1 INTRODUCTION

1.1 Overview

The Personal Expense Tracker application is a web-based financial management system designed to help individuals track and manage their personal expenses effectively. The application is built using the React framework for the frontend, Java Spring for the backend, and MySQL database for data storage.

1.2 Purpose

The Personal Expense Tracker application aims to empower individuals in managing their personal finances more efficiently. It offers a user-friendly interface, robust backend functionality, and comprehensive reporting capabilities to help users gain control over their expenses, budget effectively, and make informed financial decisions.

2 LITERATURE SURVEY

2.1 Existing problem

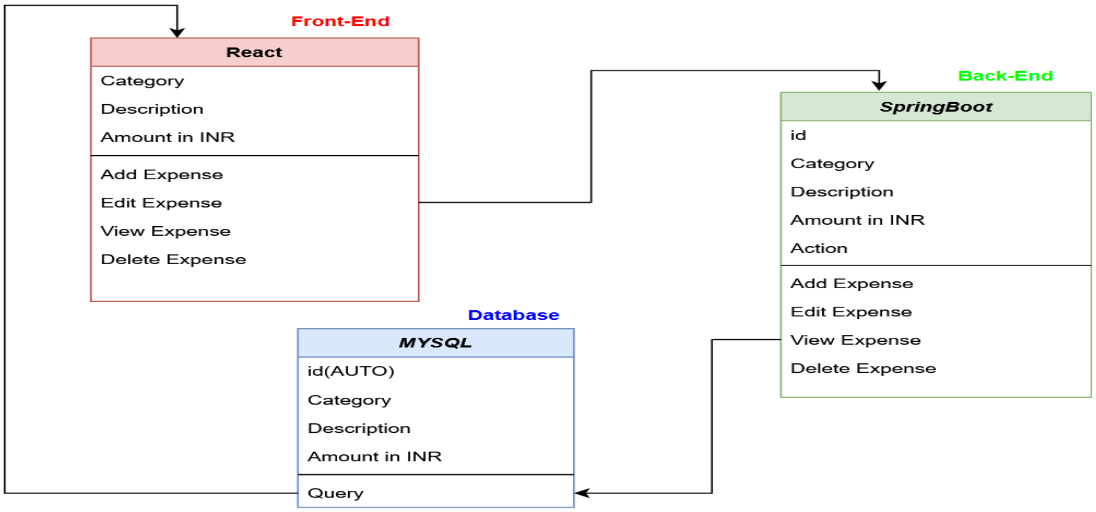
Lack of Mobile Accessibility,Limited Data Visualization Options,Insufficient Expense Categorization,Manual Data Entry,Lack of Real-time Notifications.

2.2 Proposed solution

One of the key features of our expense tracker application is the ability to export expense records. With this feature, users can conveniently extract their expense data and save it as a file for further analysis or record-keeping purposes. Whether it's for personal reference, tax reporting, or budget analysis, exporting expense records provides users with the flexibility to manipulate and utilize their financial data in a way that suits their needs.

3 THEORITICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software designing

- Front-end (React)
- Back-end (Spring Boot)
- Database (MySQL)
- Dependency Management (Maven)

The software design encompasses the front-end React components for the user interface, the Spring Boot backend for handling requests and interacting with the MySQL database, and the database itself to store and retrieve application data. Maven is used as the dependency provider for managing project dependencies and facilitating the build process.

Multi-core processor for efficient request handling and database operations.Adequate memory (RAM) for concurrent user requests and optimal performance.Sufficient storage capacity for storing code, dependencies, and backups.Stable network interface for seamless communication with clients.

4 EXPERIMENTAL INVESTIGATIONS

During the development of a Personal Expenses Tracker Application, several experimental investigations and analyses can be conducted to assess the solution. Here are some areas to focus on during the investigation:

Performance Analysis:

- Measure the application's response time for key operations, such as adding expenses, retrieving expense reports, or generating analytics
- Analyze the application's memory usage and CPU utilization during different usage scenarios.

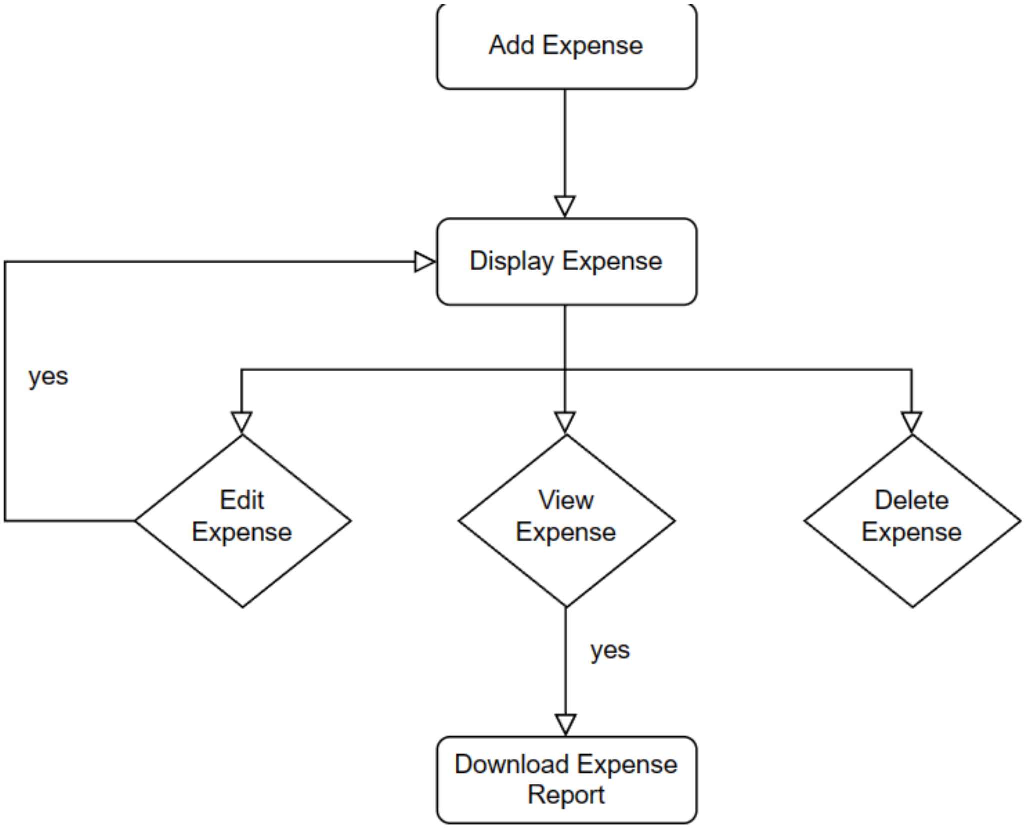
Usability Analysis:

- Conduct user feedback sessions or surveys to gather insights on the usability of the application.
- Identify areas where users might encounter difficulties in navigating the interface, inputting expenses, or understanding the presented data.
- Analyze user interaction patterns and identify potential areas for improvement, such as simplifying workflows or providing clearer instructions.

Feature Analysis:

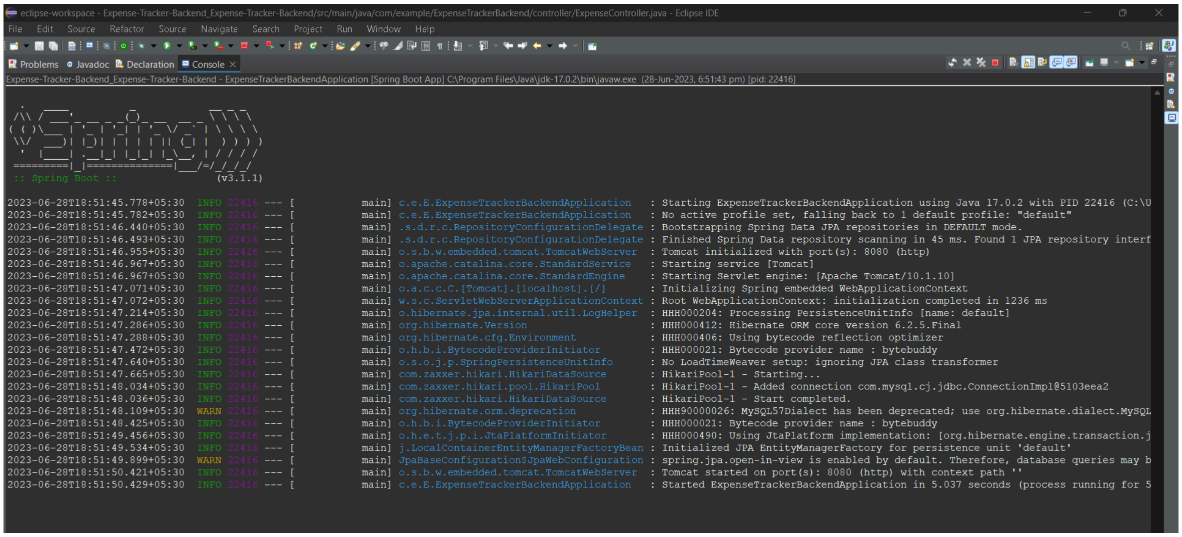
- Evaluate the functionality of different features, such as expense categorization, budget tracking.
- Identify and address any bugs or inconsistencies in feature behavior.

5 FLOWCHART



6 RESULT

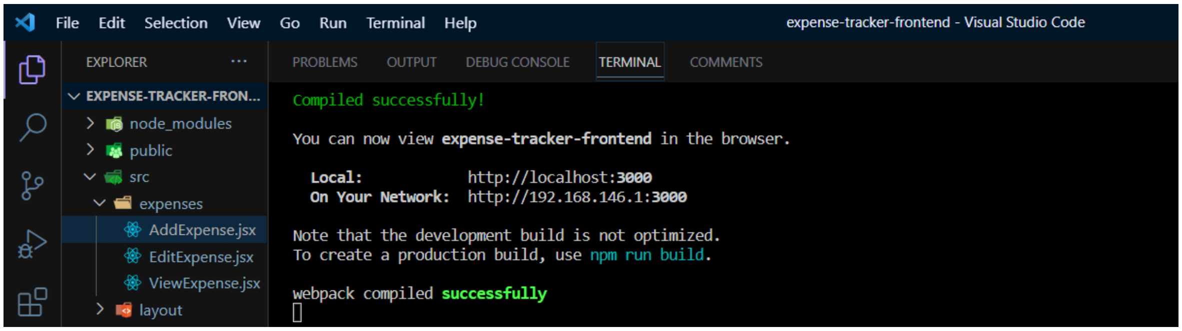
Image of the Backend application running on tomcat server 8080:


eclipse-workspace - Expense-Tracker-Backend_Expense-Tracker-Backend/src/main/java/com/example/ExpenseTrackerBackend/controller/ExpenseController.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Problems
Javadoc
Declaration
Console

