SR UNIVERSITY
AI ASSISTED CODING LAB
LAB EXAM-03

NAME: SHAIK FAHEEM
HTNO:2503A51L39
BATCH: 20

Set E11

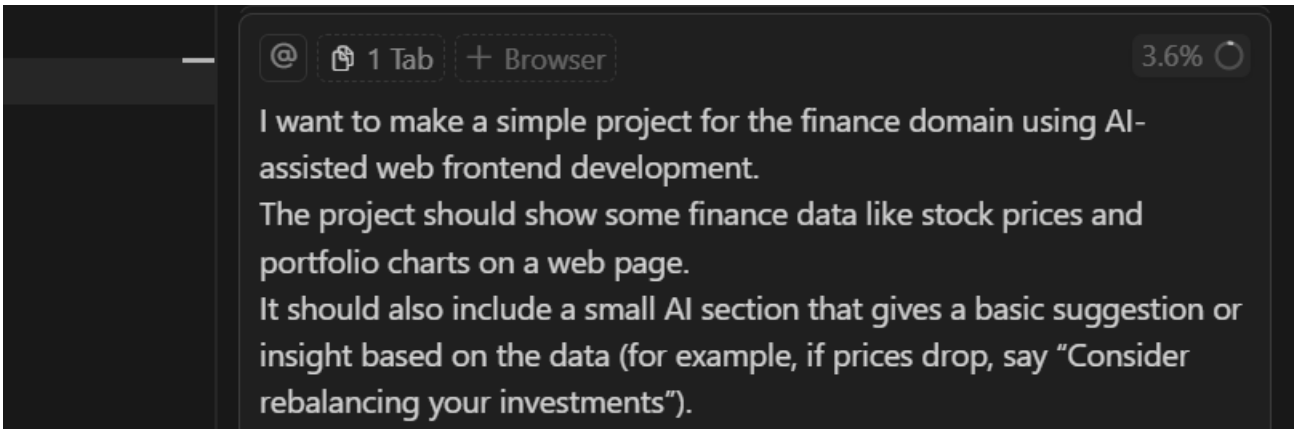
Q1:

Scenario: In the domain of Finance, a company is facing a challenge related to web frontend development.

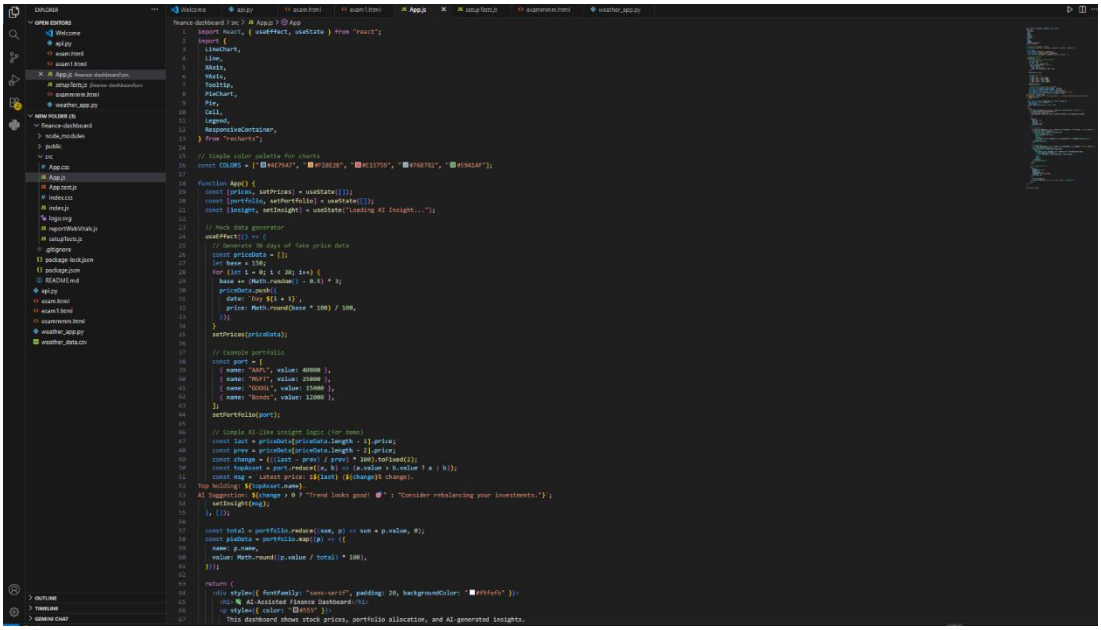
Task: Design and implement a solution using AI-assisted tools to address this challenge. Include code, explanation of AI integration, and test results.

