**East West University**

**CSE412: Software Engineering**

**Project Report (Software Testing)**

**PET : Personal Expense Tracker**

**Section: 2**

**Group: 08**

**Group Name: Team Noob**

**Name & ID of Students:**

| Abu Kahhar Mohammad Sakib | 2021-1-60-040 |
| --- | --- |
| Prioti Kar Tithy | 2022-1-60-082 |
| Budrun Nahar Bristy | 2022-1-60-144 |
| Fatema Tuz Zannat | 2022-1-60-153 |

**Course Instructor Information:**

Yasin Sazid

Lecturer

Department of Computer Science and Engineering

East West University

**1. Introduction:**

**Project Overview:**

## The Personal Expense Tracker is a software application that enables users to manage their daily financial transactions efficiently. Through this tracker user can track the budget via a web-based application. For starters, he sets up an account via his email and password. Upon sign-up, he securely logs in to access his saved financial data on the website. With the dashboard, he can create new spending and income transactions by inputting fields like description, date, category, and amount. If they make a mistake, they can revise or remove transactions to maintain their records accurately.

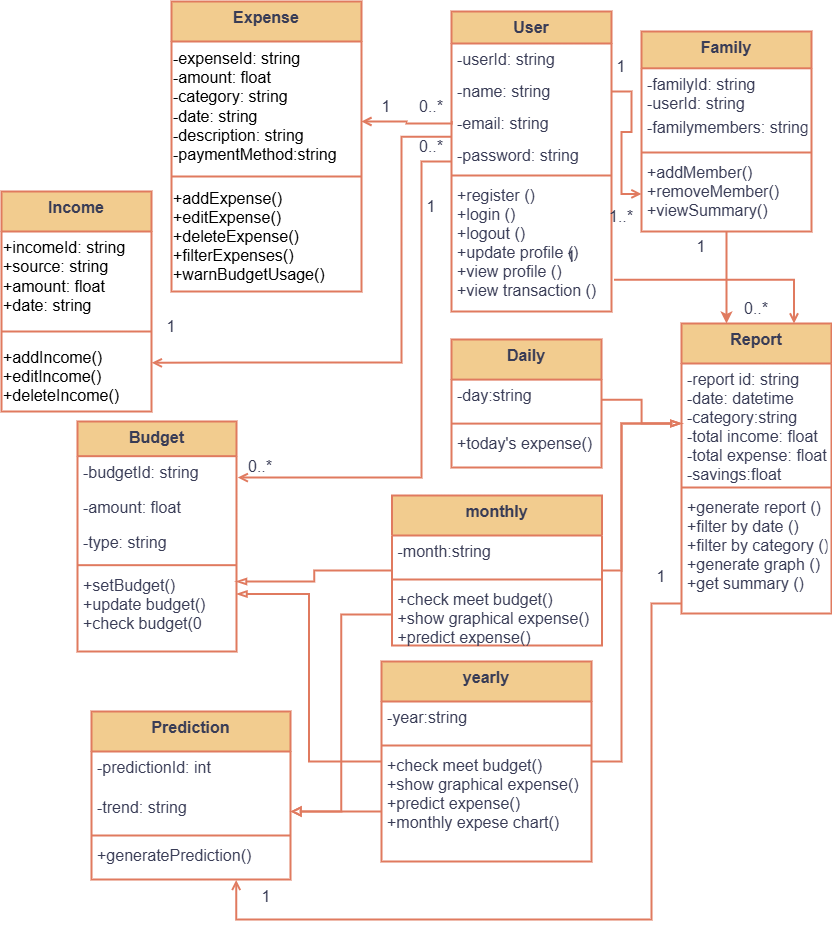
To better control expenses, the user allocates transactions to predefined categories like Food, Rent, and Entertainment. They can define monthly budget limits for certain categories or all together to limit spending and get notified when they are near or have crossed their budget. The system also offers AI-driven forecasts for next month's spending based on historical spending patterns, enabling them to plan their finances.

The user can also invite family members to the platform, making it a family-shared expense tracker. Every member can monitor their income and expenses, which will create a combined financial summary. The system will update all transactions in real-time, so everyone will be made aware of the economic condition of the family. To check their spending, users visit the reports feature, where they get interactive graphs and charts displaying money trends. Users can filter reports based on date, category, or transaction type to receive precise information. The system should further provide a breakdown of total income, spending, and savings over a specified time and give a general financial overview for the user or the entire family.

**List of Components and Associated Classes:**

| **Components** | **Associated Classes** |
| --- | --- |
| Authentication | Admin and User |
| ExpenseManager | Expense |
| IncomeManager | Income |
| BudgetManager | Budget |
| ReportGenerator | Report |
| ChartVisualizer | Income, Expense, Budget |
| ExpensePredictor | Prediction |
| FamilyBudeget | Family, Budget |

