Python Developer Internship – Task 7 Report

Name: Khushi Maurya
Internship Title: Python Developer
Company: Main Flow Services and Technologies Pvt. Ltd.
Task: Task 7 – Problem Solving and Web Development

✅ 1. Task Overview

This task focused on solving classic algorithmic and data structure problems in Python, along with developing a real-world Flask web app: **Personal Budget Advisor**.

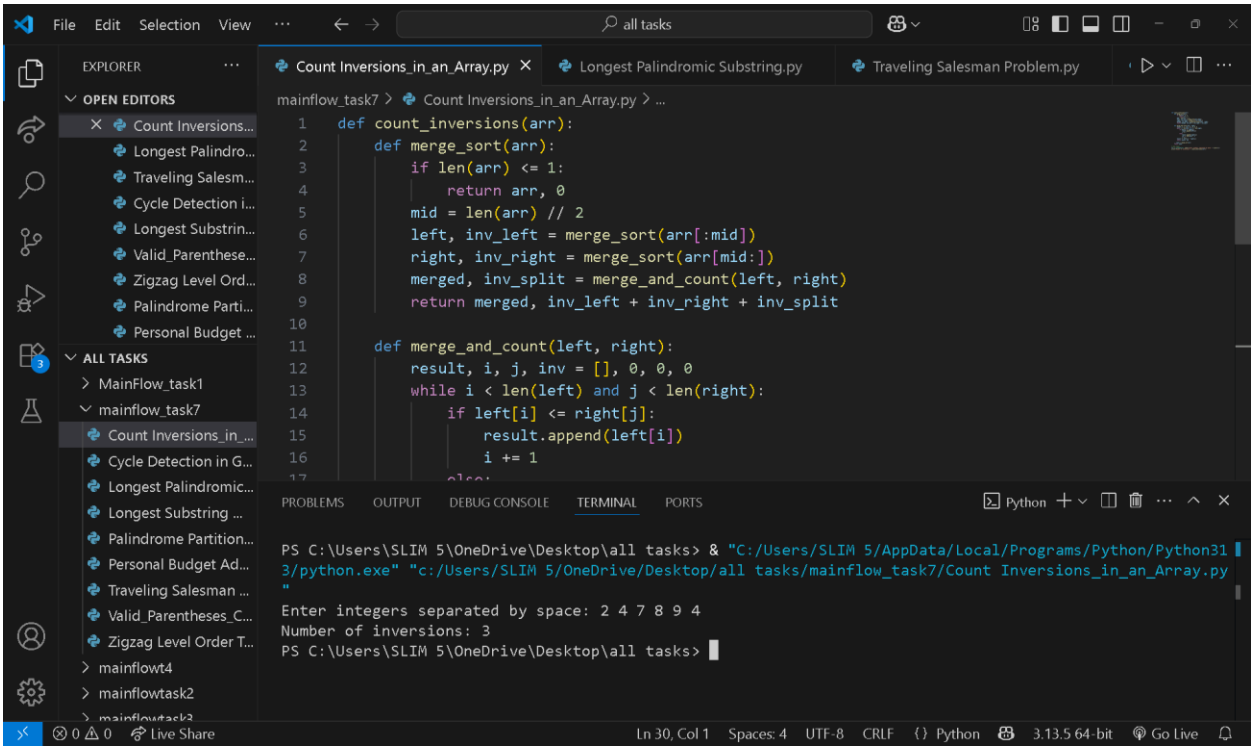
✅ 2. List of Problems Solved

---

47. Count Inversions

- **Objective:** Count how many pairs (i, j) exist where i < j and a[i] > a[j].
- **Technologies:** Python
- **Approach:** Modified Merge Sort
- **Input:** List of integers
- **Output:** Integer (number of inversions)

