University of Westminster

# Software Development – 4COSC006C

## COURSE WORK 03

## TEST PLAN

| | |
|---|---|
| Name | M.G.G.W.Tharaki Dimithri |
| Module | 4COSC006C - Programming |
| Type of Assignment | Individual Coursework 03 Specification |
| UOW ID | W2081980 |
| IIT ID | 20232778 |

## I. Abstract

The purpose of this code is to improve the Personal Finance Tracker usefulness by creating an intricate graphical user interface (GUI) using Python Tkinter. To improve modularity and scalability, I aim to develop GUI components as objects by embracing the concepts of object-oriented programming (OOP). Financial transactions will be loaded and displayed by the application with seamless integration from JSON files. I created a comprehensive search function that will enable users to discover transactions quickly based on multiple parameters, so empowering them to manage their finances. In addition, my program will have a sorting function similar to a file explorer, which will let users effectively organize and evaluate financial data. My goal is to provide a user-friendly and effective tool for managing personal finances by integrating these components.

## II. Acknowledgement

I want to extend my heartfelt appreciation to Mr. Guhanathan Poravi, our lecturer and module leader, and Mr. Lakshan Costa, our tutorial lecturer whose unwavering support and guidance helped me complete this assignment with excellence. I am also grateful for the prompt response to my queries and his guidance and support was invaluable and greatly appreciated. In addition, I would like to express my gratitude to my colleagues for their constant encouragement and support in completing this assignment.

## Contents

## List of Figures

## Table of contents

# Coursework 03 Specification: Enhanced Personal Finance Tracker (GUI Implementation with Tkinter and OOP)

## Overview:

Building on your knowledge of Python, dictionaries, and file I/O, your next challenge is to enhance the Personal Finance Tracker by developing a graphical user interface (GUI) using Tkinter. This advanced version should not only display the information from a provided JSON file but also incorporate object-oriented programming (OOP) concepts for the GUI components. Additionally, your application will include a search function and a sorting feature, similar to a file explorer, to manage and analyze financial transactions more effectively.

## Objectives:

1. Integrate a GUI using Tkinter and OOP concepts.
2. Load and display data from a JSON file upon GUI invocation.
3. Implement search and sorting functionalities within the GUI.

# 4. Design

## 4.1 Data Structure

**Search Functionality**: Allow users to search for transactions based on attributes such as date, amount, or type of expense. Implement filtering to display only the transactions that match the search criteria.

**Sorting Feature**: Implement a feature where clicking on a column heading in the transaction display table sorts the data based on that column. The sorting should toggle between ascending and descending order.

**JSON Integration**: Use the JSON file format for loading and saving transactions. Ensure your application correctly parses the JSON structure as specified in the original assignment.

## Functions

- **Create widgets:** This method organizes the GUI elements into sections for adding transactions, searching transactions, and displaying transactions. It ensures that the user interface is functional for managing personal finance data.
- **Load Transactions:** This function ensures that transaction data is loaded from a JSON file if it exists, and returns an empty dictionary otherwise, providing a clean way to handle file loading errors.
- **Save Transactions:** This function writes the transactions stored in the self.transactions dictionary to the specified JSON file. It overwrites the content of the file if it already exists.

