**Smart Expense Splitter**

**Name:** Suhani Singh [RA2411003011538]

Vaishnavi Jagatap [RA2411003011544]

Ali Ahmed Flexwala [RA2411003011560]

Hastik Pangi [RA2411003011561]

**Course:** Advanced Programming Practice[21CSC203P]  
**Semester:**3rd semester  
**Institution:** SRM UNIVERSITY OF SCIENCE AND TECHNOLOGY

**Faculty name**: Girirajan S  
**Submission Date:** October 17, 2025

**Abstract:**  
Smart Expense Splitter is a full-stack web application designed to simplify the management and splitting of expenses among individuals and groups. Built using React for the frontend and Spring Boot with MySQL for the backend, the app allows users to add, view, and delete personal and group expenses, automatically calculates splits, and supports bill scanning via OCR. The system ensures persistent data storage, user-friendly interaction, and transparent expense tracking, making it ideal for roommates, friends, or teams sharing costs.

**Keywords:**  
Expense management, Group splitting, React, Spring Boot, MySQL

**Problem Statement, Objectives, Scope**

**1. Introduction / Background**

Managing shared expenses in groups can be tedious and error-prone, especially when tracking contributions and calculating splits manually. Digital solutions can automate this process, reduce disputes, and provide transparency.

**2. Problem Statement**

Manual expense tracking leads to confusion and errors in group settings. There is a need for an automated, user-friendly system to manage and split expenses efficiently.

