INFORMATICS INSTITUTE OF TECHNOLOGY

UNIVERSITY OF WESTMINSTER

Module: Software development

Module Code: 4COSC006C

Module leader: Mr. Poravi Guganathan

Assessment Type: Individual

Assessment Number: 2

K.K Poorna Madusith

20222166

W2083117

Table of contents

Table of Contents

Table of contents....2
Table of figures....3
About program....4
Code....6
Design decisions and functionality....18
Design decisions....18
Class structure....18
Functionality....18
Test cases....19
Screen shots of the test cases....23
1.1....23
1.2....26
1.3....29
1.4....32
1.5....33
2....34
3.1....35
3.2....39
3.3....40
4....41
5.1....43
5.2....44
5.3....45
6....46
7.1....47
7.2....49
7.3....50
9....51
Summary of test cases....52

Table of figures

Figure 1 23
Figure 2 24
Figure 3 25
Figure 4 26
Figure 5 27
Figure 6 28
Figure 7 29
Figure 8 30
Figure 9 31
Figure 10 32
Figure 11 33
Figure 12 34
Figure 13 35
Figure 14 36
Figure 15 37
Figure 16 38
Figure 17 39
Figure 18 39
Figure 19 40
Figure 20 40
Figure 21 41
Figure 22 41
Figure 23 41
Figure 24 42
Figure 25 43
Figure 26 43
Figure 27 43
Figure 28 43
Figure 29 44
Figure 30 44
Figure 31 45
Figure 32 45
Figure 33 45
Figure 34 46
Figure 35 47
Figure 36 47
Figure 37 48
Figure 38 48
Figure 39 49
Figure 40 49
Figure 41 49
Figure 42 50
Figure 43 50
Figure 44 51

About program

This is a simple personal finical tracker made by using python. It has a simple GUI which made by python tkinter. at the beginning in main screen program shows these buttons to user.

1.add transaction

2.veiw transactions

3.update transaction

4.delete transaction

5.search transactions

5.display summary

6.read transactions from a file

7.exit

8.about

If user click on the add transaction button dialog box will appear and ask the category of transaction then another dialog box come and ask the type of transaction. type of the transaction should be income or expense. Then another dialog box will appear and ask the amount of the transaction. final dialog box will ask the date of the transaction. if user enters the transactions correctly program will shows a message by a message box. Otherwise it will shows the error messages .

If user select the view transactions button program will shows all the transactions user entered by using another interface with a tree veiw

If user select the update transaction button first program will ask the category of transaction user want to update by using a dialog box. If there is a category as user mentioned program show the transaction history of that category with a index before each transaction. then program ask from

user what is the index of transaction user want update. After user entering the index program allow user to enter the new transaction details(type, amount, date). Then program replace old details with new details

If user select delete transaction like the previous update transaction program will ask category and the index. If there any index as user mentioned that transaction will deleted.

if user select the search transactions button first program will ask the criteria of the transaction. it should be category, type, amount or date . other wise program will display an error message. Then program will ask the value from user. Then program will display the all matched transactions by using a message box.

If select display summary button program will display total amount of each category, total income, total expense and net balance as a summary of all transactions.

When user select the button read transactions from a file user can create a text file and store transaction details inside the text file according to this format category, type, amount, date. Then program ask from the user name of the text file user created . the program read the text file and store the data inside the dictionary and Json file.

If user select the exit button all the changes will be saved in the JSON file and program will be close

Bu clicking the about button user can know what each button does and how use the program.

Code

```
import json #import json for save trasnaction in json formt

import tkinter as tk#import tkinter to create gui for the program

from tkinter import messagebox, simpledialog#from tkinter import messagebox and simpledialog
to handle gui operations

from tkinter import ttk


class FinancialTracker:

    def __init__(self):

        #initialize transactions dictionary

        self.transactions = {}

        #load existing transactions from the JSONfile

        self.load_transactions()


    def load_transactions(self):#create laod trasnactions function to load trasnactions from the json
file

        try:

            with open("transactions.json","r") as file:#open transactions file in r mode

                self.transactions = json.load(file)

        except FileNotFoundError:

            self.transactions = {}


    def save_trasactions(self):#create save transactions for the save transaction inside the json file

        with open("transactions.json","w") as file:#open trasnactions.json file in write mode and
save the transaction details inside the json file

            json.dump(self.transactions,file, indent=4)


    def transactiontype_input(self):#create a function to handle the transaction type  inputs
```

```
        while True:
            user_input = simpledialog.askstring("input","Enter the type of transaction")#get user input usong tkinter simpldialog
            if user_input is None:#chiek user input is none
                return
            else:
                user_input = user_input.lower()#if not make user input to lowercase
            if user_input in ["income","expense"]:#check whether user input in mentioned list
                return user_input
            else:#if not show error message usiing tkinter messagebox
                messagebox.showerror("Error","Please enter income or expense")


    def instructions(self):#cretae a fucntion to show intrctions to the user to use the program
        messagebox.showinfo("Instructions","1. To add Transactions click add button and enter the transaction detais\n2. to veiw your transaction history click veiw transactions button\n3. to update transactiosnc lick update transaction button and enter category and index of the trasnaction you want to update and then enter the new transaction details\n4. to delete a trasnaction enter the category and index number of the transaction then ur transaction will be deleted\n5. to search trasnaction based on criteria enter the criteria and then program will shw=ow the all details\n6. to see summary of your all trabsactions select the summary button\n7. if you want to add more than one transaction at the same time select the read transactions from a file button. first you have to create a text file and enter the trasnaction details in this format one under other category,type,amount,date. then enter the file name with txt extenstion.then all details will be added and saved\n7. if you click select exit button your all transactions will be saved and you can exit from the financial tracker")



    def search_transactions(self):#creta a function to search transactions
        search_criteria = simpledialog.askstring("input","Enter the search ctiteria (category,type,amount,date): ")#ask the criteria using tkinter simpledialog
        #check search criteria is none or nor if none return and if not make search criteira input to lowercase
```

```
        if search_criteria is None:
            return
        else:
            search_criteria = search_criteria.lower()



        if search_criteria in ["category","type","amount","date"]:#check search criteria inside the
mentioned list
            search_value = simpledialog.askstring("input","Enter the value you want to search:
")#ask the search value using tkinter simpledialog
            #check search value is none or nor if none return and if not make search value input to
lowercase
            if search_value is None:
                return
            else:
                search_value = search_value.lower()
            output = "search results\n"#make a varialble name output and assingned it into search
results string
            for category,trans in self.transactions.items():#go theourh wach category and trasactions
                for t in trans:
                    if search_criteria == "category" and category == search_value:#check the search
criteria is category and matches the search value
                        output += f"Category: {category}, Type: {t["type"]}, Amount:{t["amount"]},
Date: {t["date"]}\n"
                    elif str(t.get(search_criteria)) == search_value:#check if search criteria matches any
transaction attribute and matches the search value
                        output += f"Category: {category}, Type: {t["type"]}, Amount:{t["amount"]},
Date: {t["date"]}\n"


            if output == "search results\n":#check if any results are found
```

```
            output = "No transactions found maching in the search criteria"
        messagebox.showinfo("search results",output)#display the search results using tkinter
messagebox
    else:#if search criteria not in metioned list display error message using message boc
        messagebox.showerror("Error","There is no such a criteria.only exist
category,type,amount and date")




  def read_bulk_transactions_fromfile(self):#cretae a function read transactions from a text file
    filename = simpledialog.askstring("input","Enter file name (e.g:- filename.txt): ")#ask the
file name from user using simple dialog
    if filename is None:
        return
    try:
        with open(filename,"r") as file:#open the user mentioned text file in read mode and read
it
            for line in file:
                category,type,amount,date = line.strip().split(',')
                amount = float(amount)
                if category in self.transactions:#if category already exist add deatils in to that
category
                    self.transactions[category].append({"type": type, "amount": amount, "date":
date})
                else:#if not create a new category and add the details
                    self.transactions[category] = [{"type": type, "amount": amount, "date": date}]
            messagebox.showinfo("succsess","succsessfully read from the file")
    except FileNotFoundError:#if filenotfounderror comes display a error message using
messagebox
```

```
        messagebox.showerror("error","there is no such a file")
    except ValueError:#if there any value error inside the text file display an error message
        messagebox.showerror("error","Error reading transactions form the file\nplease enter transactions inside the file this format\ncategory,type,amount,date")


  def add_transactions(self):#create a function to add a transaction
    category = simpledialog.askstring("input","Enter the category of transaction: ")#ask the category
    if category is None:
        return
    else:
        category = category.lower()


    type = self.transactiontype_input()#call transactiontype_function to get the trasnaction type input
    if type is None:
        return


    amount = simpledialog.askfloat("input","Enter the amount of your transaction")#ask amount from the user
    if amount is None:
        return


    date = simpledialog.askstring("input","Enter the date")#ask date from user
    if date is None:
        return
    #cretate a dictioanry to to store the transaction details
    trans = {"type": type, "amount": amount, "date": date}
```

```
        if category in self.transactions:#check category is already exsist
            self.transactions[category].append(trans)#if yes add details in to that category
        else:
            self.transactions[category] = [trans]#if not cretae a new  category and add the details

        messagebox.showinfo("Succsess","Transaction added succsessfully")#displaya message
using tkinter messagebox

    def veiw_transactions(self):#create a fucntion to veiw the transactions
        #create a new windows for display transactions
        window = tk.Tk()
        window.title("Transactions History")

        #create tree veiw widget
        tree = ttk.Treeview(window)
        tree["columns"] = ("Type", "Amount", "Date")

        #define column headings
        tree.heading("#0", text = "Category")
        tree.heading("Type", text = "Type")
        tree.heading("Amount", text = "Amount")
        tree.heading("Date", text = "Date")

        #populate tree veiw with data
        for category, tra in self.transactions.items():
            for trans in tra:
```

```
            tree.insert("",'end',text = category, values=(trans["type"], trans["amount"],
trans["date"]))


        #display tree veiw
        tree.pack(expand = True,fill="both")


    def update_transacions(self):#create a function to update a transaction
        category = simpledialog.askstring("input","Enter the category you want to update")#ask
category from user
        if category is None:#check category is none
            return#if yes return from the function
        else:
            category = category.lower()#if not make the category input to lower


        if category in self.transactions:#check category in trasnactions dictionary
            output = ""#assign output variable to emplty string
            for index,tra in enumerate(self.transactions[category]):#go throuth the trasnactions
dictionary and index each transaction using enumarate
                output += f"{index+1}. Type:{tra['type']}, Amount: {tra['amount']}, Date:
{tra['date']}\n"


            choice = simpledialog.askinteger("input",f"Please select the index of the transaction you
want to update: \n{output}")#ask the index of the trasnaction user want to update
            if choice is None:#if choice is none return from the fucntion
                return



            if choice and 0 < choice <= len(self.transactions[category]):#check the index is valid
                type = self.transactiontype_input()#callctype function to ask the typr of trasnaction
```

```
            if type is None:
                return

            amount = simpledialog.askfloat("input","Enter the amount of your transaction: ")#ask
amount from the user
            if amount is None:#if amount is none return from the function
                return

            date = simpledialog.askstring("input","Enter the date")#ask the date from user
            if date is None:#if date is none return from the function
                return

            self.transactions[category][choice-1] = {"type": type, "amount": amount, "date":
date}#update the trasnaction with new details
            messagebox.showinfo("Succsess","Trasnactions updated succsessfully")#display a
message to user using tkinter messagebox
        else:
            messagebox.showerror("Error","Invalid index")#if index is not valid displaya message
to user
    else:
        messagebox.showerror("Error",f"There is no such a category named{category}")#if
cateogry not in trasnaction display error message



  def delete_trasnaction(self):#crreate a function to delete a trasnaction
    category = simpledialog.askstring("input","Enter the category you want to delete: ")#ask the
category from user using tkinter simpledialog
    if category is None:#if category is none return from the function
        return
```

```
        else:#if not turn category input into lowercase
            category = category.lower()
        if category in self.transactions:#check category in transactions dictionary
            output = ""#if yes assign output variable into empty string
            for index,tra in enumerate(self.transactions[category]):#based on the category user enterd go through the dictionary and display all the transaction details with index before each transaction
                output += f"{index+1}. Type:{tra['type']}, Amount: {tra['amount']}, Date: {tra['date']}\n"

            choice = simpledialog.askinteger("input",f"Please select the index of the transaction you want to delete: \n{output}")#ask the index number user want to delete using tkinter simpledialog
            if choice is None:#if choice is none return from the function
                return

            if choice and 0 < choice <= len(self.transactions[category]):#check if index is valid
                del self.transactions[category][choice-1]#if yes delete the indexed transaction
                messagebox.showinfo("Success","Trasnaction deleted succsessfully")#displaya message using tkinter messagebox
            else:
                messagebox.showerror("Error","Invalid index")#if index is invalid display an error message
        else:
            messagebox.showerror("Error",f"There is no category named {category}")#if category not in transactions display error message using tkinter messagebox

    def display_summary(self):#create a function to display summary of transactions
        total_income = 0#assign total income to zero
        total_expense = 0#assign total expense to zero
```

```
        output = "Summary of trasnactions\n"#assingn output in to summary of transactions string


        for category,trans in self.transactions.items():#go throuth the transactions dictionary
            total = sum(tra['amount'] for tra in trans)#get the sum of all transactions amounts
            total_income += sum(tra['amount'] for tra in trans if tra['type'] == "income")#get the total
amount of each transactions where type equals to income
            total_expense += sum(tra['amount'] for tra in trans if tra['type'] == "expense")#get the
total amount of each transactions where type equals to expense


            output += f"{category}: total amount{total}\n"#add cateogry and total amounts of each
catgory to output vairable


        output += f"\nTotal income: {total_income}\n"#add total income to output
        output += f"Total expense: {total_expense}\n"#add total expense to output
        output += f"Net balance: {total_income- total_expense}\n"#add net balance by
substractiong total expense from total income and add it to ouput


        messagebox.showinfo("Summary",output)#dispplay output using tkinter messagebox


def main_menu():#create a function for main menu
    tracker = FinancialTracker()


    #create the mainmenu window using tkinter
    window = tk.Tk()
    window.geometry("400x500")
    window.configure(bg="lightblue")
    window.title("Personal Financial Tracker")
```

```
    #create and pack the welcome label

    label1 = tk.Label(window,text="WELCOME!!!\nTO YOUR\nPERSONAL FINANCIAL
TRACKER", font=("Comic Sans Ms",16),bg="lightblue")

    label1.pack()


    #create and pack the buttons for varius functionaliteis

    add_button = tk.Button(window,text="Add
Transactions",command=tracker.add_transactions,fg="white",bg="black")

    add_button.pack(pady=10)


    veiw_button = tk.Button(window,text="Veiw
Transactions",command=tracker.veiw_transactions,fg="white",bg="black")

    veiw_button.pack(pady=10)


    update_button = tk.Button(window,text="Update
Transaction",command=tracker.update_transacions,fg="white",bg="black")

    update_button.pack(pady=10)


    delete_button = tk.Button(window,text="Delete
Transaction",command=tracker.delete_trasnaction,fg="white",bg="black")

    delete_button.pack(pady=10)


    search_button = tk.Button(window,text="Search
Transactions",command=tracker.search_transactions,fg="white",bg="black")

    search_button.pack(pady=10)


    summary_button = tk.Button(window,text="Display
Summary",command=tracker.display_summary,fg="white",bg="black")

    summary_button.pack(pady=10)
```

```
    read_button = tk.Button(window,text="Read Transactions From A
File",command=tracker.read_bulk_transactions_fromfile,fg="white",bg="black")

    read_button.pack(pady=10)


    exit_button =
tk.Button(window,text="Exit",command=lambda:[tracker.save_trasactions(),window.destroy()],f
g="white",bg="black")

    exit_button.pack(pady=10)


    about_button =
tk.Button(window,text="About",command=tracker.instructions,fg="white",bg="black")

    about_button.pack(side = "bottom",padx=10, pady=10, anchor="se")


    #run the main loop of the tikinter window

    window.mainloop()


main_menu()#call main_menu function to strat the program
```