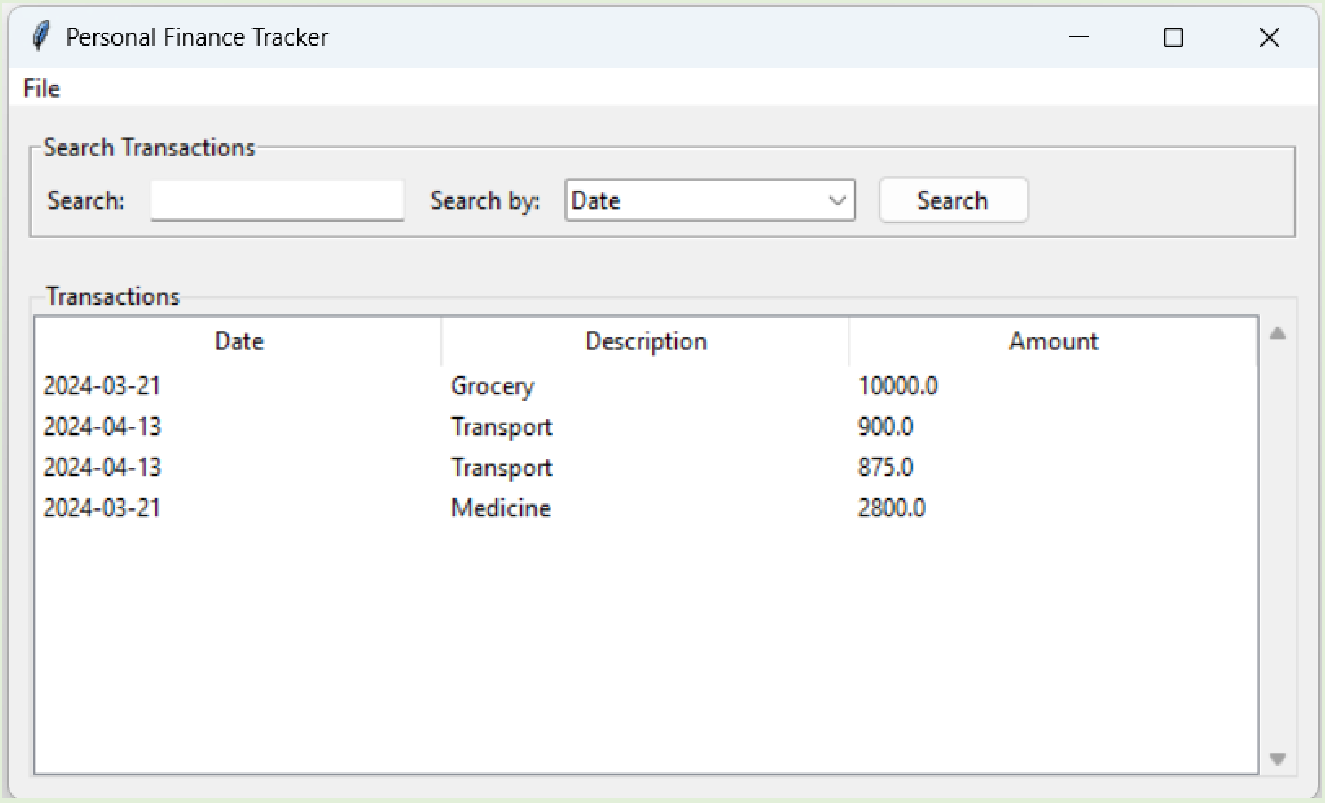
- **Display Transactions:** This function updates the display of transactions in the GUI by removing existing entries from the Treeview and adding new entries based on the provided transaction data.
- **Search Transactions:** Enables users to search for transactions based on a query and a selected attribute, and then updates the display to show the search results.
- **Sort Column:** This function sorts the items in a Treeview widget based on the values of a specified column and toggles the sorting order each time it's called.
- **Launch CLI Menu:** This function for saves any transactions made in the GUI application, closes the GUI window, and returns control to a main menu in a command-line interface. It's essentially transitioning from a graphical interface to a command-line interface.

## 5. Test Plan

### Test Cases for Main Program.

| Test No. | Test Case | Input | Expected Output | Actual Output | Pass or fail |
|---|---|---|---|---|---|
| 1 | Run in IDLE | Check the program run in IDLE correctly | Must display the Personal Finance Tracker GUI | Displayed The program and ran correctly. | Pass |
| 2 | Exit the program | Close window button clicked | Must Exit the program | Exited the program correctly. | Pass |

