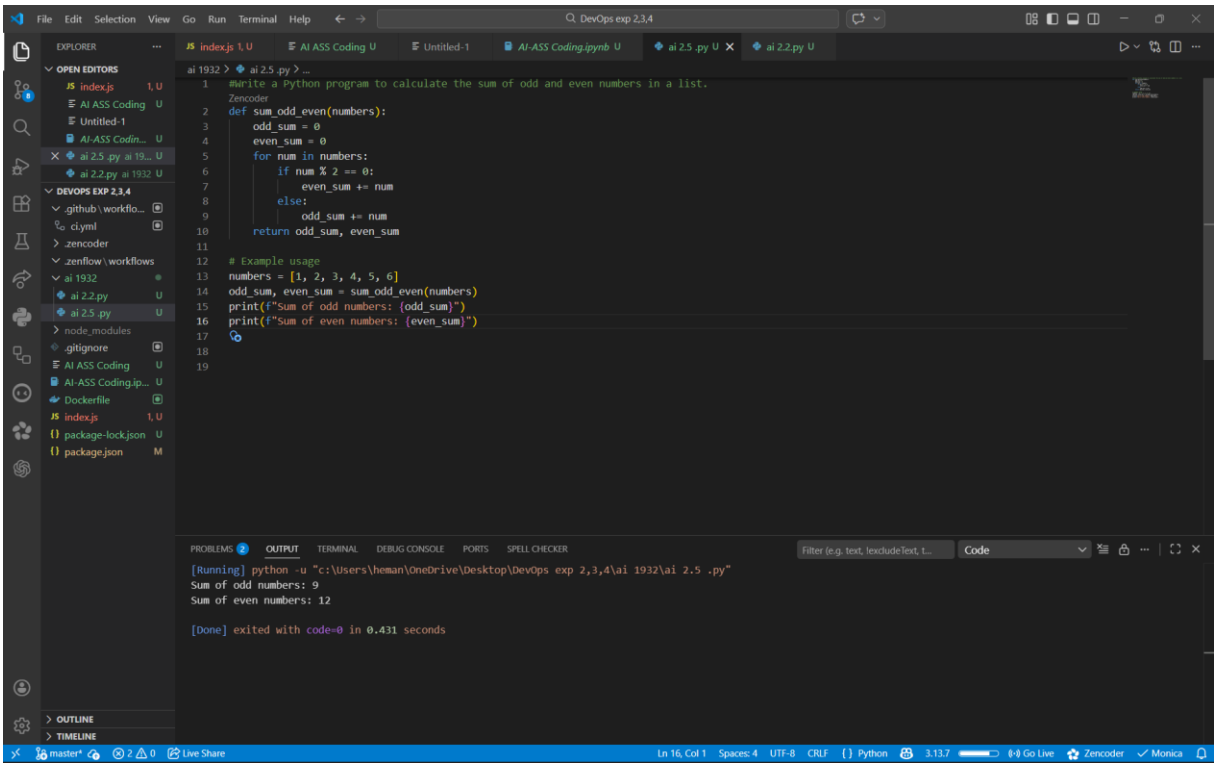
Name-R.Teeneswari

Roll no – 2303A51932

Batch-30 Task-1:

Prompt: Write a program to calculate the sum of odd and even numbers in a list

Code:



Observation:

The **original code** works correctly but is written as a single block, making it harder to reuse and test.

The **refactored (AI-improved) code** separates logic into a function, improving:

- Readability
- Reusability
- Maintainability

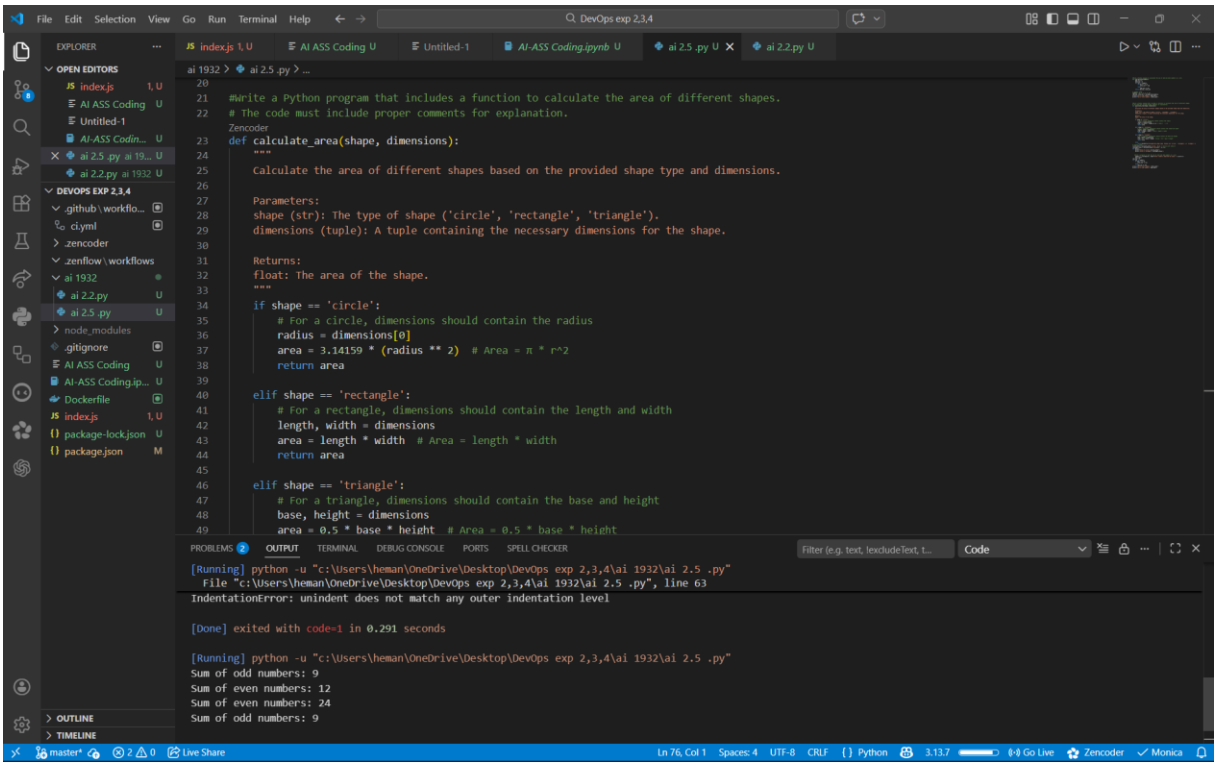
Using a function allows the same logic to be reused with different lists without rewriting code.

