import json

import datetime

# Global dictionary to store transactions

transactionsB = {"Groceries" : [], "Salary" : [], "Utilities" : [], "Transport" : [], "Other Income" : [], "Other Expense" : []}

'''Open a JSON file in read mode and if it is not, handle the error'''

# File handling functions

def load\_transactions():

global transactionsB

while True:

try:

with open("transactionsB.json", "r") as file:

return json.load(file)

except FileNotFoundError:

return{}

'''Open the JSON file in write mode and create a function to save the transactions to the JSON file '''

def save\_transactions():

with open("transactionsB.json", "w") as file:

json.dump(transactionsB, file)

def read\_bulk\_transactions\_from\_file(filename):

# Open and read the file, then parse each line to add to the transactions dictionary

global transactionsB

try:

with open(filename, "r") as file:

for line in file:

date, add\_choice, amount = line.strip.split(",")

add\_transaction()

except FileNotFoundError:

return{}

# Feature implementations

def add\_transaction():

global transactionsB

while True:

print("1. Groceries")

print("2. Salary")

print("3. Utilities")

print("4. Transport")

print("5. Other Expense")

print("6. Other Income")

try:

add\_choice = input("\nPlease select the category to add the transaction : ")

except ValueError:

print("\nInvalid Input! Please input valid number\n")

if add\_choice == "1":

while True:

try:

amount = float(input("Enter amount of groceries : "))

except ValueError:

print("\nPlease input numeric value!\n")

continue

else:

break

while True:

is\_today = input("Did you make this transaction today(Yes / No)? : ")

is\_today = is\_today.lower()

if is\_today == "yes":

today = datetime.datetime.now().date()

date = str(today)

break

elif is\_today == "no":

date = input("Enter the date in YYYY-MM-DD format : ")

break

else:

print("Please input yes or no")

transactionsB["Groceries"].append({"amount" : amount, "date" : date})

save\_transactions()

print("\nTransaction added successfully!\n")

print(transactionsB)

break

elif add\_choice == "2":

while True:

try:

amount = float(input("Enter amount of salary : "))

except ValueError:

print("\nPlease input numeric value!\n")

continue

else:

break

while True:

is\_today = input("Did you make this transaction today(Yes / No)? : ")

is\_today = is\_today.lower()

if is\_today == "yes":

today = datetime.datetime.now().date()

date = str(today)

break

elif is\_today == "no":

date = input("Enter the date in YYYY-MM-DD format : ")

break

else:

print("Please input yes or no")

transactionsB["Salary"].append({"amount" : amount, "date" : date})

save\_transactions()

print("\nTransaction added successfully!")

break

elif add\_choice == "3":

while True:

try:

amount = float(input("Enter amount of utilities : "))

except ValueError:

print("\nPlease input numeric value!\n")

continue

else:

break

while True:

is\_today = input("Did you make this transaction today(Yes / No)? : ")

is\_today = is\_today.lower()

if is\_today == "yes":

today = datetime.datetime.now().date()

date = str(today)

break

elif is\_today == "no":

date = input("Enter the date in YYYY-MM-DD format : ")

break

else:

print("Please input yes or no")

transactionsB["Utilities"].append({"amount" : amount, "date" : date})

save\_transactions()

print("\nTransaction added successfully!")

break

elif add\_choice == "4":

while True:

try:

amount = float(input("Enter amount of transport : "))

except ValueError:

print("\nPlease input numeric value!\n")

continue

else:

break

while True:

is\_today = input("Did you make this transaction today(Yes / No)? : ")

is\_today = is\_today.lower()

if is\_today == "yes":

today = datetime.datetime.now().date()

date = str(today)

break

elif is\_today == "no":

date = input("Enter the date in YYYY-MM-DD format : ")

break

else:

print("Please input yes or no")

transactionsB["Transport"].append({"amount" : amount, "date" : date})

save\_transactions()

print("\nTransaction added successfully!")