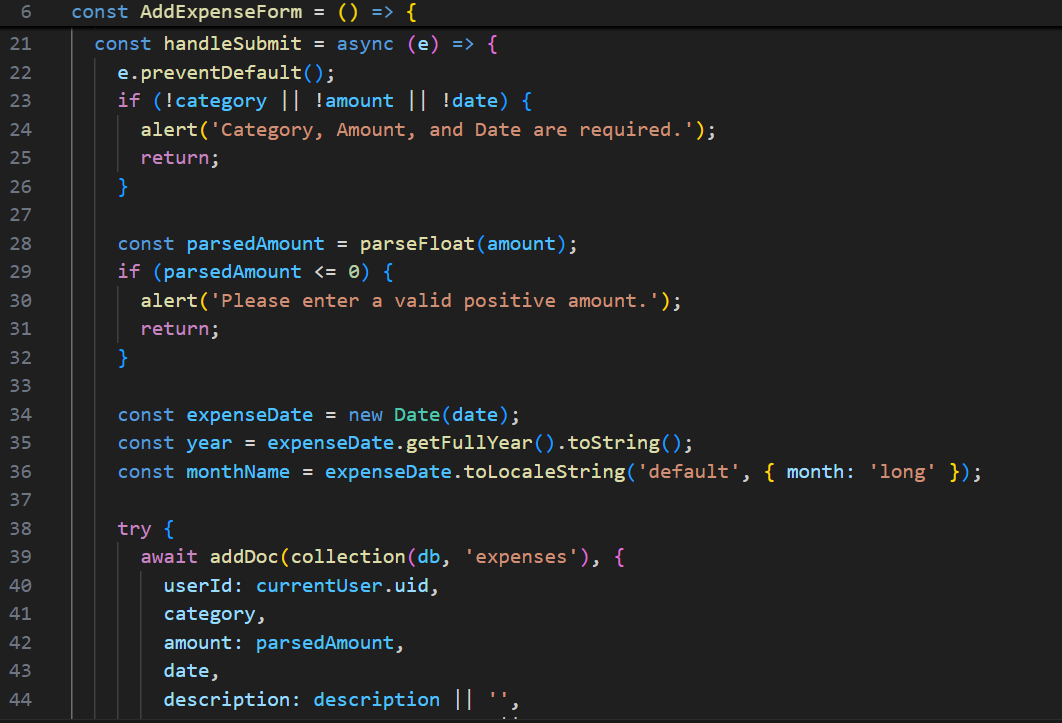
**Class Diagram:**

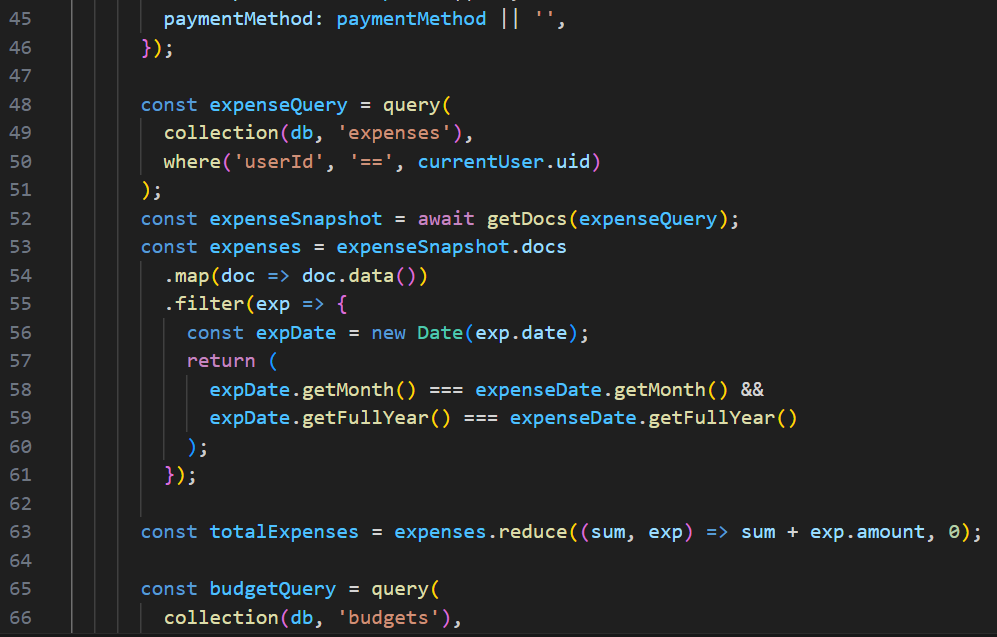
****

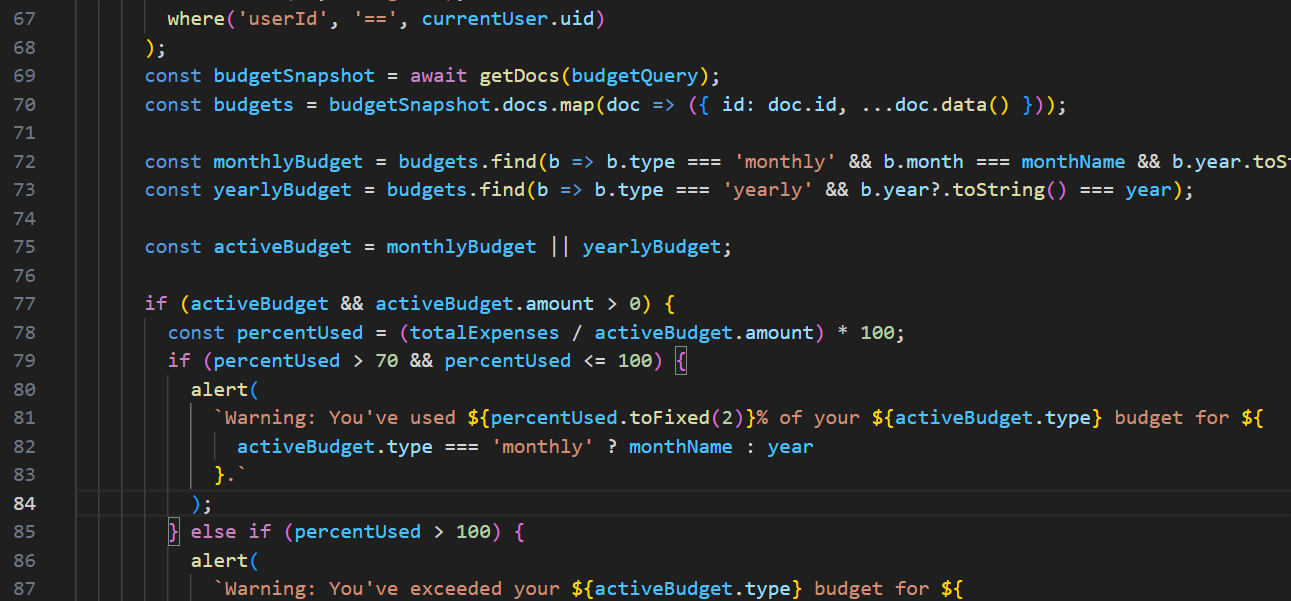
**2. White Box Testing**

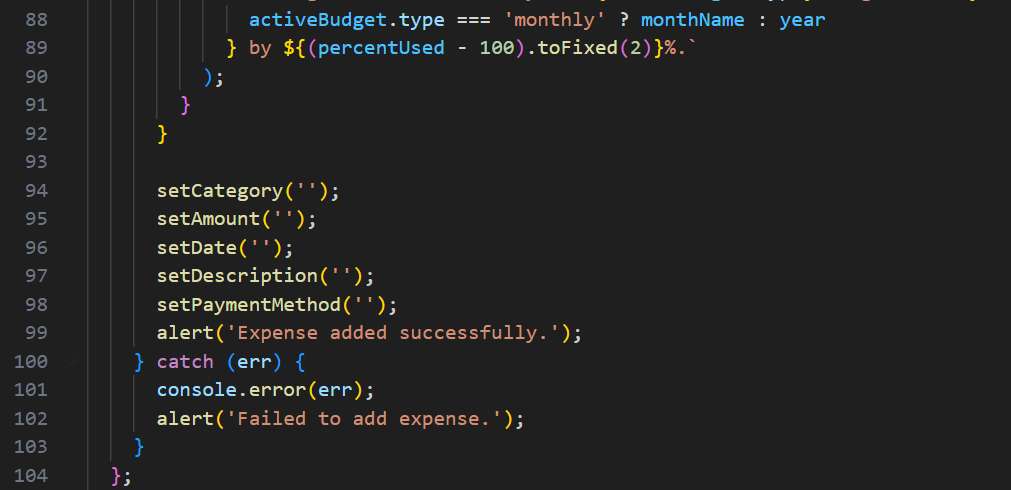
**For Expense Class:**

**handleSubmit( )**

****

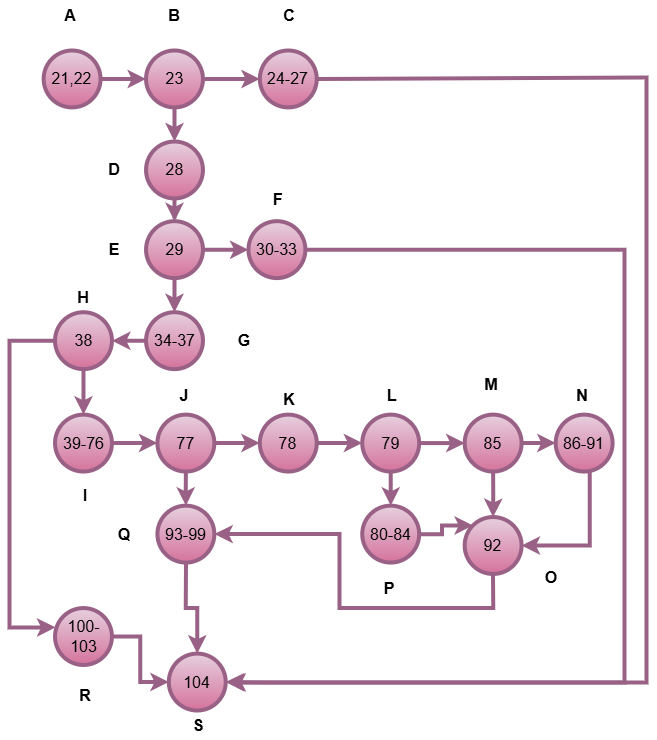
****

****

****

**Control Flow Graph**

**DD Graph (Decision-to-Decision Graph) of handleSubmit( ):**

****

**Cyclomatic complexity of handleSubmit( ):**

Using McCabe’s formula: V(G) = E - N + 2P

Where:

E = Number of edges = 24

N = Number of nodes = 19

P = Number of connected components = 1

V(G) = E - N + 2P

= 24 – 19 + 2\*1

= 7

So, there are 7 independent paths.

**Independent Paths:**

* A → B → C → S
* A → B → D → E → F → S
* A → B → D → E → G → H → R → S
* A → B → D → E → G → H → I → J → Q → S
* A → B → D → E → G → H → I → J → K → L → M → N → O → Q → S
* A → B → D → E → G → H → I → J → K → L → M → O → Q → S
* A → B → D → E → G → H → I → J → K → L → P → O → Q → S

Test Cases:

| Test Case ID | Component | Class | Input | Expected Output' | Actual Output | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- |
| WTC1 | ExpenseManager | Expense | category = "", amount = "", date = "" | Please fill out this Form | Please fill out this Form | Pass |
| WTC2 | ExpenseManager | Expense | category = "Food", amount = "-100", date = "2025-05-01" | Alert: "Please enter a valid positive amount." | Alert: "Please enter a valid positive amount." | Pass |
| WTC3 | ExpenseManager | Expense | Valid input, no budget in DB | Expense added successfully, no warning | Expense added successfully, no warning | Pass |
| WTC4 | ExpenseManager | Expense | Valid input, budget exists, >=70% used | Expense added successfully, warning shown | Expense added successfully, warning shown | Pass |
| WTC5 | ExpenseManager | Expense | Valid input, budget exceeded | Expense added successfully, exceed warning | Expense added successfully, exceed warning | Pass |
| WTC6 | ExpenseManager | Expense | Valid input, budget < 70% used | Expense added successfully, no warning | Expense added successfully, no warning | Pass |
| WTC7 | ExpenseManager | Expense | DB write fails (mock error) | Alert: "Failed to add expense." | Alert: "Failed to add expense." | Pass |

