**Project Report**

on

“EXPENSE TRACKER APP”

***Submitted in partial fulfillment of the***

***requirement for the award of the degree of***

B.TECH



**Under the Supervision of:**

**Mr. JANARTHANAN.S**

**Assistant Professor**

Submitted By:

**KARANJEET SINGH 21SCSE1011644**

**ABHINAV KUMAR CHOUDHARY 21SCSE1011615**

**SHIVANI KUMARI 21SCSE1011614**

**EBAD ZAFAR 21SCSE1011484**

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING / DEPARTMENT OF COMPUTER APPLICATION, GALGOTIAS UNIVERSITY, GREATER NOIDA**

**INDIA**

**MAY,2023**

**Table of Contents**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Title** | |  | | | **Page No.** |
| **Abstract** | | | | | **III** |
| **Chapter 1** | **Introduction** | | | | **1** |
|  | 1.1 | | Problem Statement | | **2** |
|  | 1.2 | | Project Objective | | **3** |
|  | 1.3 | | Project Scope | |  |
|  | 1.4 | | Assumptions | |  |
|  | 1.5 | | Requirements | |  |
| **Chapter 2** | **Literature Survey** | | | | **6** |
| **Chapter 3** | **Tools and Technology used** | | | | **7** |
| **Chapter 4** | **Product Feature** | | | | **8** |
| **Chapter 5** | **User & Communication Interface** | | | | **9-10** |
| **Chapter 6** | **Project Architecture** | | | | **11** |
| **Chapter 7** | **JWT** | | | | **12** |
| **Chapter 8** | **Conclusion** | | | | **13** |
|  | **Reference** | | | | **14** |
|  |  | |  |  |  |

**ABSTRACT**

### The purpose of this project was to build an expense tracker application using the MERN stack. MERN stands for MongoDB, Express, React, and Node.js, and it is a popular technology stack for building full-stack web applications. The expense tracker application allows users to keep track of their daily expenses, categorize their spending, and generate reports for their spending habits. The application has a user-friendly interface that makes it easy for users to add, edit, and delete expenses, view their spending history, and generate reports. The data is stored in a MongoDB database, which is accessible through a RESTful API built with Express. The front-end is built using React, providing a responsive and interactive user interface, and the application is hosted on a Node.js server. The project also covered the testing and deployment process, and the challenges that were faced during the development process. Overall, the expense tracker application provides a simple and effective solution for managing daily expenses, and serves as a good example of how the MERN stack can be used to build full-stack web applications.

# Introduction

## Problem Statement

### The problem that the expense tracker application built using the MERN stack aims to solve is the difficulty that many people face in tracking their daily expenses. In today's fast-paced world, it is easy to lose track of how much money is being spent on a daily basis, making it difficult to budget and save. Additionally, many traditional methods of tracking expenses, such as pen and paper, can be time-consuming and prone to errors. The expense tracker application provides a simple and efficient solution to this problem by allowing users to keep track of their daily expenses in one central location.

## Project Objectives

### The expense tracker application solves these challenges by providing users with a convenient, accurate, and easy-to-use platform for tracking their expenses. With the ability to add, edit, and delete expenses in real-time, users can have complete control over their spending and make informed decisions about budgeting and saving. Additionally, the application's reporting capabilities provide valuable insights into users' spending habits, making it easier to identify areas where changes can be made to improve financial stability.

## Project Scope

### The problem scope for the expense tracker application built using the MERN stack is to provide a solution for individuals and households to effectively manage and track their daily expenses. The following are some of the specific problems addressed by the application:

### Lack of Accessibility: Many individuals find it challenging to keep track of their expenses, especially when they are on the go. The expense tracker application solves this problem by providing a platform that is accessible from any device with an internet connection.

### Inefficient Spending Habits: People often spend money without realizing how much they are spending on specific items or categories. The expense tracker application provides users with a comprehensive view of their spending patterns and helps them make informed decisions about their finances.

### Unsecured Data:Storing financial information on paper or spreadsheets can be risky, as it can be lost, damaged, or stolen. The expense tracker application solves this problem by securely storing all user data in a MongoDB database, which is encrypted and only accessible through a secure API.

## **Assumptions**

Here are some of the assumptions made during the development of the expense tracker application built in the MERN stack:

* User familiarity with the MERN stack: It is assumed that the users are familiar with the concepts and technologies used in the MERN stack, such as MongoDB, Express, React, and Node.js.
* Internet connectivity: It is assumed that the users have a stable internet connection to access the application and use its features.
* Device compatibility: The application is designed to work on a variety of devices, including smartphones, tablets, and desktops. However, it is assumed that the users have a device with a modern web browser that is capable of running the application.
* Data privacy: The application stores all user data in a MongoDB database, which is encrypted and secure. However, it is assumed that users understand that no system is completely secure and they should use the application at their own risk.
* User data accuracy: The accuracy of the user data is dependent on the users themselves. The application assumes that the users will accurately enter and categorize their expenses to get the most value from the reports generated by the application.
* Deployment environment: It is assumed that the application will be deployed on a Node.js server in a secure and stable environment. The application assumes that the deployment environment is properly configured and maintained to ensure its stability and security.
* Technical support: The application assumes that users will have access to technical support if they encounter any issues during the use of the application. The support may come from the developers of the application, onlineresources,orothersources.