Design decisions and functionality

Design decisions

GUI framework: the program uses “tkinter” for building the GUI , providing a native look and feel on most platforms

Data storage: Transactions are stored in a dictionary (“transactions”) in memory and saved/loaded from a JSON file for persistence

A “FinancilaTracker” class was designed to encapsulate all transactions-related functionalities.

Class structure

“FinancilaTracker” this class manes all the operations related to transactions like adding, viewing, updating and deleting transactions.it also handles file operations for saving and loading transactions.

Functionality

Loading: Transactions are loaded from a “transactions.json” file on startup. The the file doesn’t exist , an empty dictionary is used

Adding transactions: users can add transactions specifying category, type (income/expense), amount and date

Viewing transactions: users can view all transactions in a tree view

Updating transactions: users can update transactions by specifying category and transaction index

Deleting transaction: users can delete transactions by specifying category and transaction index.

Searching transactions: users can search transactions based on category, type, amount or date

Summary display: users can view a summary of all transactions, including total income , total expense and net balance

Test cases

Test plans and executions					
	Test no	**Input**	**Expected output**	**Actual output**	**Pass/ Fail**
Add transaction	1.1	Click add transaction Salary Income 1000000 2024-01-01	Message box with transaction added successfully. Save the transaction details inside the JSON file	Message box with transaction added successfully. Saved the transaction details inside the JSON file	Pass
	1.2	Click add transaction Grossary Expense 5000 2024-01-03	Message box with transaction added successfully. Save the transaction details inside the JSON file	Message box with transaction added successfully. Saved the transaction details inside the JSON file	Pass
	1.3	Click add transaction Grossary Expense 6000 2024-01-04	Message box with transaction added successfully. Add the transaction in previous grossary category and save inside the JSON file	Message box with transaction added successfully. Added the transaction in previous grossary category and saved inside the JSON file	Pass
	1.4	Click add transaction Fuel exp	Display an error message using tkinter message box.	Displayed an error message using tkinter message box.	Pass
	1.5	Click add transaction Fuel Expense thousand	Display error message because amount should be a floating value.	Displayed an error message string values are not allowed	Pass

View transactions	2	Click view transactions button	Display all transactions history using a another interface with a tree view	Displayed all transactions history using a another interface with a tree view	Pass
Update transaction	3.1	Click update transaction Grossary 1 7000 Expense 2024-01-03	Update the indexed transaction with new details and save it inside the JSON file	Updated the indexed transaction with new details and save it inside the JSON file	Pass
	3.2	Click update transaction Grossary 5	Display an error message invalid index by using message box	Displayed an error message invalid index by using message box	Pass
	3.3	Click update transaction button rent	Display error message there is no category named rent using message box	Displayed error message there is no category named rent using message box	Pass
Delete transaction	4	Click delete transaction Grossary 2	Delete the indexed transaction from the dictionary and save it inside the JSON file	Deleted the indexed transaction from the dictionary and save it inside the JSON file	Pass
	5.1	Click search transactions Date 2024-01-03	Display all the transactions did in mentioned date by using message box	Displayed all the transactions did in mentioned date by using message box	Pass

Search transactions	5.2	Click search transactions time	Display an error message there is no such a criteria	Displayed an error message there is no such a criteria	Pass
	5.3	Click search transactions Amount 9000000	Display an error message no transactions found	Displayed an error message no transactions found	Pass
Display summary	6	Click display summary button	Display summary of all transaction by message box	Displayed summary of all transaction by message box	Pass
Read transactions from a file	7.1	Click read transactions from a file Tc.txt	Read all transactions from the mentioned file and save it inside the JSON file	Read all transactions from the mentioned file and save it inside the JSON file	pass
	7.2	Click read transactions from a file Xt.txt	There is error format in the mentioned file because of that display an error message.	Displayed an error message saying format is wrong	Pass
	7.3	Click read transactions from a text file Kt.txt	There is no file name kt because of that display an error message	Displayed an error message	pass

About	8	Click about button	Display instructions to use the financial tracker using message box	Displayed the instructions using message box	pass
Exit	9	Click the exit button	Exit from the application	Exit from the application	Pass

Screen shots of the test cases

1.1

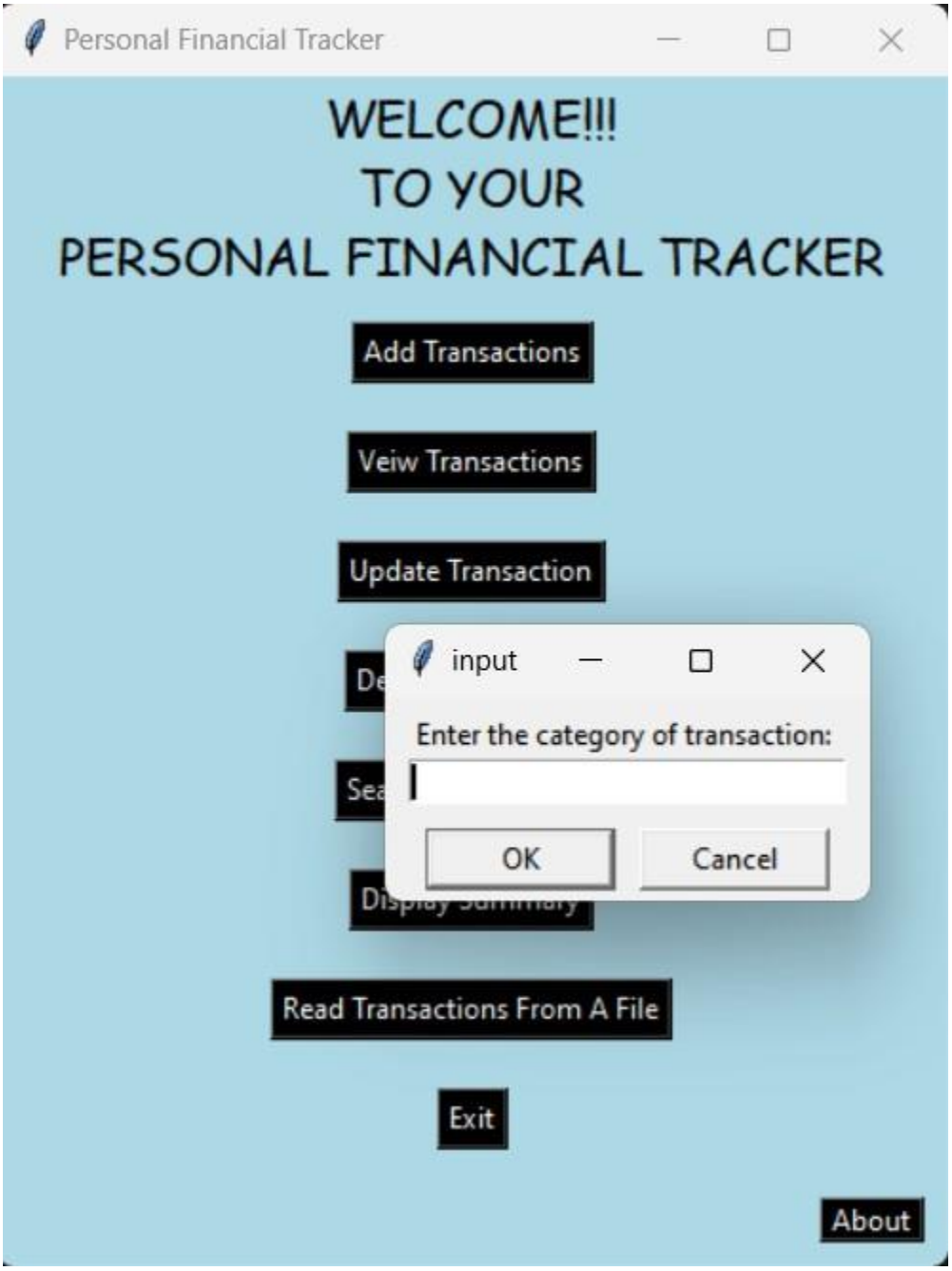


Figure 1


Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Succsess
Transaction added succsessfully
OK
Read Transactions From A File
Exit
About

