WEEK 2

VAULTOFCODES

PYTHON PROGRAMMING INTERNSHIP

NAME: HIMANSHU KUNDAN TAPDE

Mini Project: Build a Personal Expense Tracker that allows users to log their daily expenses, view summaries, and track their spending over time.

ALGORITHM:

Program Overview:

1. Initialize the Expense class.
2. Load existing expense data from the **expense.json** file.
3. Display a main menu for user interaction with various options.
4. Repeat actions based on user selection until the user chooses to exit.

Steps:

1) **Start Program:**
   - Initialize the Expense class.
   - Call the **load_expenses** method to load any existing data from the **expense.json** file.

2) **Load Existing Data:**
   - Check if expense.json file exists and is not empty:
   - If it exists, read and load the data into the expenseList attribute.

- If it doesn't exist or is empty, create an empty dictionary for expenseList.

3) **Display Main Menu:**
   - Present the user with a list of options to choose from:

     - Add Expense
     - View Category-Wise Expenses
     - View Total Expense of a Day
     - Display All Expenses
     - Plot Expenses by Category
     - Save and Exit

4) **User Input for Menu Selection:**
   - Prompt the user to select one of the options from the menu.

5) **If User Chooses "Add Expense":**
   - Call the addExpense method.
   - Prompt the user to enter the date in the format dd-mm-yyyy.
   - Validate the entered date format.
   - Initialize an empty dictionary dic to store expense details.
   - Ask the user if they want to add expense items (Y/N).

6) **If user chooses 'Y':**
   - Display available expense categories (e.g., food, transport, entertainment).
   - Ask the user to
   - Select a category.
   - Enter a description of the expense.
   - Enter the amount spent.
   - Store the expense details in the dictionary.
   - Check if the date already exists in expenseList:
   - If it does, update the existing entry.
   - If it doesn't, create a new entry for that date.
   - Print a confirmation message that the expense has been added successfully.

7) **If User Chooses "View Category-Wise Expenses":**
   - Call the calculate ExpensesCategorywise method.
   - Iterate through expenseList

- Calculate and print the total expenses for each category (e.g., food, transport) on each date.

8) **If User Chooses "View Total Expense of a Day":**
   - Call the calculate TotalExpensesADay method.
   - Prompt the user to enter a date in dd-mm-yyyy format.
   - Validate the entered date format.
   - Check if there are any expenses for the entered date.
   - If there are expenses, calculate the total amount spent and display it.
   - If no expenses are found, inform the user that there are no expenses recorded for that date.

9) **If User Chooses "Display All Expenses":**
   - Call the display AllExpensesTillNow method.
   - Check if any expenses have been recorded:
   - If there are recorded expenses, display them categorized by date and category.
   - If there are no expenses, inform the user.

10) **If User Chooses "Plot Expenses by Category":**
   - Call the plot_expenses_by_category method.
   - Calculate total expenses for each category.
   - Use matplotlib to create a bar chart that displays the total expenses by category (e.g., food, transport, entertainment).

11) **If User Chooses "Save and Exit":**
   - Call the save_expenses method to save the current expenseList data into expense.json in JSON format.
   - Exit the program.

12) **Repeat or Exit:**
   - After completing an action (adding an expense, viewing expenses, etc.), return to the main menu.
   - Repeat steps until the user chooses Save and Exit to finish the program.