✅ Example:
Input: [2, 4, 1, 3, 5] → Output: 3

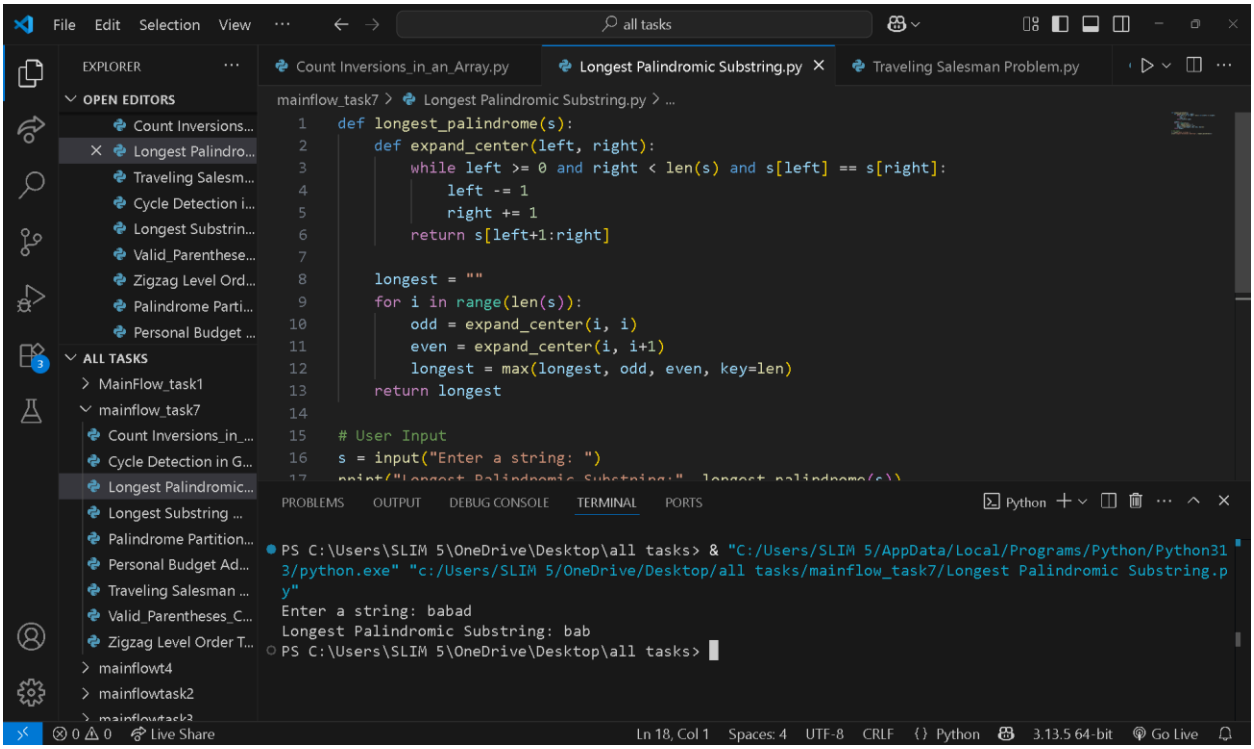


---

48. Longest Palindromic Substring

- **Objective:** Find the longest palindromic substring in a given string.
- **Approach:** Expand around center
- **Input:** A string
- **Output:** Longest palindrome substring