break

elif add\_choice == "5":

while True:

try:

amount = float(input("Enter amount of groceries : "))

except ValueError:

print("\nPlease input numeric value!\n")

continue

else:

break

while True:

is\_today = input("Did you make this transaction today(Yes / No)? : ")

is\_today = is\_today.lower()

if is\_today == "yes":

today = datetime.datetime.now().date()

date = str(today)

break

elif is\_today == "no":

date = input("Enter the date in YYYY-MM-DD format : ")

break

else:

print("Please input yes or no")

transactionsB["Other Expense"].append({"amount" : amount, "date" : date})

save\_transactions()

print("\nTransaction added successfully!")

break

elif add\_choice == "6":

while True:

try:

amount = float(input("Enter amount of other income : "))

except ValueError:

print("\nPlease input numeric value!\n")

continue

else:

break

while True:

is\_today = input("Did you make this transaction today(Yes / No)? : ")

is\_today = is\_today.lower()

if is\_today == "yes":

today = datetime.datetime.now().date()

date = str(today)

break

elif is\_today == "no":

date = input("Enter the date in YYYY-MM-DD format : ")

break

else:

print("Please input yes or no")

transactionsB["Other Income"].append({"amount" : amount, "date" : date})

save\_transactions()

print("\nTransaction added successfully!")

break

else:

print("\nPlease Enter valid range number!")

def view\_transactions():

global transactionsB

while True:

print("1. Groceries")

print("2. Salary")

print("3. Utilities")

print("4. Transport")

print("5. Other Expense")

print("6. Other Income")

try:

view\_choice = int(input("\nPlease select the category to view the transactions : "))

except ValueError:

print("\nInvalid Input! Please input valid number\n")

continue

number = 1

if view\_choice == 1:

if "Groceries" in transactionsB:

print("|Groceries Transactions|")

num = 1

for transactions in transactionsB["Groceries"]:

print(number, ".", "Date:",transactions["date"], ", Amount:", transactions["amount"])

number += 1

else:

print("No transactions found for groceries")

break

if view\_choice == 2:

if "Salary" in transactionsB:

print("|Salary Transactions|")

num = 1

for transactions in transactionsB["Salary"]:

print(number, ".", "Date:",transactions["date"], ", Amount:", transactions["amount"])

number += 1

else:

print("No transactions found for salary")

break

if view\_choice == 3:

if "Utilities" in transactionsB:

print("|Utilities Transactions|")

num = 1

for transactions in transactionsB["Utilities"]:

print(number, ".", "Date:",transactions["date"], ", Amount:", transactions["amount"])

number += 1

else:

print("No transactions found for utilities")

break

if view\_choice == 4:

if "Transport" in transactionsB:

print("|Transport Transactions|")

num = 1

for transactions in transactionsB["Transport"]:

print(number, ".", "Date:",transactions["date"], ", Amount:", transactions["amount"])

number += 1

else:

print("No transactions found for transport")

break

if view\_choice == 5:

if "Other Expense" in transactionsB:

print("|Other Expense Transactions|")

num = 1

for transactions in transactionsB["Other Expense"]:

print(number, ".", "Date:",transactions["date"], ", Amount:", transactions["amount"])

number += 1

else:

print("No transactions found for other expense")

break

if view\_choice == 6:

if "Other Income" in transactionsB:

print("|Other Income Transactions|")

num = 1

for transactions in transactionsB["Other Income"]:

print(number, ".", "Date:",transactions["date"], ", Amount:", transactions["amount"])

number += 1

else:

print("No transactions found for other income")

break

else:

print("Please input valid range number")

def update\_transaction():

global transactionsB

while True:

print("1. Groceries")

print("2. Salary")

print("3. Utilities")

print("4. Transport")

print("5. Other Expense")

print("6. Other Income")

try:

update\_choice = int(input("\nPlease select the category to update the transactions : "))

except ValueError:

print("\nInvalid Input! Please input valid number\n")

continue

if update\_choice == 1:

if "Groceries" in transactionsB:

print("Groceries Transactions:")

for idx, transaction in enumerate(transactionsB["Groceries"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to update: "))

if 1 <= idx <= len(transactionsB["Groceries"]):

amount = float(input("Enter the updated amount: "))

date = input("Enter the updated date (YYYY-MM-DD format): ")

transactionsB["Groceries"][idx - 1] = {"amount": amount, "date": date}

save\_transactions()

print("Transaction updated successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Groceries.")

