

**TEST PLAN SUMMARY**

STUDENT NAME : S.GOKULA NANDHAN

STUDENT ID : 20233027

UOW ID : 20820910

EMAIL : gokula.20233027@iit.ac.lk

**COURSEWORK DETAILS**

Course : B.Eng(Hons) Software Engineering

Level : 04

Module : 4COS006C0.2 Software Development

Coursework No : 03

Coursework type : Individual

Issued date : 04th April 2024

Submission due date : on or before 29th of April 2024

Module Leader : Mr.Guhanathan Poravi

**CONTENTS**

1. Abstract

2. Acknowledgement

3. Specification of the task

-Task overview

-Task executing steps

4. Detailed steps of tasks

-Analyse the task

-Design the programme

-Implimentation and documentation

-Test

**ABSTRACT**

The module leader,Mr.Guhanadhan Poravi has tasked to enhance the Personal Finance Tracker by developing a graphical user interface (GUI) using Tkinter utilizing the software development module.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.This report serves as comprehensive documentation for the project. In the documentation, the problem statement is reviewed and a feasible solution is offered to solve it. To ensure the accuracy and resilience of the GUI application, internal testing procedures have also been conducted, and the report includes comprehensive test cases and validations.

**ACKNOWLEDGEMENT**

All of the people that helped create this test report for the Personal Financial Tracker GUI project have my sincere gratitude.I want to thank Sir Mr. Guhanathan Poravi, the module leader, and his team sincerely for giving me this wonderful opportunity to work on this assignment. They enabled the assignment to be completed successfully by offering direction, motivation, and unwavering support throughout.   
Additionally, I want to thank my peers and colleagues for their active participation in the review process and for their insightful comments and constructive criticism. Their insightful comments and ideas improved the report's quality and accuracy.   
I have made every effort to convey my appreciation to each and every references and contributors.

Thankyou

**SPECIFICATION OF THE TASK**

**TASK OVERVIEW**

The task 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,The application will include a search function and a sorting feature to manage and analyze financial transactions more effectively.

**TASK EXECUTING STEPS**

1.Analyse the task

2.Design the programme

3.Implimentation and documentation

4.Test

**DETAILED STEPS OF THE TASK**

**1.ANALYSE THE TASK**

Key objectives of the task

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. Ensure the application is user-friendly and robust.

**2.DESIGN THE PROGRAMME**

After analyzing above key objectives I have designed a documentation plan with pseudocode.

Pseudocode as follows

FinanceTrackerGUI CLASS :

Function \_\_init\_\_

Root folder

Title “Personal Finance Tracker”

Call Function create\_widgets

Load json file

Button for sort column

Function create\_widgets

#frame for table and scrollbar

Widget frame fill

#treeview

Widget headings of treeview

Write text topic for every heading

Insert widget headings data

#scroll bar

Scroll bar size,orientation

Insert scroll bar

#search bar and button

Search bar orientation,place

Insert search bar

Search button button text,place

Insert search button

Function load\_transactions

Try

Open file in read mode

transactions=load the file

return transactions

except

return { }

Function display\_transactions

For entry in self.treeview.get\_children():

Delete(entry)

for transaction\_type,transactions\_data in self.transactions.items();

for data in transactions\_data:

insert(values=(transaction\_type,["Date"],["Amount"])

Function search\_transactions

INITIALIZE query=self.search\_entry.get()

processed\_transactions = {}

for transaction\_type,transactions\_data in self.transactions.items():

processed\_data=[data for data in transactions\_data if

query in data["Amount"] or

query in transaction\_type or

query in data["Date"]]

IF processed\_data THEN

processed\_transactions[transaction\_type]=processed\_data

display\_transactions(processed\_transactions)

i=1

for transaction\_type,transactions\_data in processed\_transactions.items():

print(f"{i}.Expense\_type :{transaction\_type}")

j=1

for details in transactions\_data:

print(f"{j}.Amount :{data['Amount']})\n Date :{data['Date']}")

j+=1

i+=1

Function column\_click()

col\_id=treeview.identify\_column(event.x)

IF col\_id THEN

col=col\_id.split("#")[-1]

self.sort\_by\_column(col)