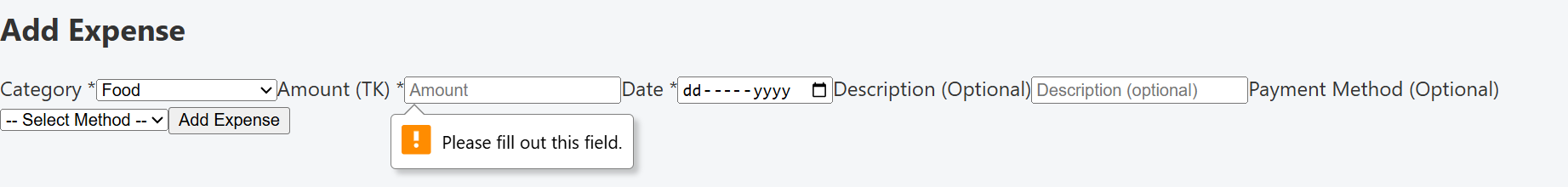


Fig 1: WTC1

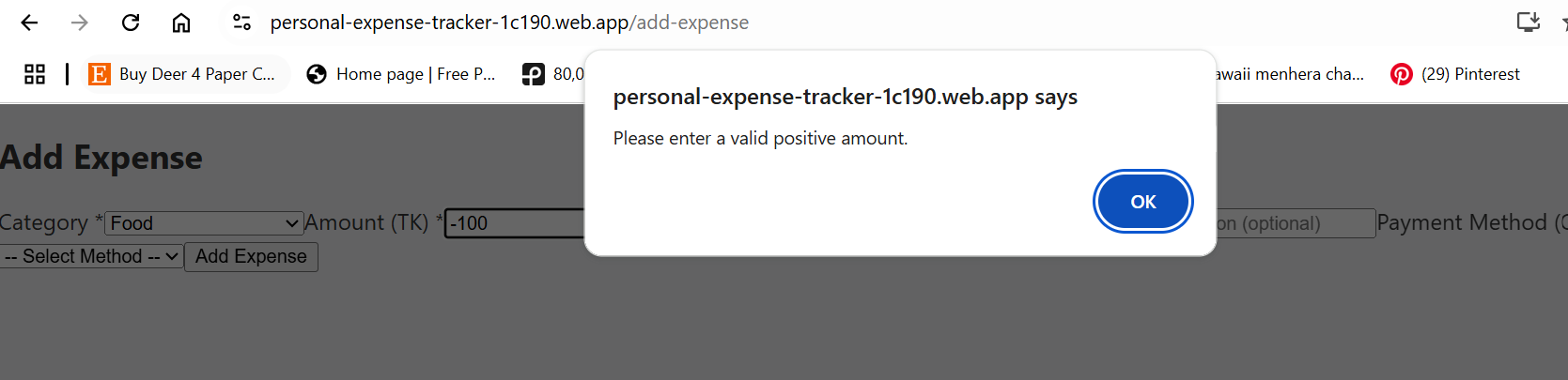


Fig 2: WTC2

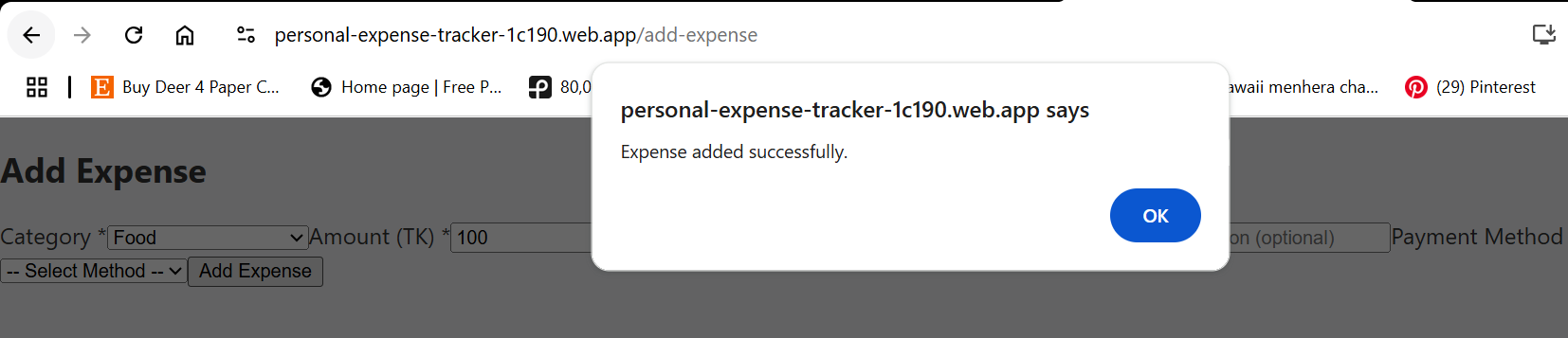
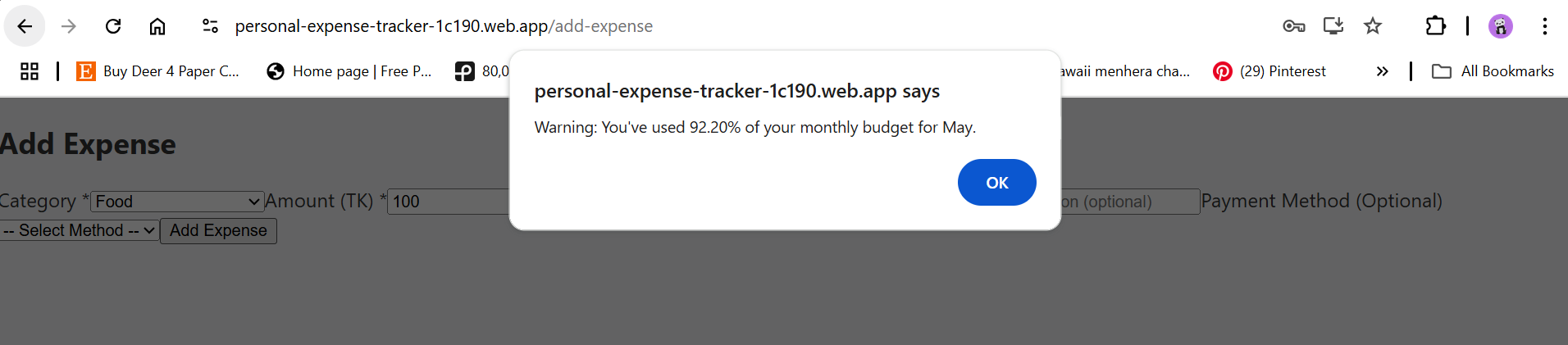