*Table 1 – Test Cases for the main program*

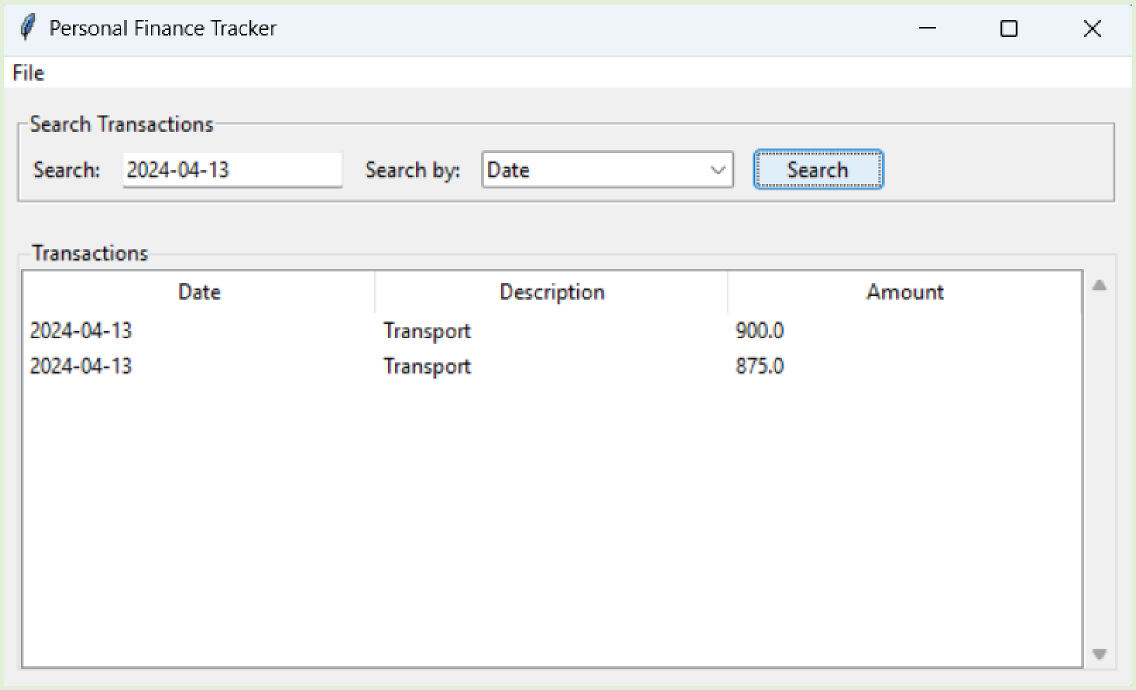


| Date | Description | Amount |
|---|---|---|
| 2024-03-21 | Grocery | 10000.0 |
| 2024-04-13 | Transport | 900.0 |
| 2024-04-13 | Transport | 875.0 |
| 2024-03-21 | Medicine | 2800.0 |

*Figure 1- Test Case 1 - Run In IDLE*

## Test Case for Searching Transactions:

| Test No. | Test Case | Input | Expected Output | Actual Output | Pass or fail |
|---|---|---|---|---|---|
| 3 | Searching Transactions by Description | Enter a search query in the search field and select search criteria from the dropdown and select “Type”. Then, click the "Search" button. | Transactions matching the search criteria should be displayed in the Treeview widget. | Displayed matching transactions related to Type | Pass |
| 4 | Searching Transactions by Date | Enter a search query in the search field and select search criteria from the dropdown and select “Date”. Then, click the "Search" button. | Transactions matching the search criteria should be displayed in the Treeview widget. | Displayed matching transactions related to the date | Pass |
| 5 | Searching Transactions by Amount | Enter a search query in the search field and select search criteria from the dropdown and select “Amount”. Then, click the "Search" button | Transactions matching the search criteria should be displayed in the Treeview widget. | Displayed matching transactions related to the amount | Pass |