Function sort\_by\_column

tems=treeview.get\_children("")

data=[(self.treeview.set(child,col),child)for child in items]

from data sort

for index,(val,child) in enumerate(data):

from treeview move(child,"",index)

Function main

root = tk.Tk()

app = FinanceTrackerGUI(root)

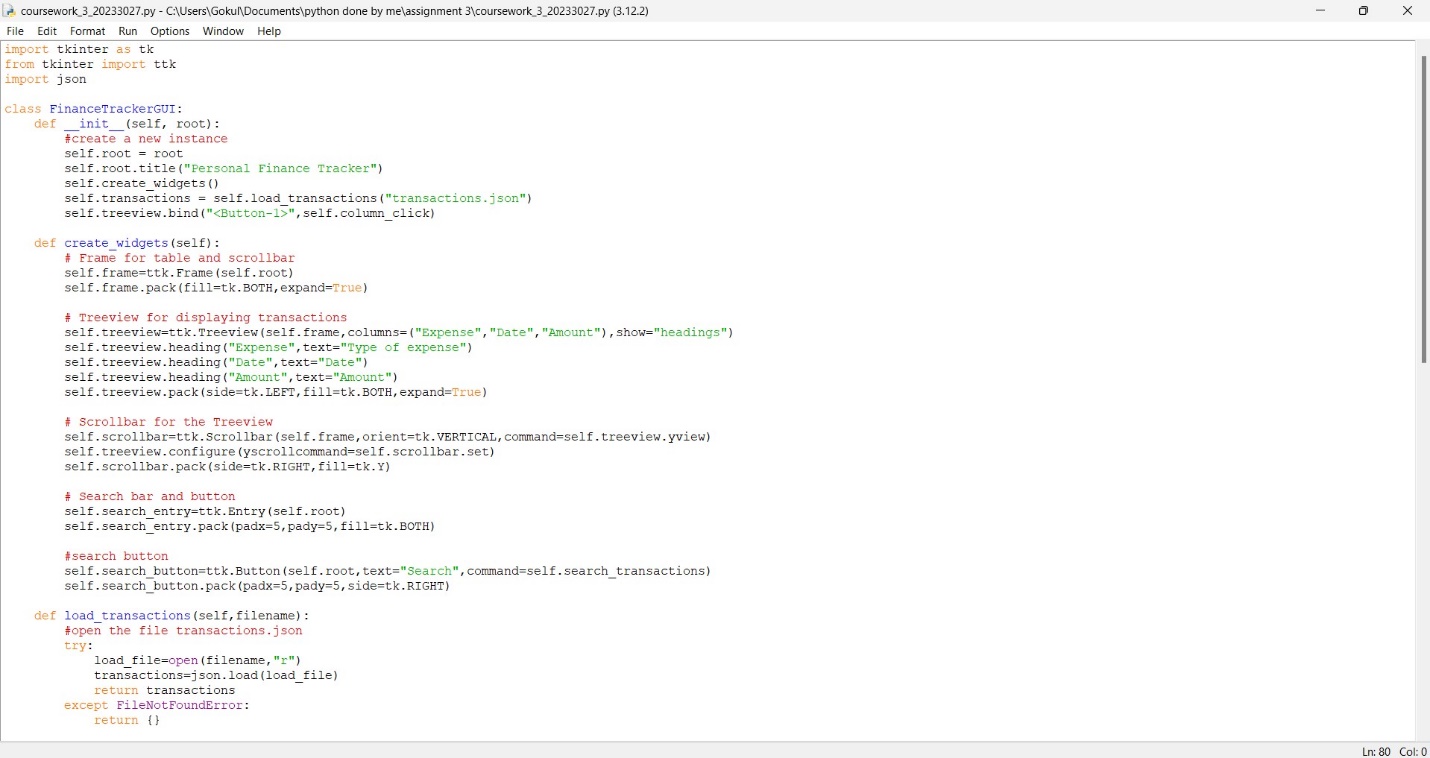
app.display\_transactions(app.transactions)

root.mainloop()