Expense-Tracker-Backend_Expense-Tracker-Backend - ExpenseTrackerBackendApplication [Spring Boot App] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe (28-Jun-2023, 6:51:43 pm) [pid: 22416]

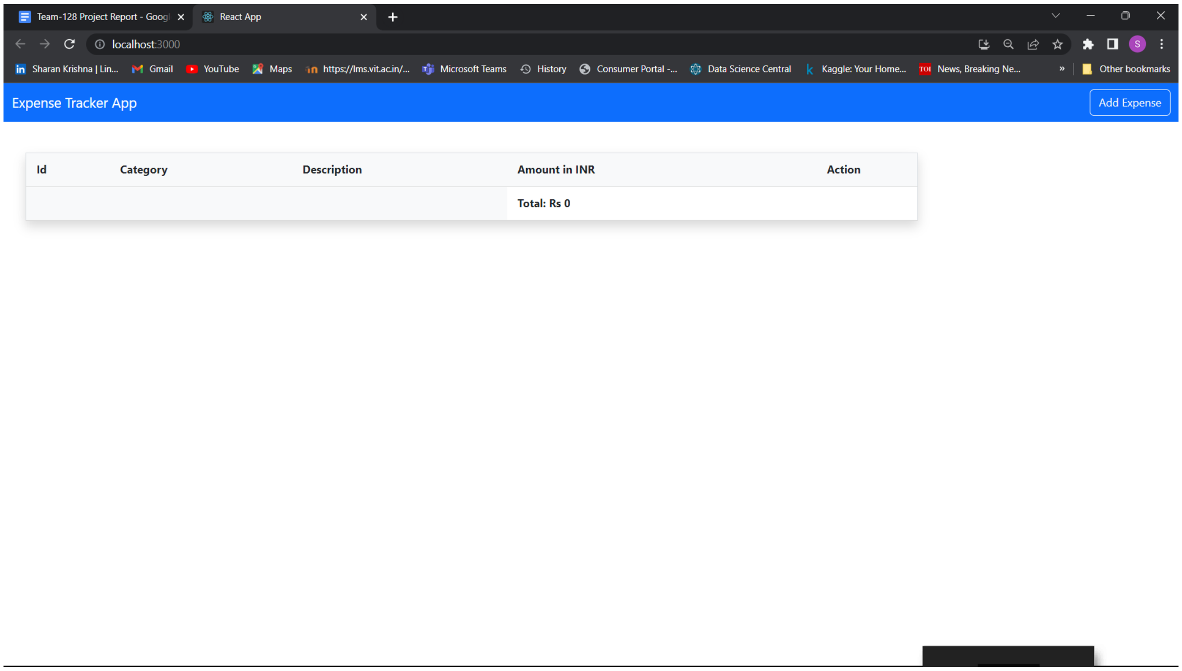
```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v3.1.1)

2023-06-28T18:51:45.778+05:30  INFO 22416 --- [           main] c.e.E.ExpenseTrackerBackendApplication   : Starting ExpenseTrackerBackendApplication using Java 17.0.2 with PID 22416 (C:\U
2023-06-28T18:51:45.782+05:30  INFO 22416 --- [           main] c.e.E.ExpenseTrackerBackendApplication   : No active profile set, falling back to 1 default profile: "default"
2023-06-28T18:51:46.440+05:30  INFO 22416 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2023-06-28T18:51:46.493+05:30  INFO 22416 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 45 ms. Found 1 JPA repository interf
2023-06-28T18:51:46.955+05:30  INFO 22416 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2023-06-28T18:51:46.967+05:30  INFO 22416 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2023-06-28T18:51:46.967+05:30  INFO 22416 --- [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10.1.10]
2023-06-28T18:51:47.071+05:30  INFO 22416 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2023-06-28T18:51:47.072+05:30  INFO 22416 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1236 ms
2023-06-28T18:51:47.214+05:30  INFO 22416 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2023-06-28T18:51:47.286+05:30  INFO 22416 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 6.2.5.Final
2023-06-28T18:51:47.288+05:30  INFO 22416 --- [           main] org.hibernate.cfg.Environment            : HHH000406: Using bytecode reflection optimizer
2023-06-28T18:51:47.472+05:30  INFO 22416 --- [           main] o.h.b.i.BytecodeProviderInitiator        : HHH000021: Bytecode provider name : bytebuddy
2023-06-28T18:51:47.640+05:30  INFO 22416 --- [           main] o.s.o.j.p.SpringPersistenceUnitInfo      : No LoadTimeWeaver setup: ignoring JPA class transformer
2023-06-28T18:51:47.665+05:30  INFO 22416 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2023-06-28T18:51:48.034+05:30  INFO 22416 --- [           main] com.zaxxer.hikari.pool.HikariPool        : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@5103eea2
2023-06-28T18:51:48.036+05:30  INFO 22416 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2023-06-28T18:51:48.109+05:30  WARN 22416 --- [           main] org.hibernate.orm.deprecation            : HHH90000026: MySQL57Dialect has been deprecated; use org.hibernate.dialect.MySQL
2023-06-28T18:51:48.425+05:30  INFO 22416 --- [           main] o.h.b.i.BytecodeProviderInitiator        : HHH000021: Bytecode provider name : bytebuddy
2023-06-28T18:51:49.456+05:30  INFO 22416 --- [           main] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.j
2023-06-28T18:51:49.534+05:30  INFO 22416 --- [           main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2023-06-28T18:51:49.899+05:30  WARN 22416 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may b
2023-06-28T18:51:50.421+05:30  INFO 22416 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with context path ''
2023-06-28T18:51:50.429+05:30  INFO 22416 --- [           main] c.e.E.ExpenseTrackerBackendApplication   : Started ExpenseTrackerBackendApplication in 5.037 seconds (process running for 5
```