Figure 2

JSON file

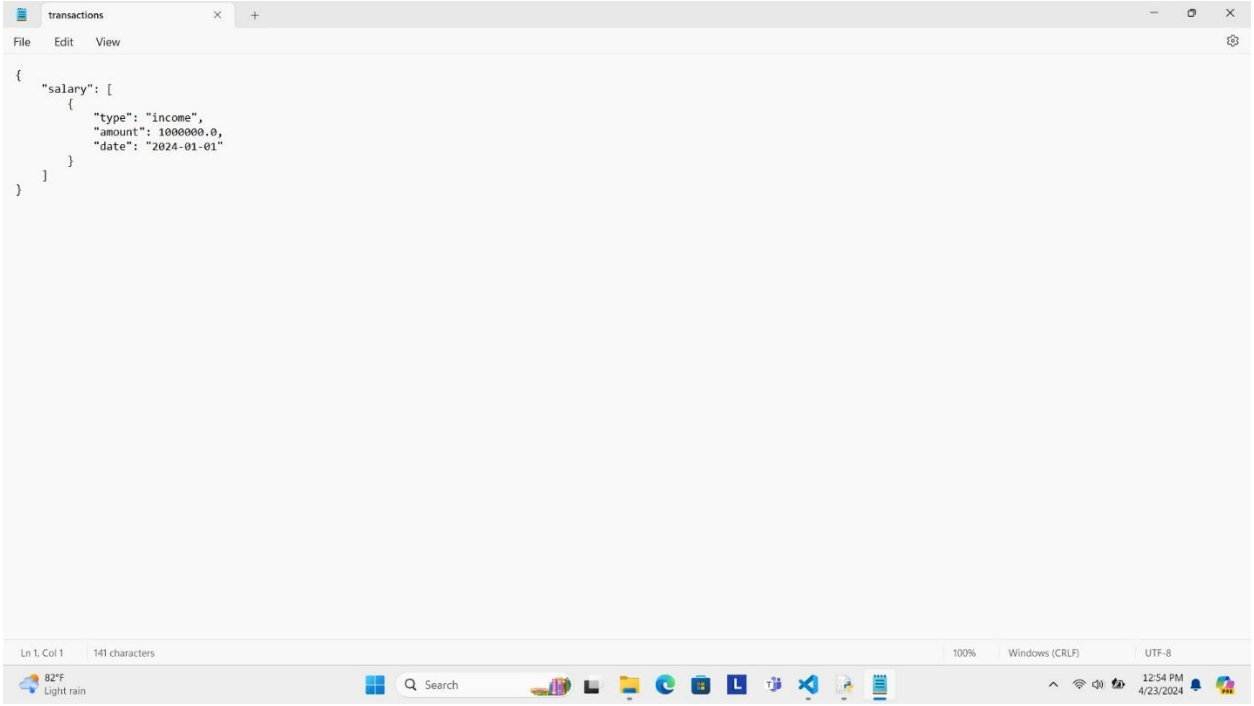


Figure 3

1.2

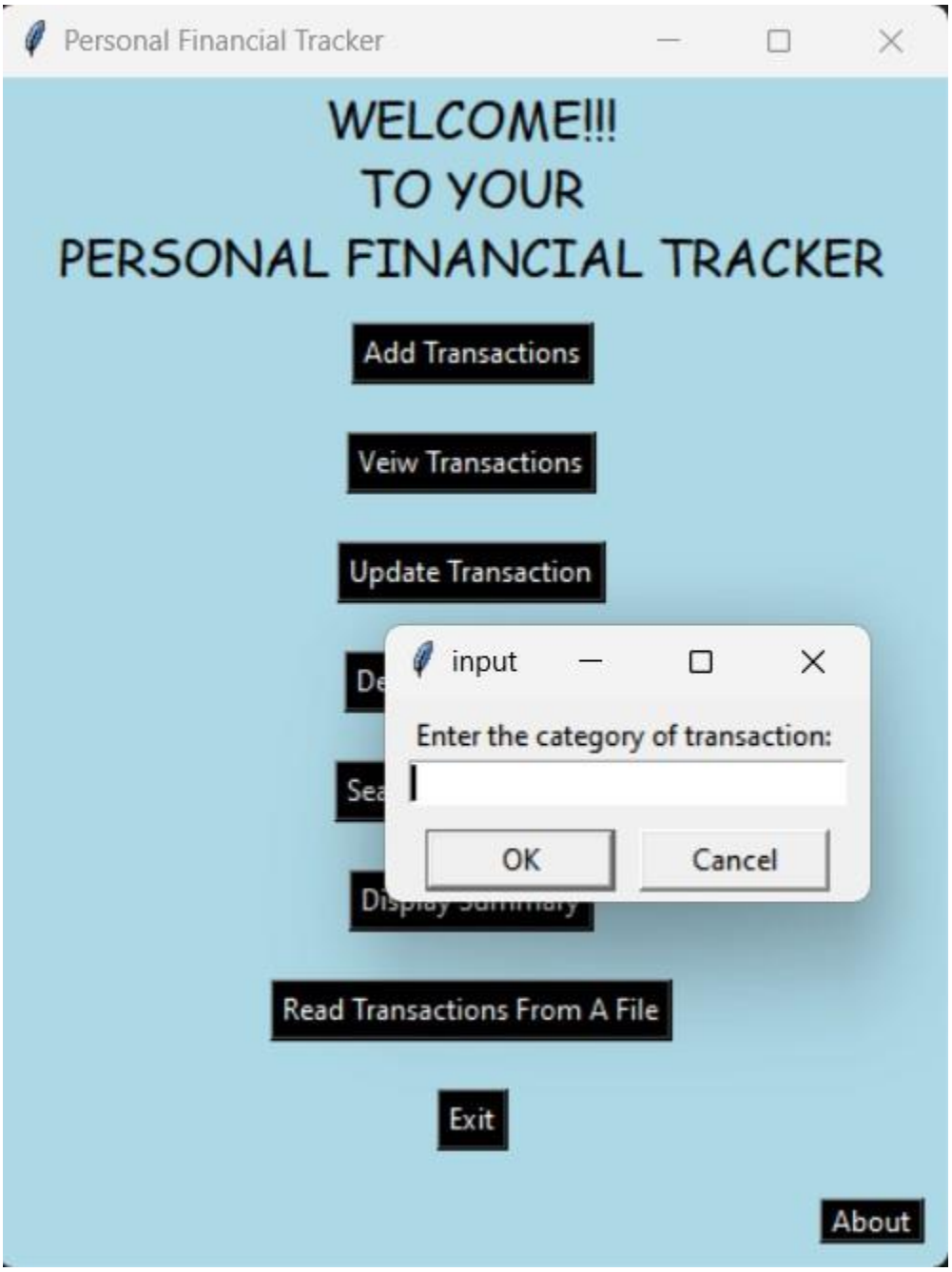

Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Update Transaction
input
Enter the category of transaction:
OK
Cancel
Read Transactions From A File
Exit
About

Figure 4


Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Succsess
Transaction added succsessfully
OK
Read Transactions From A File
Exit
About

Figure 5

JSON file

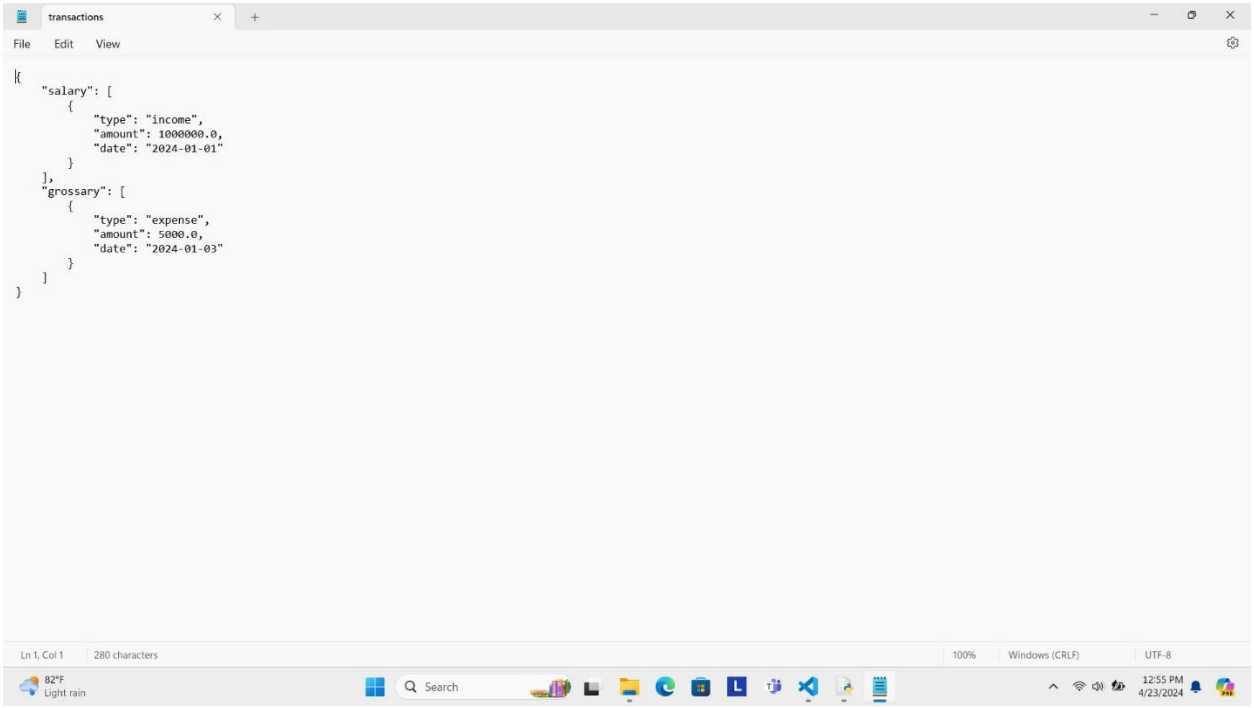


Figure 6

1.3

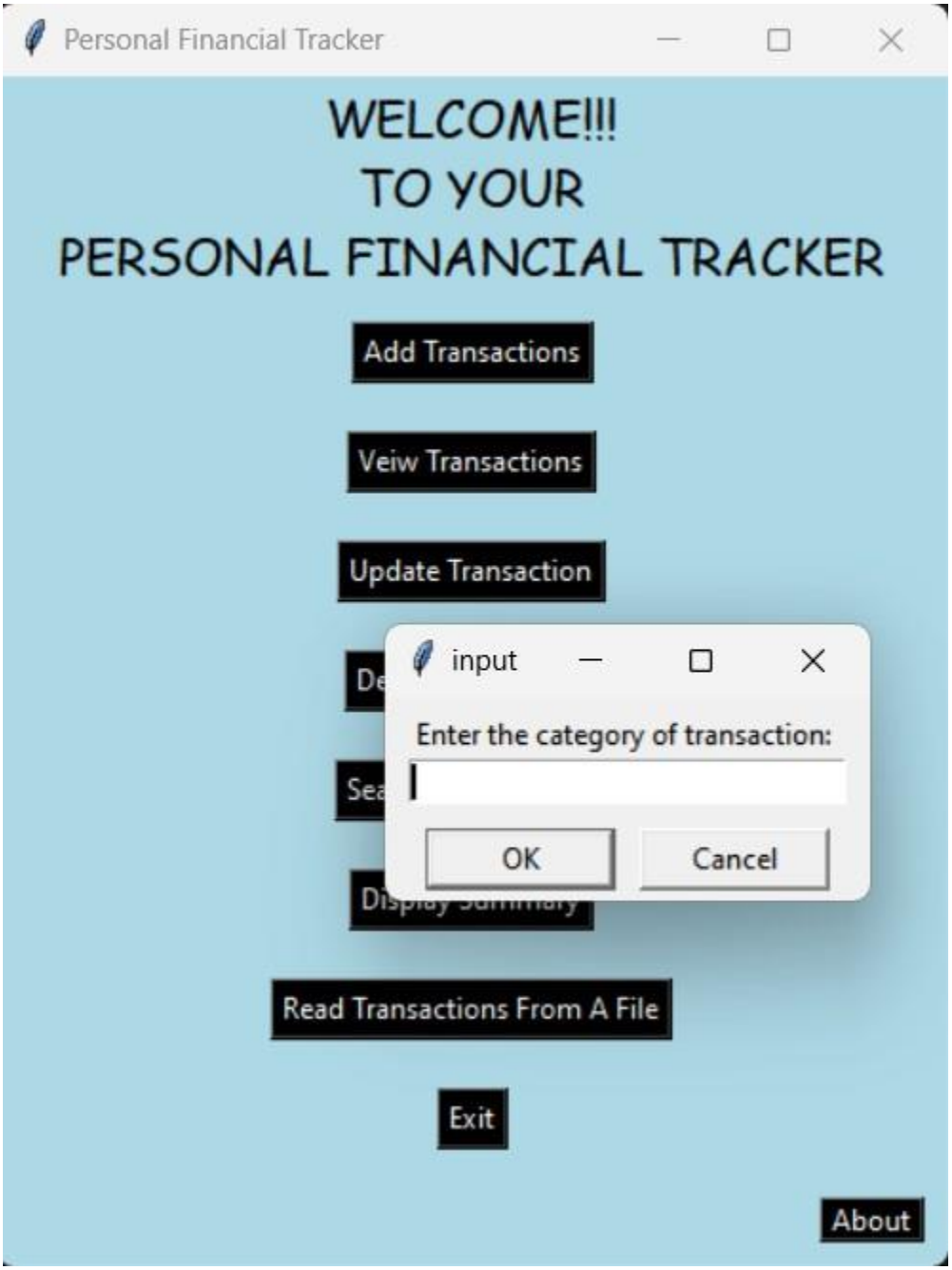

Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Update Transaction
input
Enter the category of transaction:
OK
Cancel
Read Transactions From A File
Exit
About