Fig 3: WTC3



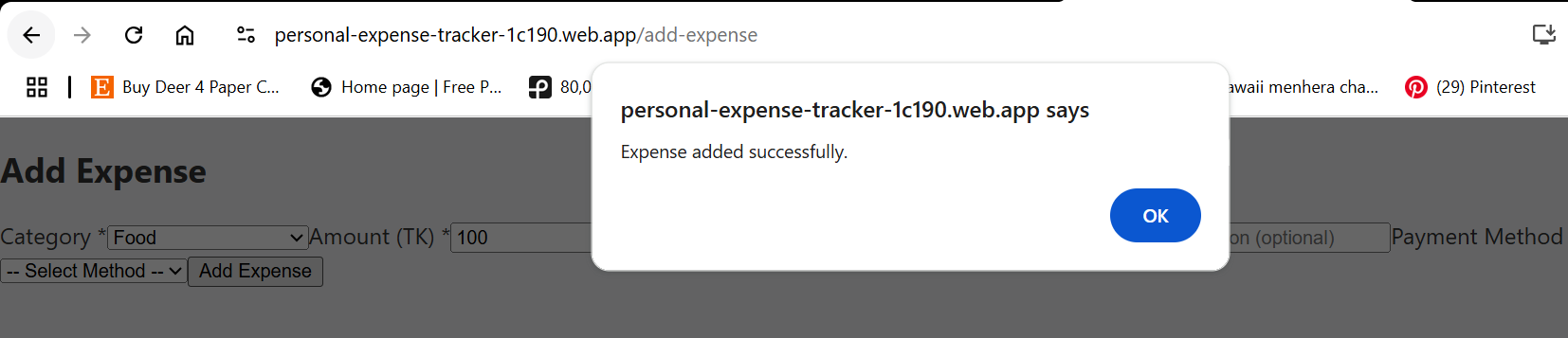
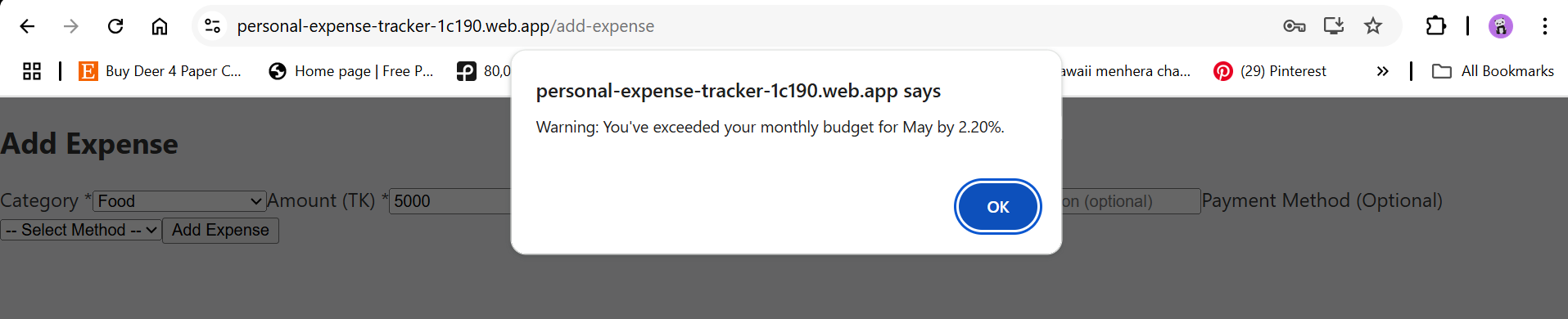