Image of the terminal of starting the React frontend app:


File Edit Selection View Go Run Terminal Help
expense-tracker-frontend - Visual Studio Code
EXPLORER
EXPENSE-TRACKER-FRON...
node_modules
public
src
expenses
AddExpense.jsx
EditExpense.jsx
ViewExpense.jsx
layout
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

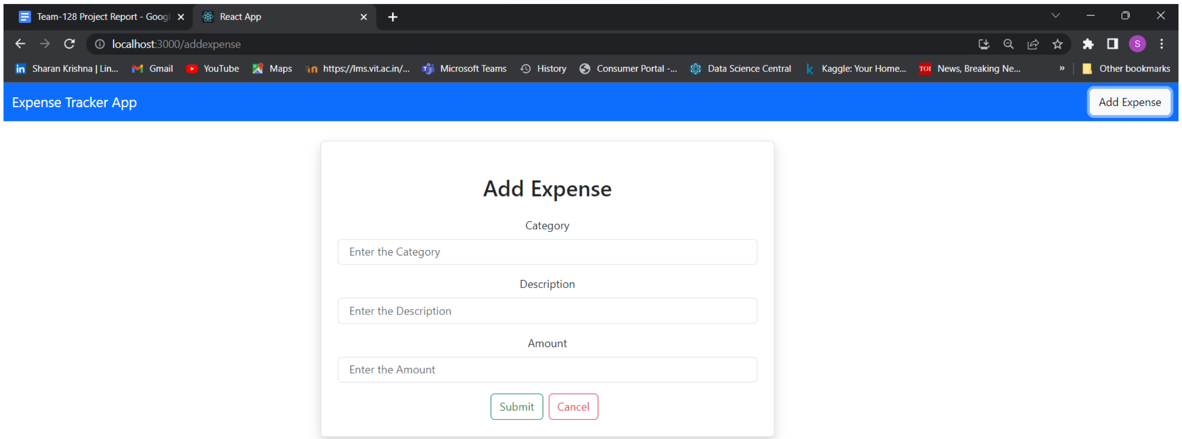
```
Compiled successfully!

You can now view expense-tracker-frontend in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.146.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

Home page:


Team-128 Project Report - Goog
React App
localhost:3000
Sharan Krishna | Lin... Gmail YouTube Maps https://lms.vit.ac.in/... Microsoft Teams History Consumer Portal -... Data Science Central Kaggle: Your Home... News, Breaking Ne... Other bookmarks
Expense Tracker App
Add Expense

Id	Category	Description	Amount in INR	Action
			Total: Rs 0	

Add Expense:


Team-128 Project Report - Goog
React App
localhost:3000/addexpense
