PERSONAL EXPENSE TRACKER APP

# INTRODUCTION

Welcome to our expense tracker app! Our app is designed to help you easily track and manage your expenses, making it convenient for you to stay on top of your financial transactions.

Whether you are a student, a working professional, or a business owner, our app is designed to meet your expense tracking needs.

# DESCRIPTION:

The Expense Tracker app offers a comprehensive solution for users to manage their finances efficiently. With a user-friendly interface, users can easily record their expenses, categorize them, and set budget limits. The app provides real-time notifications to alert users when they approach or exceed their budget limits. Additionally, users can generate detailed reports to analyze their spending patterns and make informed financial decisions. The app prioritizes security and privacy, ensuring users' financial data remains protected. With robust backend functionalities, including user authentication and database management, the Expense Tracker app offers a seamless and hassle-free experience for users to take control of their finances.

With a user-friendly interface and intuitive design, our expense tracker app allows you to effortlessly record your expenses, categorize them, and analyze your spending patterns. You can easily add new transactions, specify the category, and add relevant details such as date, vendor, and payment method.

We prioritize user satisfaction and aim to provide a seamless and hassle-free experience for managing your finances. Our app enables you to set budgets, track your spending against those budgets, and receive notifications when you exceed your limits. You can also generate reports and visualize your expenses through graphs and charts for better financial insights.

For business users, our app offers robust backend functionalities. You can manage multiple accounts, track expenses for different projects or departments, and generate detailed expense reports for accounting purposes. Administrators can efficiently handle user inquiries, ensure data accuracy, and monitor the overall performance of the app.

Our expense tracker app prioritizes your security and privacy. We employ encryption and password protection to safeguard your financial data, ensuring confidentiality. With our secure platform, you can manage expenses with peace of mind.

We are excited to have you on board and look forward to providing you with a reliable and user- friendly expense tracking experience. Start tracking your expenses with our app today and take control of your finances.

# SCENARIO BASED CASE STUDY

**Scenario 1: Personal Finance Management**

**User Profile:** Alice - Working Professional

**Scenario Overview:** Alice, a working professional, aims to take control of her finances using the Expense Tracker app. She diligently tracks her expenses, sets budget limits, and generates reports to analyze her spending patterns.

**Key Actions:**

1. **Registration and Login:**
   * Alice downloads the Expense Tracker app and completes the registration process.
   * Upon successful registration, she logs in to the app and accesses the dashboard.
2. **Expense Tracking:**
   * Alice starts adding her expenses for the month into the app's interface.
   * She categorizes expenses into groceries, utilities, entertainment, transportation, etc.
   * Alice monitors her spending and receives notifications when approaching or exceeding budget limits.
3. **Budget Analysis:**
   * Throughout the month, Alice continues tracking expenses and adjusting her budget.
   * At the month-end, she generates a report to analyze spending patterns.
   * Alice identifies areas to cut back on expenses and sets new financial goals for the next month.

**Scenario 2: Business Expense Management**

**User Profile:** Bob - Small Business Owner

**Scenario Overview:** Bob, a small business owner, utilizes the Expense Tracker app to manage his company's expenses efficiently. He sets up categories, tracks employee expenses, and generates detailed reports for accounting purposes.

**Key Actions:**

1. **Account Setup:**
   * Bob creates an account for his business on the Expense Tracker app.
   * He logs in and accesses the dashboard to set up expense categories and budget limits.
2. **Expense Tracking for Employees:**
   * Bob encourages his employees to use the app to record expenses.
   * Employees add expenses, specify categories, and upload receipts or invoices.
3. **Backend Functionality:**
   * As the administrator, Bob has access to backend functionalities.
   * He can view all expense records, track spending for different projects/departments, and generate reports.

**Scenario 3: Student Budgeting**

**User Profile:** Charlie - College Student

**Scenario Overview:** Charlie, a college student, adopts the Expense Tracker app to manage his finances responsibly while studying. He categorizes expenses, sets budget limits, and utilizes notifications to avoid overspending.

**Key Actions:**

1. **Expense Categorization and Budgeting:**
   * Charlie categorizes expenses into textbooks, dining out, transportation, etc.
   * He sets realistic budget limits based on his monthly allowance and academic expenses.
2. **Expense Monitoring:**
   * Throughout the month, Charlie records expenses promptly using the app.
   * He receives notifications when approaching or exceeding budget limits, helping him prioritize expenses.
3. **Financial Analysis and Adjustment:**
   * At the end of the month, Charlie reviews spending patterns and expense reports.
   * He identifies areas to save money and adjusts his budget for the upcoming month accordingly.

**Scenario 4: Family Budget Planning**

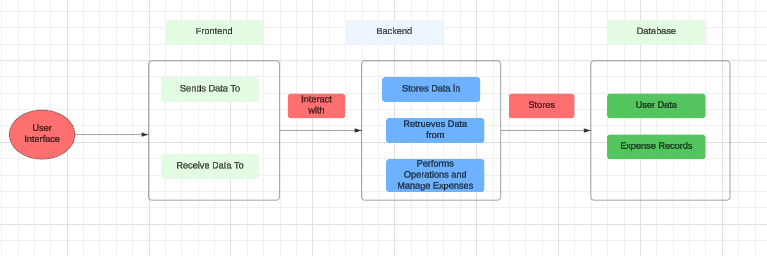
**User Profile:** David and Emily - Parents with children

**Scenario Overview:** David and Emily, a couple with children, seek to organize their family finances effectively using the Expense Tracker app. They aim to manage household expenses, save for future goals, and teach their children financial responsibility.