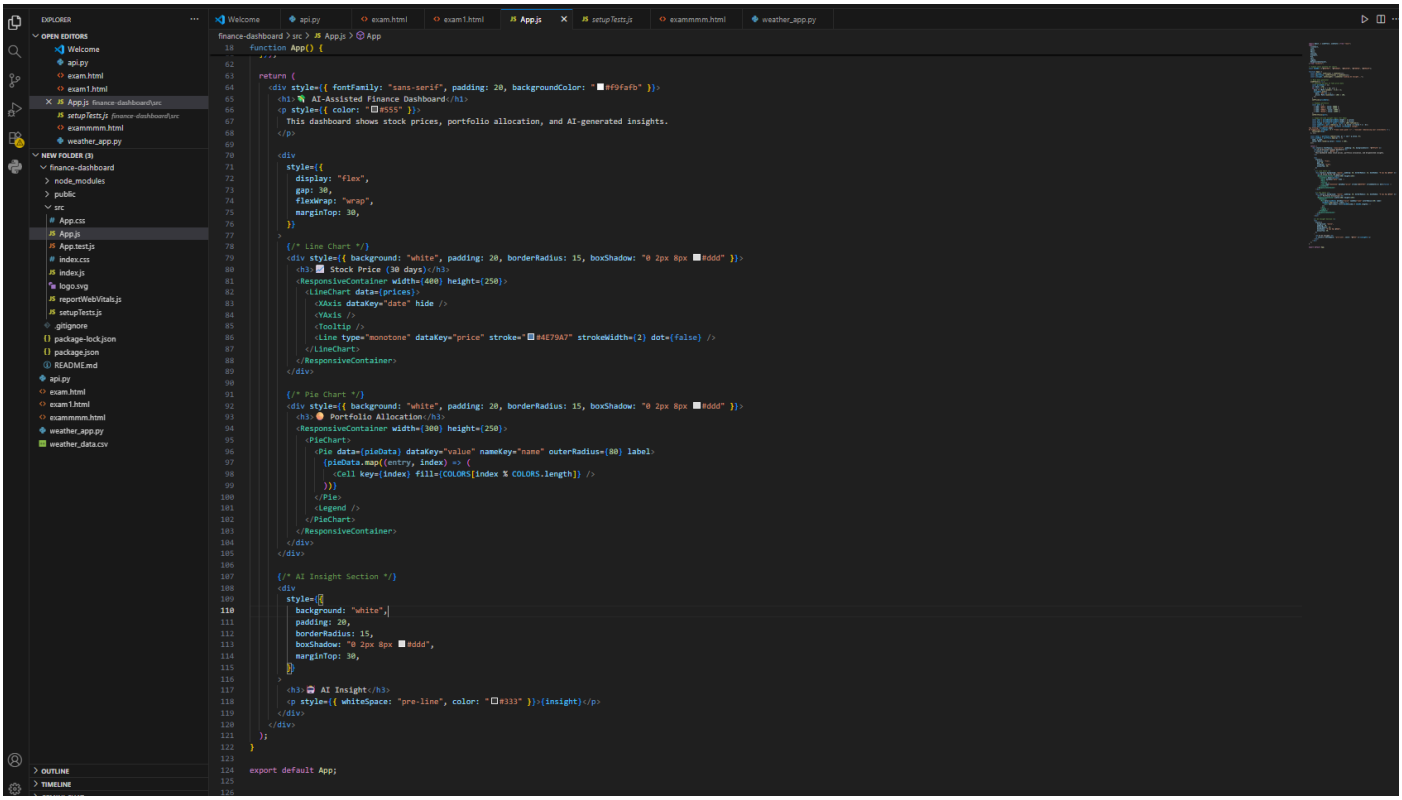
Deliverables: Source code, explanation, and output screenshots

