Expense Manager Project Report

A Java-Based Application for Efficient Expense Tracking and Management

---

Developed By:

INDRAGANTI KRISHNA MOHANA VARSHITHA
B.Tech in Electronics & Communication Engineering
Guru Nanak Institutions Technical Campus, Hyderabad

Project Supervisor:

Independent Project

Project Duration:

[13/09/24] – [21/09/24]

Date of Completion:

[22/09/2024]

Table of Contents

Expense Manager

1. Introduction

The **Expense Manager** is a comprehensive, Java-based application designed to help individuals and groups manage and track shared expenses effectively. In everyday life, whether it's a group of friends splitting a dinner bill, roommates sharing rent, or colleagues organizing a work trip, managing shared expenses can be a tedious and complex task. The Expense Manager simplifies this process by offering a user-friendly system to record, track, and manage shared costs.

With this application, users can create detailed expense records, specify the total cost, and choose from various methods to split the cost among participants. The three split methods provided by the system are **Equal Split**, **Percentage Split**, and **Custom Split**, giving users the flexibility to choose the method that suits their situation best.

Equal Split divides the total amount equally among the participants, making it ideal for scenarios where everyone contributes the same amount. **Percentage Split** allows each user to pay a predefined percentage of the total expense, which is useful when different participants agree to contribute differently based on their ability or agreement. **Custom Split** gives users the option to enter specific amounts for each participant, offering the highest level of flexibility for complex scenarios.

This application is designed not only to split expenses but also to maintain a running tally of what each participant owes or is owed. Users can track their balances and easily determine if they need to pay or receive money from others. The system’s ability to dynamically calculate and update balances ensures that no one is left out, and everyone contributes their fair share.

The **Expense Manager** project is built using **Object-Oriented Programming (OOP)** principles, making the code modular, easy to understand, and maintain. Through the use of classes, inheritance, and polymorphism, the program can be extended and scaled easily to include more features in the future. By employing a clear and systematic approach to organizing user data and

expenses, the application offers a high level of usability and accuracy in expense tracking.

This project is not only a practical tool for managing expenses but also an excellent learning experience for anyone interested in Java programming, object-oriented principles, and software development. The application's simplicity, flexibility, and extensibility make it suitable for personal use as well as small group settings, making it a valuable tool for real-world applications.

2. Abstract

The **Expense Manager** is a comprehensive Java-based application that employs object-oriented programming (OOP) principles to help users manage shared expenses effectively. The project provides a structured approach to creating and managing users, recording and splitting expenses, and calculating individual balances based on various splitting strategies. The core objective of the application is to simplify the process of managing group or shared expenses, providing clarity on who owes what and how costs should be divided.

The system supports three primary expense splitting strategies to suit different scenarios:

1. **Equal Split**: This method divides the total expense evenly among all participants. It is a straightforward approach ideal for situations where all participants agree to share the cost equally, such as in group outings or collective purchases.
2. **Percentage Split**: This method allows for a more customized approach by assigning predefined percentages of the total expense to each participant. This is useful in scenarios where different participants are expected to contribute in different proportions. For example, in a business setting where some participants have more significant shares or responsibilities, they can pay a larger portion of the expense.
3. **Custom Split**: This method offers the highest level of flexibility. Users can specify exactly how much each participant should pay, independent of percentages or equal shares. This option comes in handy when expenses vary significantly among participants or when some individuals may be paying for more or fewer items than others.

The **Expense Manager** application enables dynamic user creation, allowing users to register and manage their accounts easily. Each user can create expenses, which can then be split among multiple participants according to their preferred strategy. The system calculates each user's balance after an

expense is recorded, keeping track of who owes whom and how much, based on the splits.

The application is designed with scalability in mind, making it adaptable for various use cases, including shared household expenses, group travel costs, joint business expenses, and more. By integrating these flexible splitting strategies, the project aims to address a wide range of expense-sharing scenarios, ensuring that everyone's contributions are accurately tracked and calculated.

The use of object-oriented programming principles, such as inheritance, polymorphism, encapsulation, and abstraction, ensures that the application is modular, extensible, and easy to maintain. Each component, such as users, expenses, and split strategies, is encapsulated in its respective class, making the system robust and easy to update or extend in the future.

3. Features / Requirements

Features

- **User Management**: Create and manage users with unique IDs, names, and emails.
- **Expense Management**: Create and track expenses, including splitting costs among multiple users.
- **Flexible Split Methods**:
    - **Equal Split**: Splits the total cost evenly among participants.
    - **Percentage Split**: Allows specifying percentages for each user.
    - **Custom Split**: Lets users specify exactly how much each person will pay.
- **Balance Calculation**: Displays the current balance for each user based on the expenses and splits.

Requirements

- **Java Development Kit (JDK)** version 8 or higher
- **IDE** (Eclipse, IntelliJ IDEA, or any Java IDE)
- Basic knowledge of Java and object-oriented programming concepts.

