

SMART EXPENSE TRACKER

A MINI PROJECT REPORT

Submitted by

AYUSHMAN RAJ (22BCA10021)

in partial fulfillment for the award of the degree of

BACHELOR OF COMPUTER APPLICATION

UNIVERSITY INSTITUTE OF COMPUTING



**CHANDIGARH
UNIVERSITY**

Discover. Learn. Empower.

APRIL 2025

Abstract

In today's fast-paced and economically driven world, managing personal finances efficiently is a skill that holds paramount importance. As individuals become more financially independent, the necessity for tools that support proper expense management becomes increasingly critical. With the advent of technology, digital solutions are not only replacing traditional pen-and-paper methods of recording expenditures but are also offering intuitive, automated, and intelligent ways to handle finances. The **Smart Expense Tracker** is one such digital solution designed as a mini-project using **Java**, **Swing**, and **MySQL**—targeting users who seek an efficient and user-friendly platform to monitor their day-to-day expenses. The project provides a basic but robust personal finance system that emphasizes ease of use, security, and functionality.

The central objective of this application is to empower users to log, monitor, and analyze their financial transactions in a simple desktop environment. Upon launching the application, users can register themselves or log into an existing account. Post-authentication, users gain access to a personalized dashboard where they can perform three main tasks: add a new expense, view all recorded expenses, or delete previous entries. Each expense record consists of a category (e.g., food, travel, shopping), an amount, a date of expenditure, and the associated user ID, ensuring all entries are securely tied to individual accounts. This eliminates the risk of unauthorized access and enables secure multi-user handling.

At the heart of this application lies its use of **Java Swing**, a part of the Java Foundation Classes (JFC), which provides a powerful and versatile GUI framework for building interactive desktop applications. Swing's lightweight components enable the development of a responsive interface without compromising on functionality. The graphical layout includes components such as buttons, text fields, labels, and tables that offer intuitive interaction for users. The Swing GUI is designed in such a way that users—even those with limited technical experience—can easily navigate the system. Every button click is linked to an action listener, which, in turn, interacts with the backend logic defined in separate service classes.

The backend functionality is powered by **MySQL**, a relational database management system known for its reliability and performance. All user data and expense records are persistently stored in a MySQL database using **JDBC (Java Database Connectivity)**. JDBC acts as a bridge between the Java application and the MySQL server, allowing smooth and efficient data operations such as insertion, retrieval, and deletion. The database schema includes two major tables: users and expenses, linked via foreign key constraints to ensure data consistency and integrity. This modular architecture separates data management from UI interaction, following the MVC (Model-View-Controller) design principles.

Security is another key consideration in the Smart Expense Tracker. Every user has a unique username and password, stored securely in the database. Although this mini-project does not yet implement advanced encryption methods, the groundwork has been laid to support secure hashing mechanisms in future iterations. The login and registration system ensures that expense data remains private and user-specific, enhancing trust and data confidentiality.

The **design** of the application follows a modular and scalable approach. The source code is organized into distinct packages: models for defining data structures, services for business logic, ui for user interface components, and database for handling database connections. This separation of concerns improves maintainability and allows future enhancements to be implemented without significant overhauls. For instance, adding functionality such as budget alerts or data visualization would only require extending the current services and UI components.

From a developmental perspective, the Smart Expense Tracker project provides students and developers with valuable insights into full-stack desktop application development. It demonstrates how GUI-based Java applications can seamlessly integrate with SQL databases to deliver real-world functionality. The project also reinforces key programming concepts such as object-oriented design, exception handling, event-driven programming, and CRUD (Create, Read, Update, Delete) operations. Moreover, by working with Java Swing and JDBC, developers gain experience in designing software systems that are responsive, data-driven, and user-centric.

In terms of practical usability, this application caters to anyone seeking a digital record-keeping tool for personal finances. Users no longer need to depend on memory or external tools like spreadsheets, which can become cumbersome over time. Instead, they have at their disposal a neatly structured application that keeps their financial information organized and readily accessible. Whether it's a student trying to track monthly allowances or a working professional managing bills and subscriptions, the Smart Expense Tracker can adapt to diverse financial needs.