✅ Example:
Input: babad → Output: bab or aba

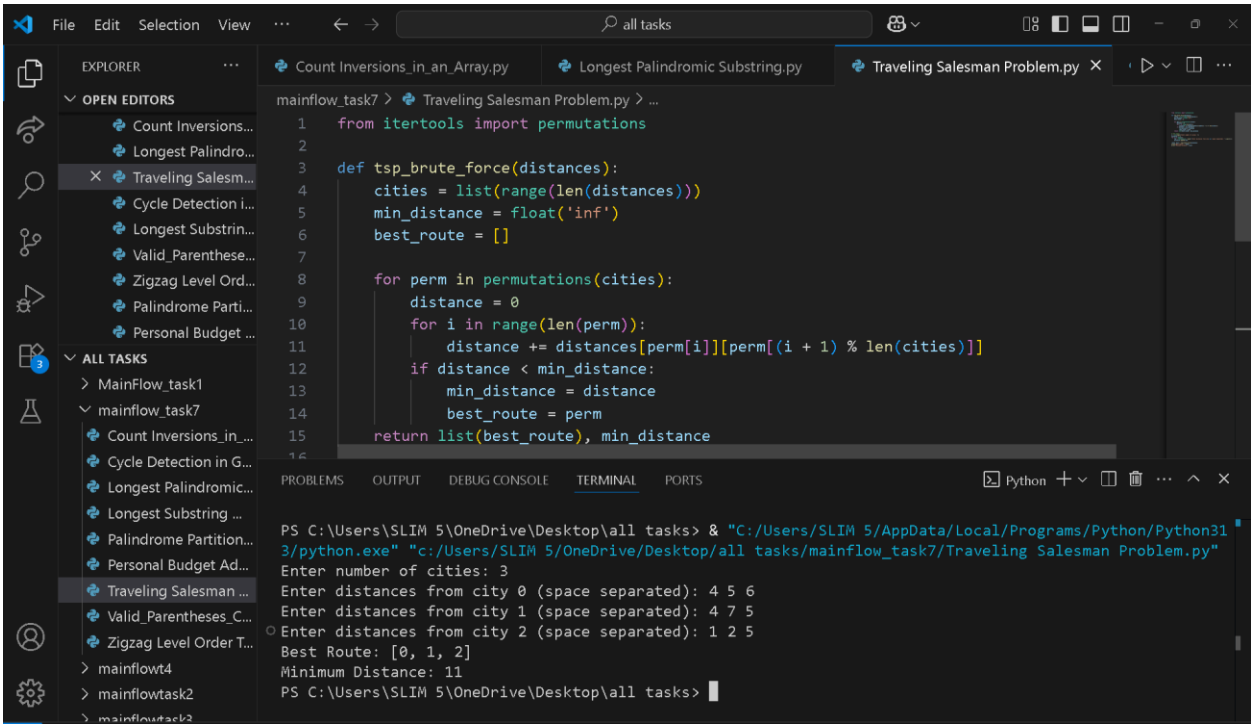


---

49. Traveling Salesman Problem (TSP)

- **Objective:** Find the shortest path visiting each city once and returning to the start.
- **Technologies:** Python
- **Approach:** Dynamic programming
- **Input:** Distance matrix
- **Output:** Shortest route and distance

✅ Example Output:
Best Route: [0, 1, 3, 2]
Minimum Distance: 80

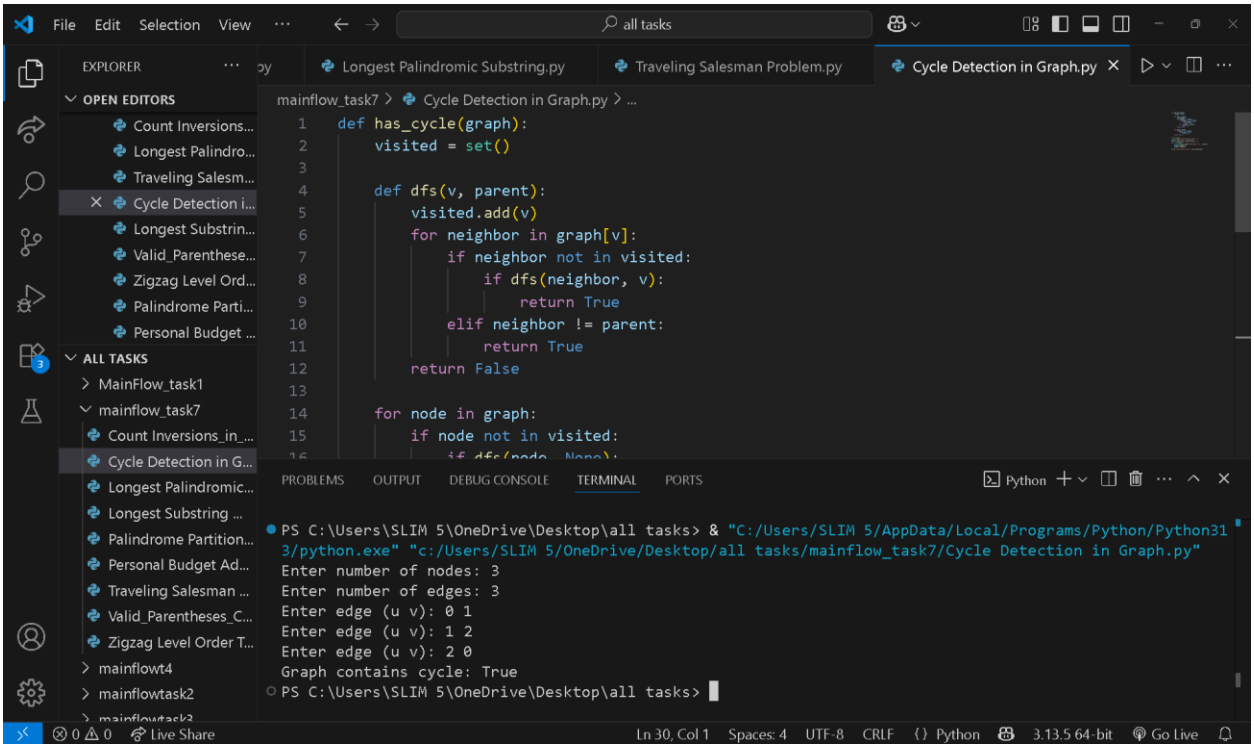


---

50. Graph Cycle Detection

- **Objective:** Detect if an undirected graph contains a cycle.
- **Approach:** Depth-First Search (DFS)
- **Input:** Adjacency list
- **Output:** True / False