**3.Implimentation and documentation**

Screen shots of implemented code

**PAGE 01**

****

**PAGE 02**

**A screenshot of a computer

Description automatically generated**

**PAGE 03**

**A screenshot of a computer

Description automatically generated**

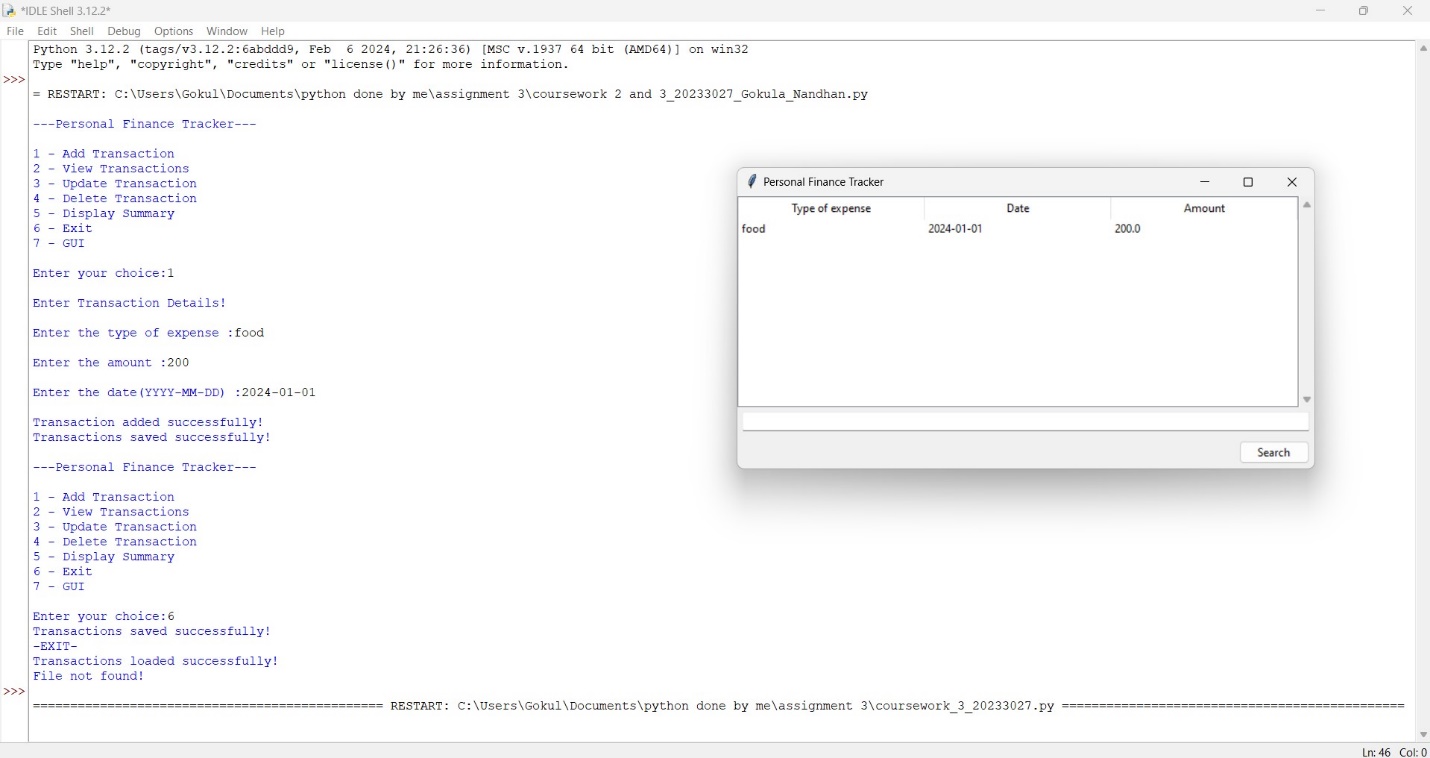
**4.Test**

THE FOLLOWING CHARTS SHOWS THE ENTIRE TEST PHASE OF THE EXECUTED CODE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test case** | **Discription** | **Inputs** | **Expected**  **Output** | **Actual**  **Output** | **Remarks** |
| 01 | Open the file in GUI | Input transactions in coursework 2 python code | If run GUI python code Display in GUI | Did expected output | Test pass |
| 02 | Add more transactions | Input transactions in coursework 2 python code | If run GUI python code Display in GUI | Did expected output | Test pass |
| 03 | Check the scroll bar in GUI | Inpu more transactions to the coursework 2 python code | Newly created a scroll bar | Did expected output | Test pass |
| 04 | Sort function | Click the type of expense and date in GUI | Sorted by date | Did expected output | Test pass |
| 05 | Search a type of expense | Type in searchbar  And search | Display searched row | Did not expected  output | Test Fail |
| 06 | Open GUI in coursework 2 | Click the coursework file with python launcher and choice=7 | Open GUI | Did expected output | Test pass |