Looking ahead, the scope for enhancing this application is substantial. Integration with mobile platforms such as Android or iOS can significantly broaden its accessibility. Adding features such as charts for graphical representation of expenses, category-wise expenditure analysis, and budget setting with alerts can make the application more insightful. Implementation of security practices like password encryption, role-based access, and cloud storage would further align it with modern software development standards.

Introduction

In a digital era characterized by innovation and automation, individuals are becoming increasingly aware of the importance of managing personal finances with accuracy and efficiency. While traditional methods of maintaining expense records—such as handwritten logs or basic spreadsheets—have served their purpose, they are now being rapidly replaced by digital systems that offer better usability, data organization, and security. The Smart Expense Tracker emerges as a solution to this modern requirement. It is a desktop application built using Java, Swing, and MySQL, designed to assist users in monitoring, categorizing, and managing their day-to-day financial expenditures with ease.

The Smart Expense Tracker addresses a common pain point: the challenge of remembering and organizing all personal spending data. Many people face difficulties in tracking their expenses over time, often leading to overspending or misallocation of funds. This software allows users to log their expenses by inputting key information such as category, amount, and date of the transaction. By maintaining a digital record that is both searchable and editable, users can quickly review their financial activities and make informed decisions.

The application's graphical user interface (GUI) is crafted using Java Swing, a lightweight toolkit that enables the creation of rich and interactive desktop interfaces. Swing provides the flexibility to implement features like text fields, buttons, tables, and menus with custom styling. The interface is deliberately designed to be clean and intuitive, ensuring that even users with minimal technical expertise can operate the system without difficulty. Whether the user wants to log a new expense, browse past records, or delete outdated entries, the interface provides a seamless experience.

Behind the interface lies a robust data management system powered by MySQL. This relational database management system stores user credentials and expense data securely. Using JDBC (Java Database Connectivity), the application connects to the MySQL server to perform all backend operations, including insertions, updates, deletions, and queries. JDBC serves as a bridge between Java and SQL, enabling smooth communication between the frontend interface and the backend database. This layered architecture supports data integrity and separation of concerns, which is a vital principle in software engineering.

From a usability standpoint, the application follows a multi-user model, where each user has a unique account. Upon logging in, users are granted access to a dashboard where their personalized data is displayed. Each expense record is linked to the user's unique ID, ensuring that one user's data is not accessible to another. This feature enhances privacy and supports concurrent usage without risk of data leaks or duplication. While the current implementation allows basic authentication with username and password, future versions could enhance security through hashed passwords, two-factor authentication, and session management.

One of the notable highlights of this project is its ability to simplify complex financial tracking through automation and structured data. Users are not required to remember when and where they spent their money; the system keeps a chronological and categorical log for them. Moreover, with the "view expenses" feature, users can analyze their spending habits over a period. This analysis forms the basis for smarter budgeting and helps promote financial discipline.

The Smart Expense Tracker project is built following the Model-View-Controller (MVC) architecture. The model layer defines core data structures like User and Expense, encapsulating data attributes and behaviors. The view layer, constructed using Swing, handles all the graphical elements and user interactions. The controller layer comprises services and event listeners that process user input, validate

data, and communicate with the database. This separation ensures modularity, reusability, and easier maintenance.

The application's modular codebase is divided into various packages for clarity. The ui package contains classes responsible for GUI screens such as login, registration, and the main dashboard. The models package defines data structures, while the services package manages database interactions and business logic. The database package includes the utility class for establishing connections to the MySQL server. Such a structured setup makes it easy to debug, test, and extend the software.

This project also has significant educational value. It offers hands-on experience with key programming concepts like GUI development, event handling, and SQL queries. It familiarizes students with real-world software development practices such as modularization, object-oriented design, and exception handling. Moreover, by working with Java Swing and MySQL, developers enhance their understanding of full-stack development on desktop platforms.

Although this version of the Smart Expense Tracker is designed for desktop environments, the concept is scalable and adaptable. A future extension of this project could involve developing a web-based or mobile version, allowing users to track their expenses from anywhere and on any device. Integration with APIs for automatic bill tracking, QR code scanning, or online payments can significantly enhance the user experience. Additionally, incorporating data visualization tools could help users understand their spending patterns through charts and graphs.