break

if update\_choice == 2:

if "Salary" in transactionsB:

print("Salary Transactions:")

for idx, transaction in enumerate(transactionsB["Salary"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to update: "))

if 1 <= idx <= len(transactionsB["Salary"]):

amount = float(input("Enter the updated amount: "))

date = input("Enter the updated date (YYYY-MM-DD format): ")

transactionsB["Salary"][idx - 1] = {"amount": amount, "date": date}

save\_transactions()

print("Transaction updated successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Salary.")

break

if update\_choice == 3:

if "Utilities" in transactionsB:

print("Utilities Transactions:")

for idx, transaction in enumerate(transactionsB["Groceries"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to update: "))

if 1 <= idx <= len(transactionsB["Utilities"]):

amount = float(input("Enter the updated amount: "))

date = input("Enter the updated date (YYYY-MM-DD format): ")

transactionsB["Utilities"][idx - 1] = {"amount": amount, "date": date}

save\_transactions()

print("Transaction updated successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Utilities.")

break

if update\_choice == 4:

if "Transport" in transactionsB:

print("Transport Transactions:")

for idx, transaction in enumerate(transactionsB["Transport"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to update: "))

if 1 <= idx <= len(transactionsB["Transport"]):

amount = float(input("Enter the updated amount: "))

date = input("Enter the updated date (YYYY-MM-DD format): ")

transactionsB["Transport"][idx - 1] = {"amount": amount, "date": date}

save\_transactions()

print("Transaction updated successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Transport.")

break

if update\_choice == 5:

if "Other Expense" in transactionsB:

print("Other Expense Transactions:")

for idx, transaction in enumerate(transactionsB["Other Expense"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to update: "))

if 1 <= idx <= len(transactionsB["Other Expense"]):

amount = float(input("Enter the updated amount: "))

date = input("Enter the updated date (YYYY-MM-DD format): ")

transactionsB["Other Expense"][idx - 1] = {"amount": amount, "date": date}

save\_transactions()

print("Transaction updated successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Other Expense.")

break

if update\_choice == 6:

if "Other Income" in transactionsB:

print("Other Income Transactions:")

for idx, transaction in enumerate(transactionsB["Other Income"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to update: "))

if 1 <= idx <= len(transactionsB["Other Income"]):

amount = float(input("Enter the updated amount: "))

date = input("Enter the updated date (YYYY-MM-DD format): ")

transactionsB["Other Income"][idx - 1] = {"amount": amount, "date": date}

save\_transactions()

print("Transaction updated successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Other Income.")

break

else:

print("Invalid Input. Please input valid range number.")

def delete\_transaction():

global transactionsB

while True:

print("1. Groceries")

print("2. Salary")

print("3. Utilities")

print("4. Transport")

print("5. Other Expense")

print("6. Other Income")

try:

delete\_choice = int(input("\nPlease select the category to delete the transaction from: "))

except ValueError:

print("\nInvalid Input! Please input valid number\n")

continue

if delete\_choice == 1:

if "Groceries" in transactionsB:

print("Groceries Transactions:")

for idx, transaction in enumerate(transactionsB["Groceries"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to delete: "))

if 1 <= idx <= len(transactionsB["Groceries"]):

del transactionsB["Groceries"][idx - 1]

save\_transactions()

print("Transaction deleted successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Groceries.")