**SCREEN SHOT OF THE TEST CASES**

TEST 01



TEST 02A screenshot of a computer

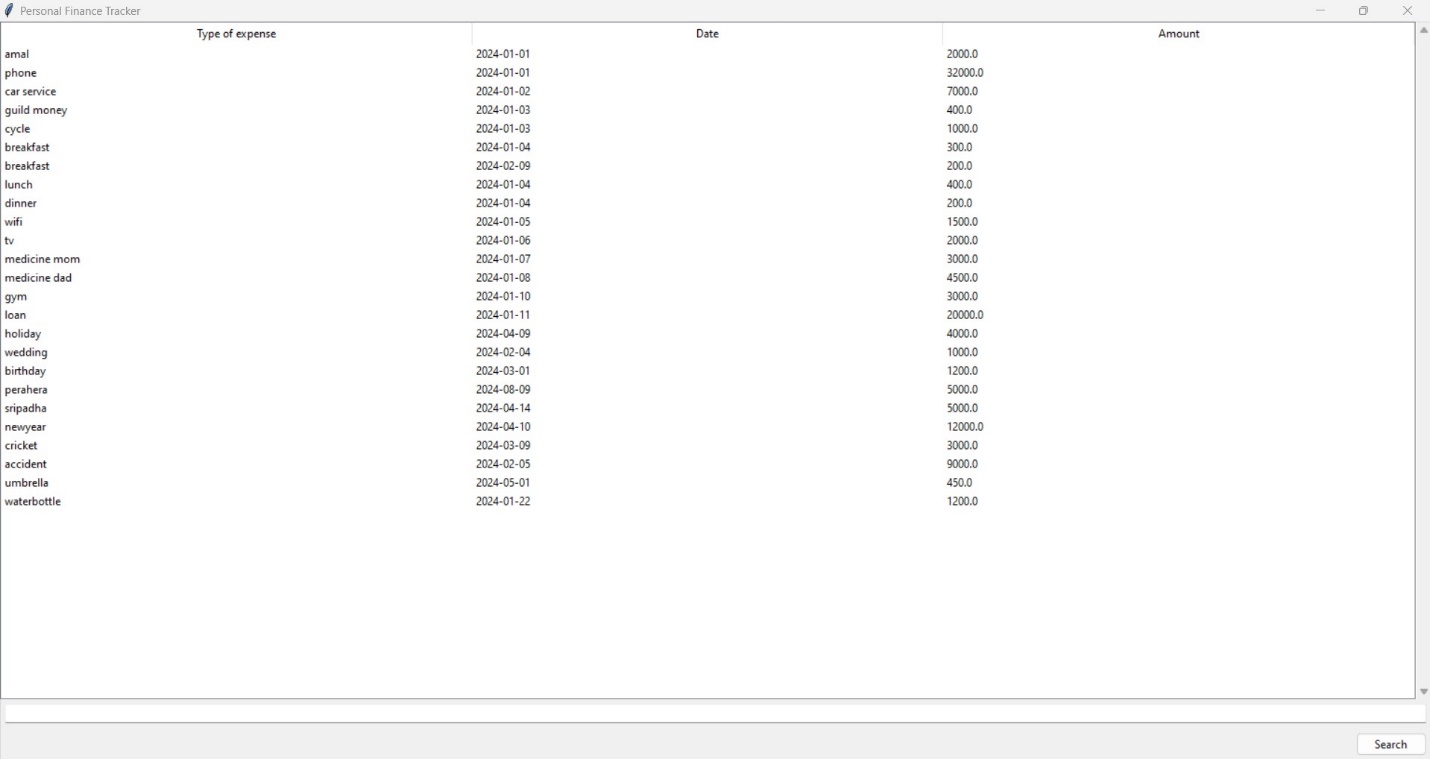
Description automatically generated

A computer screen with a white screen

Description automatically generated

TEST 03

**Overview**

****

**After creation of scroll bar**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

TEST 04