**Key Actions:**

1. **Family Account Setup:**
   * David and Emily create a joint account for their family on the Expense Tracker app.
   * They customize the account settings to include categories relevant to their family's expenses, such as groceries, childcare, education, utilities, and family outings.
2. **Expense Tracking for Family Activities:**
   * David and Emily record all family-related expenses, including groceries, children's activities, and household bills, in the app.
   * They involve their children in the process, teaching them the importance of tracking spending and budgeting.
3. **Budget Allocation and Saving Goals:**
   * David and Emily set monthly budgets for each expense category based on their family's financial goals and priorities.
   * They allocate a portion of their income towards savings for emergencies, education funds for their children, and family vacations.
4. **Financial Education for Children:**
   * David and Emily use the Expense Tracker app as a tool to educate their children about money management.
   * They involve their children in budget discussions, encourage them to track their own spending for allowances or savings, and discuss the family's financial decisions openly.
5. **Regular Financial Reviews:**
   * At regular intervals, David and Emily review their family's financial status using the app's reporting features.
   * They assess their spending patterns, evaluate progress towards savings goals, and make adjustments to their budget as necessary to ensure financial stability and growth for their family.

## TECHNICAL ARCHITECTURE:



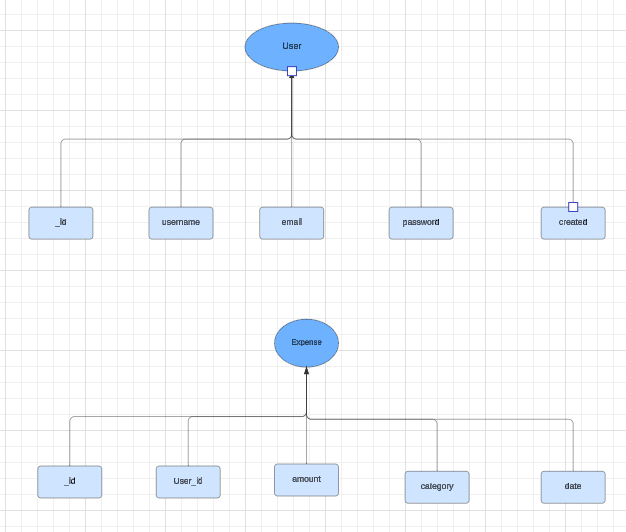
The technical architecture for an expense tracker typically consists of three main components: the User Interface, the Backend/API, and the Database.

**User Interface:** This component is responsible for providing a user-friendly interface for users to interact with the app. It includes features such as input forms for adding expenses, displaying expense lists, and generating charts or visualizations of expense data. The User Interface communicates with the Backend/API to send and receive data.

**Backend/API:** The Backend or API serves as the core logic and functionality of the expense tracker app. It handles business operations, data processing, and communication between the User Interface and the Database. The Backend/API includes modules for authentication, expense management, and any other necessary functionalities.

**Database:** The Database stores and manages the data for the expense tracker app. It includes tables or collections to store expense records, user information, and any other relevant data. The Database interacts with the Backend/API to store and retrieve data as needed.

# ER-DIAGRAM:

****

### User entity:

* user\_id (Primary Key): A unique identifier for each user.
* username: The username chosen by the user.
* email: The email address associated with the user.
* password: The password for the user's account.
* created\_at: The timestamp indicating when the user account was created.

### Expense entity:

* expense\_id: A unique identifier for each expense.
* user\_id (Foreign Key): A reference to the user who made the expense.
* amount: The cost or amount spent for the expense.
* category: The category or type of expense (e.g., food, transportation, entertainment).
* date: The date when the expense was incurred.

The relationship between the "User" and "Expense" entities is represented by the user\_id attribute in the "Expense" entity, which serves as a foreign key referencing the primary key (user\_id) of the "User" entity. This indicates that each expense is associated with a specific user

#### PRE REQUISITES:

To develop a full-stack expense tracker app using React js, Node.js, and MongoDB, there are several prerequisites you should consider. Here are the key prerequisites for developing such an application:

**Node.js and npm:** Install Node.js, which includes npm (Node Package Manager), on your development machine. Node.js is required to run JavaScript on the server side.

* + Download: <https://nodejs.org/en/download/>
  + Installation instructions: <https://nodejs.org/en/download/package-manager/>

**MongoDB:** Set up a MongoDB database to store hotel and booking information. Install MongoDB locally or use a cloud-based MongoDB service.

* + Download: <https://www.mongodb.com/try/download/community>
  + Installation instructions: <https://docs.mongodb.com/manual/installation/>

**Express.js:** Express.js is a web application framework for Node.js. Install Express.js to handle server-side routing, middleware, and API development.

* + Installation: Open your command prompt or terminal and run the following command: **npm install express**

**React**: React is a JavaScript library for building user interfaces. To create and manage your React project, you can use Create React App, a popular tool for bootstrapping React applications.

#### Install Create React App:

React offers a command-line tool called Create React App that simplifies project setup and development.

#### To install Create React App globally, run the following command:

npm install -g create-react-app

#### Create a new React project:

Choose or create a directory where you want to set up your React project. Open your terminal or command prompt.

Navigate to the selected directory using the cd command.

#### Create a new React project by running the following command:

npx create-react-app my-react-app

Replace my-react-app with your preferred project name. Wait for the project to be created. Navigate into the project directory:

After the project creation is complete, navigate into the project directory by running the

following command:

my-react-app

cd

#### Start the development server:

To launch the development server and see your React app in the browser, run the following command:

npm start

Create React App will compile your app and start the development server.

Open your web browser and navigate to http://localhost:3000 to see your React app running.

You have successfully set up React on your machine and created a new React project. You can now start building your app by modifying the generated project files in the src directory.