In summary, the Smart Expense Tracker serves as a practical solution to a very real problem. It digitizes the process of financial tracking, brings efficiency into everyday budgeting, and empowers users with better control over their finances. By leveraging Java Swing for the interface and MySQL for data handling, this project not only meets academic criteria but also holds the potential for real-world application. Its user-friendly design, modular structure, and scalable architecture make it a comprehensive example of personal finance management software.

Objective

The main objective of the Smart Expense Tracker project is to offer a streamlined, digital solution for managing personal finances in a desktop environment. In a world where financial discipline is vital for personal and professional success, users often find it challenging to maintain an organized record of their daily expenses. This project seeks to empower individuals with a user-friendly tool to track, manage, and evaluate their financial activities. Below are the detailed objectives of the project:

1. Develop a Reliable Expense Tracking System:

The primary goal is to build a reliable desktop application where users can log and monitor their daily expenses. The system must provide accurate, real-time recording capabilities for every transaction a user wishes to enter.

2. Ensure User-Friendly GUI:

The application must be intuitive and easy to navigate, even for users with minimal technical knowledge. By using Java Swing components, we aim to create a responsive and accessible interface that clearly presents expense records and user operations.

3. Implement Secure Authentication:

A secure login and registration system is essential. Each user should be able to access only their own data, protected by username and password credentials. This prevents data leakage and ensures personal financial information is safe.

4. Integrate with MySQL Database:

All data, including user credentials and expense records, should be stored in a structured and efficient manner in a MySQL relational database. The system should support dynamic SQL operations like insertion, deletion, retrieval, and updates through JDBC connectivity.

5. Enable Core Functionalities:

Users should be able to perform essential tasks such as:

- Adding a new expense (category, amount, date)
- Viewing a list of recorded expenses
- Deleting entries when necessary
- Logging out and securely ending their session

6. Facilitate Personalized Tracking:

The system should associate every expense record with a specific user ID, ensuring personalized data tracking. It helps in creating a private workspace for every individual using the software.

7. **Support Expense Categorization:**

Expenses must be categorized (e.g., food, transport, bills), allowing users to analyze spending habits and better manage their finances.

8. **Build for Modularity and Maintainability:**

The application should follow the MVC (Model-View-Controller) architecture, which divides the project into distinct layers. This ensures clean code, easier debugging, and future scalability.