CODE:

```
import sys
import json
import datetime
import os
import matplotlib.pyplot as plt

class Expense:

    def __init__(self, expenseList=None):
        if expenseList is None:
            self.expenseList = {}
        else:
            self.expenseList = expenseList

    # Load the data from the json file
    def load_expenses(self):
        def is_file_empty(file_path):
            try:
                return os.path.getsize(file_path) == 0
            except FileNotFoundError:
                return True  # If the file doesn't exist, it's considered empty

        if is_file_empty('expense.json'):
            print("No expense record found!!")
```

```
    else:
        try:
            with open('expense.json', 'r') as file:
                print("Status: Data is Loaded successfully!!")
                self.expenseList = json.load(file)
        except FileNotFoundError:
            print("File not found. Starting with an empty expense list.")



# Add an expense with a date
def addExpense(self):
    while True:
        try:
            date = input("Enter the date (dd-mm-yyyy): ")
            datetime.datetime.strptime(date, '%d-%m-%Y')  # Validate date format
            break
        except ValueError:
            print("Invalid Date Format!!")

    dic = {}

    while True:
        ch = input("Do you want to add items (Y/N): ").strip().upper()

        categories = [
            "Education", "Food & Dining", "Transportation", "Travel", "Healthcare",
            "Entertainment", "Housing", "Shopping", "Others"
        ]
```

```
    if ch == 'Y':
        print("\nThe Categories of Expenses:")
        for val in range(len(categories)):
            print(f"{val + 1}. {categories[val]}")
        print()

        try:
            expenseTypeName = int(input("Enter the Expense Category: "))
            itemName = input("Enter the Description: ").strip()
            itemPrice = float(input("Enter the Amount Spent: ").strip())
        except (ValueError, TypeError):
            print("\nStatus: Invalid Choice Or Wrong Data Entered!!")
            continue

        if categories[expenseTypeName - 1] not in dic:
            dic[categories[expenseTypeName - 1]] = []

        dic[categories[expenseTypeName - 1]].append(itemName)
        dic[categories[expenseTypeName - 1]].append(itemPrice)

        print(f"Status: The Expense on {date} is added successfully!")

    elif ch == 'N':
        break
    else:
        print("\nInvalid input, please enter 'Y' or 'N'.")
        continue

# If the date already exists in the dictionary
```

```
        if date in self.expenseList:
            for key, value in dic.items():
                if key in self.expenseList[date]:
                    self.expenseList[date][key].extend(value)
                else:
                    self.expenseList[date][key] = value
        else:
            self.expenseList.update({date: dic})


    # Calculate category-wise expenses for each day
    def calculateExpensesCategorywise(self):
        for dates, expenses in self.expenseList.items():
            print(f"-----------------------------------------------------\nDate: {dates}\n-----------------
-----------------------------------")
            for item, pricelist in expenses.items():
                total = sum(pricelist[i] for i in range(1, len(pricelist), 2))
                print(f"Total Amount Spent on {item}: Rs. {total}")


    # Calculate total expense cost of a day
    def calculateTotalExpensesADay(self):
        while True:
            try:
                inp_date = input("\nEnter the date that you want to view total amount spent:
")
                datetime.datetime.strptime(inp_date, '%d-%m-%Y')  # Validate date format
                break
            except ValueError:
                print("Invalid Date Format!!")


        total = 0
```

```
        for dates, expenses in self.expenseList.items():
            if dates == inp_date:
                for pricelist in expenses.values():
                    total += sum(pricelist[i] for i in range(1, len(pricelist), 2))
                print(f"\nTotal Amount Spent on {inp_date}: Rs. {total}")
                break
        else:
            print(f"No Expenses found for the date: {inp_date} !!")

    # Display all expenses till now
    def displayAllExpensesTillNow(self):
        if not self.expenseList:
            print("No expenses recorded yet.")
            return

        for dates, expenses in self.expenseList.items():
            print(f"-------------------------------------------------------\nDate: {dates}\n-----------------
-----------------------------------")
            for category, items in expenses.items():
                print(f"{category}:")
                for i in range(0, len(items), 2):
                    print(f"  - {items[i]}: Rs. {items[i + 1]}")
            print("-------------------------------------------------------")

    # Save expenses to a JSON file
    def save_expenses(self):
        try:
            with open('expense.json', 'w') as file:
                json.dump(self.expenseList, file, indent=4)
                print("Expenses saved successfully!")
```

```
        except IOError:
            print("Error saving expenses to file.")

    # Plot a graph for expenses by category
    def plot_expenses_by_category(self):
        category_totals = {}
        for dates, expenses in self.expenseList.items():
            for category, items in expenses.items():
                total = sum(items[i] for i in range(1, len(items), 2))
                if category not in category_totals:
                    category_totals[category] = total
                else:
                    category_totals[category] += total

        categories = list(category_totals.keys())
        amounts = list(category_totals.values())

        plt.bar(categories, amounts)
        plt.xlabel('Categories')
        plt.ylabel('Total Expense (Rs.)')
        plt.title('Total Expenses by Category')
        plt.xticks(rotation=45, ha='right')
        plt.tight_layout()
        plt.show()

# Main code to interact with the Expense class
def main():
    expense_tracker = Expense()
    expense_tracker.load_expenses()
```

```
    while True:
        print("\nExpense Tracker Menu:")
        print("1. Add Expense")
        print("2. View Category-Wise Expenses")
        print("3. View Total Expense of a Day")
        print("4. Display All Expenses")
        print("5. Plot Expenses by Category")
        print("6. Save and Exit")

        choice = input("Enter your choice (1-6): ").strip()

        if choice == '1':
            expense_tracker.addExpense()
        elif choice == '2':
            expense_tracker.calculateExpensesCategorywise()
        elif choice == '3':
            expense_tracker.calculateTotalExpensesADay()
        elif choice == '4':
            expense_tracker.displayAllExpensesTillNow()
        elif choice == '5':
            expense_tracker.plot_expenses_by_category()
        elif choice == '6':
            expense_tracker.save_expenses()
            break
        else:
            print("Invalid choice, please try again.")

if __name__ == "__main__":
```

main()