Figure 7


Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Succsess
Transaction added succsessfully
OK
Read Transactions From A File
Exit
About

Figure 8

JSON file

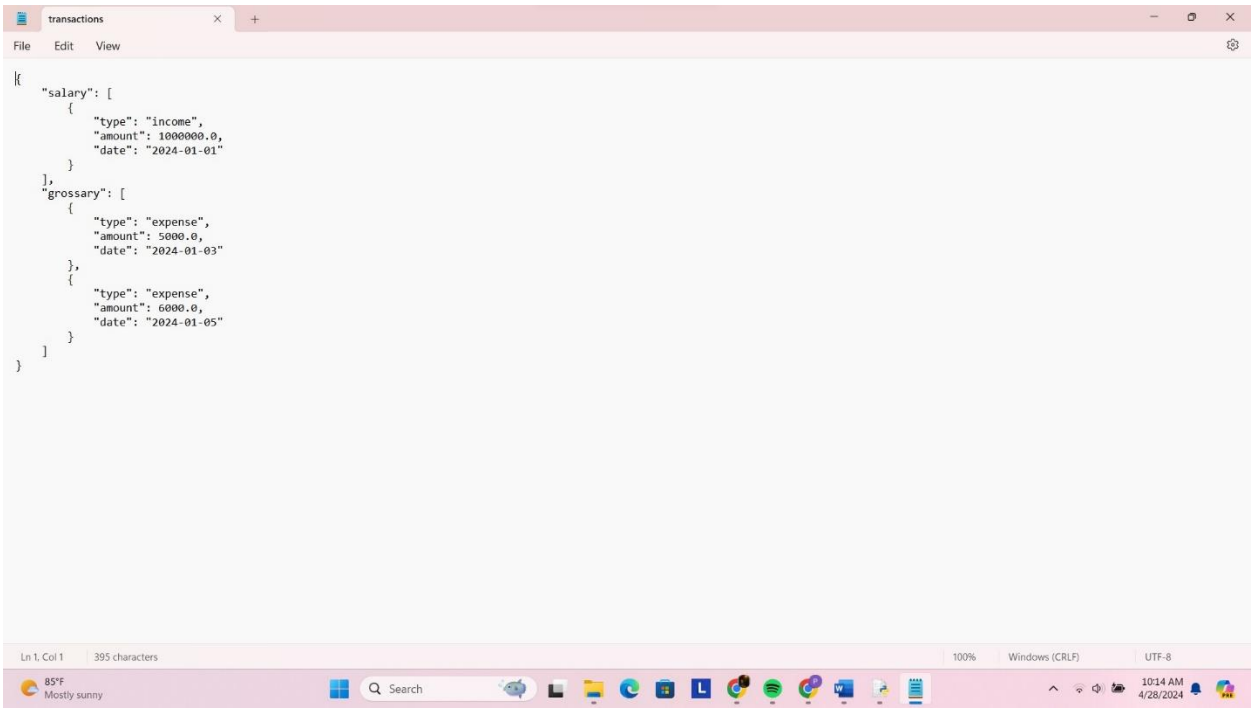


Figure 9

1.4

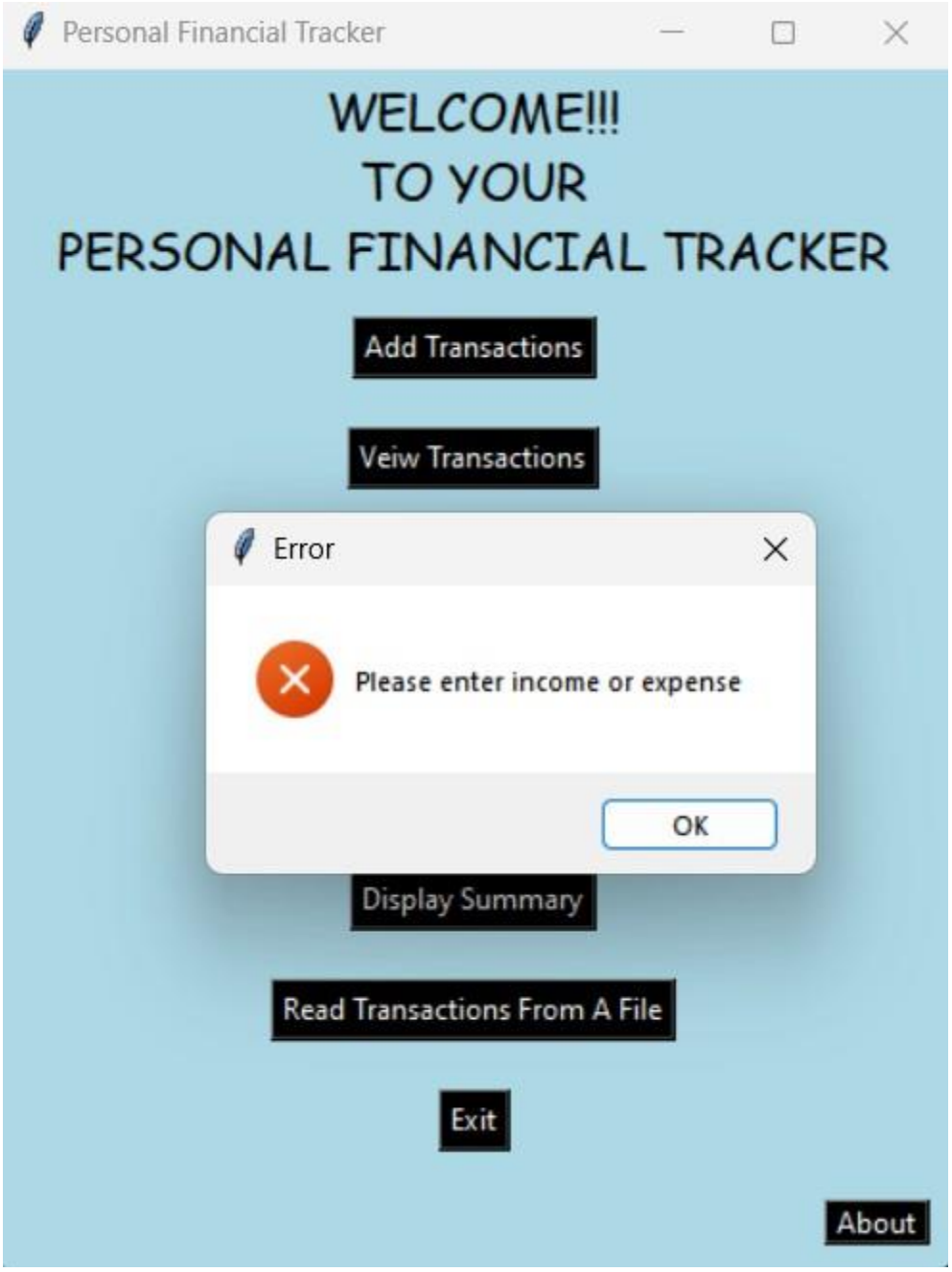

Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Error
Please enter income or expense
OK
Display Summary
Read Transactions From A File
Exit
About

Figure 10

1.5

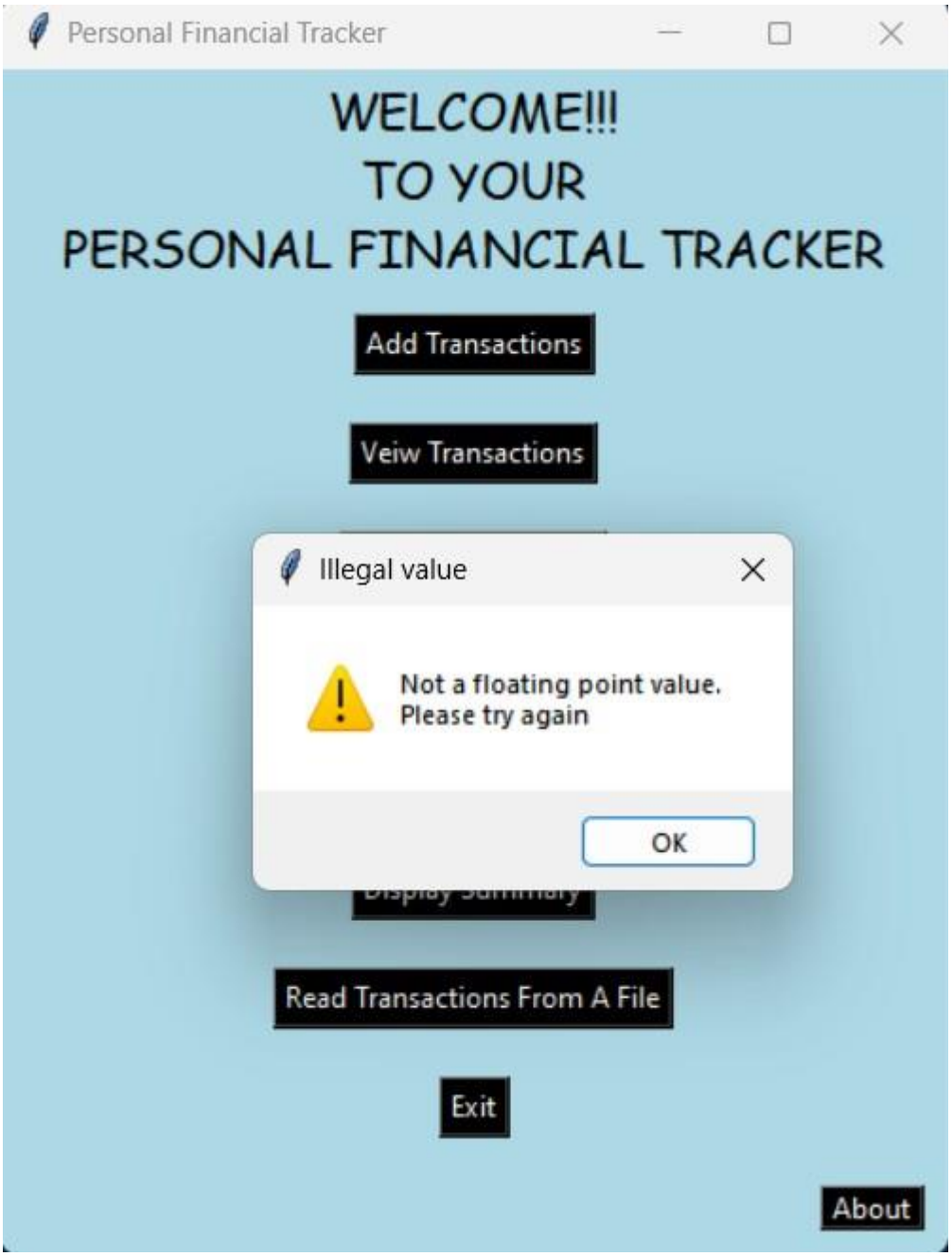
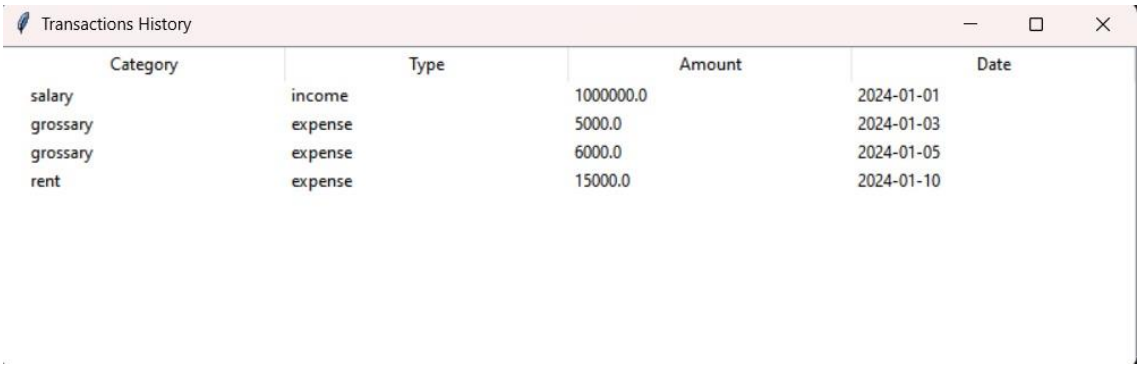

Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Illegal value
Not a floating point value.
Please try again
OK
Read Transactions From A File
Exit
About

Figure 11

2



Category	Type	Amount	Date
salary	income	1000000.0	2024-01-01
grossary	expense	5000.0	2024-01-03
grossary	expense	6000.0	2024-01-05
rent	expense	15000.0	2024-01-10

Figure 12

3.1

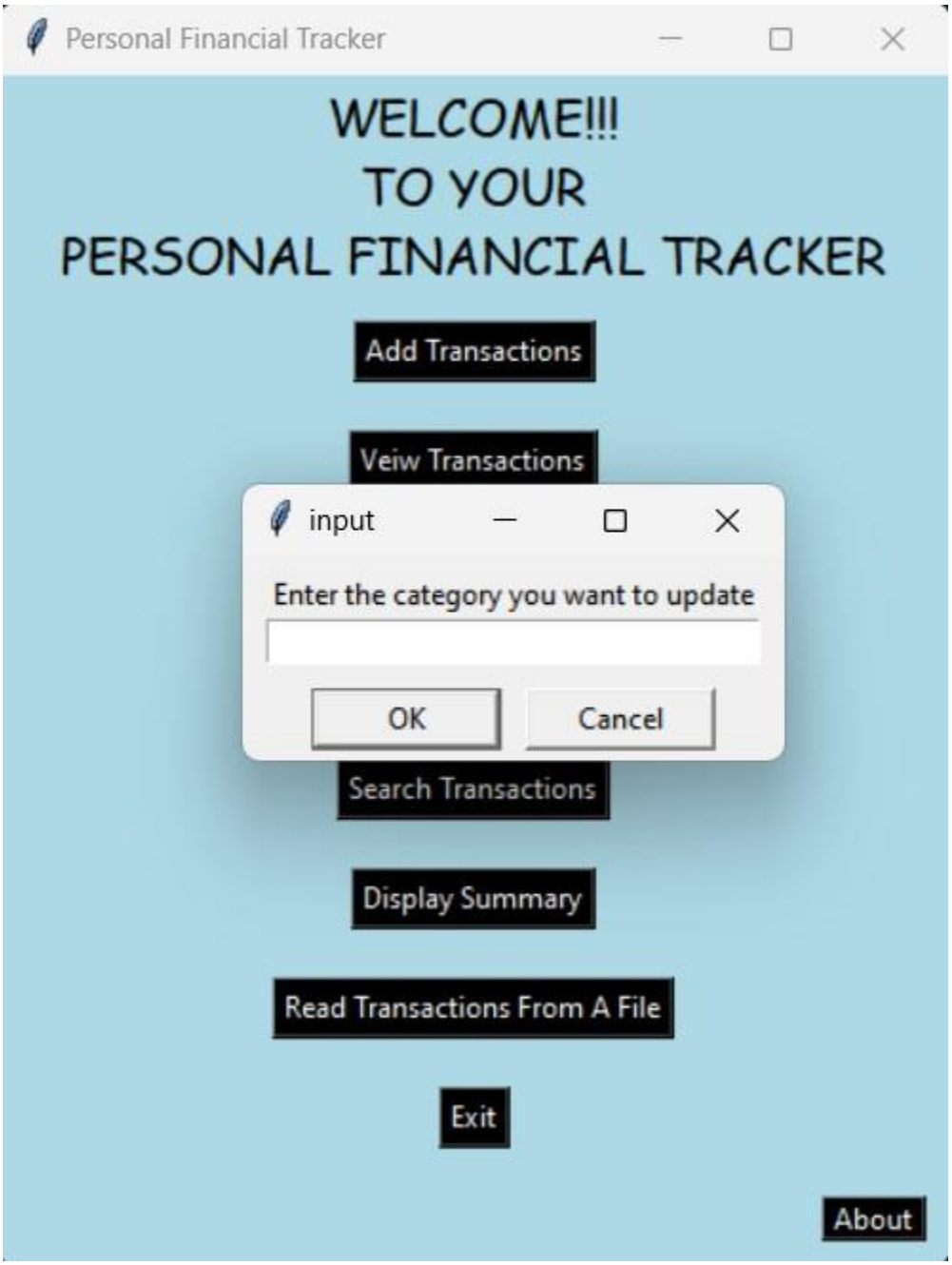

Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
input
Enter the category you want to update
OK
Cancel
Search Transactions
Display Summary
Read Transactions From A File
Exit
About

Figure 13

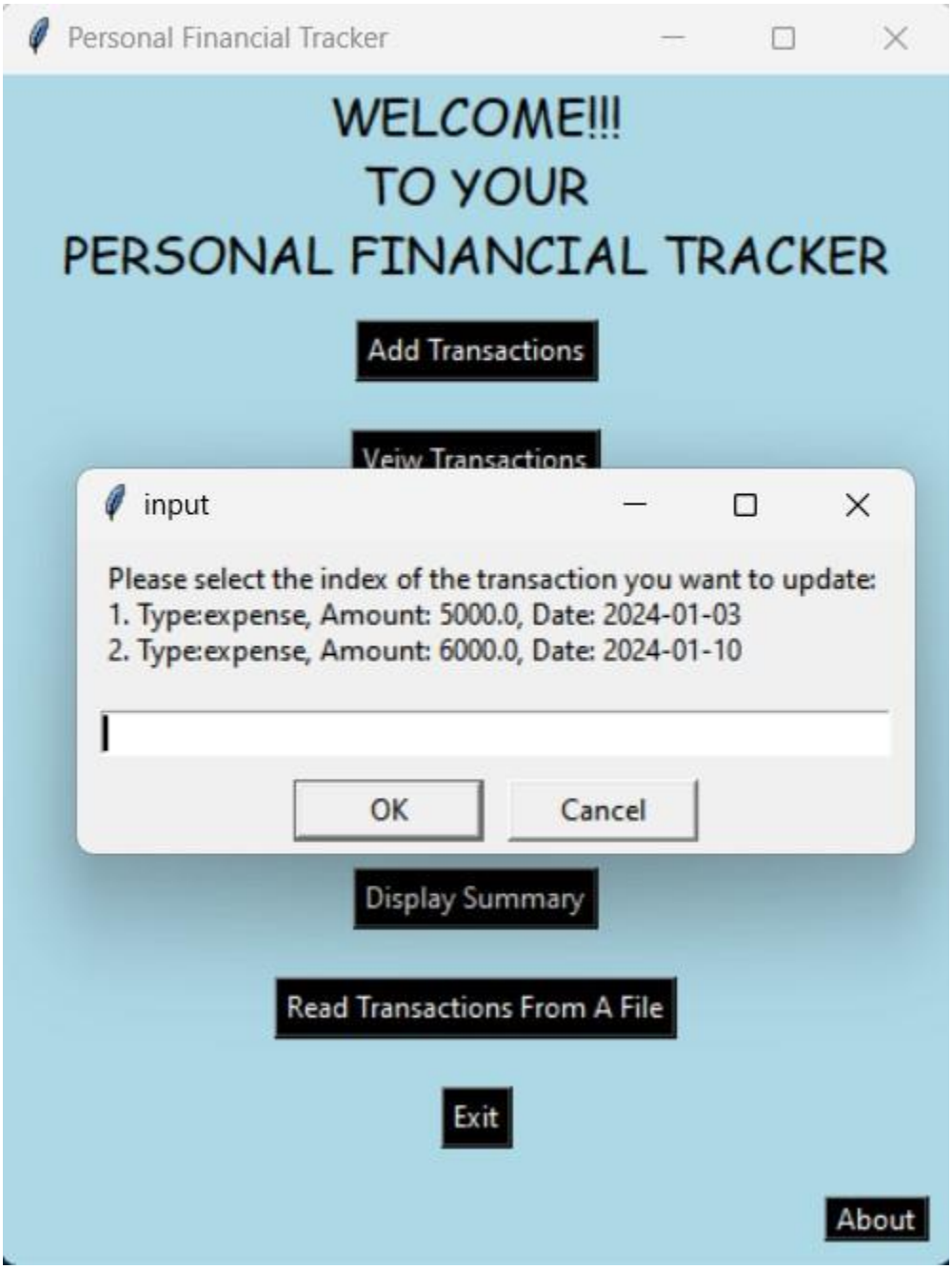

Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
input
Please select the index of the transaction you want to update:
1. Type:expense, Amount: 5000.0, Date: 2024-01-03
2. Type:expense, Amount: 6000.0, Date: 2024-01-10
OK
Cancel
Display Summary
Read Transactions From A File
Exit
About

Figure 14

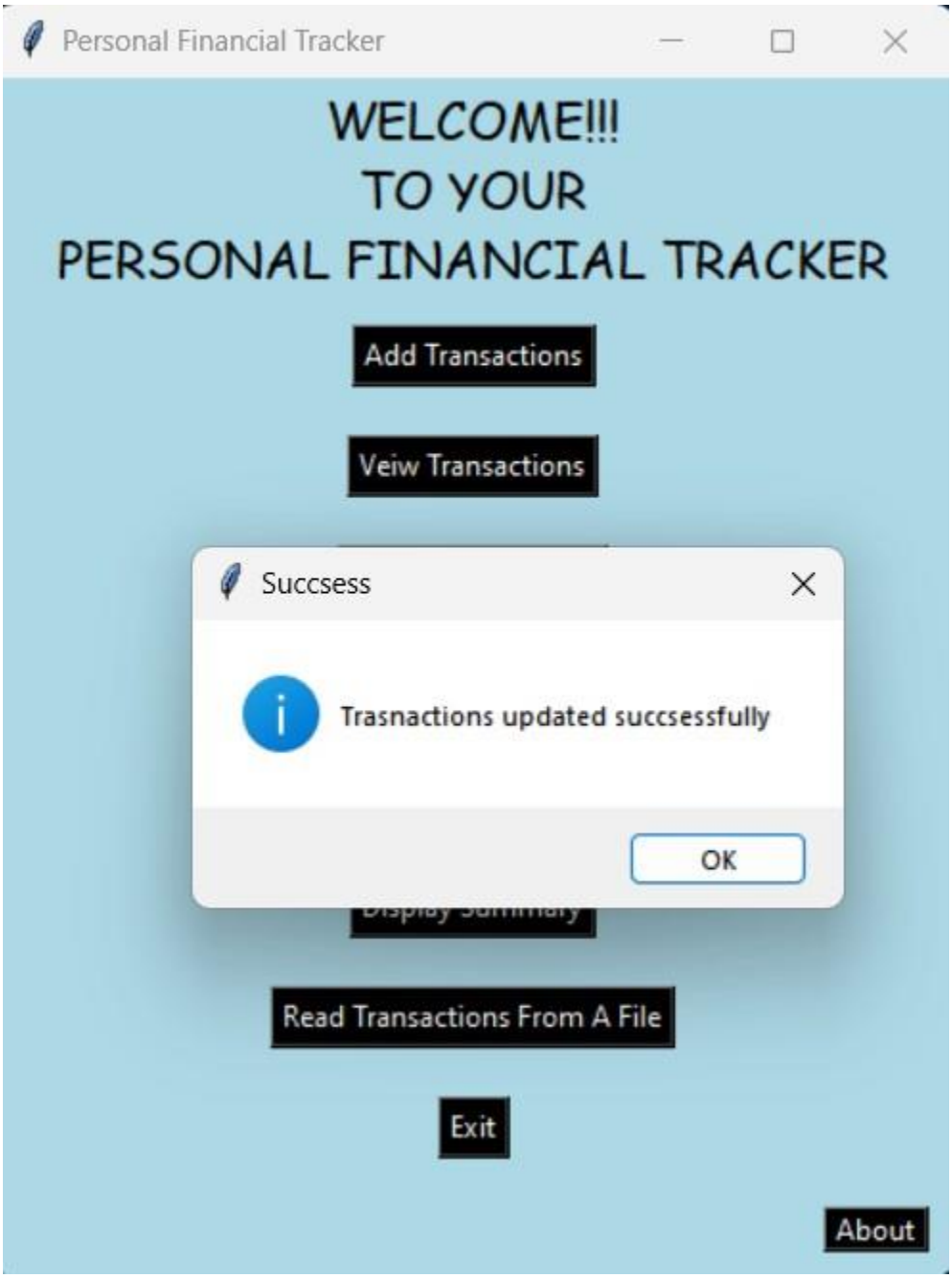

Personal Financial Tracker
WELCOME!!!
TO YOUR
PERSONAL FINANCIAL TRACKER
Add Transactions
Veiw Transactions
Succsess
Trasnactions updated succsessfully
OK
Read Transactions From A File
Exit
About