Sharan Krishna | Lin... Gmail YouTube Maps https://lms.vit.ac.in/... Microsoft Teams History Consumer Portal -... Data Science Central Kaggle: Your Home... News, Breaking Ne... Other bookmarks
Expense Tracker App
Add Expense

Add Expense

Category

Enter the Category

Description

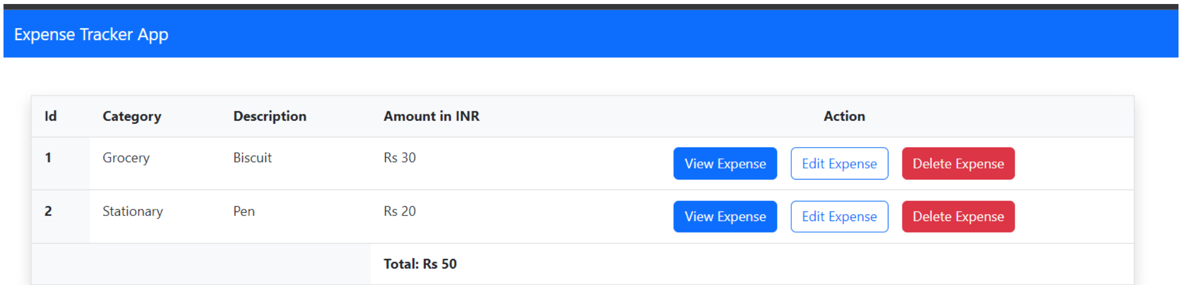
Enter the Description

Amount

Enter the Amount

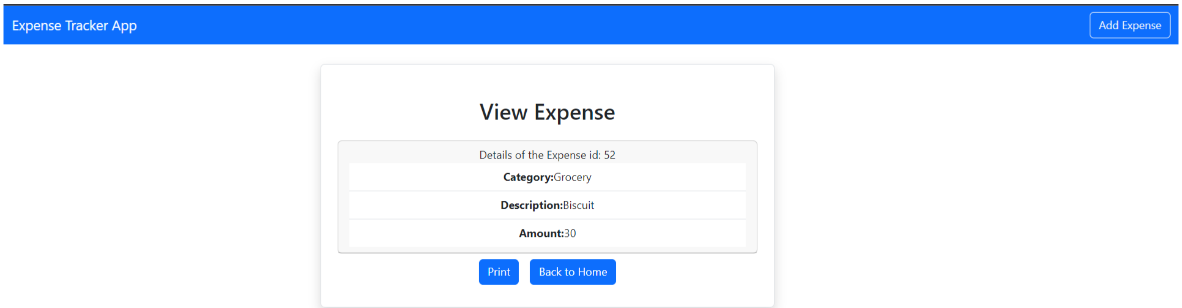
Submit Cancel

Home page after adding expense:



Id	Category	Description	Amount in INR	Action
1	Grocery	Biscuit	Rs 30	View Expense Edit Expense Delete Expense
2	Stationary	Pen	Rs 20	View Expense Edit Expense Delete Expense
			Total: Rs 50	

View Expense page:



View Expense

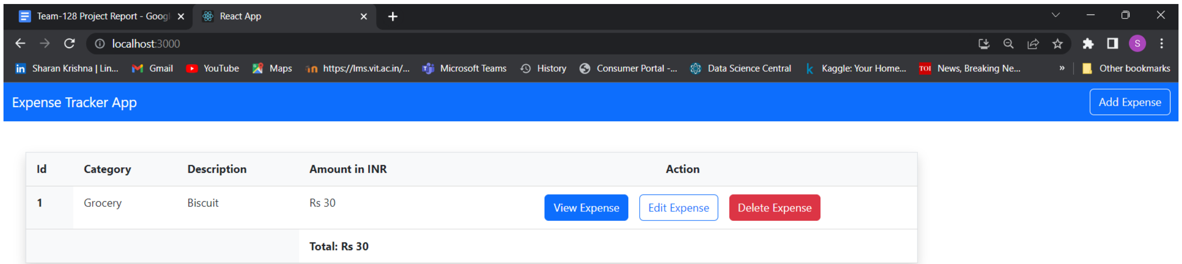
Details of the Expense id: 52

Category: Grocery

Description: Biscuit

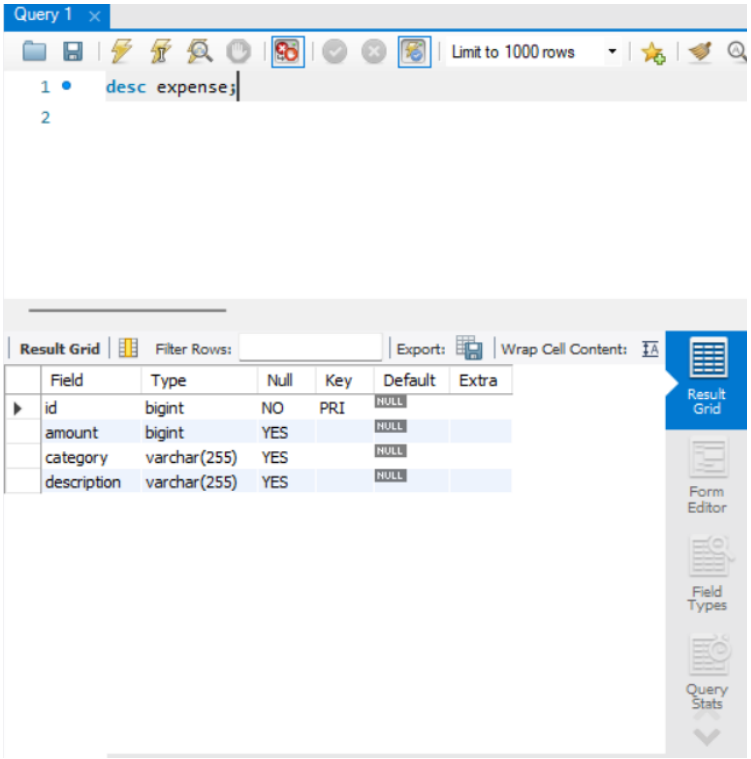
Amount:30

Home page after deleting the Expense:



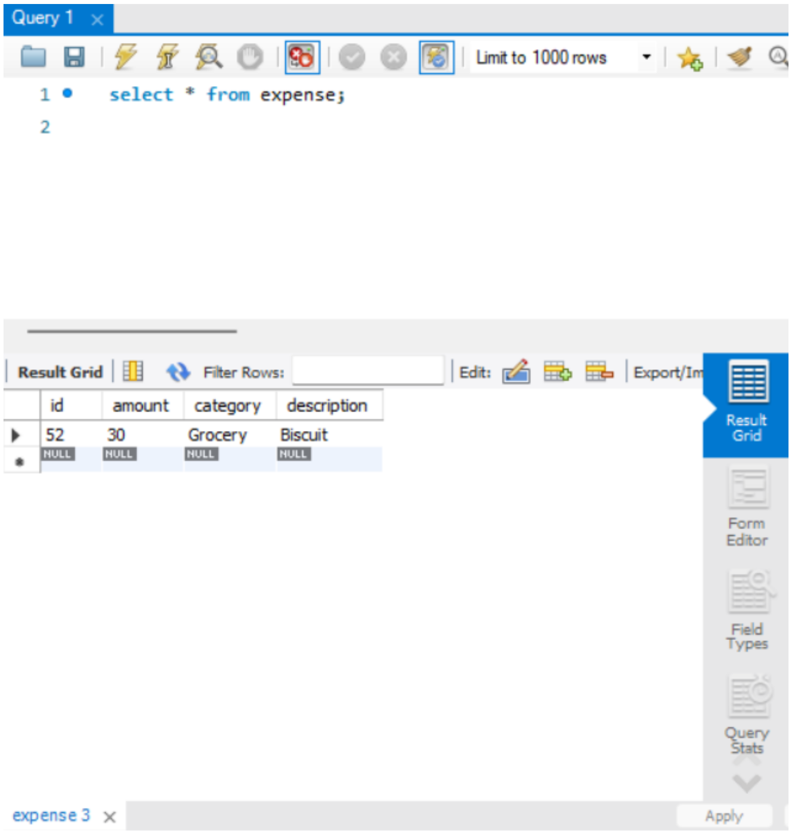
Id	Category	Description	Amount in INR	Action
1	Grocery	Biscuit	Rs 30	View Expense Edit Expense Delete Expense
			Total: Rs 30	

SQL table images:



```
desc expense;
```

Field	Type	Null	Key	Default	Extra
id	bigint	NO	PRI	NULL	
amount	bigint	YES		NULL	
category	varchar(255)	YES		NULL	
description	varchar(255)	YES		NULL	



```
select * from expense;
```

id	amount	category	description
52	30	Grocery	Biscuit
NULL	NULL	NULL	NULL

7 ADVANTAGES & DISADVANTAGES

- Advantages of Personal Expenses Tracker Application:

1. Financial Awareness: A personal expenses tracker application provides users with a clear view of their spending habits, helping them become more aware of where their money is going. It enables users to track and categorize expenses, identify areas of overspending, and make informed decisions to improve their financial management.

2. Budgeting and Goal Setting: Such an application allows users to set budgets and financial goals. It provides real-time tracking of expenses against the defined budget, helping users stay on track and avoid overspending. By setting specific goals, users can work towards saving for specific milestones or reducing unnecessary expenses.

3. Expense Categorization and Analysis: Personal expenses tracker applications offer the ability to categorize expenses, making it easier to analyze spending patterns. Users can gain insights into their spending by category, identify areas where they can cut costs, and make adjustments to align their expenses with their financial goals.

4. Simplified Expense Tracking: These applications streamline the process of recording and managing expenses. Instead of relying on manual methods like pen and paper or spreadsheets, users can quickly enter their expenses using the application, saving time and reducing the likelihood of errors.

- Disadvantages of Personal Expenses Tracker Application:

1. Learning Curve: Some users may initially find it challenging to adapt to a new application and its interface. It may take time to become familiar with the features and functionalities of the personal expenses tracker application, potentially leading to a learning curve that could discourage some users from fully utilizing its benefits.

2. Data Security and Privacy Concerns: Storing personal financial data in an application requires robust security measures to protect against unauthorized access or data breaches. Users must trust that their sensitive information, such as bank account details or transaction history, is properly safeguarded. Concerns about data security and privacy may discourage some individuals from using such applications.

3. Reliance on Technology: Personal expenses tracker applications heavily rely on technology, including internet connectivity, servers, and software. Any technical issues, such as server outages or software bugs, may temporarily hinder users from accessing or utilizing the application, impacting their ability to track expenses effectively.

4. Limited Customization: Depending on the specific application, users may have limited options for customization. Some individuals may prefer more flexibility in tailoring expense categories, budgeting methods, or reporting formats to align with their unique financial needs and preferences.

8 APPLICATIONS

Personal Expenses Tracker applications find applications in various scenarios and for individuals in different roles who seek to effectively manage their personal finances. Here are some specific applications of Personal Expenses Tracker applications:

- Individuals and Families: Personal Expenses Tracker applications are commonly used by individuals and families to manage their day-to-day expenses and gain better control over their finances. They can track income, monitor spending, set budgets, and analyze expenses to make informed financial decisions.

- Budgeting and Financial Planning: These applications help users create and maintain budgets. They provide tools for categorizing expenses, setting spending limits, and tracking actual expenses against budgeted amounts. Users can identify areas of overspending, make adjustments, and plan their finances accordingly.

- Saving for Financial Goals: Personal Expenses Tracker applications assist users in setting and achieving financial goals. Whether it's saving for a down payment on a house, a vacation, or retirement, these applications help users track their progress, allocate savings, and stay on track towards their objectives.

- Expense Reimbursement and Expense Tracking for Professionals: Professionals, such as freelancers, consultants, or employees who need to track and submit expenses for reimbursement, can benefit from Personal Expenses Tracker applications. These applications provide a convenient way to capture and categorize expenses, attach receipts, and generate reports for reimbursement or tax purposes.

- Small Business Owners and Entrepreneurs: Personal Expenses Tracker applications can be used by small business owners and entrepreneurs to track business-related expenses separate from personal expenses. This allows for better financial management, tax preparation, and monitoring of business expenditures.