Please note that these instructions provide a basic setup for React. You can explore more advanced configurations and features by referring to the official React documentation: https://reactjs.org

**HTML, CSS, and JavaScript:** Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

**Database Connectivity:** Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Node.js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations.

**Front-end Framework:** Utilize React to build the user-facing part of the application, including products listings, booking forms, and user interfaces for the admin dashboard.

**Version Control**: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.

* Git: Download and installation instructions can be found at: [https://git-](https://git-scm.com/downloads) [scm.com/downloads](https://git-scm.com/downloads)

**Development Environment:** Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

* Visual Studio Code: Download from<https://code.visualstudio.com/download>
* Sublime Text: Download from <https://www.sublimetext.com/download>
* WebStorm: Download from <https://www.jetbrains.com/webstorm/download>

#### To Connect the Database with Node JS go through the below provided link:

* Link: [https://www.section.io/engineering-education/nodejs- mongoosejs-](https://www.section.io/engineering-education/nodejs-%20mongoosejs-mongodb/) [mongodb/](https://www.section.io/engineering-education/nodejs-%20mongoosejs-mongodb/)

#### To run the existing expense-web app project downloaded from github: Follow below steps:

1. **Clone the Repository:**
   * Open your terminal or command prompt.
   * Navigate to the directory where you want to store the grocery-webapp app.
   * Execute the following command to clone the repository:

**git clone** [https://github.com/Bharath136/ExpenseTacker-MERN.git](https://github.com/Bharath136/ExpenseTacker-MERN.git%20)

Install Dependencies:

* + Navigate into the cloned repository directory: cd expense -webapp
  + Install the required dependencies by running the following command: npm install

#### Start the Development Server:

* + To start the development server, execute the following command: npm run dev or npm run start
  + The e-commerce app will be accessible at http://localhost:5100 by default. You can change the port configuration in the .env file if needed.

#### Access the App:

* + Open your web browser and navigate to http://localhost:5100.
  + You should see the grocery-webapp app's homepage, indicating that the installation and setup were successful.

**Video Tutorial Link to clone the project: -** [https://drive.google.com/file/d/1KTGK0XZj0XWOiDeNKJVRKQHXLyVWZYLM/view?usp=](https://drive.google.com/file/d/1KTGK0XZj0XWOiDeNKJVRKQHXLyVWZYLM/view?usp=sharing) [sharing](https://drive.google.com/file/d/1KTGK0XZj0XWOiDeNKJVRKQHXLyVWZYLM/view?usp=sharing)

**Project Repository Link:** <https://github.com/Bharath136/ExpenseTacker-MERN.git>

Congratulations! You have successfully installed and set up the expense-webapp app on your local machine. You can now proceed with further customization, development, and testing as needed.

## PROJECT STRUCTURE:

## 

The project structure may vary depending on the specific framework, programming language, or

development approach used. It's essential to organize the files and directories in a logical and

consistent manner to improve code maintainability and collaboration among developers.

**app/app.component.scss, src/app/app.component.spec.ts**: These files are part of the main

AppComponent, which serves as the root component for the Angular app. The component

handles the overall layout and includes the router outlet for loading different components based

on the current route.

The frontend structure assumes a React app and follows a modular approach. Here's a brief explanation of the main directories and files:

* **src/components:** This directory has minor components such as Login, Register, etc.,
* **src/pages:** The pages folder contains all the pages of the application like landing page, home page, etc.,

## APPLICATION FLOW:

1. **Account Creation and Setup:**
   * User registers for a new account by providing necessary information such as username, email, and password.
   * Upon successful registration, the user sets up and manages personal account details, including profile information like name, contact information, and preferred currency.
2. **Expense Tracking:**
   * User logs into the Expense Tracker app and accesses the dashboard.
   * From the dashboard, the user navigates to the "Add Expense" section.
   * User enters details such as amount, category, and date for each expense incurred.
   * Expenses are categorized appropriately (e.g., groceries, utilities, entertainment) for better organization and analysis.
   * User can edit or delete existing expenses if necessary to maintain accurate records.
   * The app provides real-time updates to ensure users have access to accurate and up-to-date information regarding their expenses.
3. **Budgeting and Goal Setting:**
   * User sets personal budget limits and financial goals for different expense categories (e.g., groceries, dining out, transportation).
   * User monitors and tracks expenses against the defined budget using visualizations or progress bars.
   * The app sends alerts or notifications to the user when they approach or exceed budget limits, helping them stay on track with their financial goals.
4. **Reporting and Analysis:**
   * User accesses reports and summaries of expenses from the dashboard.
   * User can analyze expenses based on different categories, time periods, or custom filters to gain insights into spending patterns and trends.
   * The app provides visual representations such as charts or graphs to facilitate easier analysis and decision-making regarding financial management.
5. **Data Security and Privacy:**
   * The Expense Tracker app ensures the security and privacy of user data by employing encryption and password protection mechanisms.
   * It follows best practices to protect personal financial information and complies with relevant regulations regarding data security and privacy.
   * Users are encouraged to use strong passwords and enable two-factor authentication for added security.
   * The app has a support or security team that users can contact to report any security concerns or suspicious activities, ensuring prompt resolution and mitigation of potential risks.
6. **Additional Functionality:**
   * **Add Income:** Similar to adding an expense, users can add income details, which are saved to the database and associated with the user.
   * **Delete Month:** Users can delete a month, removing all expenses and income records associated with that month for a specific user.
   * **Download Report:** Users can generate and download reports summarizing their expenses, income, or other relevant financial information.
   * **Add Row/Delete Row:** Users can add or delete rows within expense or income records to manage details effectively.