Figure 15

JSON file

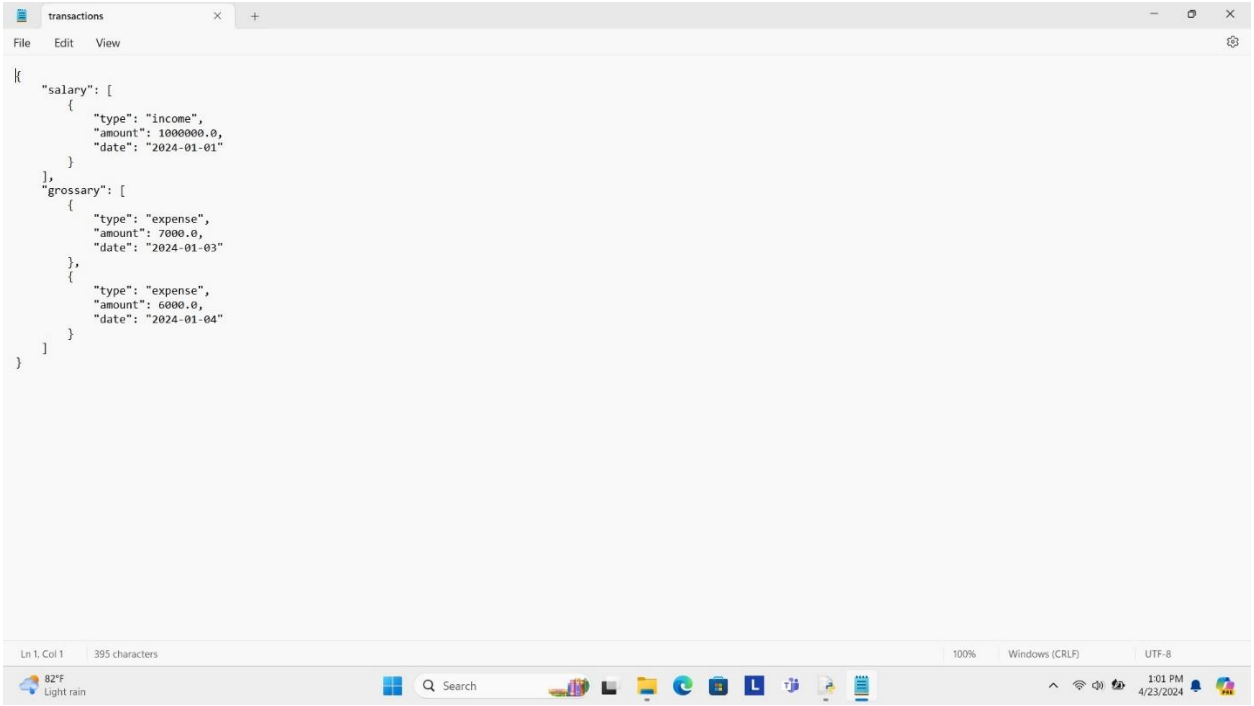


Figure 16

3.2

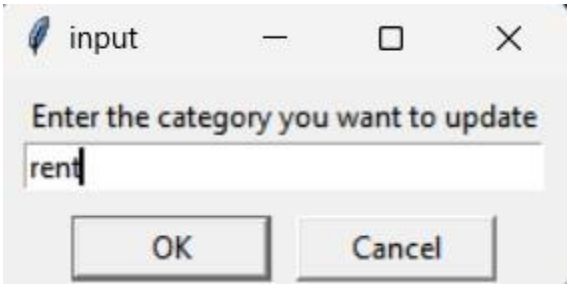


Figure 17

Figure 18

3.3

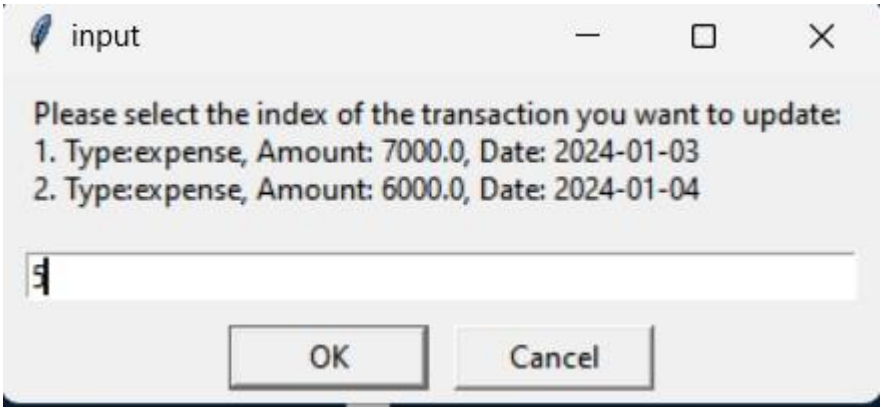


Figure 19

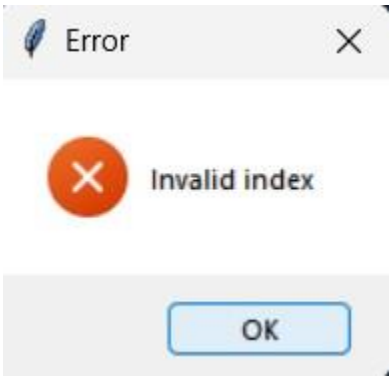


Figure 20

4.

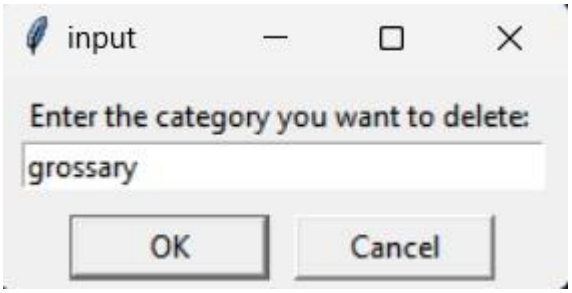


Figure 21

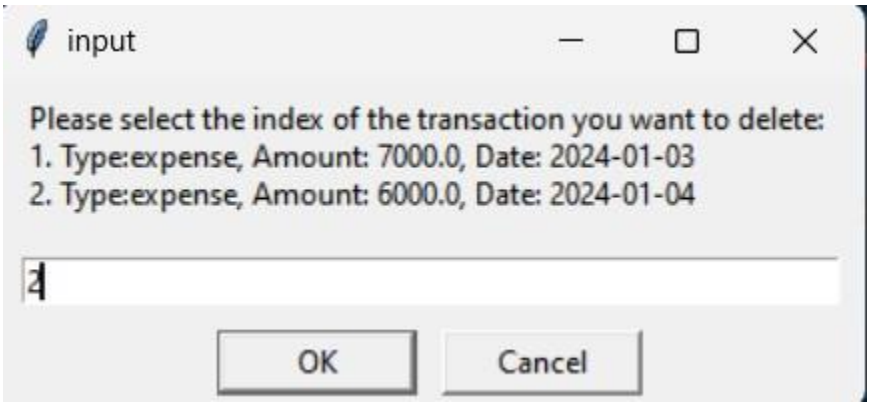


Figure 22

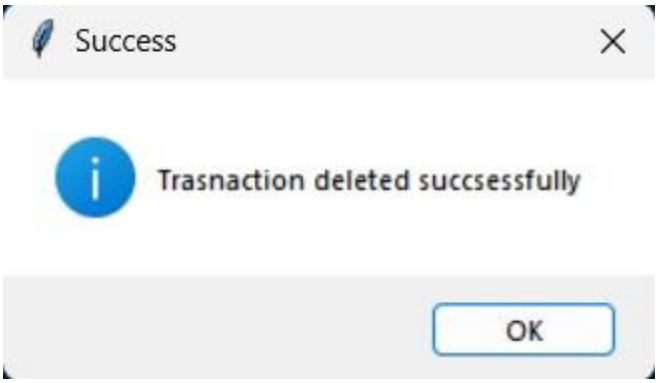
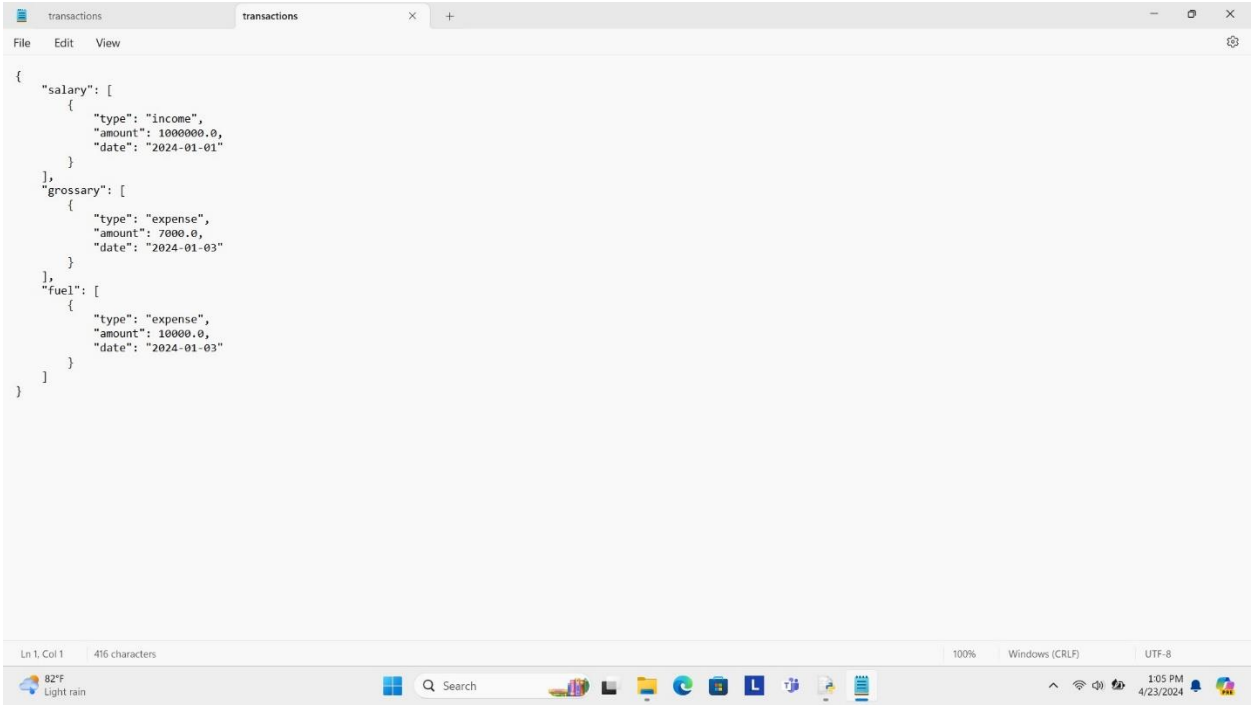


Figure 23

JSON file



```
{
    "salary": [
        {
            "type": "income",
            "amount": 1000000.0,
            "date": "2024-01-01"
        }
    ],
    "grossary": [
        {
            "type": "expense",
            "amount": 7000.0,
            "date": "2024-01-03"
        }
    ],
    "fuel": [
        {
            "type": "expense",
            "amount": 10000.0,
            "date": "2024-01-03"
        }
    ]
}
```