Search: 2024-04-13 Search by: Date

Transactions

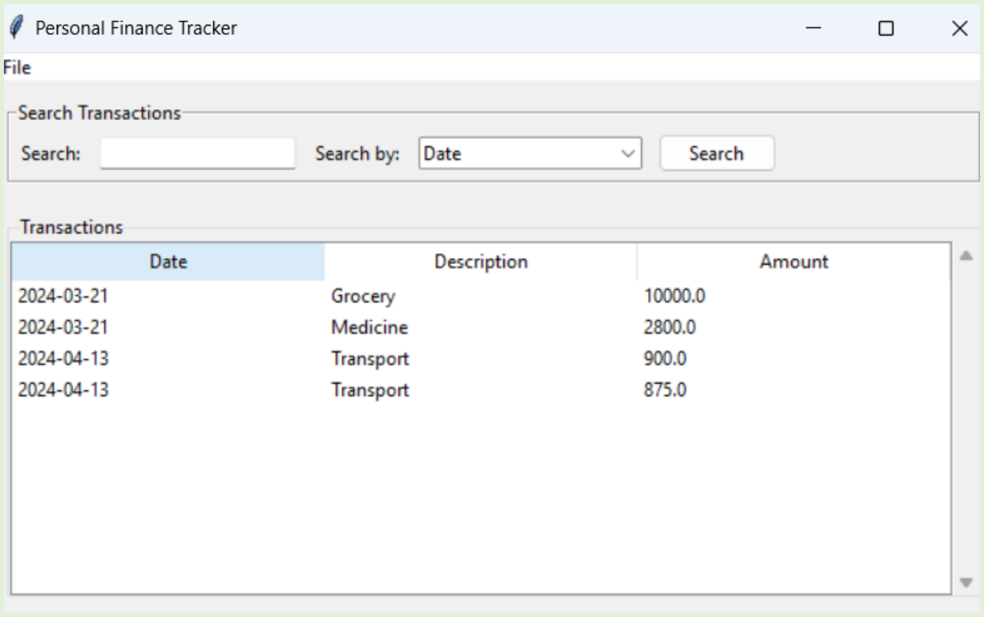
| Date | Description | Amount |
|---|---|---|
| 2024-04-13 | Transport | 900.0 |
| 2024-04-13 | Transport | 875.0 |

*Figure 2- Test Case 5 – Successful Search Transaction*

## Test Cases for Sorting Transactions:

| **Test No.** | **Test Case** | **Input** | **Expected Output** | **Actual Output** | **Pass or fail** |
|---|---|---|---|---|---|
| 6 | Sorting Transactions by Date | Click on the "Date" column header in the treeview. | Transactions should be sorted in ascending order based on the date. | Transactions are successfully sorted in ascending order based on the date. | Pass |
| 7 | Sorting Transactions by Description | Click on the headers of the “Description” in the treeview | Transactions should be sorted in ascending order based on the description. | Transactions are sorted in successfully ascending order based on the description. | Pass |
| 8 | Sorting Transactions by Amount | Click on the "Amount" column header in the treeview. | Transactions should be sorted in ascending order based on the amount. | Transactions are successfully sorted in ascending order based on the amount. | Pass |