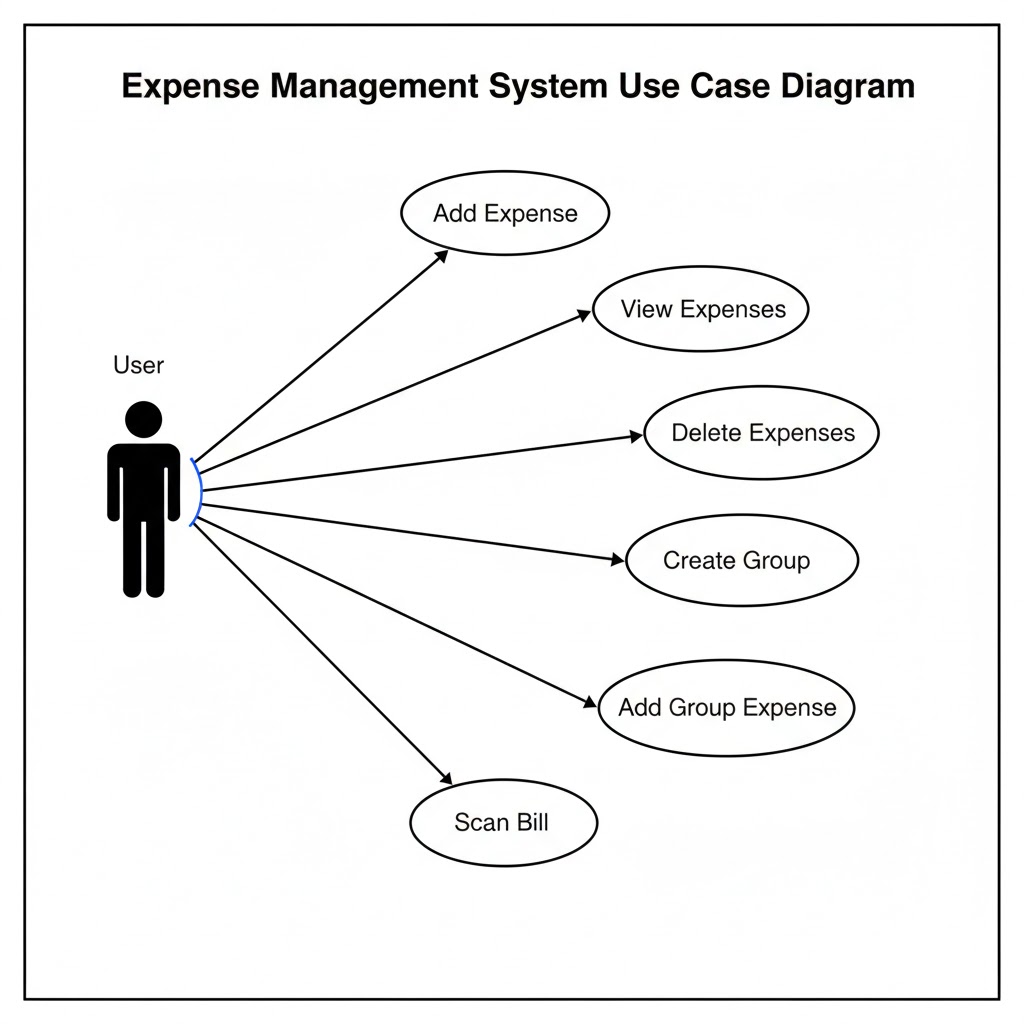
**3. Objectives**

* Develop a web app for adding, viewing, and deleting expenses.
* Enable group expense splitting with automatic calculation.
* Provide persistent storage using a relational database.
* Support bill scanning for quick expense entry.
* Ensure a responsive and intuitive user interface.

**4. Scope & Limitations**

* Web-based access only (no mobile app).
* Requires internet and database connectivity.
* OCR accuracy depends on image quality.

**5. Use Case Diagram**

****

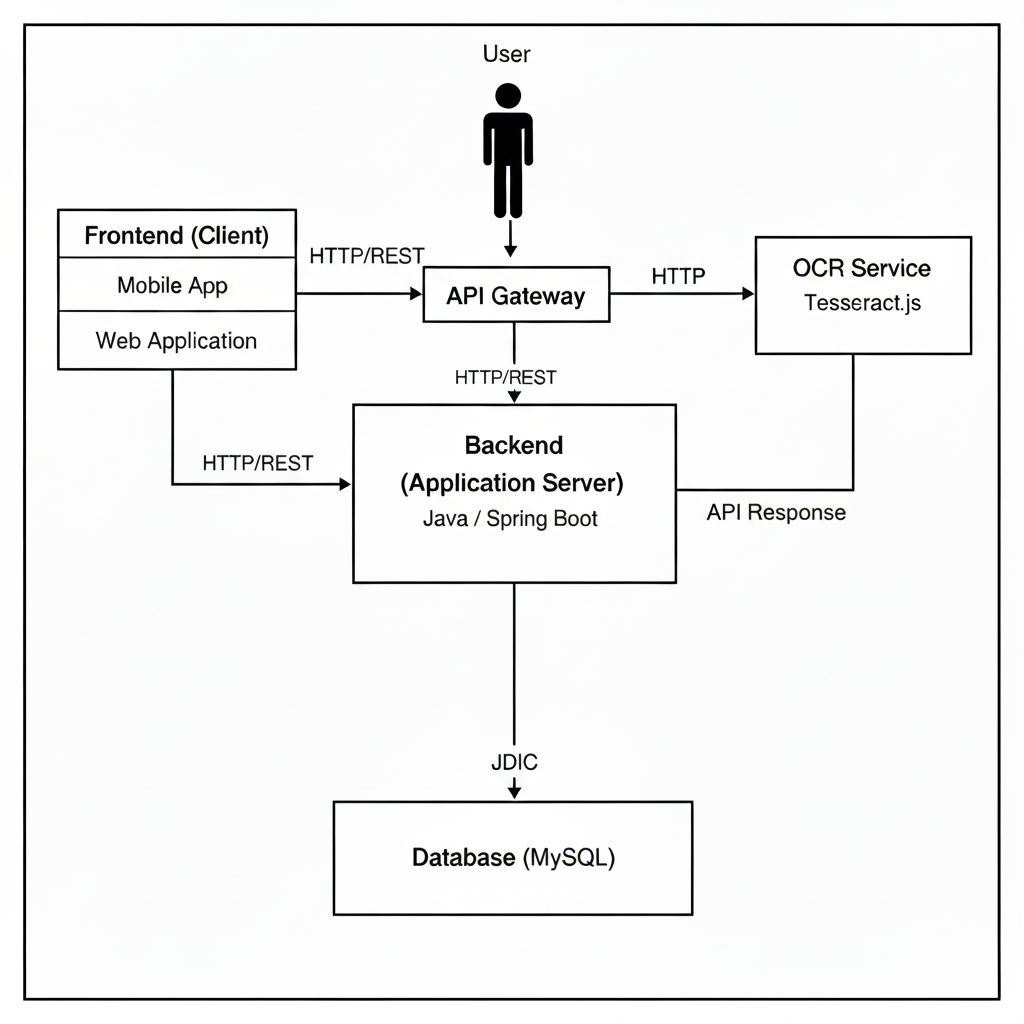
**Tools, Design & Data Model**

**5. Technology Stack**

* Java 21
* Spring Boot
* MySQL
* React.js
* Material-UI
* IntelliJ IDEA / VS Code

**6. System Architecture**

The system follows a client-server architecture. The React frontend communicates with the Spring Boot backend via REST APIs. The backend handles business logic and interacts with the MySQL database for persistent storage.



