SplitSmart App

23BCE148 Krish Bhoraniya
23BCE149 Krish Patel
Object Oriented Programming (2CS502)

1. Introduction

The SplitSmartApp Java program is an expense-sharing system designed to help groups of users record expenses, split amounts among themselves, and calculate net settlements for each user. This project, inspired by platforms like Splitwise, enables users to add expense records, specify who paid, how the cost is divided, and provides an output file detailing each user's balance to settle with others.

2. Class Structure

The program includes three primary classes: User, Expense, and SplitSmartApp. Additionally, a nested UserBalance class is used within SplitSmartApp to manage user balances in the final settlement phase.

2.1 User Class

The User class models a user with the following attributes:

- name: Name of the user.
- amt: Initial amount, defaulted to 0.0f.
- upi: UPI address of the user.

2.2 Expense Class

The Expense class inherits from User and models an expense record with these attributes:

- description: Describes the expense.

- amount: Total amount of the expense.
- paidByUsers: List of users who contributed to the payment.
- paidAmounts: Amounts contributed by each user in paidByUsers.
- splitEqually: Boolean to indicate whether the amount is split equally.
- splitAmongUsers: List of users among whom the expense is split.
- splitAmounts: Amounts each user owes as part of the split.

2.3 SplitSmartApp Class

This is the main class, containing static fields and methods to manage the application workflow:

Attributes:

- expenses: List of Expense instances, each representing an expense record.
- sharingTitle: The title for the expense-sharing session.

3. Program Flow

1. **Initialization:**

   Users are prompted to enter a sharing session title and their details (name, UPI).

2. **Expense Entry:**

   The program allows users to enter multiple expenses, specifying:

   - Description and amount of the expense.
   - Paid by information (single or multiple users).
   - Split mode (equally or custom amounts).

3. **Expense Processing:**
   - The program processes contributions and splits the amounts.
   - Each expense is recorded in the expenses list.

4. **Output Generation:**
   - generateFinalOutput calculates the net balance for each user, distinguishing creditors and debtors.

- Debtors are paired with creditors for the minimum balance, and the program writes a summary to a text file in a user-friendly format.

4. Sample Output

```
------------------------------------------------
Main menu:
1. Start New Sharing.
2. End.
Enter input: 1
------------------------------------------------
Title : FriendsTrip
Enter the number of users: 4
Users:
Enter details for user 1:
Enter user name: Alice
Enter user's UPI address: alice@upi

Enter details for user 2:
Enter user name: Bob
Enter user's UPI address: bob@upi

Enter details for user 3:
Enter user name: Charlie
Enter user's UPI address: charlie@upi

Enter details for user 4:
Enter user name: Diana
Enter user's UPI address: diana@upi
```

```
1. Add new Expense.
2. Give final output.
Enter input: 1

Description: Dinner at Beach Cafe
Enter amount: 2000
Paid by Single/Multiple (S/M): S
Select the user who paid the amount:
1. Alice
2. Bob
3. Charlie
4. Diana
Enter the index of the user who paid the amount: 1
Split equally (Y/N): Y
Split amongst how many users: 4
Expense added successfully!

1. Add new Expense.
2. Give final output.
Enter input: 1

Description: Boat Ride
Enter amount: 3000
Paid by Single/Multiple (S/M): M
Paid by how many users: 2
1. Alice
2. Bob
3. Charlie
4. Diana
Enter the indices of the users who paid the amount:
2 3
Enter the amount paid by Bob: 1000
Enter the amount paid by Charlie: 2000
Split equally (Y/N): N
Enter amount for each user:
1. Alice : 0
2. Bob : 1500
3. Charlie : 500
4. Diana : 1000
Expense added successfully!
```

```
1. Add new Expense.
2. Give final output.
Enter input: 1

Description: Lunch at Hilltop Restaurant
Enter amount: 1200
Paid by Single/Multiple (S/M): S
Select the user who paid the amount:
1. Alice
2. Bob
3. Charlie
4. Diana
Enter the index of the user who paid the amount: 4
Split equally (Y/N): Y
Split amongst how many users: 3
Enter the indices of the users amongst whom the amount must be split:
1. Alice
2. Bob
3. Charlie
4. Diana
1 2 4
Expense added successfully!
```

```
1. Add new Expense.
2. Give final output.
Enter input: 1

Description: Shopping Expenses
Enter amount: 800
Paid by Single/Multiple (S/M): M
Paid by how many users: 2
1. Alice
2. Bob
3. Charlie
4. Diana
Enter the indices of the users who paid the amount:
1 2
Enter the amount paid by Alice: 500
Enter the amount paid by Bob: 300
Split equally (Y/N): N
Enter amount for each user:
1. Alice : 400
2. Bob : 0
3. Charlie : 200
4. Diana : 200
Expense added successfully!

1. Add new Expense.
2. Give final output.
Enter input: 2

Final output has been written to FriendsTrip.txt


------------------------------------------------
Main menu:
1. Start New Sharing.
2. End.
Enter input: 2
PS E:\Coding\Java>
```

FriendsTrip.txt

```
FriendsTrip

Users:
• Alice (alice@upi)
• Bob (bob@upi)
• Charlie (charlie@upi)
• Diana (diana@upi)

Expense History:
1. Description: Dinner at Beach Cafe
   Amount: 2000.0
   Paid by: Alice
   Split:
      - Alice-- 500.0/-
      - Bob-- 500.0/-
      - Charlie-- 500.0/-
      - Diana-- 500.0/-
---------------------------------
2. Description: Boat Ride
   Amount: 3000.0
   Paid by: Bob(1000.0), Charlie(2000.0)
   Split:
      - Alice-- 0.0/-
      - Bob-- 1500.0/-
      - Charlie-- 500.0/-
      - Diana-- 1000.0/-
---------------------------------
3. Description: Lunch at Hilltop Restaurant
   Amount: 1200.0
   Paid by: Diana
   Split:
      - Alice-- 400.0/-
      - Bob-- 400.0/-
      - Diana-- 400.0/-
---------------------------------
4. Description: Shopping Expenses
   Amount: 800.0
   Paid by: Alice(500.0), Bob(300.0)
   Split:
      - Alice-- 400.0/-
      - Bob-- 0.0/-
      - Charlie-- 200.0/-
      - Diana-- 200.0/-
---------------------------------

Settlement Summary:
Bob owes Alice -- 1100.0/-
Diana owes Alice -- 100.0/-
Diana owes Charlie -- 800.0/-

```

6. Conclusion

The SplitSmartApp provides a simple and efficient solution for managing group expenses. By leveraging object-oriented principles and using file operations for persistence, this project showcases an effective implementation of inheritance, encapsulation, and aggregation. The final output, stored in a file, allows users to retain a clear record of transactions and outstanding amounts, making it useful for collaborative group expense management.