# PROJECT DESIGN

## 4.1 Problem-Solution Fit

Managing personal finances is a common challenge faced by individuals, especially students, freelancers, and young professionals. These users often rely on a mix of manual methods (spreadsheets, notes) or disconnected mobile apps that lack consistency, security, and actionable insights. This fragmented approach makes it difficult to monitor spending habits, analyze financial trends, or gain a clear overview of their income and expenses.

Through the use of **Empathy Mapping** and user-centered design techniques, we identified key pain points:

* **Lack of centralized financial tracking** – Users struggle with switching between apps or manual entries to understand their financial standing.
* **Limited data visualization** – Many solutions present data in raw formats without meaningful summaries or trend-based insights.
* **Usability concerns** – Financial apps are often cluttered or unintuitive, leading to user frustration, especially on mobile devices.
* **Data security and trust** – Users are increasingly concerned about the privacy of their financial data and expect secure authentication and storage mechanisms.

Our Expense Tracker addresses these concerns by providing a unified, visually rich, and secure platform to simplify the financial management journey. The application is structured to support intuitive navigation, responsive interactions, and real-time data updates. With features like categorized transactions, data export/import capabilities, and user profile personalization, it not only aids in tracking but empowers users to take control of their finances confidently.

## 4.2 Proposed Solution

The proposed solution is a full-stack MERN (MongoDB, Express, React, Node.js) web application that allows users to log income and expenses, view summaries and trends through charts, and manage their profile securely. Key features include:  
- Intuitive dashboard with pie, bar, and line charts  
- Secure login/signup with JWT authentication  
- Transaction management (add, delete, view)  
- Export/import financial data (Excel)  
- Responsive UI for mobile and desktop  
- Profile image upload and personalization

## 4.3 Solution Architecture

The solution follows a layered architecture with the following components:  
  
Frontend (React + Vite + Tailwind CSS):  
- Handles routing, state management (via UserContext), and rendering of UI components (charts, modals, tables).  
- Uses axios to communicate with backend APIs.  
  
Backend (Node.js + Express):  
- Manages authentication, income/expense CRUD operations, and file upload.  
- Connects to MongoDB Atlas using Mongoose for schema-based data modeling.  
- Implements route protection using JWT-based middleware.  
  
Database (MongoDB):  
- Stores users, income, and expense documents with timestamps.  
  
Static and File Upload:  
- Profile images and Excel files are stored in the server's uploads/ directory and served statically.  
  
  
  
  