**7. Data Model / Key Tables / Inputs & Outputs**

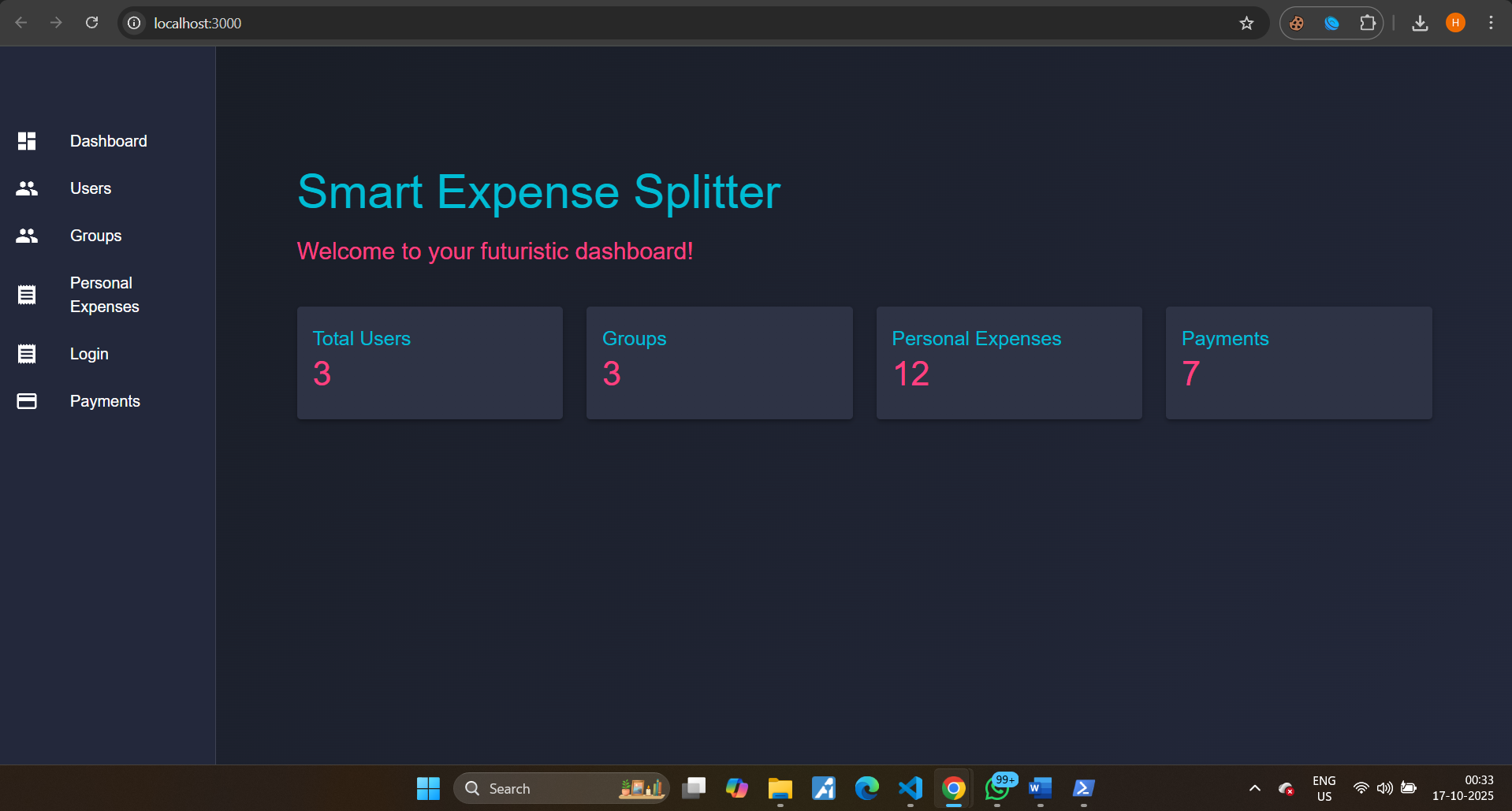
| **Entity** | **Fields** |
| --- | --- |
| User | id, name, email |
| Group | id, group Name, members |
| Expense | id, description, amount, payer Id |
| Group Expense | id, group Name, description, amount, members, split |