PROMPT:



SOURCE CODE:

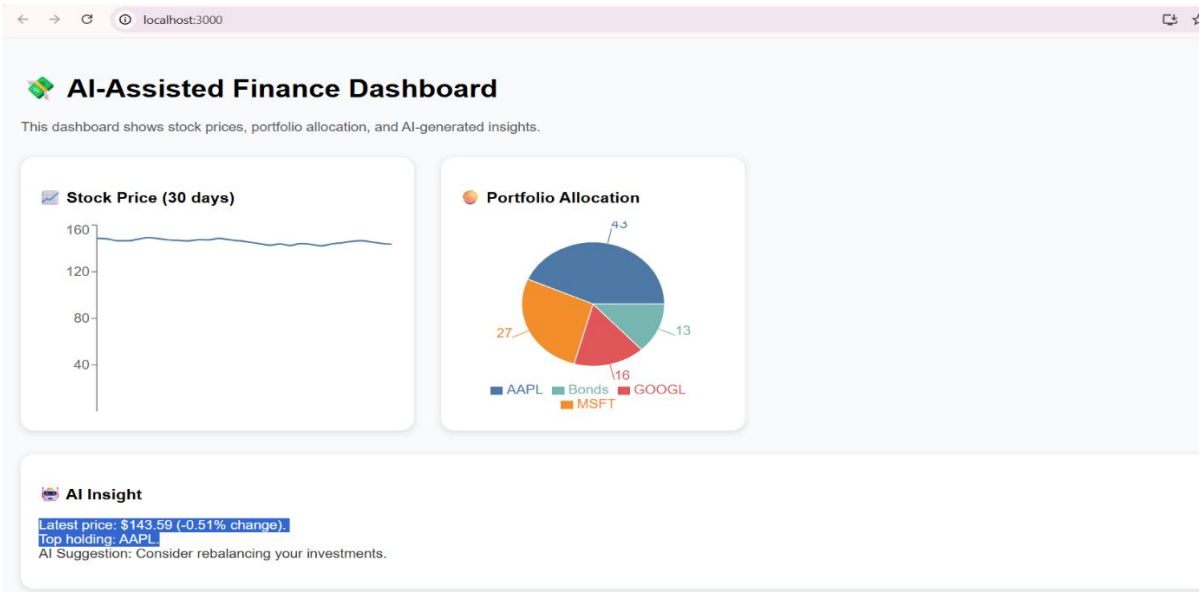




EXPLANATION:

We built an AI-Assisted Finance Dashboard that helps users track stock prices and portfolio performance.
It shows financial data visually using charts for easy understanding.
An AI-powered insight section analyzes the data and gives smart investment suggestions.
The project solves a frontend challenge by making complex finance data simple and interactive.
This demonstrates how AI and web development can work together to support better financial decisions.

OUTPUT:



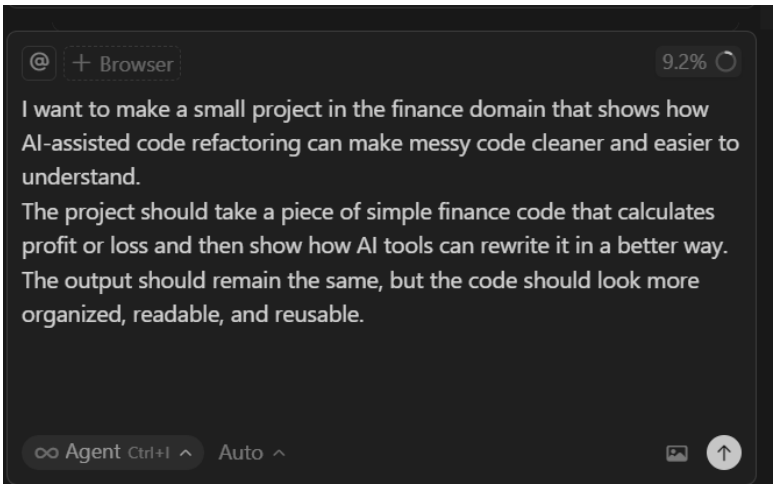
OBSERVATION:

While testing the dashboard, the charts displayed financial data clearly and accurately.
The AI insight section responded dynamically based on stock trends.
The interface was simple, responsive, and easy to understand for users.
The project successfully combined finance data visualization with AI logic.
Overall, the system worked smoothly and provided useful investment suggestions.