Fig 4: WTC4



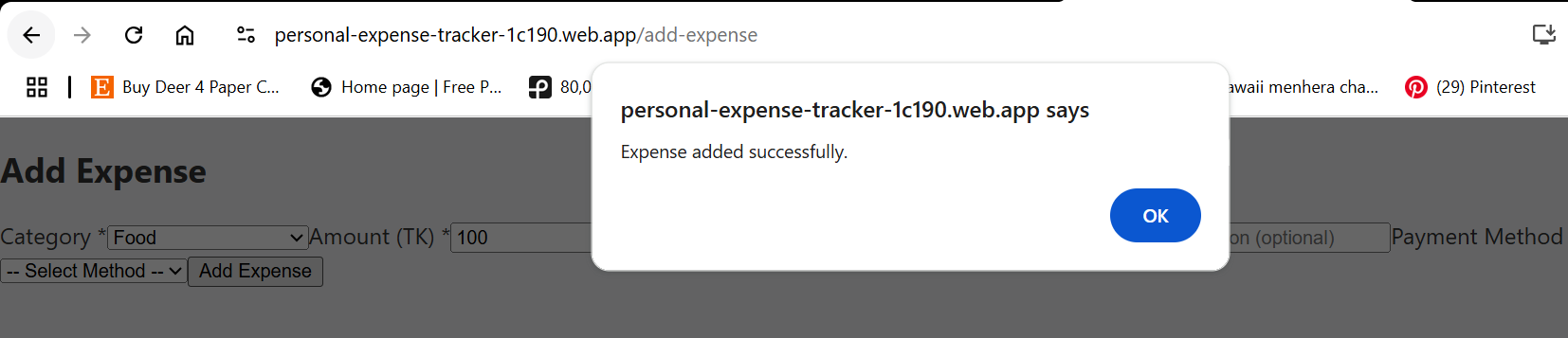


Fig 5: WTC5

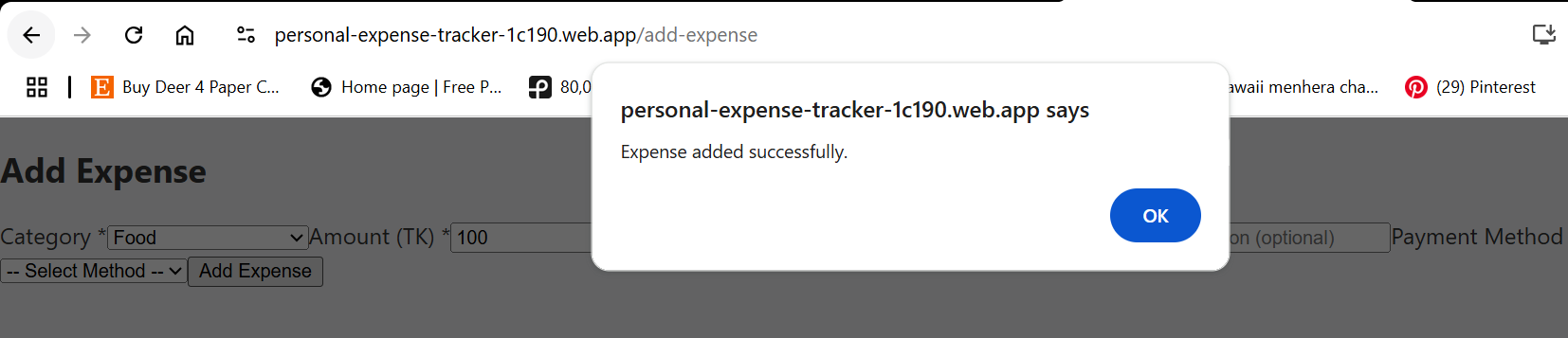


Fig 6: WTC6

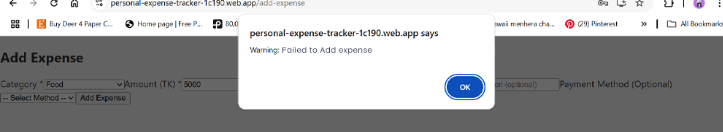
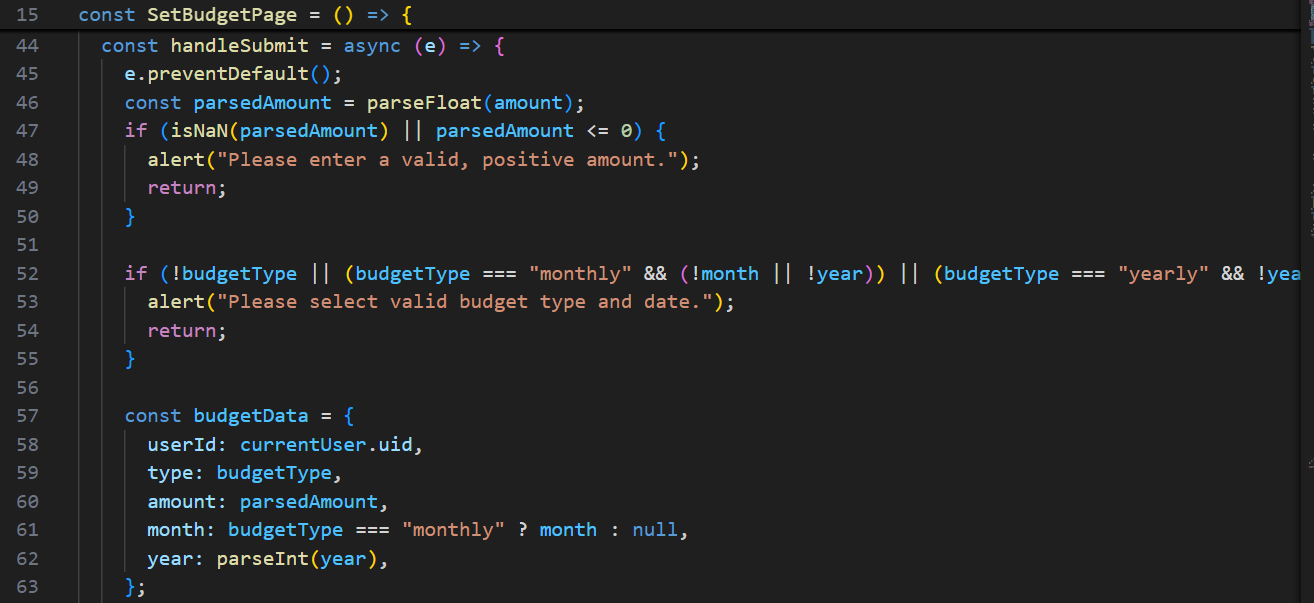


Fig 7: WTC7

**For Budget Class:**

**handleSubmit( )**



**3. Black Box Testing**

**AddExpense : User Input Field for BVA: Amount**

* Expense must be a valid positive amount

| Test Case ID | Component | Class | Input | Expected Output' | Actual Output | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- |
| BTC1 | ExpenseManager | Expense | 0.01 | Expense added successfully, no warning | Expense added successfully, no warning | Pass |
| BTC2 | ExpenseManager | Expense | 1 | Expense added successfully, no warning | Expense added successfully, no warning | Pass |
| BTC3 | ExpenseManager | Expense | 1000000 | Expense added successfully, no warning | Expense added successfully, no warning | Pass |
| BTC4 | ExpenseManager | Expense | 999999999 | Expense added successfully, no warning | Expense added successfully, no warning | Pass |