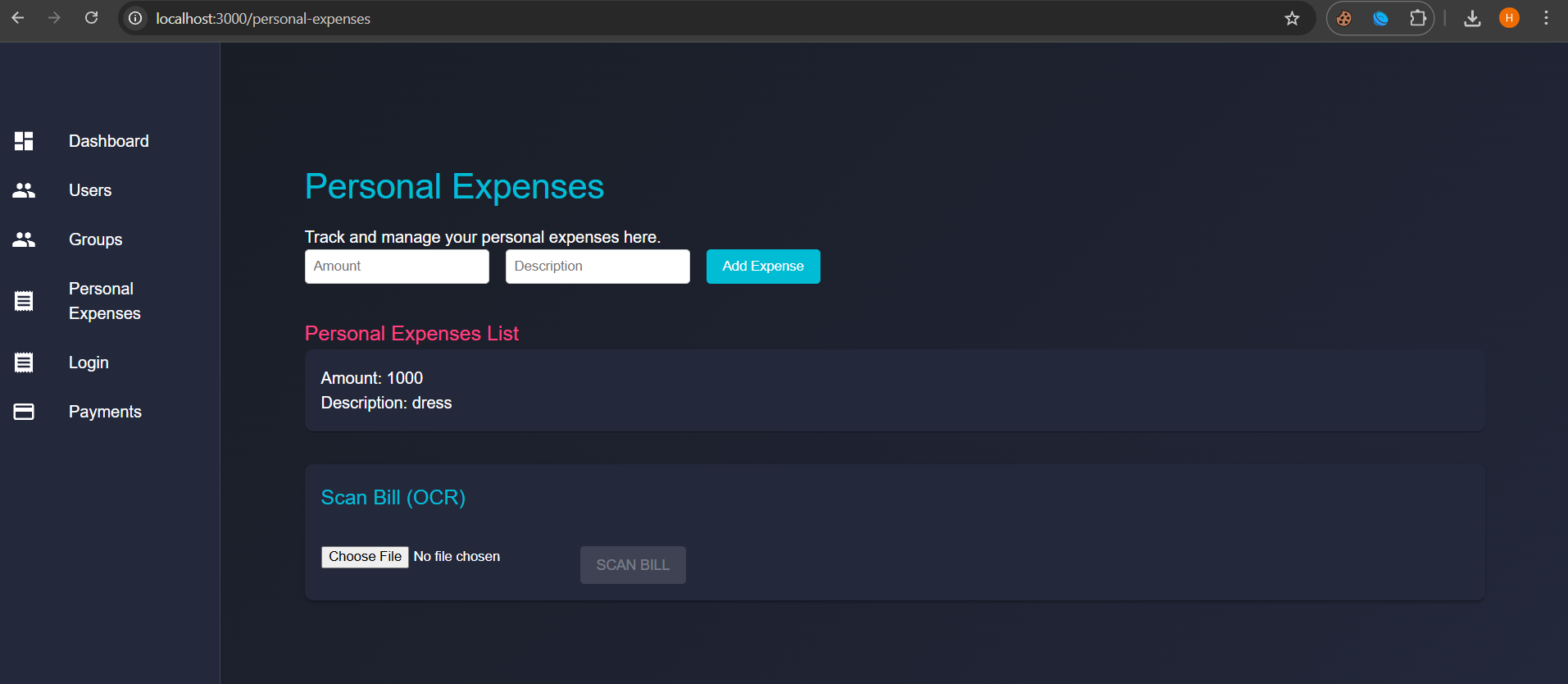
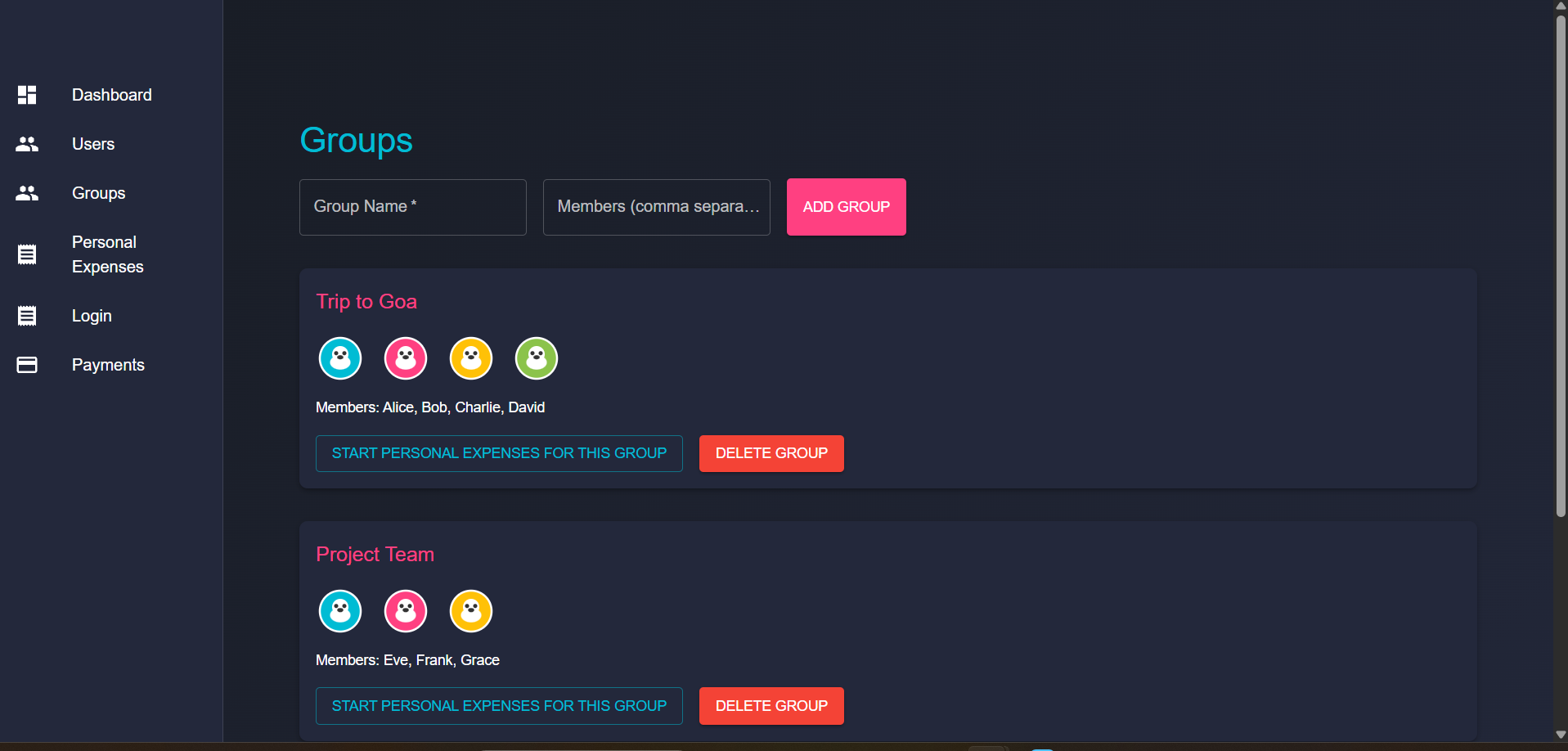
**Inputs:** User details, expense details, group details, bill images  
**Outputs:** Expense lists, split calculations, scanned text

**Step-by-Step Worked Procedure**

**Front-End Steps**

1. User logs in or registers.
2. User navigates to "Add Expense" or "Add Group Expense" page.
3. User fills in details and submits the form.
4. Expense appears in the list; group splits are calculated automatically.



****

**SQL Commands**

**Creating the database:**

CREATE DATABASE expense\_db;

**Creating tables (auto-generated by JPA if configured):**  
**(Show a screenshot of your MySQL table structure or use SHOW TABLES; output.)**

**Connectivity Setup**

**application.properties configuration:**

spring.datasource.url=jdbc:mysql://localhost:3306/expense\_db spring.datasource.username=your\_mysql\_username spring.datasource.password=your\_mysql\_password spring.jpa.hibernate.ddl-auto=update