Task-2:

Prompt: write a program explain a function that calculates the area of different shapes.

The code must include proper comments for explanation.

Code:



Observation:

This program uses **one function** to calculate the area of **multiple shapes**, which avoids code duplication.

The shape parameter decides **which formula** to apply.

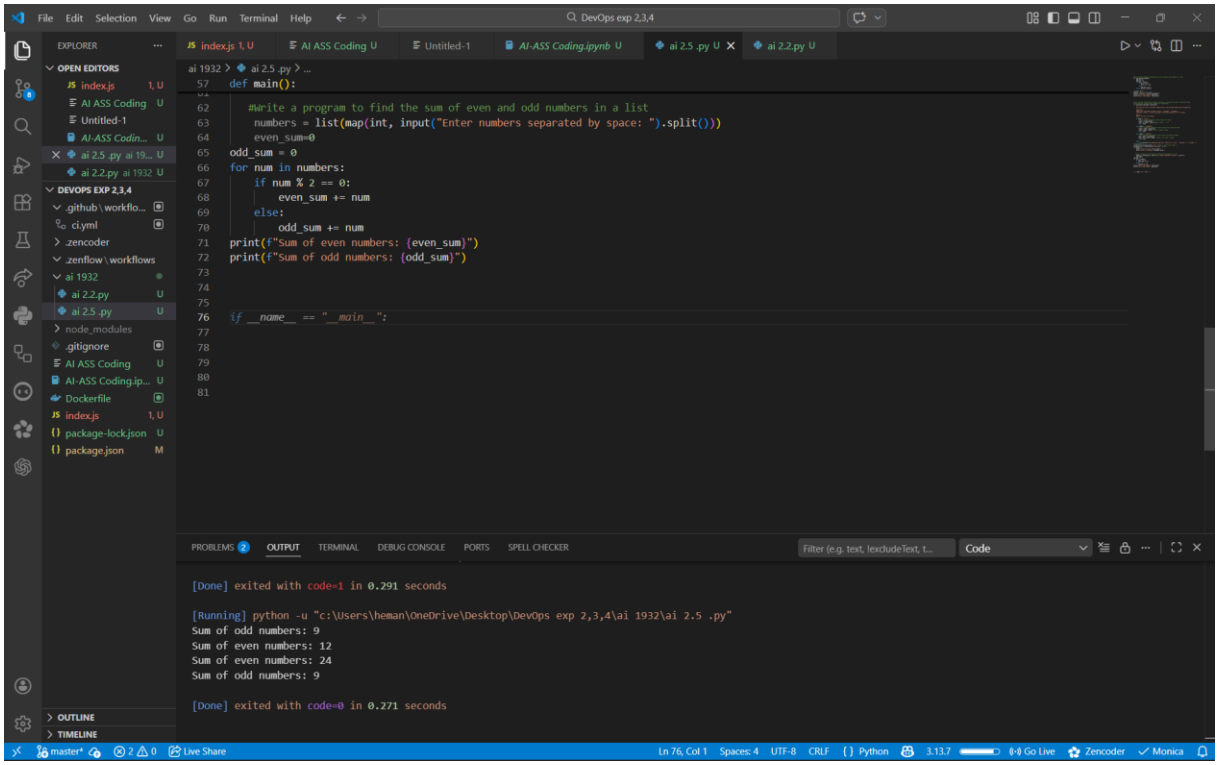
The function uses **conditional statements** (if / elif) to select the correct formula.

It improves **code clarity**, making onboarding easier and faster.

Task:3

Prompt: explain a function that calculates the area of different shapes (curser used)

Shapes. Write a program to find the sum of even and odd numbers in a list Code:



Observation:

The program demonstrates **how one function can handle multiple use cases**.

Comments clearly explain:

What the function does

Why each condition exists

What each parameter represents

Using comments makes the code **junior-developer friendly**, which is ideal for onboarding.

The main () function separates **user interaction** from **business logic**, improving structure.

This style is considered **clean, readable, and professional** in real-world projects. Task-4:

Prompt: Based on practical usage and experimentation, compare **Gemini**, **GitHub Copilot**, and **Cursor AI** in terms of **usability** and **code quality**.

Observation:

Gemini is best suited for **explanations and learning support**. It produces readable, beginner-friendly code and clear step-by-step reasoning, making it ideal for onboarding juniors and understanding concepts.

GitHub Copilot excels in **real-time coding assistance** inside IDEs. It is fast, contextaware, and highly productive for experienced developers, but its code may lack explanations.

Cursor AI stands out for **prompt sensitivity and refactoring quality**. It responds strongly to detailed prompts, generating cleaner, more structured, and optimized code, making it suitable for improving legacy codebases.

usability, Copilot integrates seamlessly into workflows, Gemini is conversational and educational, and Cursor AI offers powerful prompt-driven refactoring.

code quality, Cursor AI and Copilot generally produce more professional, productionready code, while Gemini focuses on clarity over optimization