✅ Example:
Graph with cycle → True
Graph without cycle → False

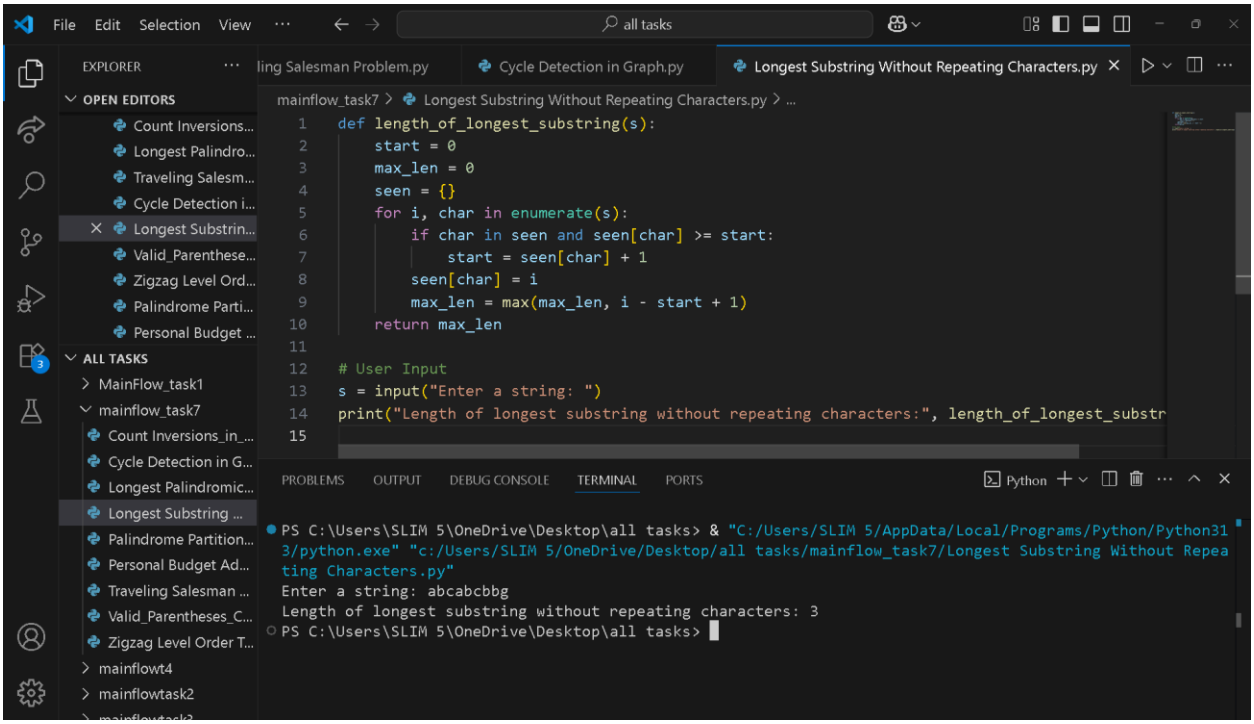


---

51. Longest Substring Without Repeating Characters

- **Objective:** Find the length of the longest substring without repeating characters.
- **Approach:** Sliding Window
- **Input:** A string
- **Output:** Integer length

✅ Example:
Input: abcabcbb → Output: 3 (abc)