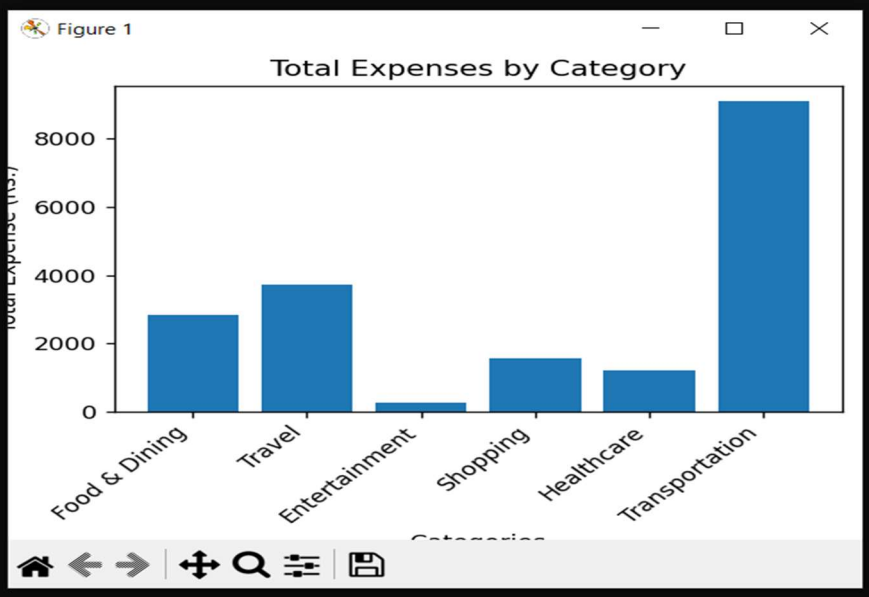
OUTPUT:

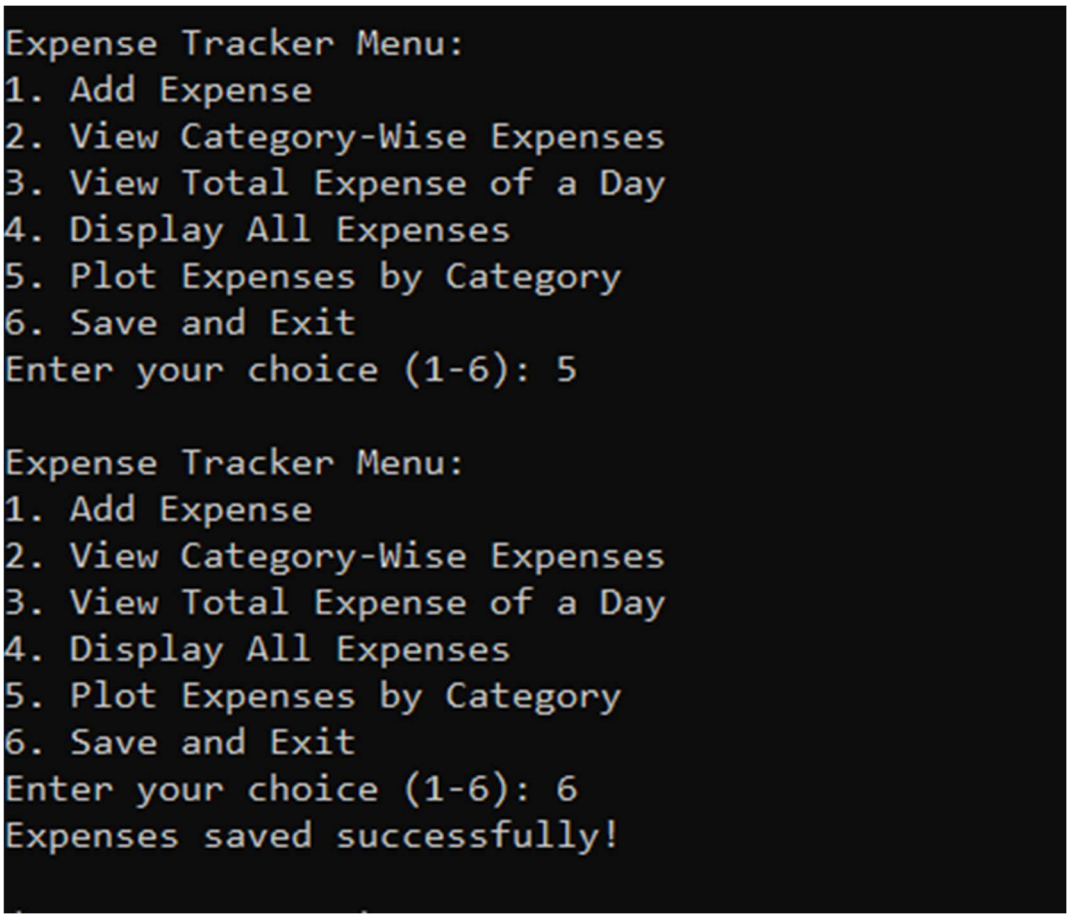
```
(expense_tracker) C:\Users\Janvi>python expense_tracker.py
Status: Data is Loaded successfully!!

Expense Tracker Menu:
1. Add Expense
2. View Category-Wise Expenses
3. View Total Expense of a Day
4. Display All Expenses
5. Plot Expenses by Category
6. Save and Exit
Enter your choice (1-6): 1
Enter the date (dd-mm-yyyy): 25-03-2025
Do you want to add items (Y/N): Y

The Categories of Expenses:
1. Education
2. Food & Dining
3. Transportation
4. Travel
5. Healthcare
6. Entertainment
7. Housing
8. Shopping
9. Others

Enter the Expense Category: 2
Enter the Description: pizza
Enter the Amount Spent: 400
Status: The Expense on 25-03-2025 is added successfully!
Do you want to add items (Y/N): N
```

```
Expense Tracker Menu:
1. Add Expense
2. View Category-Wise Expenses
3. View Total Expense of a Day
4. Display All Expenses
5. Plot Expenses by Category
6. Save and Exit
Enter your choice (1-6): 2
--------------------------------------------------
Date: 03-05-2024
--------------------------------------------------
Total Amount Spent on Food & Dining: Rs. 630.0
Total Amount Spent on Travel: Rs. 210.0
--------------------------------------------------
Date: 20-07-2024
--------------------------------------------------
Total Amount Spent on Entertainment: Rs. 250.0
--------------------------------------------------
Date: 15-07-2024
--------------------------------------------------
Total Amount Spent on Travel: Rs. 3500.0
Total Amount Spent on Shopping: Rs. 1570.0
--------------------------------------------------
Date: 05-05-2024
--------------------------------------------------
Total Amount Spent on Healthcare: Rs. 1200.0
Total Amount Spent on Food & Dining: Rs. 506.0
--------------------------------------------------
Date: 09-04-2024
--------------------------------------------------
Total Amount Spent on Transportation: Rs. 9100.0
Total Amount Spent on Food & Dining: Rs. 1300.0
--------------------------------------------------
Date: 25-03-2025
--------------------------------------------------
Total Amount Spent on Food & Dining: Rs. 400.0
```

```
Expense Tracker Menu:
1. Add Expense
2. View Category-Wise Expenses
3. View Total Expense of a Day
4. Display All Expenses
5. Plot Expenses by Category
6. Save and Exit
Enter your choice (1-6): 3

Enter the date that you want to view total amount spent:  09-04-2024
Invalid Date Format!!

Enter the date that you want to view total amount spent: 03-05-2024

Total Amount Spent on 03-05-2024: Rs. 840.0

Expense Tracker Menu:
1. Add Expense
2. View Category-Wise Expenses
3. View Total Expense of a Day
4. Display All Expenses
5. Plot Expenses by Category
6. Save and Exit
Enter your choice (1-6): 4
------------------------------------------------------
Date: 03-05-2024
------------------------------------------------------
Food & Dining:
  - Pizza: Rs. 350.0
  - Biriyani: Rs. 280.0
Travel:
  - South City Mall: Rs. 210.0
------------------------------------------------------
------------------------------------------------------
Date: 20-07-2024
------------------------------------------------------
Entertainment:
  - Movie: Rs. 250.0
------------------------------------------------------
------------------------------------------------------
Date: 15-07-2024
------------------------------------------------------
Travel:
  - Goa: Rs. 3500.0
Shopping:
  - Lake Mall: Rs. 1570.0
------------------------------------------------------
------------------------------------------------------
Date: 05-05-2024
------------------------------------------------------
```

```
Expense Tracker Menu:
1. Add Expense
2. View Category-Wise Expenses
3. View Total Expense of a Day
4. Display All Expenses
5. Plot Expenses by Category
6. Save and Exit
Enter your choice (1-6): 5
```





Github Link for Code:

https://github.com/Himanshu431-coder/VaultofCodes-Personal-Expense-Tracker