Figure 24

5.1

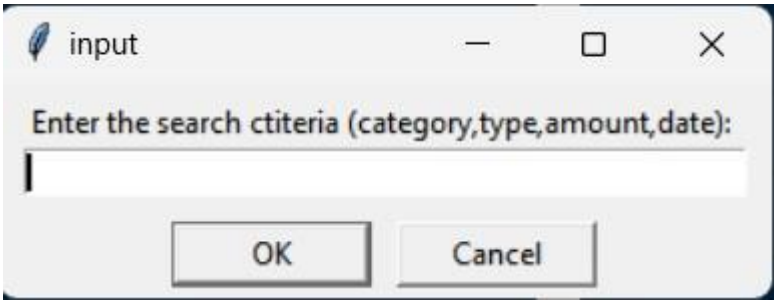


Figure 25

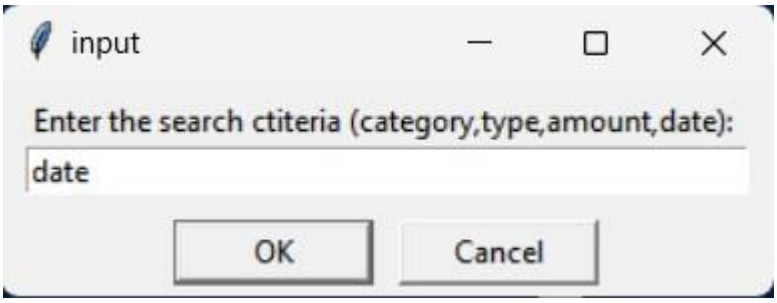


Figure 26

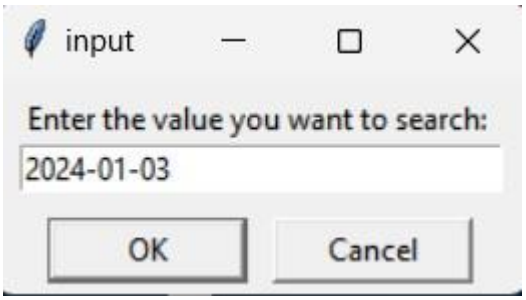


Figure 27

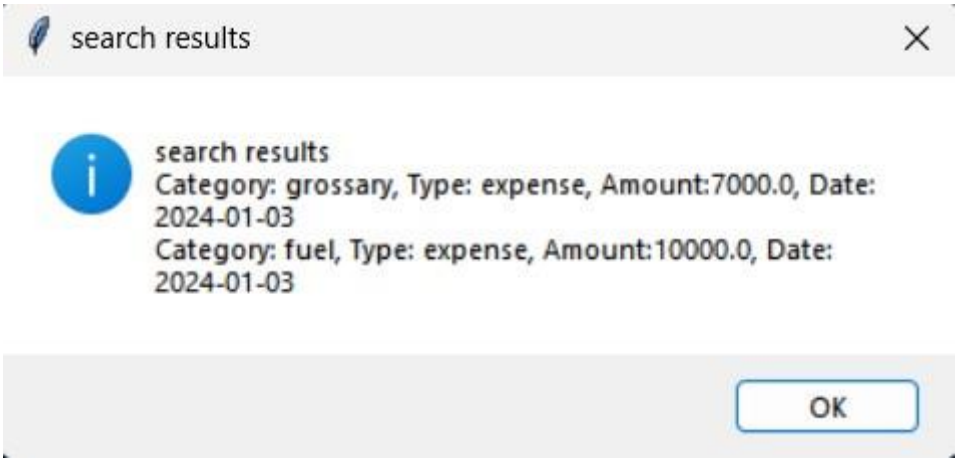


Figure 28

5.2

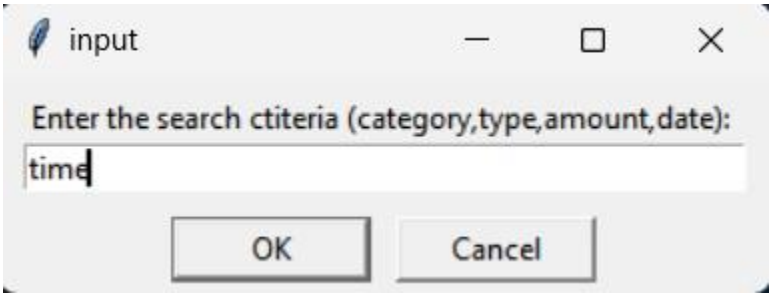


Figure 29

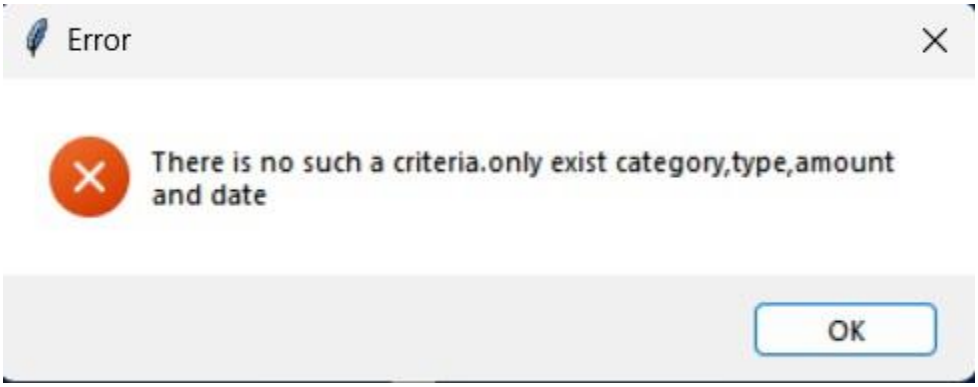


Figure 30

5.3

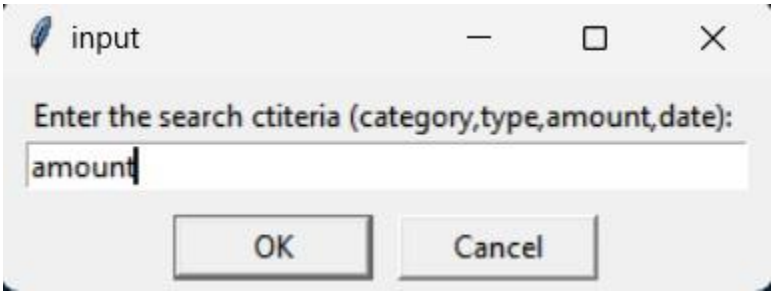


Figure 31

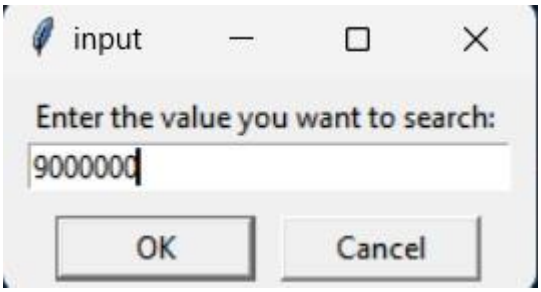


Figure 32

Figure 33

6

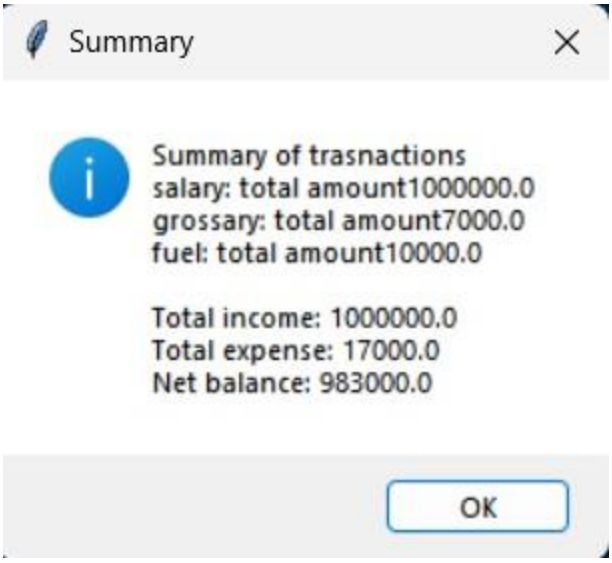

Summary
Summary of trasnactions
salary: total amount1000000.0
grossary: total amount7000.0
fuel: total amount10000.0
Total income: 1000000.0
Total expense: 17000.0
Net balance: 983000.0
OK