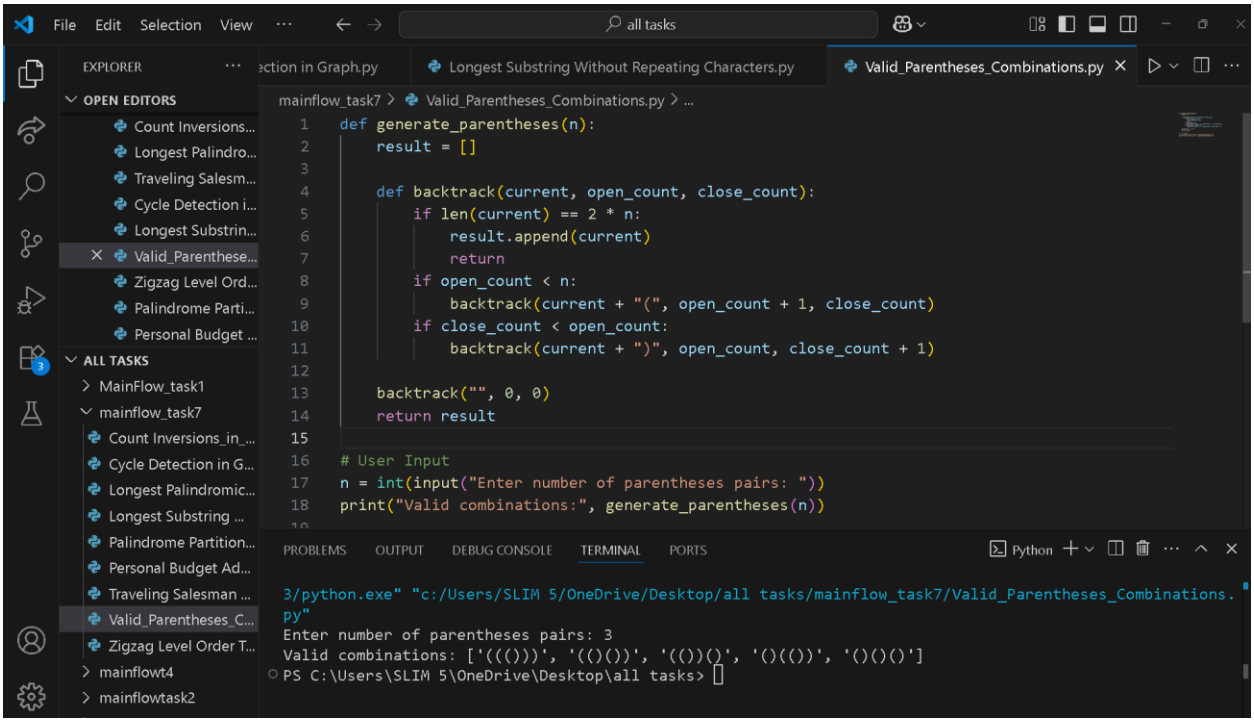


---

52. Valid Parentheses Combinations

- **Objective:** Generate all valid combinations of n pairs of parentheses.
- **Approach:** Backtracking
- **Input:** Integer n
- **Output:** List of valid strings

✅ Example:
Input: n = 3 → Output: ['((()))', '(()())', '(())()', '()(())', '()()()']