- Students and Young Adults: Personal Expenses Tracker applications can be valuable for students and young adults who are learning to manage their finances independently. These applications provide a structured approach to budgeting, expense tracking, and financial goal setting, helping them develop good financial habits early on.

- Travelers and Digital Nomads: Individuals who frequently travel or work remotely can utilize Personal Expenses Tracker applications to track their travel expenses, manage currencies, and monitor expenses in different locations. This helps them stay organized and maintain financial discipline while on the move.

- Debt Management: Personal Expenses Tracker applications can assist individuals in managing their debts, such as student loans, credit card debt, or mortgages. By tracking debt payments, interest rates, and progress toward debt reduction, users can develop effective repayment strategies and achieve financial freedom.

9 CONCLUSION

Personal Expenses Tracker applications offer a range of benefits and applications for individuals seeking to manage their personal finances effectively. These applications provide tools for budgeting, expense tracking, goal setting, and financial analysis, empowering users to gain better control over their financial health. By utilizing a Personal Expenses Tracker application, individuals can:

- Increase financial awareness: The application enables users to track and categorize expenses, providing a clear view of their spending habits and allowing them to make informed decisions about their finances.

- Set and track financial goals: Users can set specific financial goals, such as saving for a vacation or paying off debts, and monitor their progress in real-time.

- Improve budgeting and spending habits: The application helps users create budgets, track expenses against these budgets, and identify areas of overspending, allowing for better financial discipline.

- Simplify expense tracking: These applications streamline the process of recording and managing expenses, saving time and reducing the likelihood of errors compared to manual methods.

- Analyze spending patterns: Personal Expenses Tracker applications offer the ability to categorize and analyze expenses, providing insights into spending patterns and enabling users to make adjustments and optimize their financial decisions.

- Enhance financial goal collaboration: Some applications facilitate collaboration between family members or partners, allowing for joint financial planning and transparent tracking of shared financial goals.

10 FUTURE SCOPE

The future scope of Personal Expenses Tracker applications is promising, with opportunities for further enhancements and advancements. Here are some potential areas of future development and expansion:

- Integration with Financial Institutions: Personal Expenses Tracker applications can leverage open banking APIs to directly integrate with users' bank accounts, credit cards, and other financial institutions. This integration would allow for automatic expense tracking, real-time updates of transactions, and seamless reconciliation of financial data.

- Smart Expense Tracking: With the rise of Internet of Things (IoT) devices and connected systems, Personal Expenses Tracker applications could incorporate smart expense tracking features. For instance, users could capture receipts automatically using their smartphones, link expense data to smart payment devices or wearables, or integrate with smart home systems to track utility expenses.

- Enhanced Security and Privacy Features: As the importance of data security and privacy continues to grow, future applications will focus on strengthening security measures. This may include advanced encryption, biometric authentication, and compliance with stringent data protection regulations, instilling users' confidence in the security of their financial information.

- Integration with Digital Wallets and Payment Platforms: Personal Expenses Tracker applications can integrate with popular digital wallets and payment platforms, allowing users to seamlessly track their digital transactions alongside traditional expenses. This integration would provide a holistic view of all financial activities in a single application.

- Personalized Financial Advice: Personal Expenses Tracker applications can leverage user data and analytics to offer personalized financial advice and recommendations. This could include tips for reducing expenses, optimizing savings, or suggesting investment opportunities tailored to each user's financial situation and goals.

11 BIBLIOGRAPHY

- https://spring.io/projects/spring-boot
- https://hub.docker.com/
- https://kubernetes.io/docs/home/
- https://react.dev/learn
- https://maven.apache.org/

APPENDIX

A. Source Code

Code Snippets of backend:

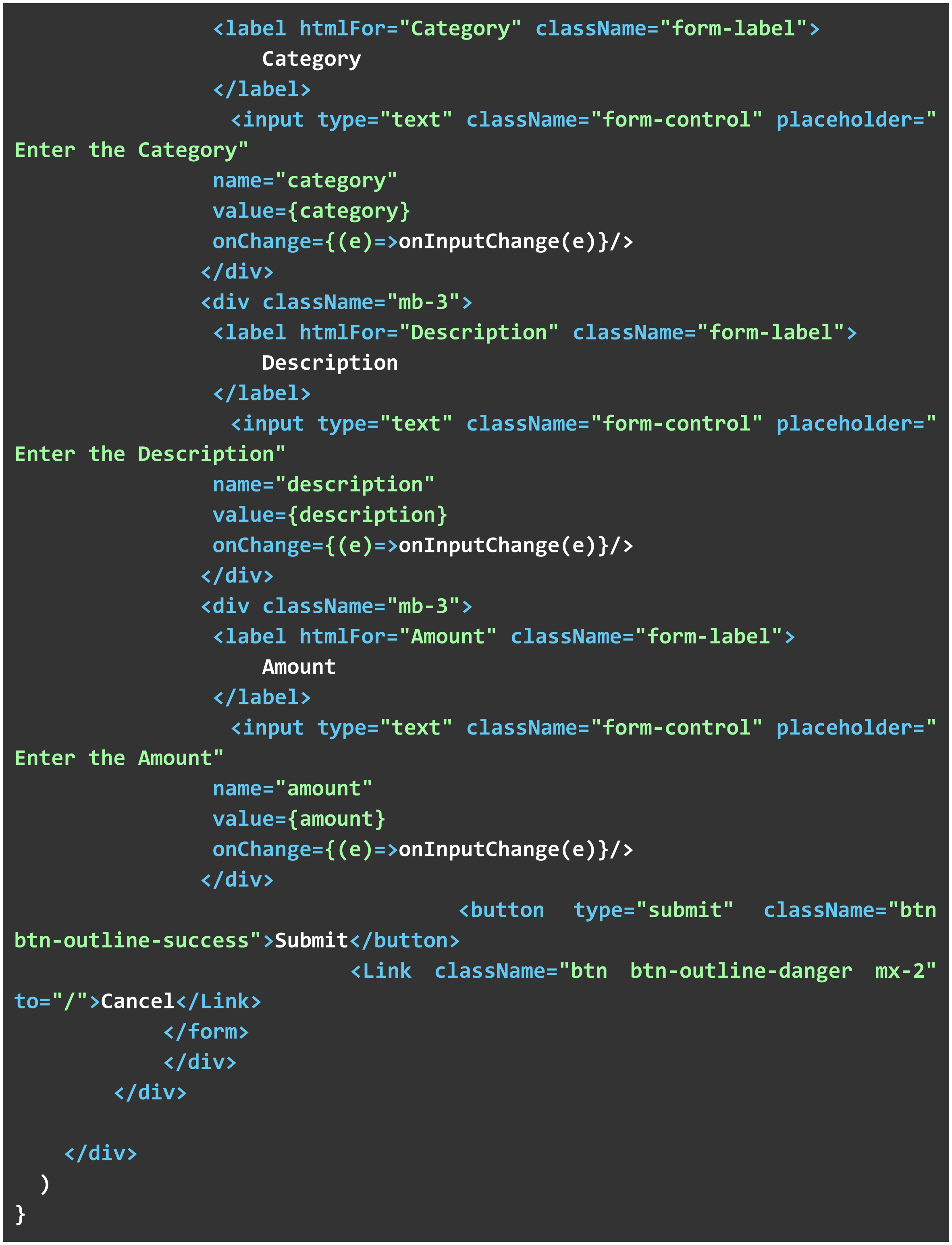
REST API Code:

```
package com.example.ExpenseTrackerBackend.controller;

import java.util.List;


import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

import
com.example.ExpenseTrackerBackend.exception.ExpenseNotFoundExcept
ion;
import com.example.ExpenseTrackerBackend.model.Expense;
import
com.example.ExpenseTrackerBackend.repository.ExpenseRepository;

@RestController
@CrossOrigin("http://localhost:3000/")

public class ExpenseController {

    @Autowired
    private ExpenseRepository expenseRepository;
    @PostMapping("/expense")
    Expense newExpense(@RequestBody Expense newExpense) {

        return expenseRepository.save(newExpense);
    }

    @GetMapping("/expenses")
    List<Expense> getAllExpenses(){

        return expenseRepository.findAll();
    }

    @GetMapping("/expense/{id}")
    Expense getExpenseById(@PathVariable Long id) {
        return expenseRepository.findById(id)
        .orElseThrow(()->new ExpenseNotFoundException(id));
    }

    @PutMapping("/expense/{id}")
    Expense          updateExpense(@RequestBody          Expense
newExpense,@PathVariable Long id) {
        return expenseRepository.findById(id)
                .map(expense -> {

expense.setCategory(newExpense.getCategory());

expense.setDescription(newExpense.getDescription());

expense.setAmount(newExpense.getAmount());
                    return
expenseRepository.save(expense);
                }).orElseThrow(()->new
ExpenseNotFoundException(id));
    }

    @DeleteMapping("/expense/{id}")
    String deleteExpense(@PathVariable Long id) {
        if(!expenseRepository.existsById(id)) {
            throw new ExpenseNotFoundException(id);
        }
        expenseRepository.deleteById(id);
        return "Expense has been deleted Succesfully!!";

    }

}
```

Front-End code :

AddExpense:

```
import React, {useState } from 'react';
import axios from "axios";
import {Link,useNavigate} from "react-router-dom";


export default function AddExpense() {

    let navigate=useNavigate();


    const[expense,setExpense]=useState({

        category:"",
        description:"",
        amount:""

    })

    const { category,description,amount}=expense

    const onInputChange=(e) => {

        setExpense({...expense,[e.target.name]: e.target.value});


    };

    const onSubmit =async (e) => {

        e.preventDefault();
        await axios.post("http://localhost:8080/expense",expense)
        navigate("/")


    };



  return (
    <div className="container">
        <div className="row">
             <div className="col-md-6 offset-md-4 border rounded p-4 mt-2
shadow ">
               <h2 className="text-center m-4">Add Expense</h2>
               <form onSubmit={(e)=>onSubmit(e)}>
               <div className="mb-3">
```



```
                <label htmlFor="Category" className="form-label">
                    Category
                </label>
                 <input type="text" className="form-control" placeholder="
Enter the Category"
                name="category"
                value={category}
                onChange={(e)=>onInputChange(e)}/>
               </div>
               <div className="mb-3">
                <label htmlFor="Description" className="form-label">
                    Description
                </label>
                 <input type="text" className="form-control" placeholder="
Enter the Description"
                name="description"
                value={description}
                onChange={(e)=>onInputChange(e)}/>
               </div>
               <div className="mb-3">
                <label htmlFor="Amount" className="form-label">
                    Amount
                </label>
                 <input type="text" className="form-control" placeholder="
Enter the Amount"
                name="amount"
                value={amount}
                onChange={(e)=>onInputChange(e)}/>
               </div>
                                    <button type="submit" className="btn
btn-outline-success">Submit</button>
                           <Link className="btn btn-outline-danger mx-2"
to="/">Cancel</Link>
            </form>
            </div>
        </div>

    </div>
  )
}
```