Figure 34

7.1

Text file

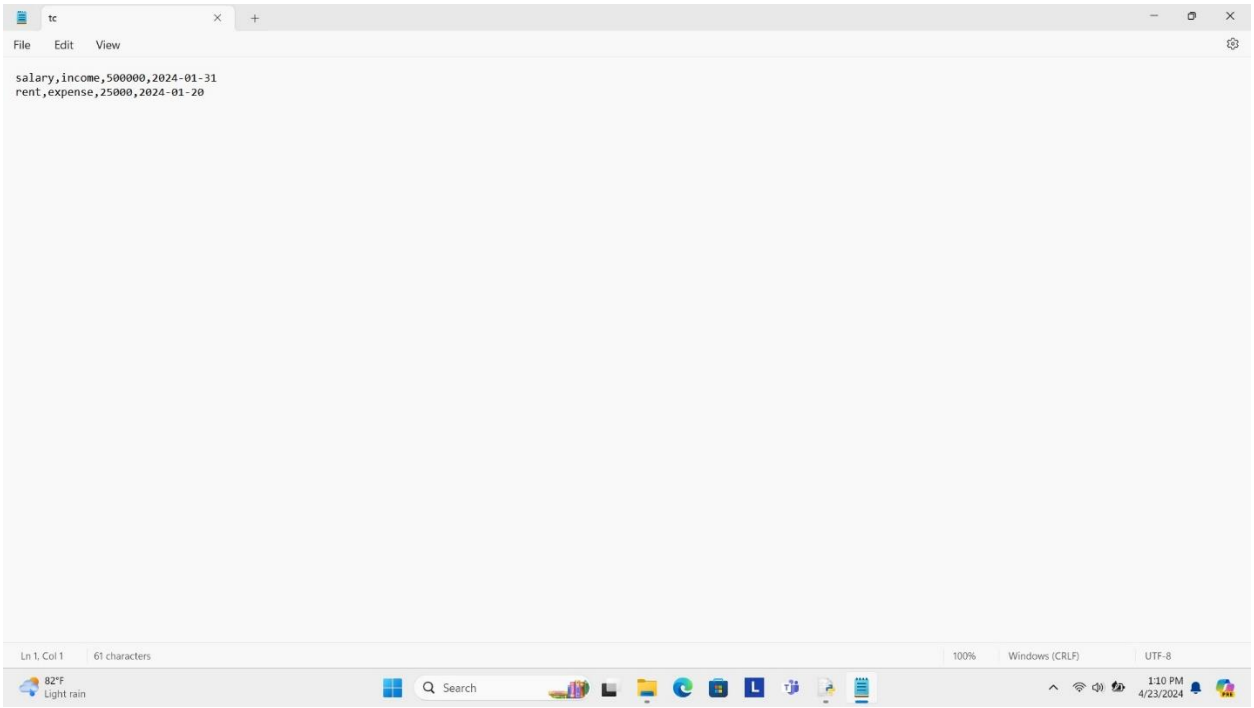


Figure 35

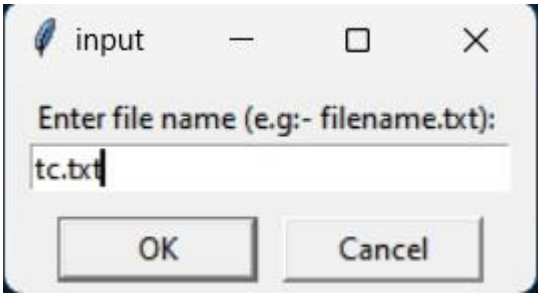


Figure 36

Figure 37

JSON file

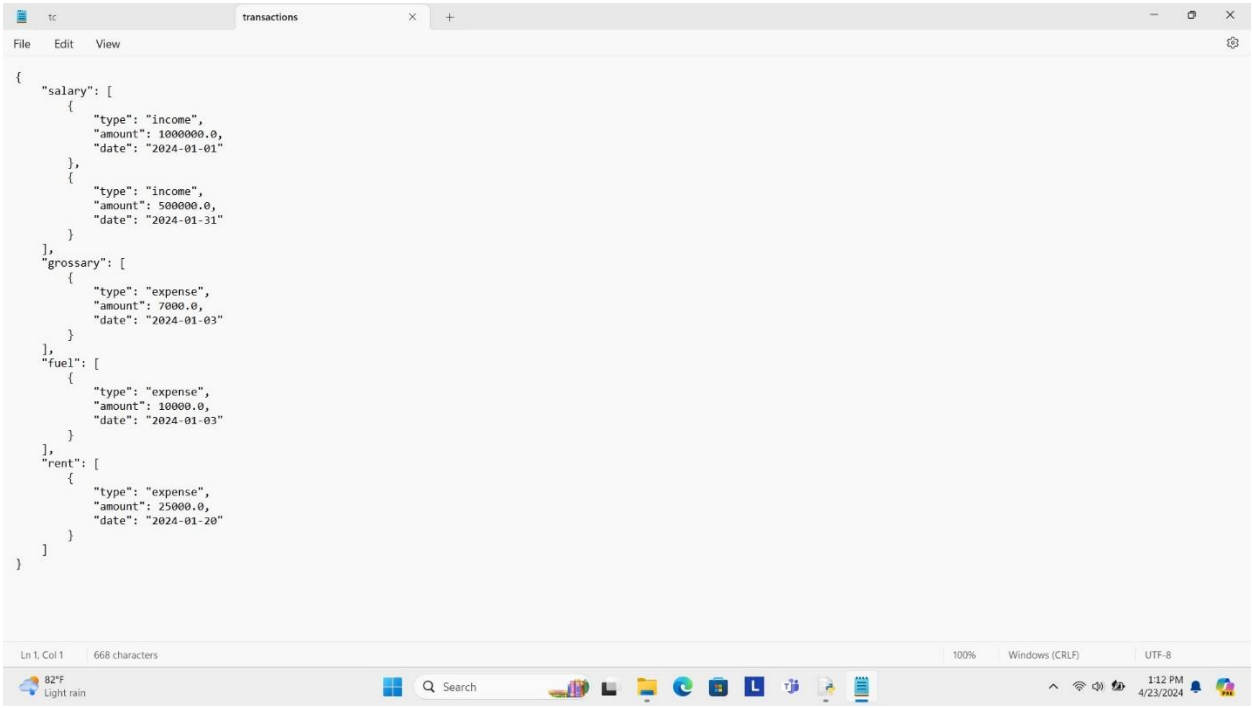


Figure 38

7.2

Text file

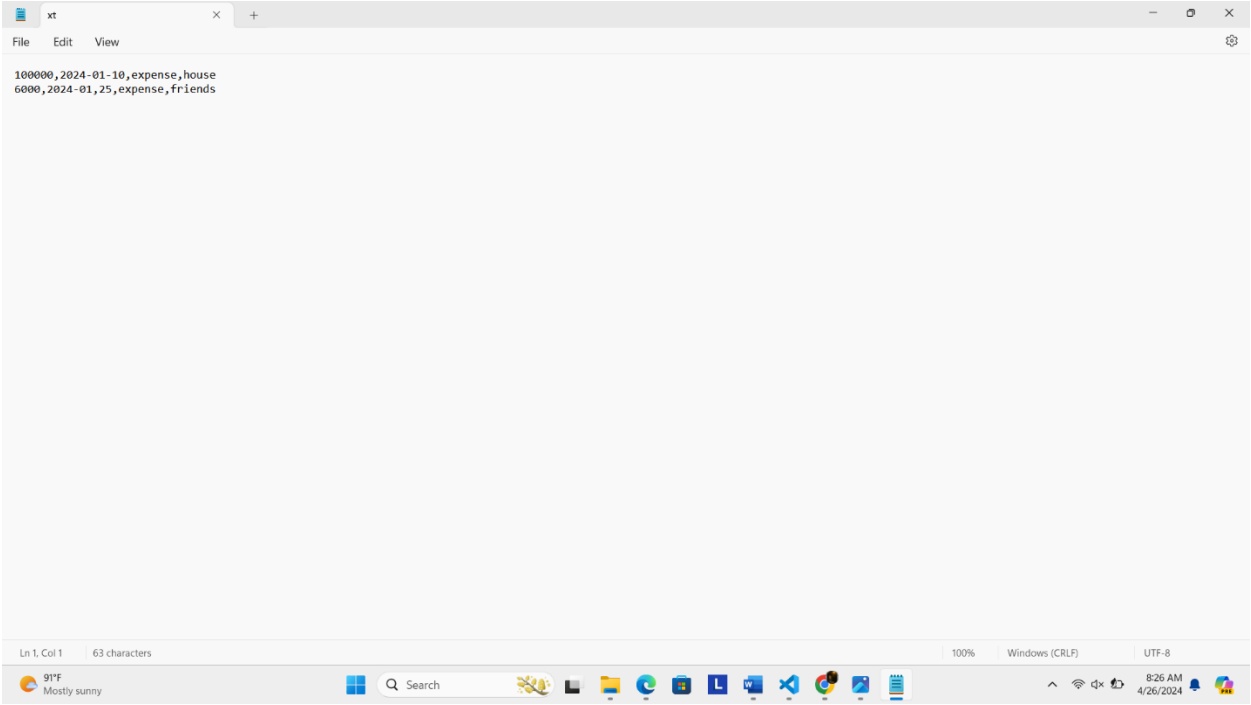


Figure 39

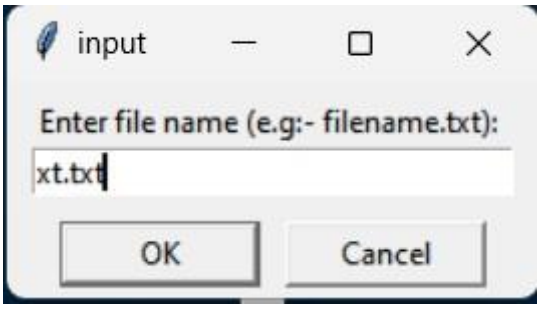


Figure 40

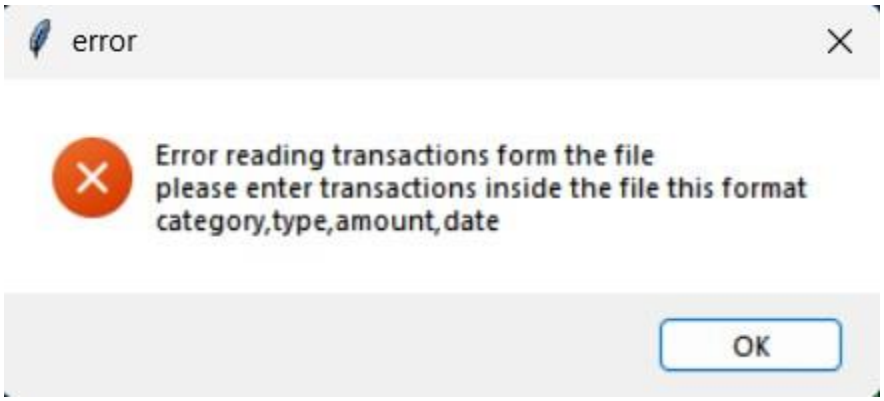


Figure 41

7.3

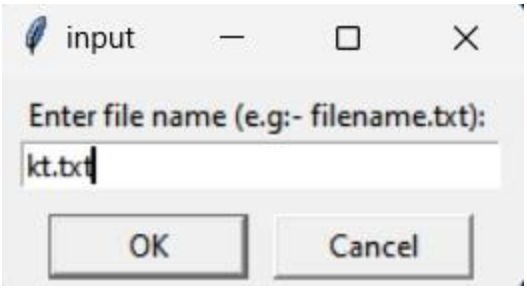


Figure 42

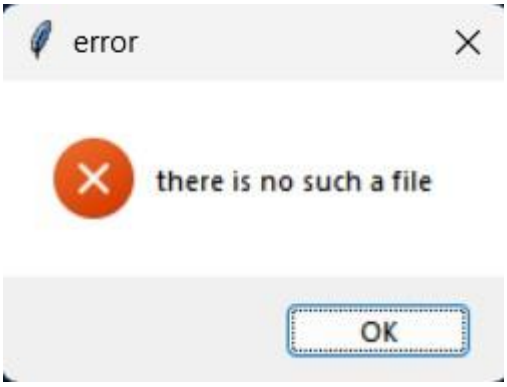


Figure 43

9

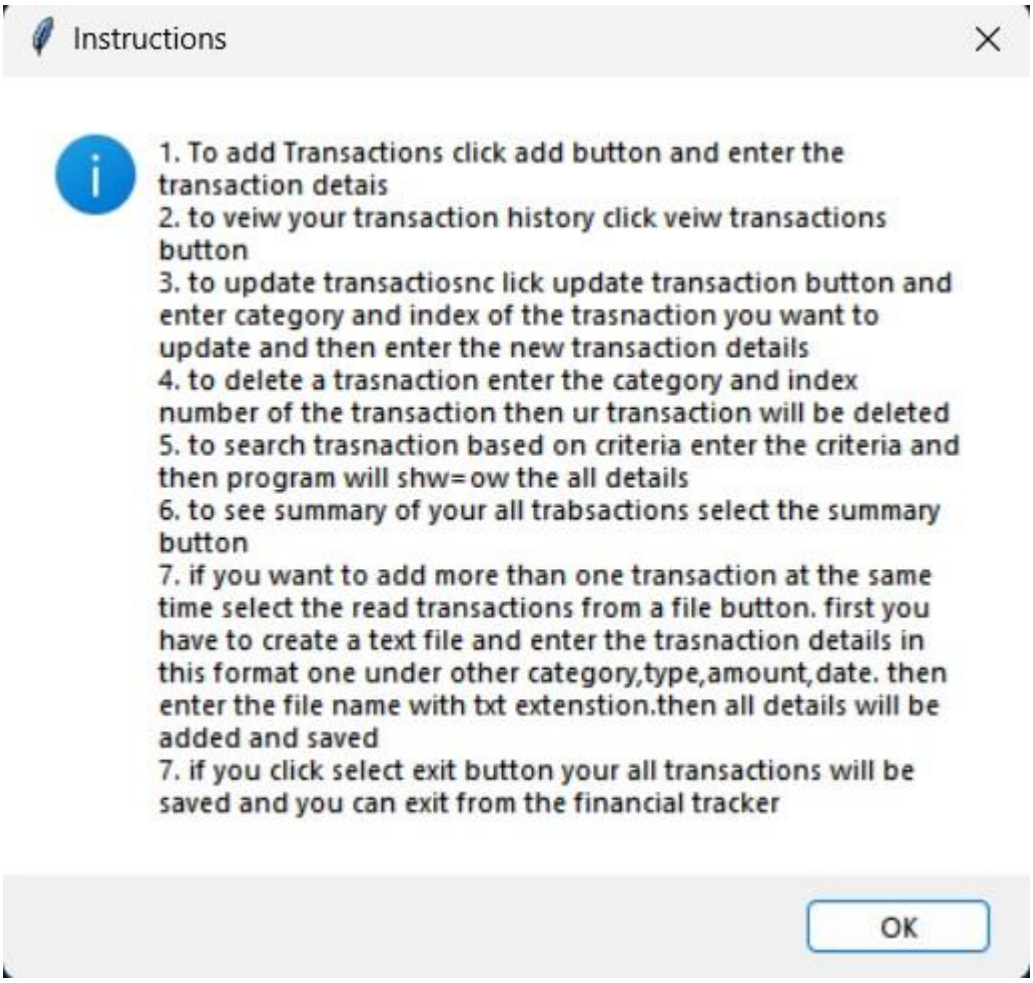

Instructions
1. To add Transactions click add button and enter the transaction detais
2. to veiw your transaction history click veiw transactions button
3. to update transactiosnc lick update transaction button and enter category and index of the trasnaction you want to update and then enter the new transaction details
4. to delete a trasnaction enter the category and index number of the transaction then ur transaction will be deleted
5. to search trasnaction based on criteria enter the criteria and then program will shw=ow the all details
6. to see summary of your all trabsactions select the summary button
7. if you want to add more than one transaction at the same time select the read transactions from a file button. first you have to create a text file and enter the trasnaction details in this format one under other category,type,amount,date. then enter the file name with txt extenstion.then all details will be added and saved
7. if you click select exit button your all transactions will be saved and you can exit from the financial tracker
OK

Figure 44

Summary of test cases

Total test cases: 19

Pass: 19

Fail: 0