# PROJECT FLOW:

## Milestone 1: Project Setup and Configuration:

## Create project folders and files:

## Now, firstly create the folders for frontend and backend to write the respective code and install the essential libraries.

* + Client folders.
  + Server folders

1. **Install required tools and software:**

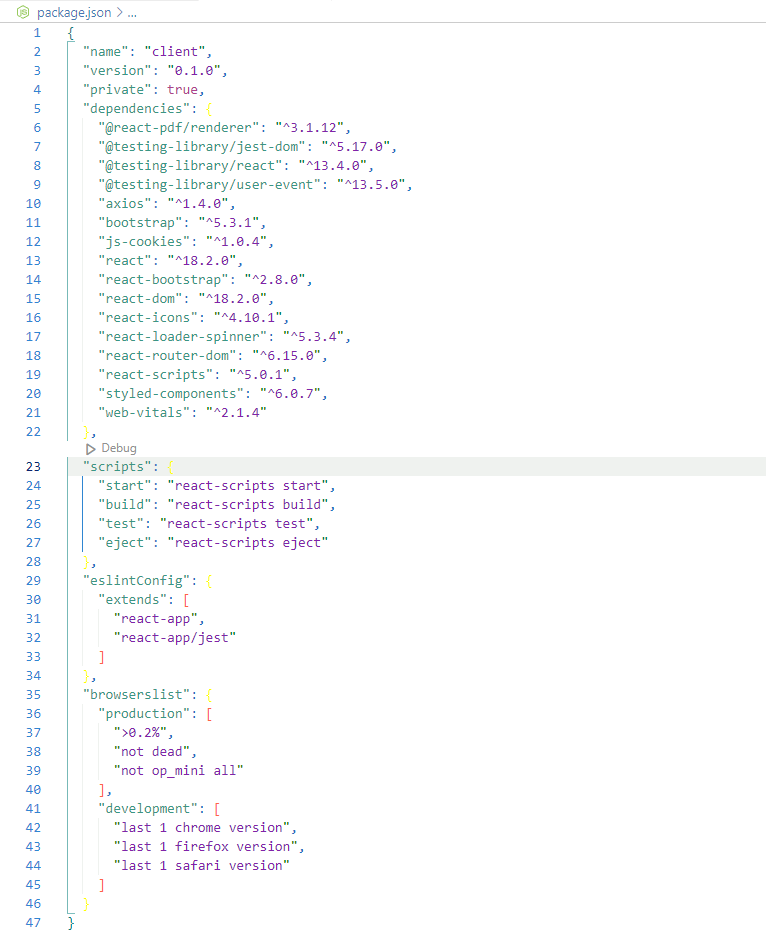
For the backend to function well, we use the libraries mentioned in the prerequisites. Those libraries includes

* + Node.js.
  + MongoDB.
  + Bcrypt
  + Body-parser

Also, for the frontend we use the libraries such as

* + React Js.
  + Material UI
  + Bootstrap
  + Axios

After the installation of all the libraries, the package.json files for the frontend looks like the one mentioned below.



After the installation of all the libraries, the package.json files for the backend looks like the one mentioned below.



## Milestone 2: Backend Development:

#### Set Up Project Structure:

* Create a new directory for your project and set up a package.json file using npm init command.
* Install necessary dependencies such as Express.js, Mongoose, and other required packages.

#### Set Up Project Structure:

* + Create a new directory for your project and set up a package.json file using npm init command.
  + Install necessary dependencies such as Express.js, Mongoose, and other required packages.

#### Create Express.js Server:

* + Set up an Express.js server to handle HTTP requests and serve API endpoints.
  + Configure middleware such as body-parser for parsing request bodies and cors for handling cross-origin requests.

#### Define API Routes:

* + Create separate route files for different API functionalities such as authentication, stock actions, and transactions.
  + Implement route handlers using Express.js to handle requests and interact with the database.

#### Implement Data Models:

* + Define Mongoose schemas for the different data entities like Add, Remove, transactions, deposits and Users.
  + Create corresponding Mongoose models to interact with the MongoDB database.
  + Implement CRUD operations (Create, Read, Update, Delete) for each model to perform database operations.

#### User Authentication:

* + Implement user authentication using strategies like JSON Web Tokens (JWT) or session-based authentication.
  + Create routes and middleware for user registration, login, and logout.
  + Set up authentication middleware to protect routes that require user authentication.

#### Handle new transactions:

* + Allow users to make transactions to other users using the user’s account id.
  + Update the transactions and account balance dynamically in real-time.

#### Admin Functionality:

* + Implement routes and controllers specific to admin functionalities such as fetching all the data regarding users, transactions, stocks and orders.

#### Error Handling:

* + Implement error handling middleware to catch and handle any errors that occur during the API requests.
  + Return appropriate error responses with relevant error messages and HTTP status codes.

Reference video for backend code:

<https://drive.google.com/file/d/1fim4-RGij-9ghk61j0ISKIxfuTPA7S9y/view?usp=drive_link>

**Milestone 3: Database Development:**

1. **Setup MongoDB:**
   * Install MongoDB locally or use a cloud-based service like MongoDB Atlas.
   * Ensure MongoDB is running and accessible from the application.
2. **Define Schemas:**
   * Create userSchema and expenseSchema using Mongoose.
   * Define the required fields for each schema, such as firstname, lastname, email, and password for users, and userId, monthYear, tables, and calculations for expenses.
3. **Create Models:**
   * Define Mongoose models for User and Expense using the schemas.
   * These models will represent collections in the MongoDB database and provide methods for interacting with the data.
4. **Connect to MongoDB:**
   * Set up a connection to MongoDB using Mongoose in the application's entry point (e.g., app.js or server.js).
   * Use the connection string to connect to the MongoDB database.
5. **Test Connection:**
   * Ensure the application successfully connects to the MongoDB database without any errors.
6. Test the connection by performing basic CRUD operations on the User and Expense models.