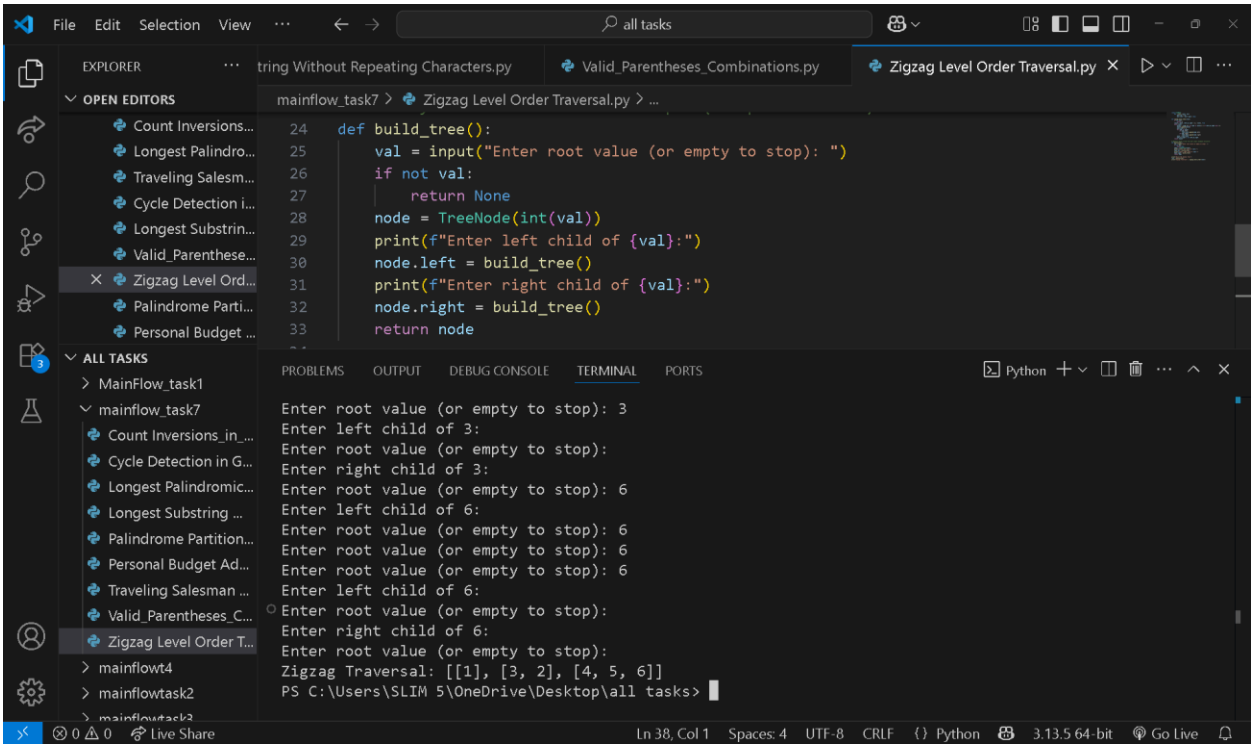


---

53. Zigzag Level Order Traversal of Binary Tree

- **Objective:** Traverse a binary tree in zigzag (left-right then right-left) level order.
- **Approach:** Breadth-first traversal using two stacks
- **Input:** Binary tree
- **Output:** List of levels in zigzag

✅ Output Example:
[[1], [3, 2], [4, 5, 6]]



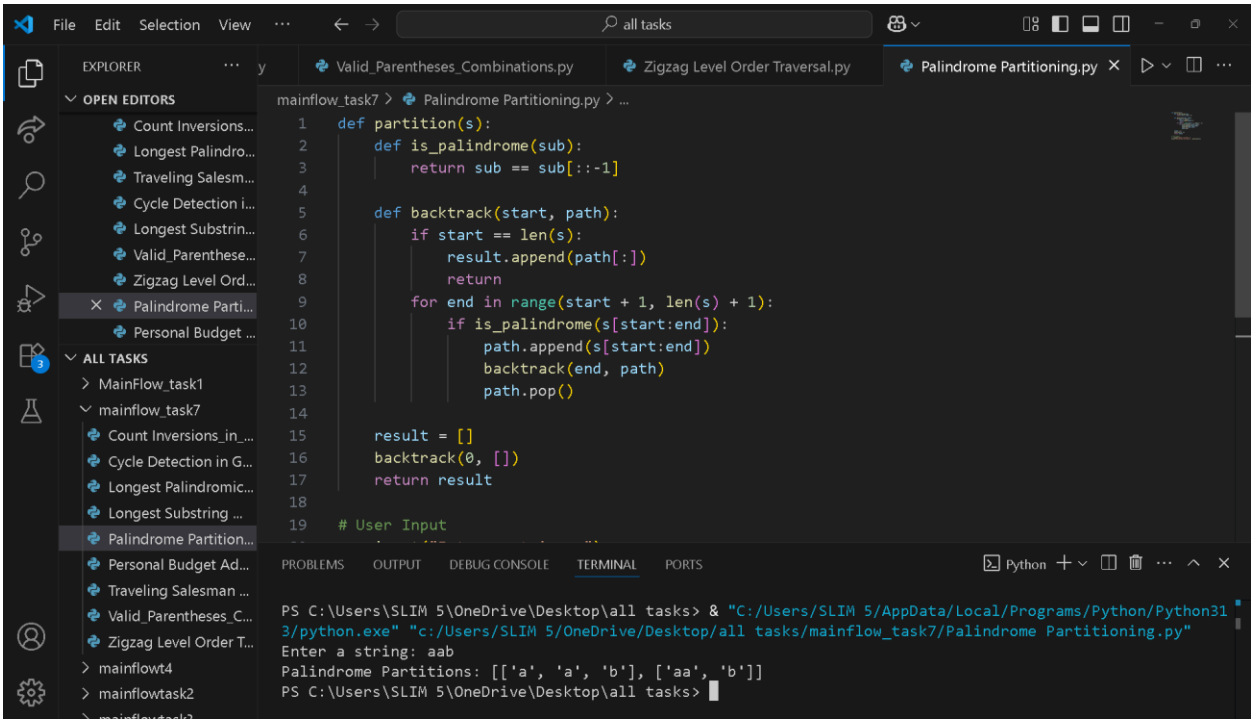
---

54. Palindrome Partitioning

- **Objective:** Partition a string such that each substring is a palindrome.
- **Approach:** Backtracking
- **Input:** A string
- **Output:** List of lists