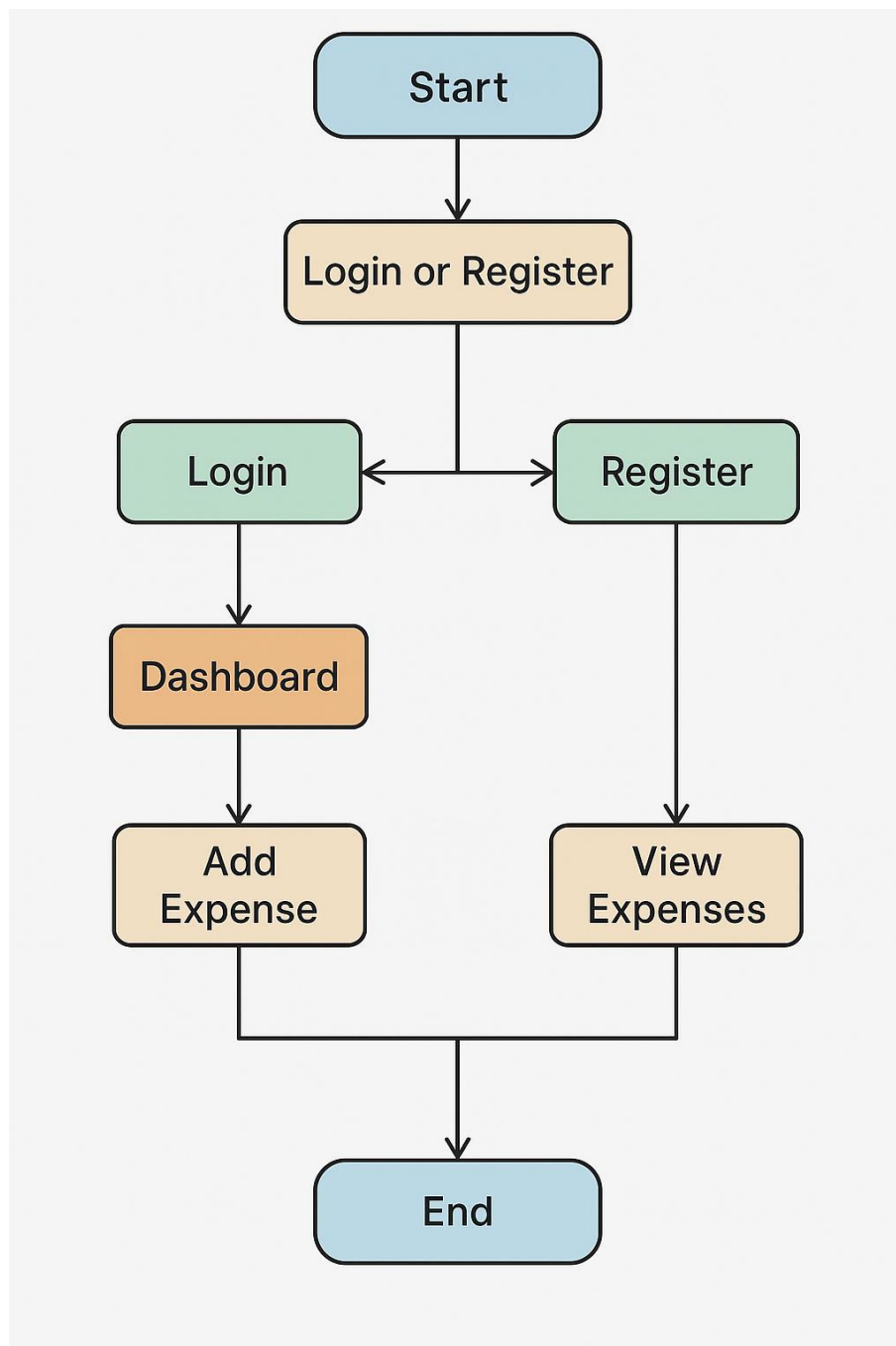
9. **Allow Real-Time Data Operations:**

All operations, including adding, viewing, and deleting records, should be reflected in real-time. This responsiveness will help users maintain up-to-date logs.