Reference video for Database connection:

<https://drive.google.com/file/d/1CQil5KzGnPvkVOPWTLP0h-Bu2bXhq7A3/view?usp=sharing>

**Milestone 4: Frontend Development**

1. **Setup React Application:**
   * Install required libraries using npm or yarn. This may include libraries for routing, state management (if needed), and any other dependencies.
   * Create the initial application structure with directories for components, containers, styles, and assets.
   * Organize project files for efficient development, ensuring a clear separation of concerns.
2. **Design UI Components:**
   * Create reusable components for interactive elements such as expense listings, input forms, buttons, and notifications.
   * Implement a layout and styling scheme to define the overall look and feel of the application. This should include consistent branding, typography, color palette, and spacing.
   * Design user interface elements with responsiveness in mind, ensuring compatibility across different screen sizes and devices.
   * Integrate a navigation system to allow seamless exploration of different sections of the expense tracker, such as adding expenses, viewing reports, and managing budgets.
3. **Implement Frontend Logic:**
   * Integrate frontend components with backend API endpoints for fetching and updating expense data.
   * Implement data binding to connect user interface elements with the underlying data model.
   * Develop logic for adding, editing, and deleting expenses, including validation to ensure data integrity.
   * Implement features for categorizing expenses, setting budgets, and generating reports based on user preferences.
4. Handle user authentication and authorization to restrict access to certain features or data based on user roles.

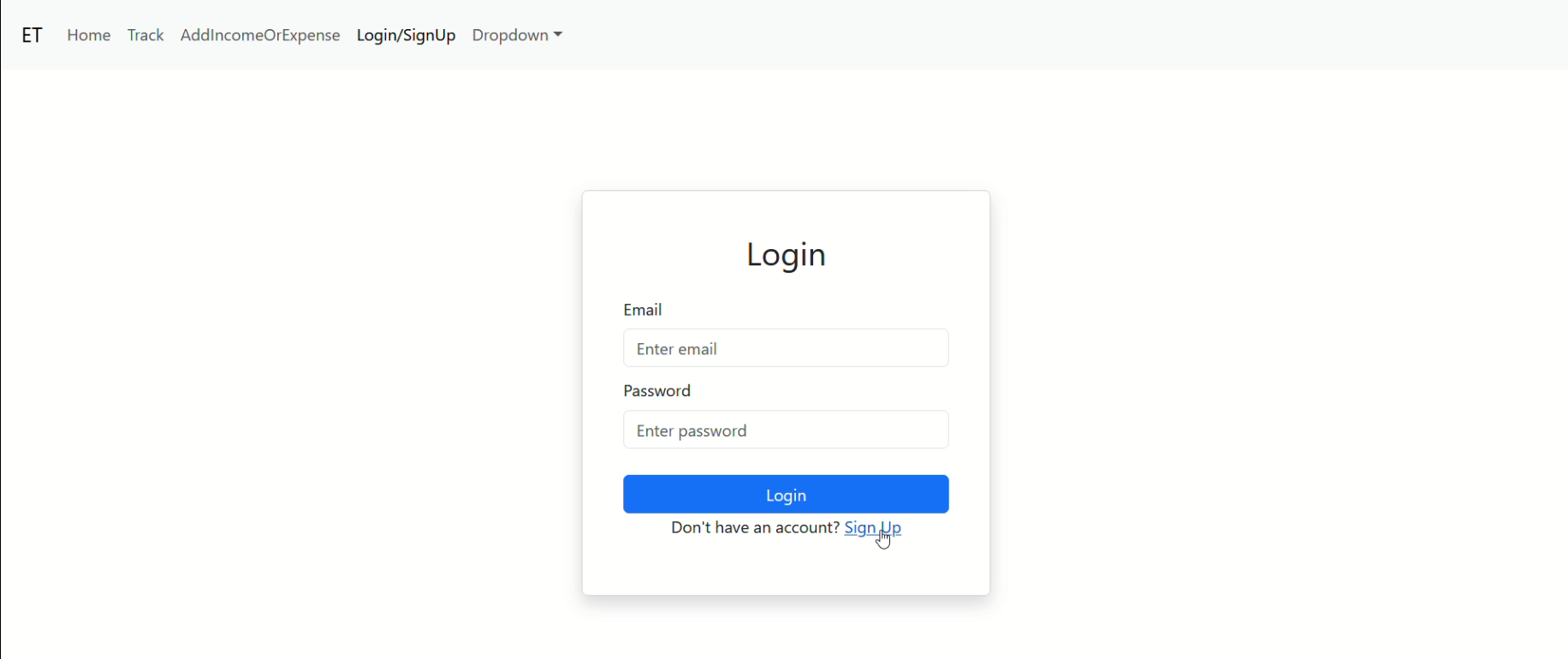
Reference video for Frontend connection:

<https://drive.google.com/file/d/1KHOI96pcvgvgkGlXwFlYQmamv2y_IpbP/view?usp=drive_link>