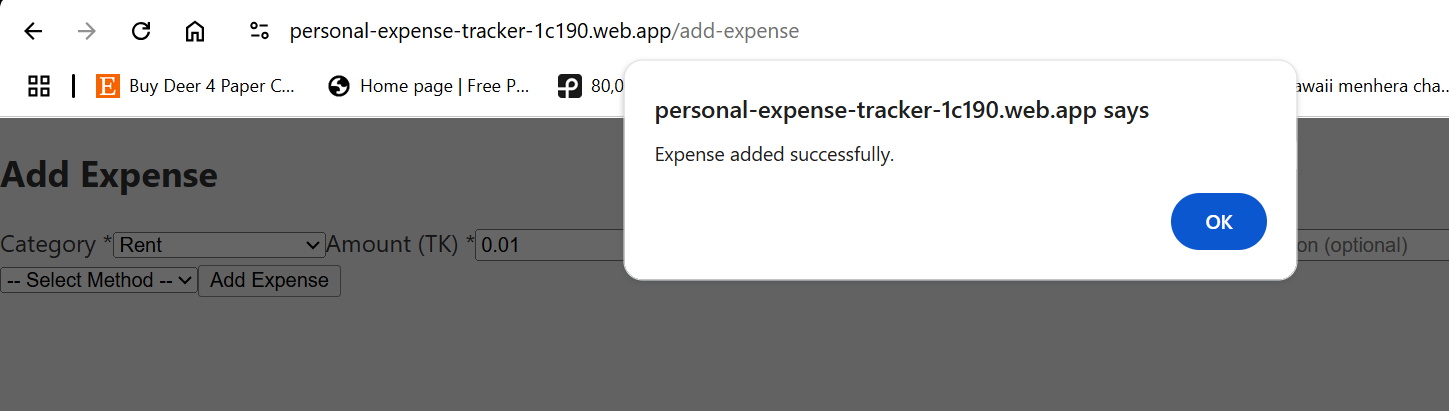
****

Fig 8: BTC1

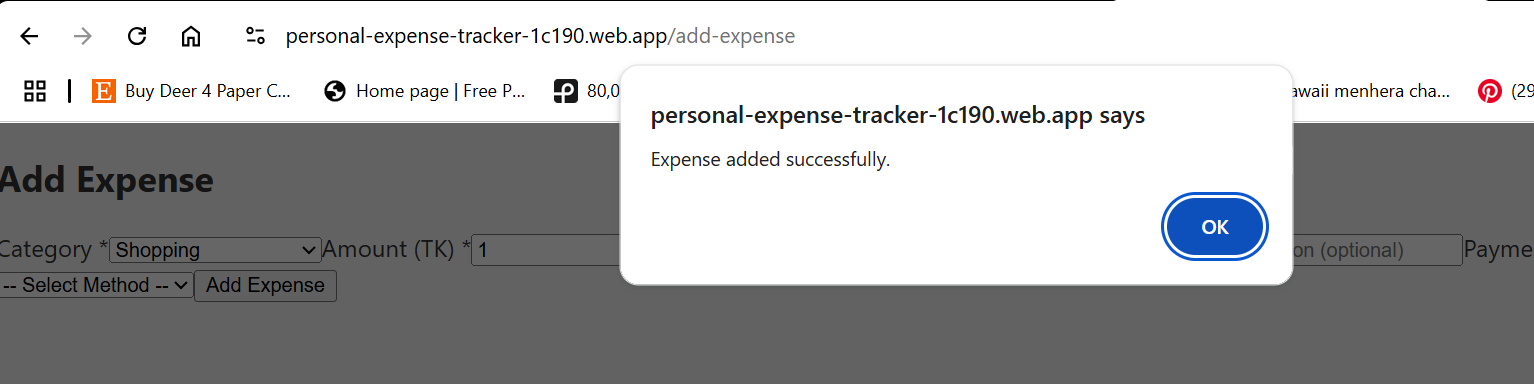
****

Fig 9: BTC2

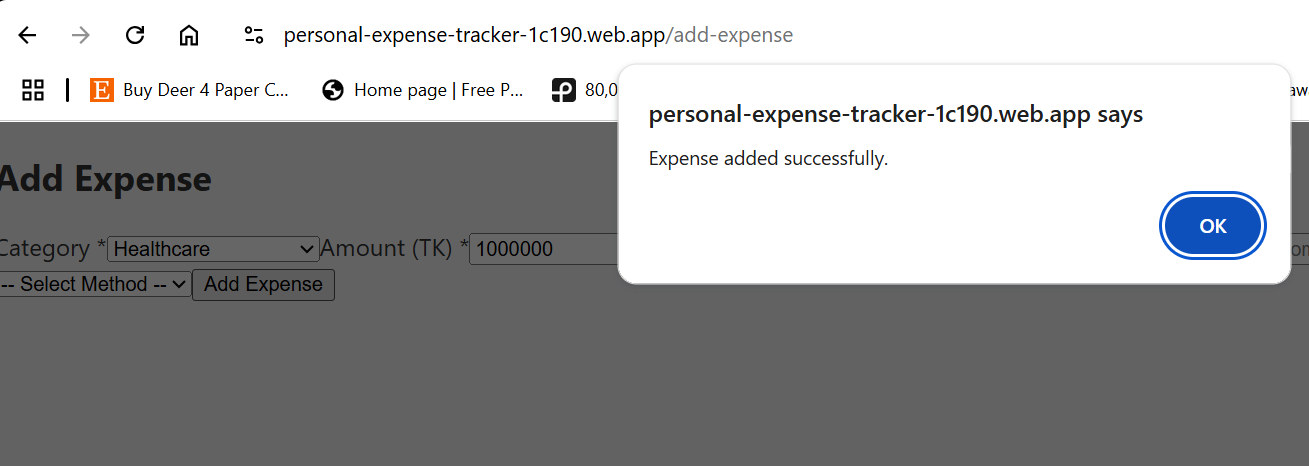
****

Fig 10: BTC3

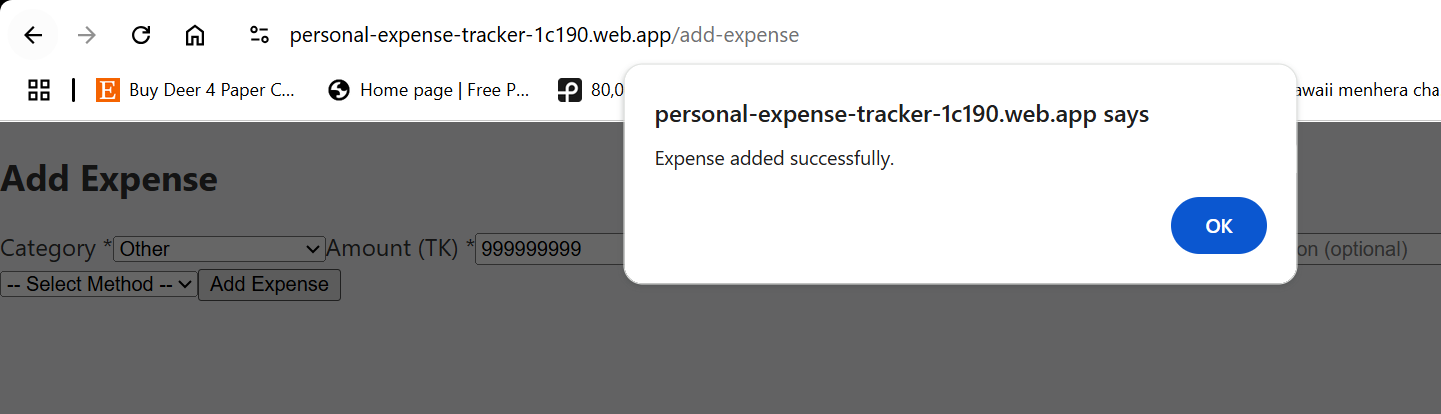
****

Fig 11: BTC4

**Yearly Budget: User Input Field for BVA: Amount**

Total monthly budget is set 55000 TK

* Yearly Budget must be a valid positive amount
* Yearly Budget cannot exceed Total monthly

| Test Case ID | Component | Class | Input | Expected Output' | Actual Output | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- |
| BTC5 | BudgetManager | Budget | 55000 | Budget added successfully to the list, no warning | Budget added successfully to the list, no warning | Pass |
