Project Report: Expense Tracker

# 1. Introduction

In today’s fast-paced world, managing personal finances is crucial for ensuring financial stability and growth. Manual methods such as spreadsheets or physical notes are prone to error, inefficient, and lack features like data insights or easy accessibility. To address this, we developed the Expense Tracker — a responsive and user-friendly web application that allows users to keep track of their daily income and expenses seamlessly.

This application provides a secure, efficient, and insightful way to monitor transactions, evaluate financial behavior, and make informed financial decisions — all from a single dashboard. The addition of a login system ensures personalized data tracking and secure access.

# 2. Objective

The primary objective of this project is to create a full-stack web application that allows users to:  
- Track their financial transactions over time.  
- Monitor their balance, income, and expenses in real-time.  
- Access their data securely through a login system.  
- Gain better insights into their spending patterns.

# 3. System Architecture

The Expense Tracker follows a client-server architecture where:  
- The Frontend (client-side) is built using React with Vite for fast development and Tailwind CSS for responsive design.  
- The Backend (server-side) uses Node.js with Express.js to create RESTful APIs.  
- MongoDB serves as the NoSQL database, storing transaction and user data.  
- Communication between frontend and backend is handled through Axios, with CORS enabling cross-origin requests.  
- Mongoose is used to define schemas and interact with MongoDB.

# 4. Features

## User Authentication

* - Secure login mechanism using username and password.
* - Authentication token stored in browser for session management.
* - Ensures that each user can only access their own transactions.

## Transaction Management

* - Add new transactions by specifying description, amount, date, and type (credit or debit).
* - View the complete transaction history sorted by date.
* - Edit existing transactions to correct or update data.
* - Delete transactions that are no longer required.

## Real-time Financial Summary

* - Automatically calculates and displays:
* - - Total Balance = Total Income − Total Expenses
* - - Total Income = Sum of all credits
* - - Total Expenses = Sum of all debits
* - Updates instantly with every transaction.

## Recent Transactions View

* - Displays the five most recent transactions for quick access.
* - Helps users stay on top of their most recent activities.

## Responsive UI

* - Clean and modern interface built with Tailwind CSS.
* - Fully responsive and mobile-friendly.
* - Real-time updates via React state management.

# 5. Technologies Used

## Frontend

* - React (Vite)
* - Tailwind CSS
* - Axios

## Backend

* - Node.js
* - Express.js
* - MongoDB
* - Mongoose
* - dotenv
* - cors

# 6. Database Design

Collections:

## 1. Users

- \_id (ObjectId)  
- username (String, unique)  
- password (String, hashed)  
- email (String, optional)

## 2. Transactions

- \_id (ObjectId)  
- userId (ObjectId, ref to Users)  
- description (String)  
- amount (Number)  
- type (String – "credit" or "debit")  
- date (Date)

# 7. API Documentation

## Authentication

- POST /login – Login user and return session token

\*(Future scope: Add registration, logout, token refresh)\*

## Expense Routes

* - POST /expenses – Add a new expense (requires login)
* - GET /expenses – Fetch all transactions for the logged-in user
* - GET /expenses/recent – Get the five most recent transactions
* - PUT /expenses/:id – Update a transaction by ID
* - DELETE /expenses/:id – Delete a transaction

# 8. Future Enhancements

* - User registration and password recovery
* - Charts and analytics for financial insights
* - Category tagging (e.g., groceries, bills, entertainment)
* - Export transactions as PDF/CSV
* - PWA support for offline tracking

# 9. Conclusion

The Expense Tracker project successfully meets its objectives of providing an easy-to-use platform for managing finances. It leverages modern full-stack technologies to deliver a robust and secure experience. With planned enhancements and scalability in mind, this project can grow into a powerful personal finance tool for individual users and small households.

# 10. Team Contribution

## Member 1 – Rishabh Paliwal (22BCE11091)

* - Developed the backend API using Node.js and Express.
* - Designed and implemented MongoDB schema with Mongoose.
* - Handled integration of API routes for transactions.

## Member 2 – Harsh Raj (22BCG10036)

* - Built the frontend interface using React and Vite.
* - Designed the user interface using Tailwind CSS.
* - Managed state and Axios integration for data communication.

## Member 3 – Aansh Garg (22BEY10002)

* - Implemented user authentication and session management.
* - Secured API routes and connected login page with backend.
* - Tested authentication workflows and handled errors.

## Member 4 – Anshul Sharma (22BEY10009)

* - Managed project documentation and system architecture design.
* - Tested features including add/update/delete transactions.
* - Prepared project presentation and future enhancement plan.