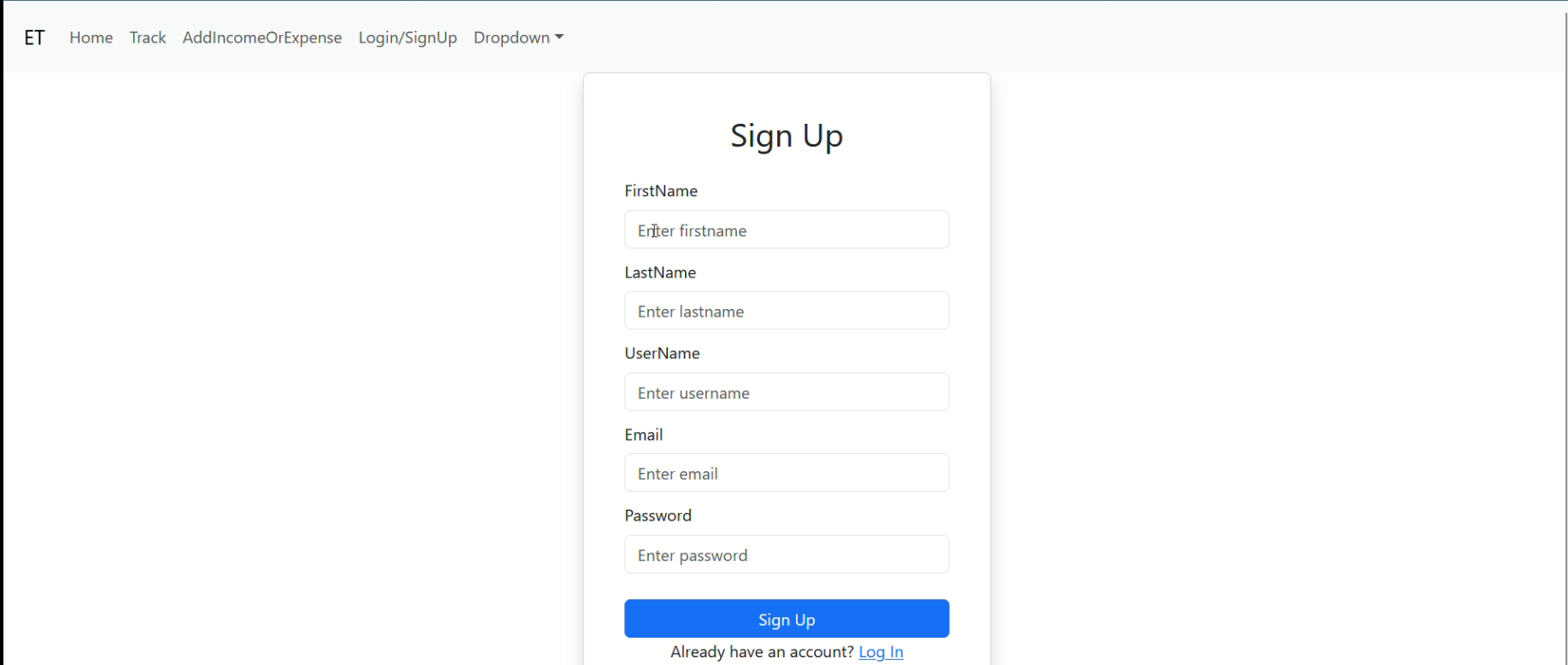
**Milestone 5: Project Implementation:**

On completing the development part, we then run the application one last time to verify all the functionalities and look for any bugs in it. The user interface of the application looks a bit like the one’s provided below.