| BTC6 | BudgetManager | Budget | 55001 | Budget added successfully to the list, no warning | Budget added successfully to the list, no warning | Pass |
| BTC7 | BudgetManager | Budget | 9999999 | Budget added successfully to the list, no warning | Budget added successfully to the list, no warning | Pass |

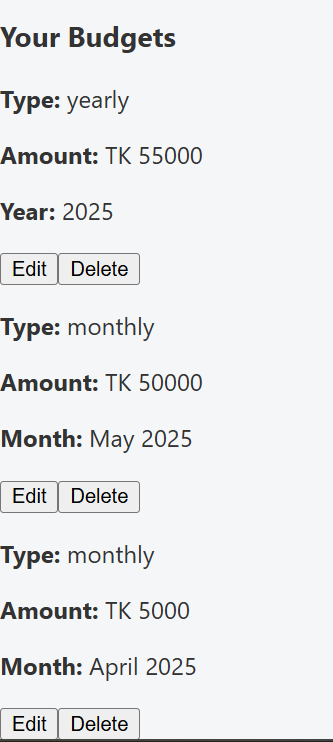
****

Fig 12: BTC5

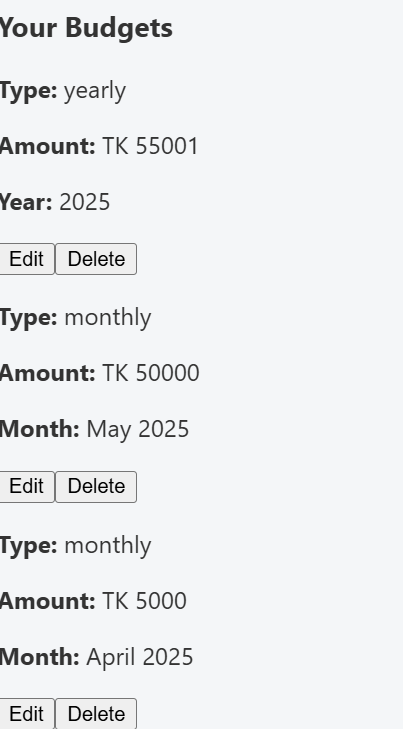


Fig 13: BTC5

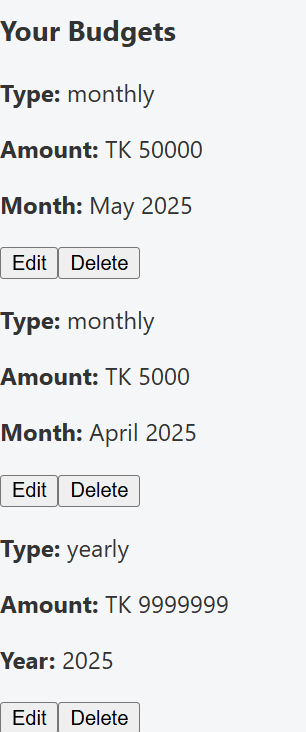


Fig 14: BTC7

**4. Bug Detection and Solution**

No bugs were detected while testing the above cases.