✅ Example:

Input: aab → Output: [['a', 'a', 'b'], ['aa', 'b']]



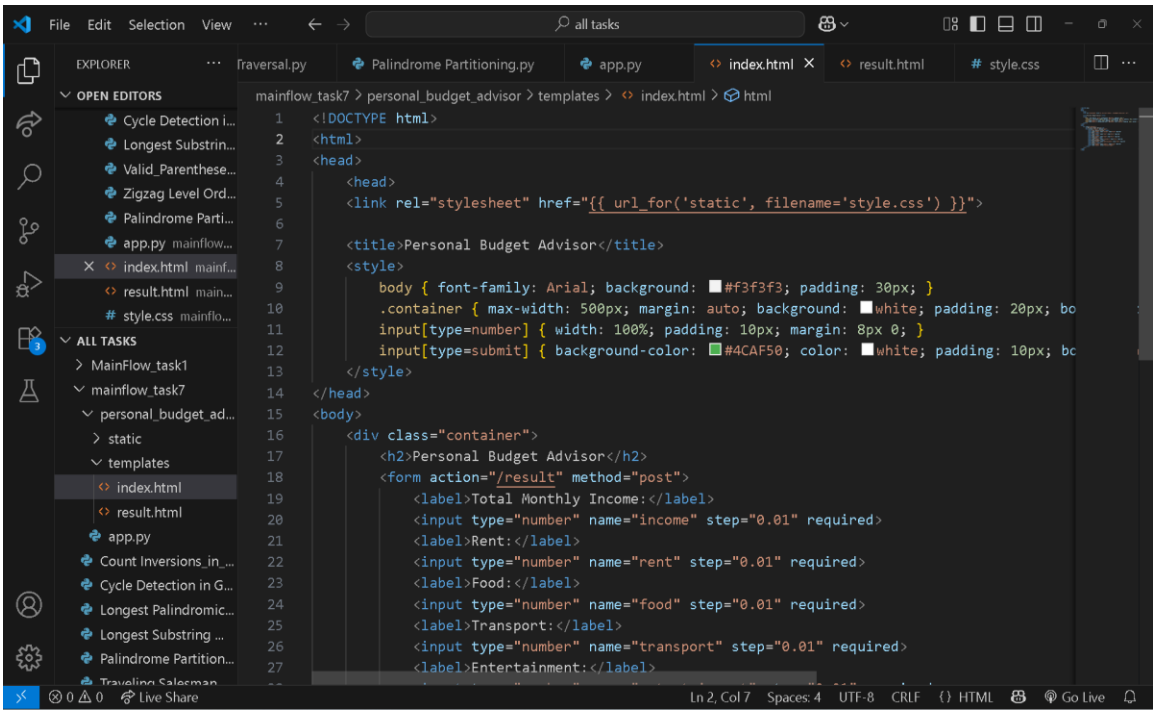
3. Personal Budget Advisor Web App (Flask Project)