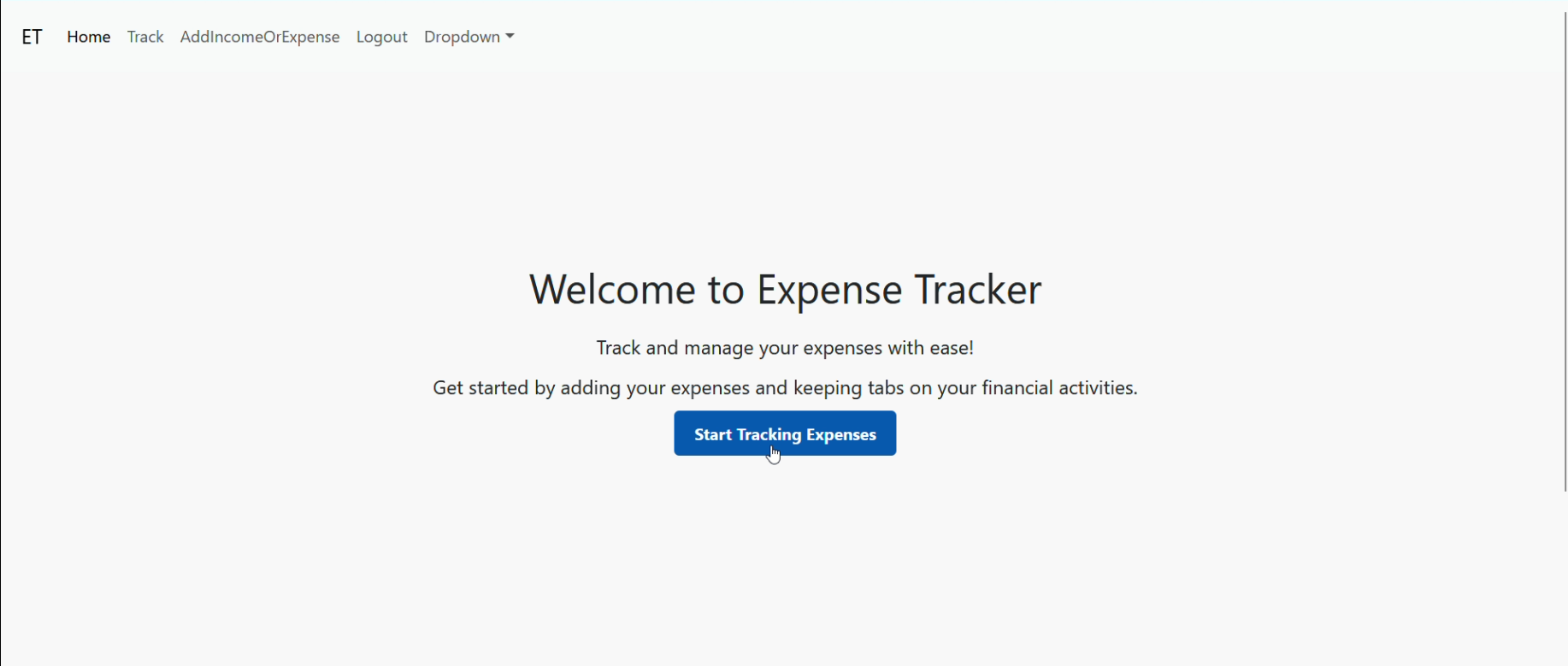
* Login Page



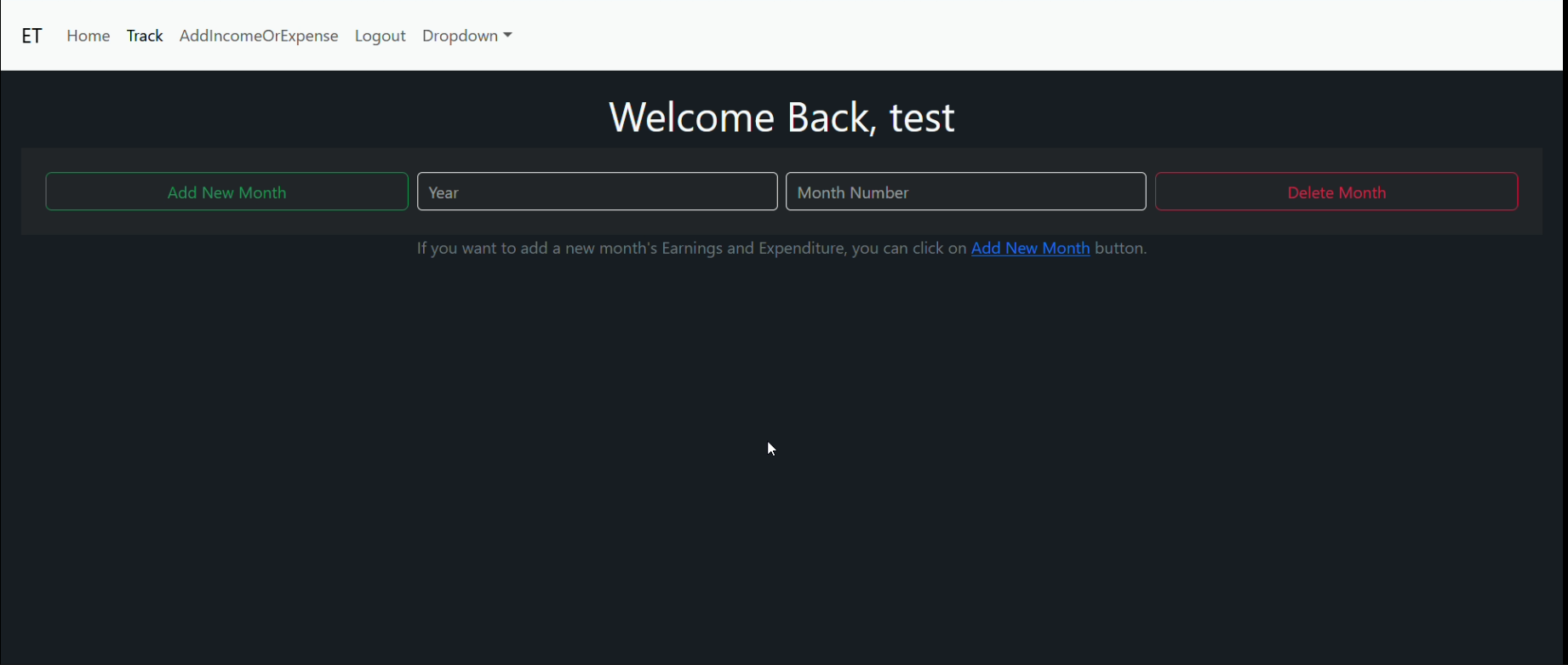
* Registration Page



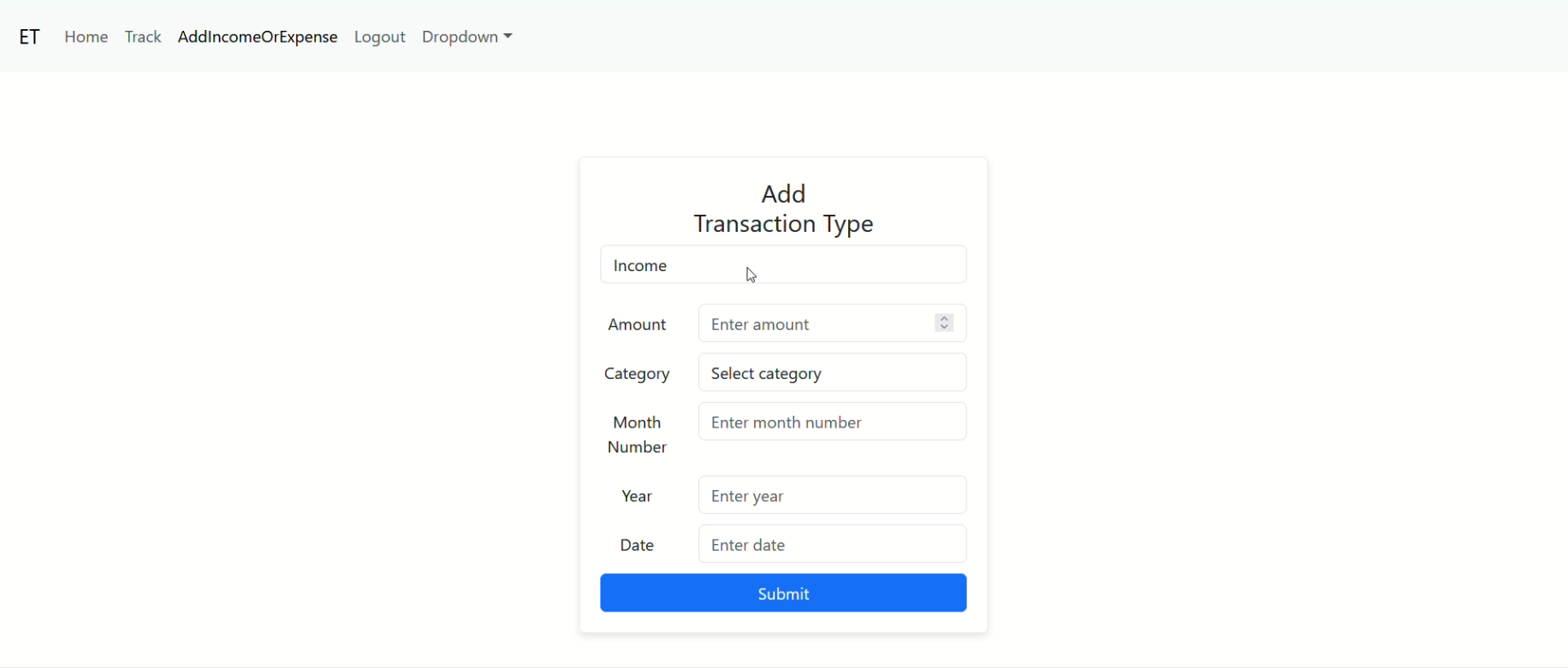
* Landing page



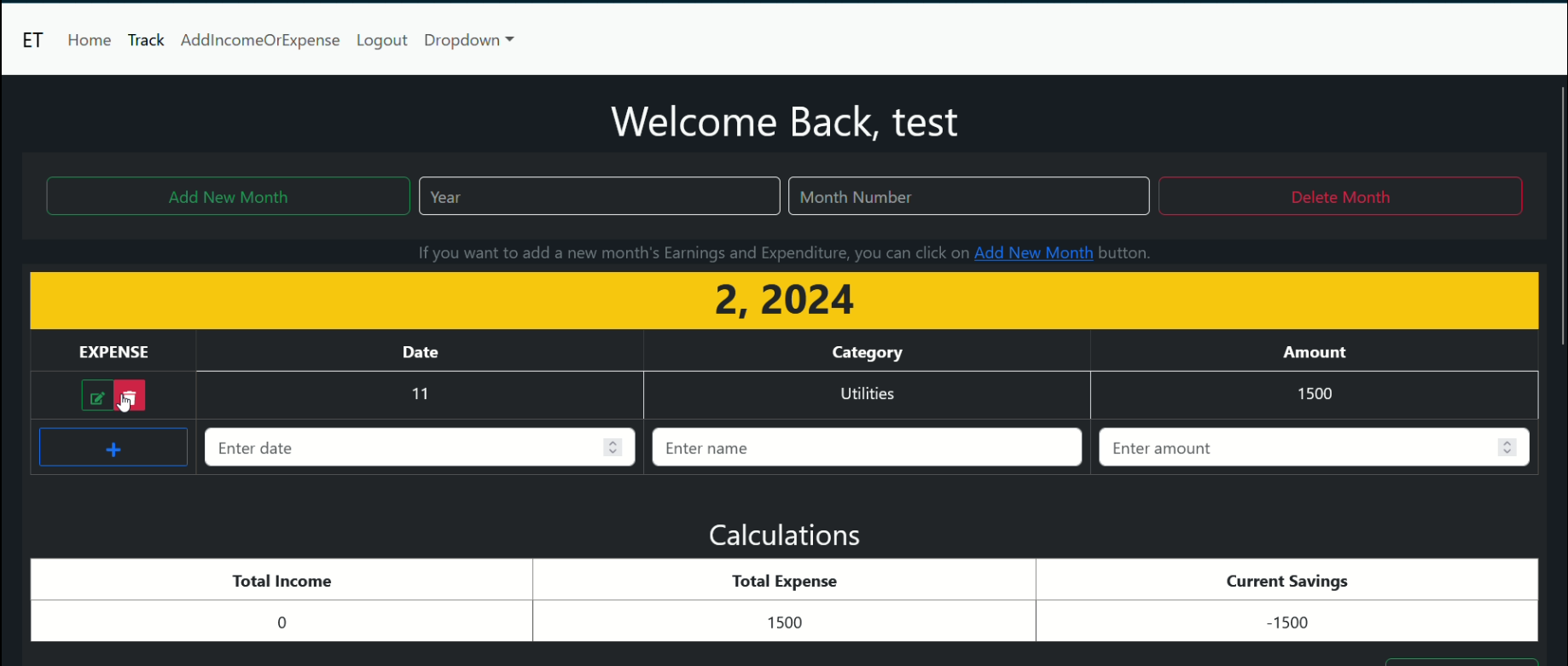
* User Page



* Adding Expense Page



* Expense tracker Page

****

For any further doubts or help, please consider the code from the google drive,

<https://drive.google.com/drive/folders/17nJJVAHSB759YqMLWC0Y3Dr1jXKOVBHd?usp=drive_link>

The demo of the app is available at:

<https://drive.google.com/file/d/1VNXvcEv_AOkUWNr2rFlw-rMLMef3YZHH/view?usp=drive_link>