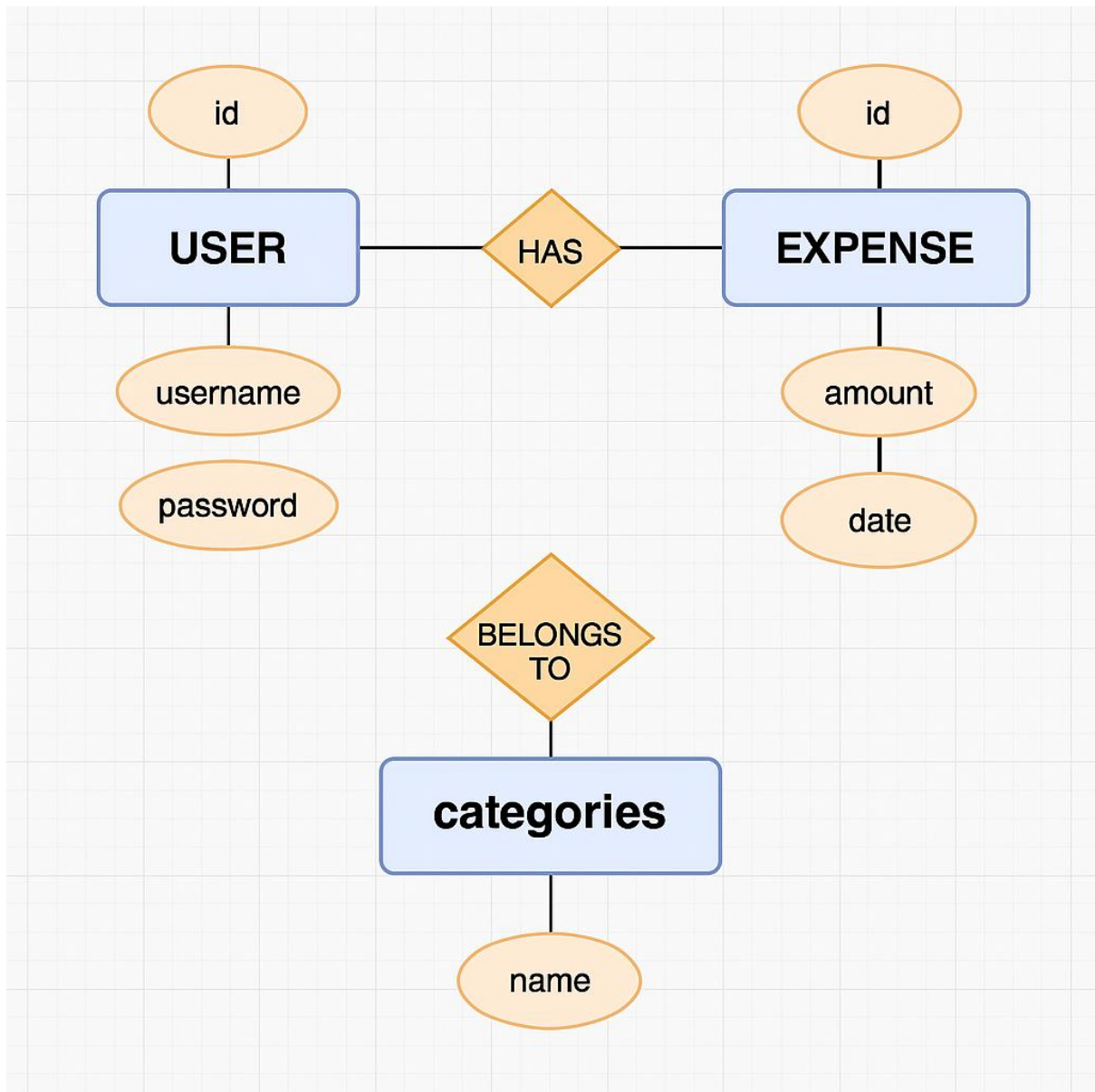
10. **Promote Financial Awareness and Responsibility:**

Ultimately, the project aims to educate users on their spending behavior. With continuous usage, users can identify financial leaks and adapt their habits accordingly. This software becomes a daily companion in achieving financial discipline.

Flow Diagram



ER Diagram



Tools and Technologies Used

For your Smart Expense Tracker mini project, the following tools and technologies would typically be used:

1. Java:
 - The core programming language for developing the application.
 - Used to implement the business logic, manage user actions, and handle interactions with the database.
2. MySQL:
 - A relational database management system (RDBMS) used to store user data and expenses.
 - Used for operations like adding, viewing, and deleting expenses, as well as managing user registrations.
3. Swing (Java GUI):
 - Java Swing is used to design the graphical user interface (GUI) of the application.
 - It provides components like buttons, labels, tables, and text fields for a user-friendly interface.
4. JDBC (Java Database Connectivity):
 - Used to connect the Java application with the MySQL database.
 - Handles queries and updates to the database, such as adding, viewing, and deleting expense records.
5. IntelliJ IDEA:
 - Integrated Development Environment (IDE) for writing, debugging, and compiling the Java application.
6. MySQL Workbench:
 - Used for managing the MySQL database, running queries, and designing the database schema.

Source Code

Databaseconnection.java

```
package database;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DatabaseConnection {
    private static final String URL =
"jdbc:mysql://localhost:3306/SmartExpenseTracker";
    private static final String USER = "root";
    private static final String PASSWORD = "Aditya514@";

    public DatabaseConnection() {
    }

    public static Connection getConnection() {
        Connection conn = null;

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/SmartEx
penseTracker", "root", "Aditya514@");
            System.out.println("Database Connected Successfully!");
        } catch (ClassNotFoundException e) {
            System.err.println("JDBC Driver not found! " +
e.getMessage());
        } catch (SQLException e) {
            System.err.println("Database connection failed! " +
e.getMessage());
        }
    }
}
```

```
    }  
  
    return conn;  
}  
}
```