*Table 4 – Test Cases for Sorting Transactions*



Personal Finance Tracker

File

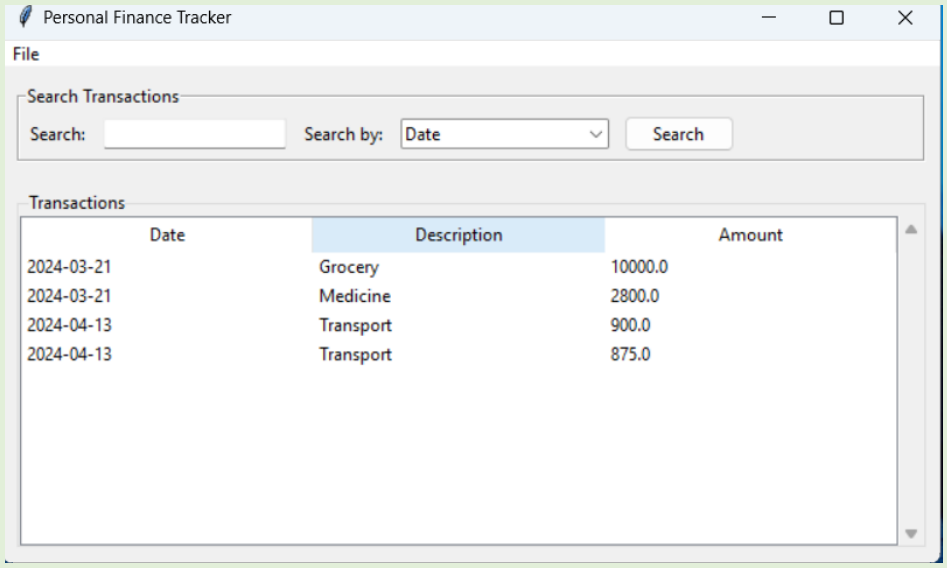
Search Transactions

Search: Search by: Date Search

Transactions

| Date | Description | Amount |
|---|---|---|
| 2024-03-21 | Grocery | 10000.0 |
| 2024-03-21 | Medicine | 2800.0 |
| 2024-04-13 | Transport | 900.0 |
| 2024-04-13 | Transport | 875.0 |