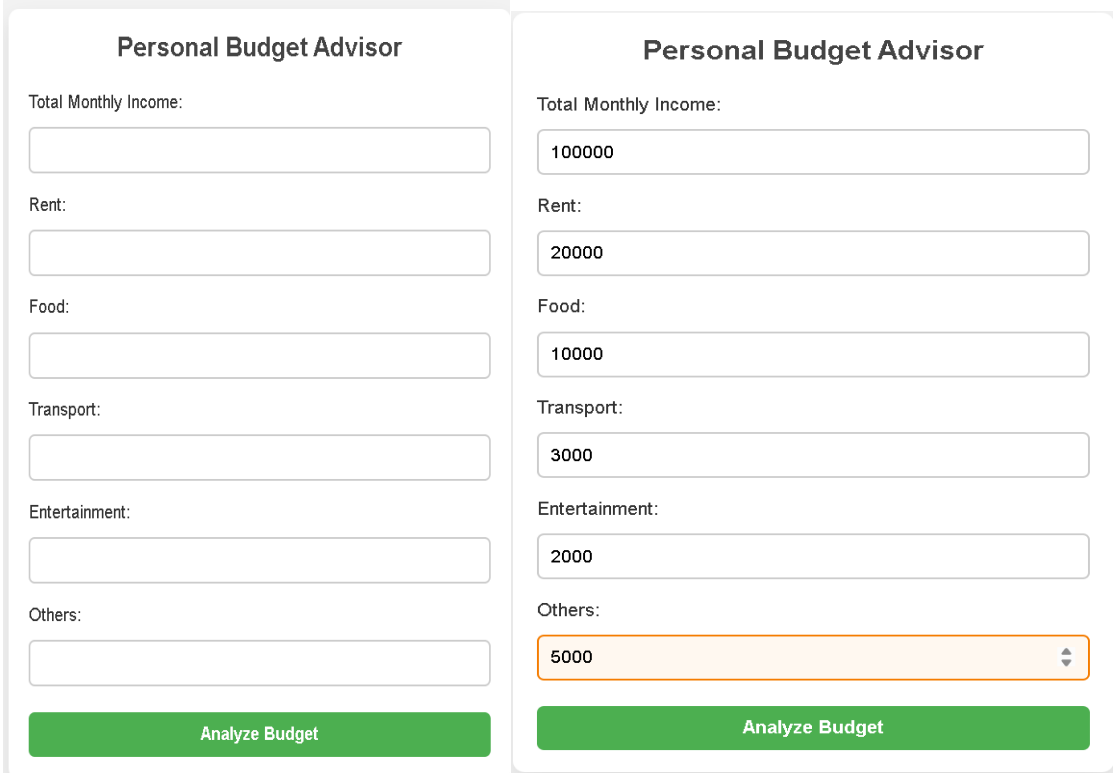
- **Objective:**

To build a web-based budget advisor tool using Flask that allows users to enter income and expenses, calculates total spending, shows savings, and gives suggestions.

- **Technologies Used:**
  - Python 3.13
  - Flask Framework
  - HTML, CSS (via static/style.css)
  - Jinja2 Templates
- **Features:**
  - Input form for income and expenses
  - Calculates expense breakdown (amount + %)
  - Computes total savings
  - Gives suggestions: save more / good job
  - Clean and modern UI

Screenshots:





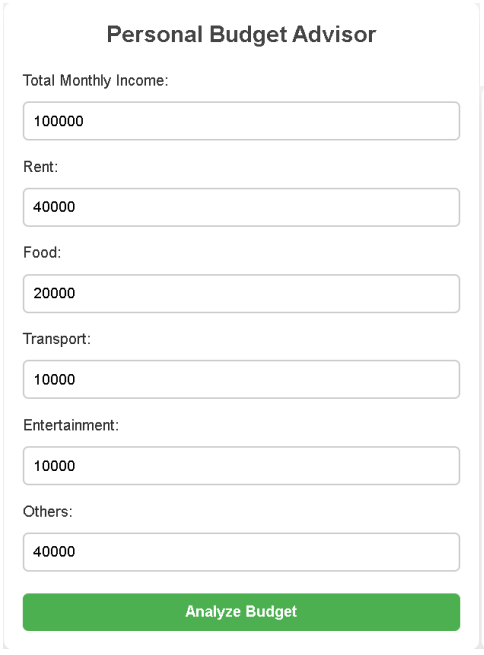
Budget Summary

Total Monthly Income: ₹100000.0

Category	Amount (₹)	Percent (%)
Rent	20000.0	20.0
Food	10000.0	10.0
Transport	3000.0	3.0
Entertainment	2000.0	2.0
Others	5000.0	5.0

Total Expenses: ₹40000.0
Total Savings: ₹60000.0

✅ Good job on your savings!



Budget Summary

Total Monthly Income: ₹100000.0

Category	Amount (₹)	Percent (%)
Rent	40000.0	40.0
Food	20000.0	20.0
Transport	10000.0	10.0
Entertainment	10000.0	10.0
Others	40000.0	40.0

Total Expenses: ₹120000.0
Total Savings: ₹-20000.0

⚠️ Try to cut down your expenses by at least 10%.

✅ 5. Pros and Cons

✅ Pros:

- All solutions implemented manually using core Python (no pandas/numpy).

- Efficient logic using optimal algorithms (merge sort, DFS, DP).
- Clean and interactive Flask UI.
- Great real-world project for resume and portfolio.

❌ **Cons:**

- No persistent database for saving data (can be improved).
- No charts or data visualization (could be added).
- Budget app doesn't store history.

✅ **6. Conclusion**

This task helped in mastering:

- Real-world problem solving with Python
- Web development with Flask and HTML/CSS
- Core concepts of algorithms, backtracking, dynamic programming
- Frontend/backend integration

It also gave a practical project (budget advisor) that can be extended into a complete finance management app.