break

if delete\_choice == 2:

if "Salary" in transactionsB:

print("Salary Transactions:")

for idx, transaction in enumerate(transactionsB["Salary"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to delete: "))

if 1 <= idx <= len(transactionsB["Groceries"]):

del transactionsB["Salary"][idx - 1]

save\_transactions()

print("Transaction deleted successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Salary.")

break

if delete\_choice == 3:

if "Utilities" in transactionsB:

print("Utilities Transactions:")

for idx, transaction in enumerate(transactionsB["Utilities"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to delete: "))

if 1 <= idx <= len(transactionsB["Groceries"]):

del transactionsB["Utilities"][idx - 1]

save\_transactions()

print("Transaction deleted successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Utilities.")

break

if delete\_choice == 4:

if "Transport" in transactionsB:

print("Transport Transactions:")

for idx, transaction in enumerate(transactionsB["Transport"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to delete: "))

if 1 <= idx <= len(transactionsB["Groceries"]):

del transactionsB["Transport"][idx - 1]

save\_transactions()

print("Transaction deleted successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Transport.")

break

if delete\_choice == 5:

if "Other Expense" in transactionsB:

print("Other Expense Transactions:")

for idx, transaction in enumerate(transactionsB["Other Expense"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to delete: "))

if 1 <= idx <= len(transactionsB["Groceries"]):

del transactionsB["Other Expense"][idx - 1]

save\_transactions()

print("Transaction deleted successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Other Expense.")

break

if delete\_choice == 6:

if "Other Income" in transactionsB:

print("Other Income Transactions:")

for idx, transaction in enumerate(transactionsB["Other Income"], start=1):

print(f"{idx}. Date: {transaction['date']}, Amount: {transaction['amount']}")

try:

idx = int(input("Enter the index of the transaction you want to delete: "))

if 1 <= idx <= len(transactionsB["Other Income"]):

del transactionsB["Other Income"][idx - 1]

save\_transactions()

print("Transaction deleted successfully!")

break

else:

print("Invalid index. Please enter a valid index.")

continue

except ValueError:

print("Invalid input. Please enter a valid index.")

continue

else:

print("No transactions found for Other Income.")

break

else:

print("Please input valid range number")

def display\_summary():

global transactionsB

total\_income = 0

total\_expense = 0

# Calculate total income

if "Salary" in transactionsB:

for transaction in transactionsB["Salary"]:

total\_income += transaction["amount"]

if "Other Income" in transactionsB:

for transaction in transactionsB["Other Income"]:

total\_income += transaction["amount"]

# Calculate total expense

if "Groceries" in transactionsB:

for transaction in transactionsB["Groceries"]:

total\_expense += transaction["amount"]

if "Utilities" in transactionsB:

for transaction in transactionsB["Utilities"]:

total\_expense += transaction["amount"]

if "Transport" in transactionsB:

for transaction in transactionsB["Transport"]:

total\_expense += transaction["amount"]

if "Other Expense" in transactionsB:

for transaction in transactionsB["Other Expense"]:

total\_expense += transaction["amount"]

# Print summary

print("\n--- Summary ---")

print("Total Income:", total\_income)

print("Total Expense:", total\_expense)

print("Net Balance:", total\_income - total\_expense)

def main\_menu():

load\_transactions()

while True:

print("1. Add transaction")

print("2. View transactions")

print("3. Update transaction")

print("4. Delete transaction")

print("5. Display summary")

print("6. Exit")

try:

choice = int(input("\nPlease enter your choice : "))

except ValueError:

print("\nInvalid input! \nPlease enter valid range numeric value.")

if choice == 1:

add\_transaction()

elif choice == 2:

view\_transactions()

elif choice == 3:

update\_transaction()

elif choice == 4:

delete\_transaction()

elif choice == 5:

display\_summary()

elif choice == 6:

print("Exiting program.")

break

else:

print("\nInvalid input!\nPlease enter valid range numeric number.")

if \_\_name\_\_ == "\_\_main\_\_":

main\_menu()