Expenditure.java

```
package models;
```

```
import java.util.Date;
```

```
public class Expense {  
    private int id;  
    private String category;  
    private double amount;  
    private Date date;  
    private int userId;  
  
    public Expense(int id, String category, double amount, Date date,  
int userId) {  
        this.id = id;  
        this.category = category;  
        this.amount = amount;  
        this.date = date;  
        this.userId = userId;  
    }  
  
    public int getId() { return id; }  
    public String getCategory() { return category; }  
    public double getAmount() { return amount; }  
    public Date getDate() { return date; }  
    public int getUserId() { return userId; }  
  
    public void setId(int id) { this.id = id; }  
    public void setCategory(String category) { this.category =
```

```

category; }
    public void setAmount(double amount) { this.amount = amount; }
    public void setDate(Date date) { this.date = date; }
    public void setUserId(int userId) { this.userId = userId; }
}

```

User.java

```

package models;

```

```

public class User {
    private int id;
    private String username;
    private String password;

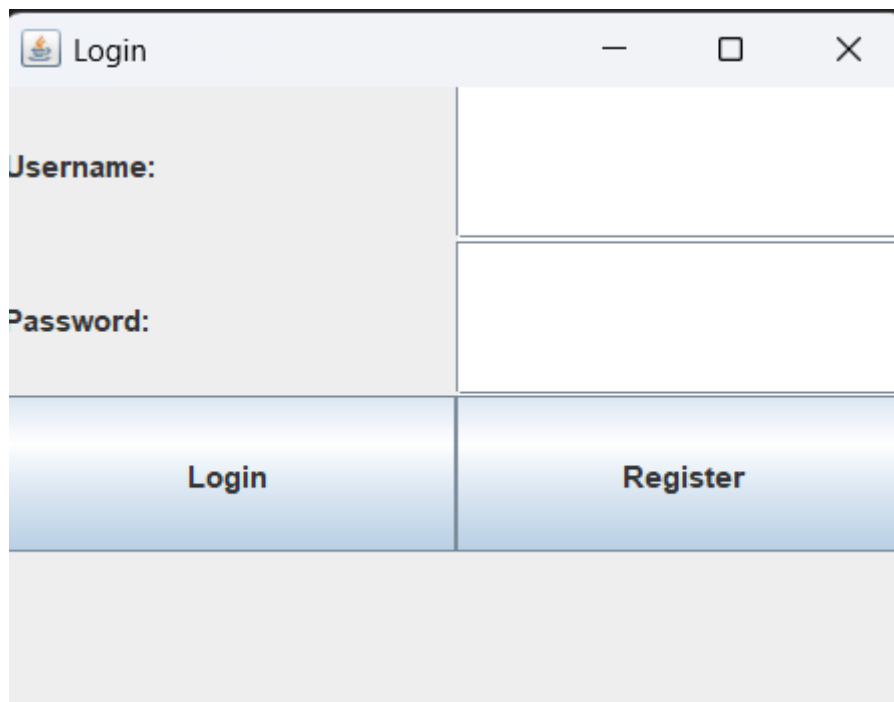
    public User(int id, String username, String password) {
        this.id = id;
        this.username = username;
        this.password = password;
    }

    public int getId() { return id; }
    public String getUsername() { return username; }
    public String getPassword() { return password; }

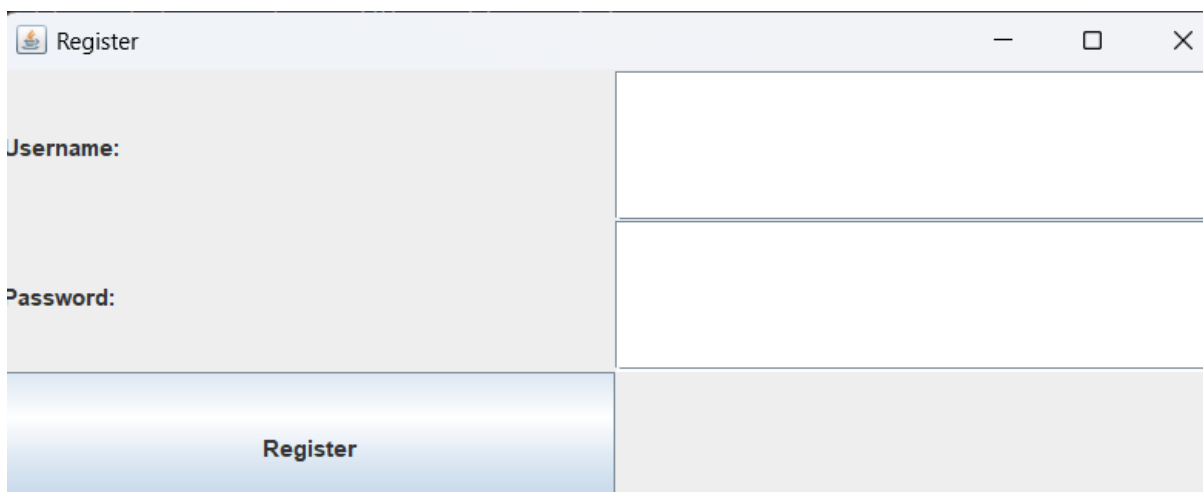
    public void setId(int id) { this.id = id; }
    public void setUsername(String username) { this.username =
username; }
    public void setPassword(String password) { this.password =
password; }
}