*Figure 3- Test Case 6 – Successful Sorting Transaction based on  date*



Personal Finance Tracker

File

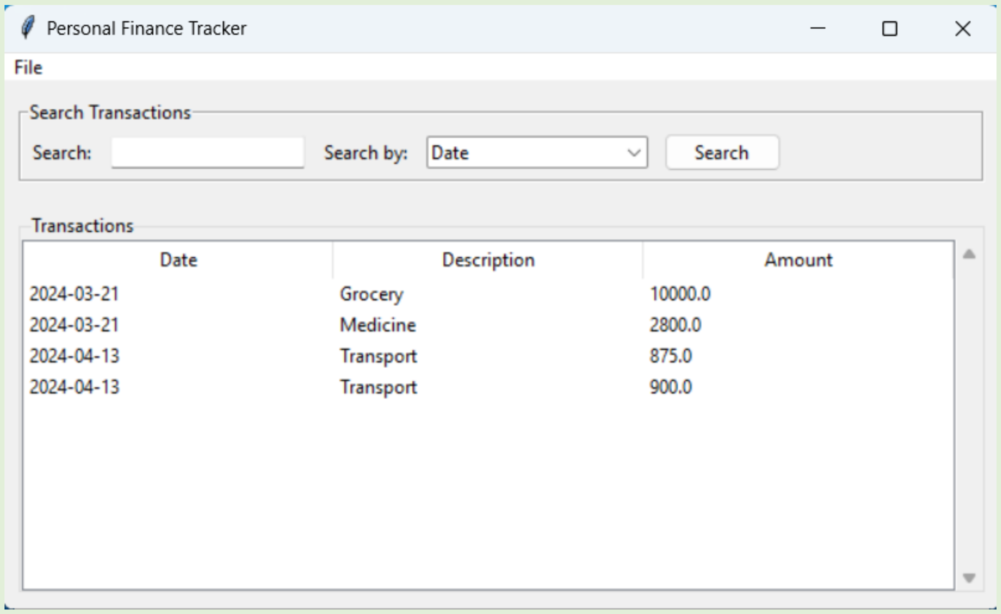
Search Transactions

Search: Search by: Date Search

Transactions

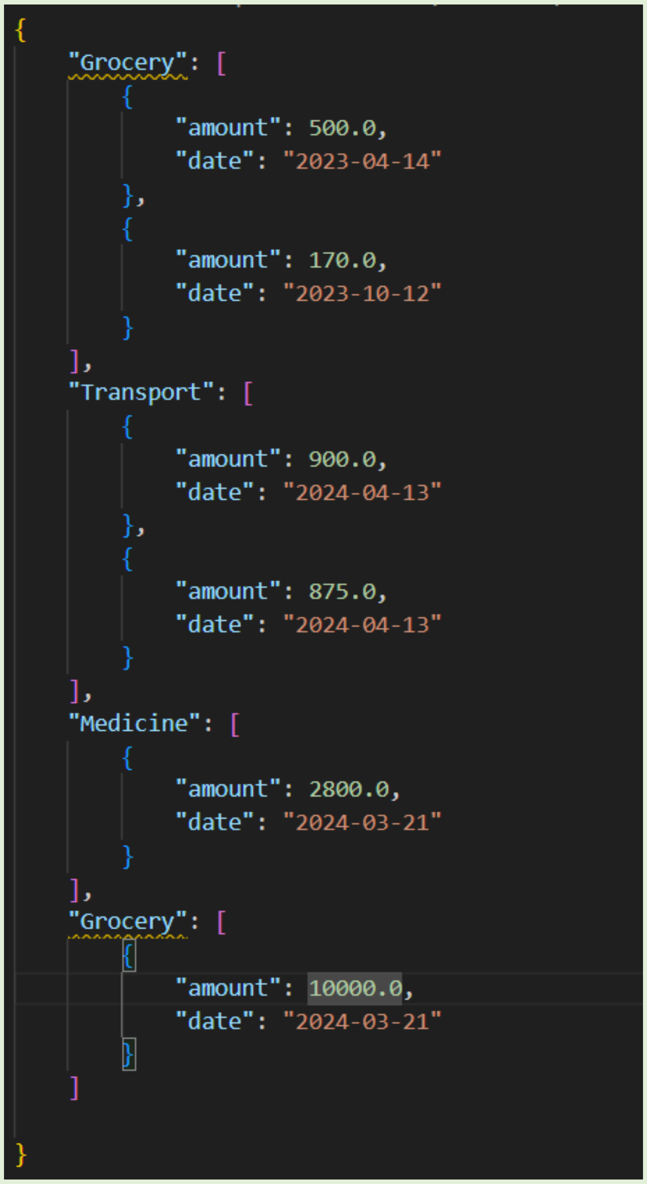
| Date | Description | Amount |
|---|---|---|
| 2024-03-21 | Grocery | 10000.0 |
| 2024-03-21 | Medicine | 2800.0 |
| 2024-04-13 | Transport | 900.0 |
| 2024-04-13 | Transport | 875.0 |

*Figure 4- Test Case 7 – Successful Sorting Transaction based on Description.*



| Date | Description | Amount |
|---|---|---|
| 2024-03-21 | Grocery | 10000.0 |
| 2024-03-21 | Medicine | 2800.0 |
| 2024-04-13 | Transport | 875.0 |
| 2024-04-13 | Transport | 900.0 |

*Figure 5- Test Case 8 – Successful Sorting Transaction based on Amount.*



```
{
    "Grocery": [
        {
            "amount": 500.0,
            "date": "2023-04-14"
        },
        {
            "amount": 170.0,
            "date": "2023-10-12"
        }
    ],
    "Transport": [
        {
            "amount": 900.0,
            "date": "2024-04-13"
        },
        {
            "amount": 875.0,
            "date": "2024-04-13"
        }
    ],
    "Medicine": [
        {
            "amount": 2800.0,
            "date": "2024-03-21"
        }
    ],
    "Grocery": [
        {
            "amount": 10000.0,
            "date": "2024-03-21"
        }
    ]

}
```

*Figure 6 - Test Case 9 - Successful creation of JSON file & save correct transaction data.*

## Test Summary

## Test Objectives

• Ensure all features function correctly according to requirements.

• Validate accurate recording, searching, and sorting of transactions.

• Confirm successful saving and loading of transaction data.

## Test Results

1. **Adding Transactions**: Ensure that transactions can be added with the correct details.
2. **Searching Transactions:** Validate the functionality of searching transactions by date, description, and amount.
3. **Sorting Transactions**: Confirm the correct sorting of transactions by date, description, and amount.
4. **Loading Transactions from File**: Verify the loading of transactions from a file and their accurate display.
5. **Handling Nonexistent File:** Ensure graceful handling of scenarios where the file does not exist.

## Conclusion

The personal finance tracker developed using Tkinter offers a robust solution for individuals seeking to manage their finances effectively. With its user-friendly interface, comprehensive features, and seamless transition between GUI and CLI, this application provides users with a convenient tool for tracking, analyzing, and organizing their financial transactions.

**************************************