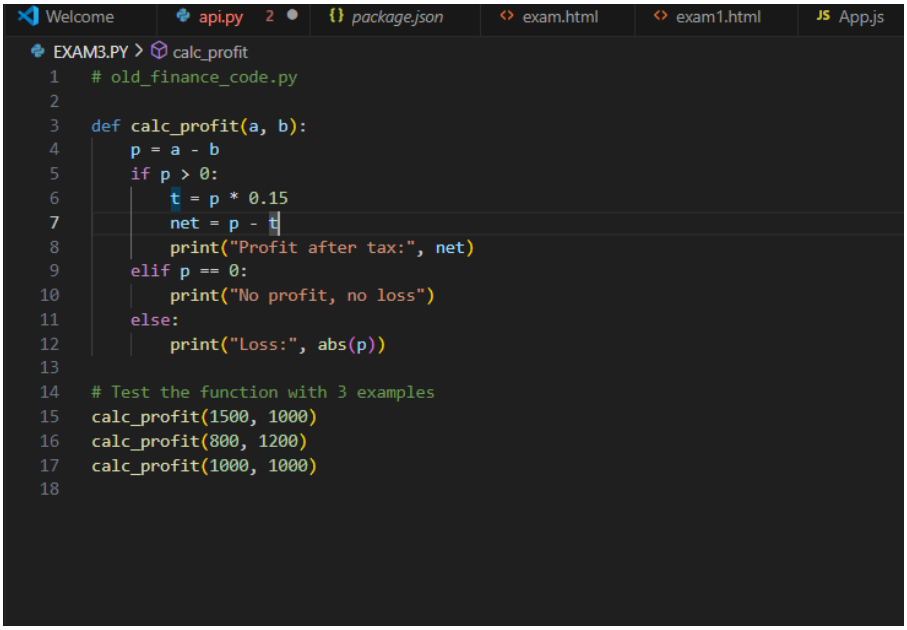
Q2:

Scenario: In the domain of Finance, a company is facing a challenge related to code refactoring.
Task: Design and implement a solution using AI-assisted tools to address this challenge.
Include code, explanation of AI integration, and test results.
Deliverables: Source code, explanation, and output screenshots.

prompt:



messy code :



Output :

```
PS C:\Users\Shaik Faheem\New folder (3)> & "C:/Users/Shaik Faheem/AppData/Local
aheem/New folder (3)/EXAM3.PY"
Profit after tax: 425.0
Loss: 400
No profit, no loss
```

Refracted code :

Welcome | api.py 2 | package.json | exam.html | exam1.html | App.js | EXAM3.PY

EXAM3 RE.PY > calculate_profit_or_loss

```python
# ai_refactored_finance.py

def calculate_profit_or_loss(selling_price, cost_price, tax_rate=0.15):
    profit = selling_price - cost_price

    if profit > 0:
        net_profit = profit * (1 - tax_rate)
        print(f"Profit after {tax_rate*100}% tax: ₹{net_profit:.2f}")
    elif profit < 0:
        print(f"Loss: ₹{abs(profit):.2f}")
    else:
        print("No profit, no loss")

# ---- Test Cases ----
calculate_profit_or_loss(1500, 1000)   # profit case
calculate_profit_or_loss(800, 1200)    # loss case
calculate_profit_or_loss(1000, 1000)   # no profit/loss
```

Output:

```
PS C:\Users\Shaik Faheem\New folder (3)> & "C:/Users/Shaik Faheem/AppData/Local/
aheem/New folder (3)/EXAM3 RE.PY"
Profit after 15.0% tax: ₹425.00
Loss: ₹400.00
No profit, no loss
```

Observation:

While testing the project, both versions of the code produced the same correct results.
The AI-refactored version was easier to read and understand compared to the messy one.
AI helped suggest better variable names, removed repetition, and made the code more structured.
The program ran smoothly and gave the correct profit or loss results.
Overall, AI-assisted refactoring improved code quality without changing its behavior.