4. Existing Method & Proposed Method

Existing Method

Typically, expense management tools in existing applications offer simple functionality for splitting costs between two users or groups, often with minimal user interaction. These applications might only support one method of cost splitting (e.g., equal split), making them inflexible for more complex scenarios.

Proposed Method

The **Expense Manager** proposes a more comprehensive solution by:

- Supporting **three types of splits**: Equal, Percentage, and Custom.
- Using **object-oriented principles** to manage users, expenses, and split methods dynamically.
- Enabling users to be created and managed easily, allowing for better user interaction.
- Offering flexibility to choose how the expenses are split, making the tool suitable for a variety of scenarios (e.g., group outings, joint bills, shared rent, etc.).

5. File Management and OOP Concepts

File Overview

The project consists of several Java files that interact with each other to perform operations. Below is a breakdown of each file and its purpose:

1. User.java

- **Purpose**: Represents an individual user in the system.
- **OOP Concepts**:
    - **Encapsulation**: The class encapsulates user details like ID, name, email, and balance. Getter and setter methods allow access to these details.
    - **Constructors**: The constructor is used to initialize the user object with a unique ID, name, email, and balance.
    - **Method**: toString() is overridden to provide a readable format of user details.

2. Expense.java

- **Purpose**: A base class to represent an expense.
- **OOP Concepts**:
    - **Abstraction**: This class is abstract and does not implement specific splitting methods, leaving the details to the subclasses.
    - **Inheritance**: The subclasses (EqualSplit, PercentageSplit, and CustomSplit) inherit from this base class and implement specific splitting methods.
    - **Method**: createExpense() allows for creating an expense and adding participants, but the actual splitting logic is defined in the subclasses.

3. EqualSplit.java, PercentageSplit.java, CustomSplit.java

- **Purpose**: These subclasses implement different strategies for splitting an expense.
- **OOP Concepts**:
    - **Polymorphism**: These classes implement the splitExpense() method, each in its own way. The correct method is invoked depending on the user's selection.
    - **Inheritance**: Each of these classes inherits from the Expense base class and implements the splitExpense() method to split the expense according to the respective strategy.

4. ExpenseManager.java

- **Purpose**: The main class that runs the application. It contains the main menu and user interface.
- **OOP Concepts**:
    - **Composition**: It contains instances of User and Expense to manage and operate on user data and expenses.
    - **Control Flow**: This class uses conditionals and loops to provide the user with options to create users, create expenses, view balances, or exit.

OOP Concepts in Action:

- **Abstraction**: The use of an abstract class (Expense.java) hides the complexity of splitting logic and only exposes the necessary methods to the user.
- **Encapsulation**: Each class keeps its data (such as user details or expense amounts) private and provides public methods to access or modify that data.
- **Inheritance**: Different types of split strategies (EqualSplit, PercentageSplit, CustomSplit) inherit from a common Expense base class, enabling code reuse and easy extension.

- **Polymorphism**: The splitExpense() method behaves differently based on the subclass (e.g., EqualSplit vs. CustomSplit), allowing flexible behavior without altering the structure.

6. Example Output

Below is an example of how the program operates during execution:

Expense Manager Menu:

1. Create User

2. Create Expense

3. Display Balances

4. Exit

Enter choice: 1

Enter User ID: 1

Enter User Name: Charan

Enter User Email: charan@email.com

User created successfully!

Expense Manager Menu:

1. Create User

2. Create Expense

3. Display Balances

4. Exit

Enter choice: 2

Enter amount: 100

Select Split Method:

1. Equal Split

2. Percentage Split

3. Custom Split

Enter choice: 1

Select participants:

1. Charan

Enter amount for Charan: 50

Enter amount for another participant: 50

Expense created successfully!

Expense Manager Menu:

1. Create User

2. Create Expense

3. Display Balances

4. Exit

Enter choice: 3

Balances:

Charan: 0.0

7. Future Prospects

- **Mobile Application**: The project can be extended to a mobile app to make it more accessible and convenient for users.

- **Advanced Split Algorithms**: Implement more advanced splitting algorithms (e.g., weighted average, bill sharing based on income).

- **Database Integration**: The project can be enhanced by integrating a database (like MySQL or MongoDB) for persistent storage of users, expenses, and transactions.

- **User Authentication**: Add a login system where users must authenticate before accessing the expense manager.

8. Conclusion

The **Expense Manager** project is a practical and useful application that allows multiple users to efficiently manage and split shared expenses. By utilizing object-oriented programming principles, it offers flexibility, scalability, and clarity. With the ability to manage multiple types of splits and track user balances, it serves as an excellent tool for personal or group expense management.

The project can be further expanded with more advanced features such as mobile access, database integration, and user authentication to enhance its functionality and usability.