**BEFORE SORT**

**A screenshot of a computer

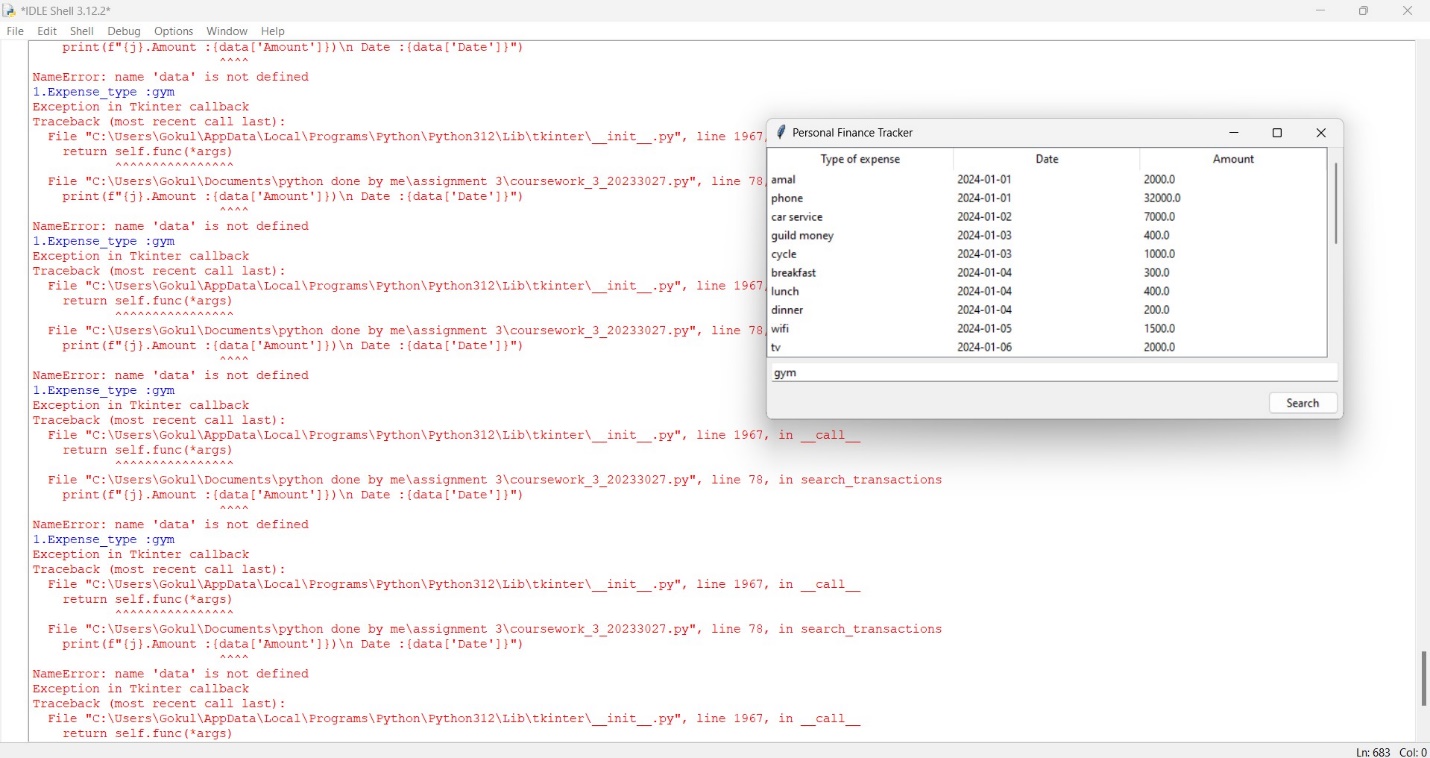
Description automatically generated**

**AFTER SORT**

**A screenshot of a computer

Description automatically generated**

Test 05



TEST 06A screenshot of a computer

Description automatically generated