```

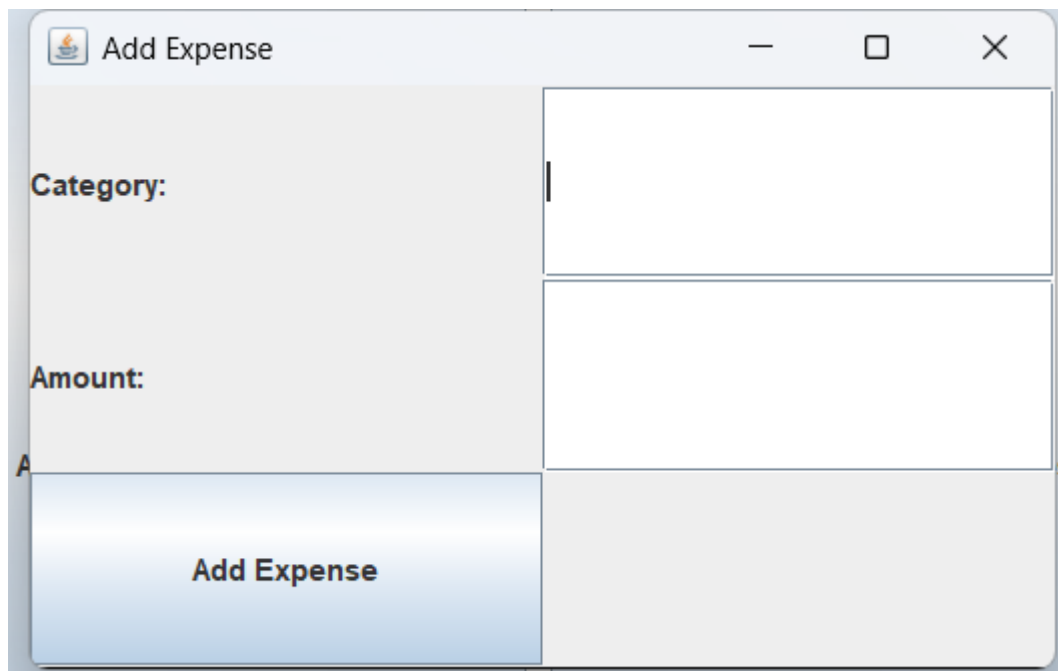
Testing and Output

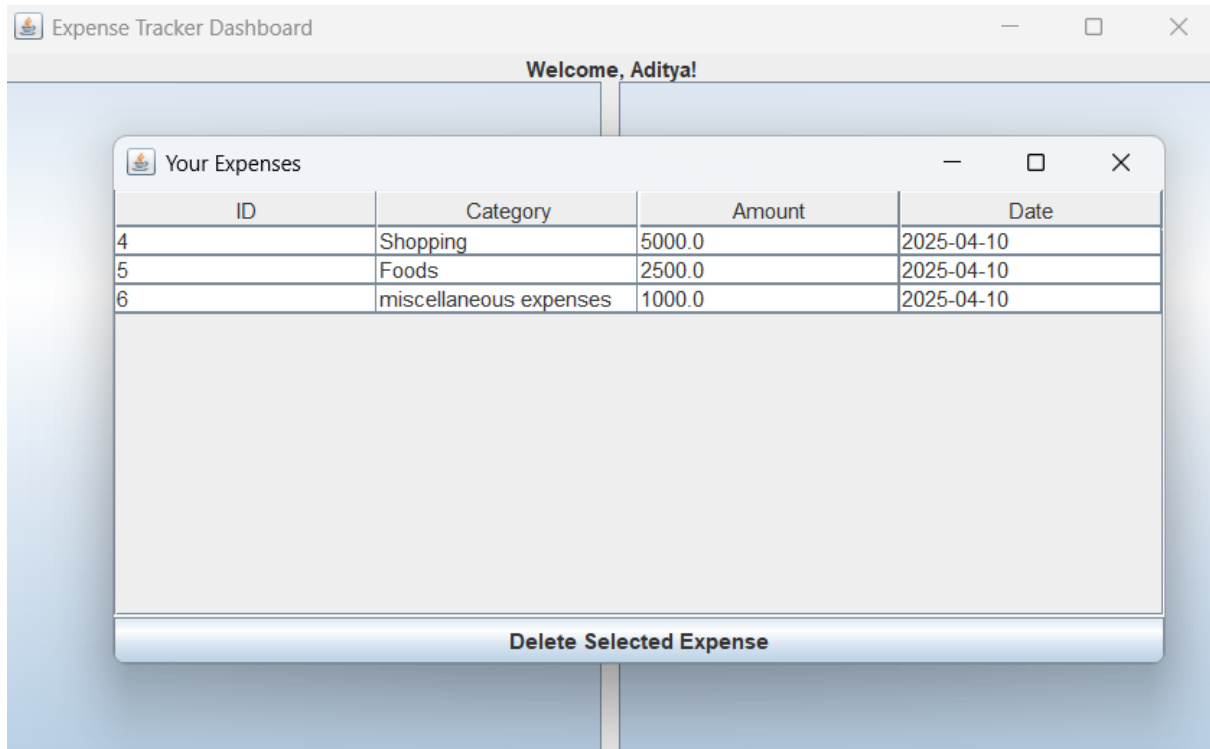


A screenshot of a Java Swing window titled "Login". The window has a standard title bar with a minimize button, a maximize button, and a close button. The main content area is divided into four sections. On the left, there are two labels: "Username:" and "Password:". To the right of "Username:" is a text input field. To the right of "Password:" is another text input field. Below these input fields are two buttons: "Login" and "Register". The "Login" button is on the left and the "Register" button is on the right. Both buttons have a blue gradient background. Below the buttons is a large, empty rectangular area with a light gray background.



A screenshot of a Java Swing window titled "Register". The window has a standard title bar with a minimize button, a maximize button, and a close button. The main content area is divided into three sections. On the left, there are two labels: "Username:" and "Password:". To the right of "Username:" is a text input field. To the right of "Password:" is another text input field. Below these input fields is a single button labeled "Register". The button has a blue gradient background. To the right of the button is a large, empty rectangular area with a light gray background.





Conclusion

In conclusion, the **Smart Expense Tracker** project successfully achieves its goal of providing an intuitive platform for managing personal finances. By utilizing **Java** and **MySQL** for the backend, along with a **Swing GUI** for the user interface, the application enables users to easily track, add, view, and delete their expenses, ensuring effective management of their finances.

Throughout the development process, we focused on creating a simple yet efficient solution that meets the needs of users looking to monitor and control their spending. Key features such as **user login, registration, expense tracking, and data management** were implemented seamlessly, ensuring both functionality and usability. The integration with MySQL allows for reliable data storage and retrieval, while the Swing-based interface ensures a smooth user experience.

The project faced some challenges, particularly in ensuring secure data handling and providing a responsive interface. However, these were addressed through careful testing and optimization, leading to a stable and user-friendly application.

Looking ahead, future enhancements could include adding features like **expense categorization, budget tracking, and generating reports**, which would further improve the application's functionality and provide users with deeper insights into their spending patterns. Additionally, expanding the system to support multi-user access or integrating with external financial APIs could make the platform even more robust.

Overall, the **Smart Expense Tracker** serves as an effective and practical tool for personal finance management, offering a reliable solution for users to track their expenses and maintain control over their financial health.