**Literature Survey**

we have came across all the finance managable apps in the market available, specifically we are going to target those people, who need more specific and user friendly interface .we let the people choose their specificaton that is more flexible than all the apps available .we will try to put some intelligence in our app . we will try to accomodate in minimum size of app as that possible in the market .

Cost: While there are free expense tracker apps available, some may require a subscription fee or in-app purchases to access all features. Limited functionality: Some expense tracker apps may not have all the features users need, which can lead to frustration and the need to use multiple apps.

Existing apps like

1.Money Manager Expense & Budget

2. Wallet – Daily Budget & Profit

3. Buddy: Budget & Save Money

4. Spending Tracker

5. Rydoo

All have some specilities but every app like these falls back in either open sourcing or functionality.so we are going to make

an app defines the optimality between these features and open source free software

**Tools and Technology**

The tools and technology for the expense tracker application built in the MERN stack typically consists of the following components:

**Server-side Environment:**

* Node.js: A JavaScript runtime environment that is used to run the server-side logic of the application.
* Express: A framework for Node.js that is used to build the RESTful API for the application.
* MongoDB: A NoSQL database that is used to store the data for the application.

**Client-side Environment:**

* React: A JavaScript library for building user interfaces that is used to build the front-end of the application.

**Development Environment:**

* Code Editor: A code editor such as Visual Studio Code, Sublime Text, or Atom can be used to write and edit the code for the application.
* Package Manager: npm (Node Package Manager) is typically used to manage the dependencies for the application.

**Deployment Environment:**

* Web Server: A web server such as Apache or Nginx can be used to host the application and make it accessible to users.

Note that the operating environment for the expense tracker application built in the MERN stack can vary depending on the specific requirements and constraints of the project

**Product Features**

### Here are some key features of an expense tracker application built using the MERN stack:

### User-friendly interface: The application provides a simple and intuitive interface for users to manage their daily expenses.

### Add, edit, and delete expenses: Users can easily add, edit, and delete their expenses, allowing them to keep their records up-to-date.

### Categorize expenses: The application allows users to categorize their expenses, making it easier to understand where their money is being spent.

### View spending history: Users can view their spending history and get a clear picture of their financial status.

### Generate reports: The application generates reports for users' spending habits, making it easy to identify areas where they can save money.

### Secure data storage: All user data is stored in a secure MongoDB database, ensuring the privacy and security of the users' financial information.

### RESTful API: The application is built with a RESTful API, making it easy to integrate with other systems and applications.

### Responsive design: The front-end is built using React, providing a responsive and interactive user interface that works seamlessly on different devices.

### Deployment and hosting: The application is hosted on a Node.js server, making it easy to deploy and access from anywhere in the world.

### Testing and debugging: The development process includes testing and debugging to ensure the application is reliable and bug-free.

**User Interface**

### The user interface of the expense tracker application is designed to be simple, intuitive, and user-friendly. The main dashboard provides an overview of the user's expenses, including a pie chart that displays the expenses categorized by category, and a bar graph that shows the expenses over time. The dashboard also includes a summary of the total expenses and a button to add new expenses.

### When adding a new expense, the user is prompted to enter the details of the expense, including the date, category, amount, and description. The user can also upload a receipt for the expense if necessary.

### The expenses can be edited or deleted by clicking on the relevant expense in the expenses list. The expenses can also be filtered by category, date range, and keywords, making it easy for users to find the information they need.

### The user interface is responsive, so it works well on both desktop and mobile devices. The application also provides a clean and modern design, with a simple color scheme and easy-to-read typography.

**Communication Interfaces**

### In this project, the expense tracker application built using the MERN stack has several communication interfaces that allow for seamless data exchange between the different components of the system. The communication interfaces include:

### MongoDB and Express API: The MongoDB database is accessed through a RESTful API built using Express, which serves as a bridge between the front-end and the database. The API allows for data retrieval, storage, and manipulation in the database.

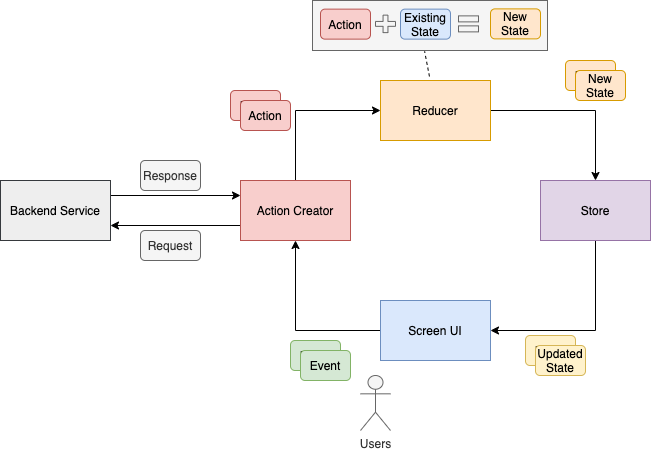
### React and Express API: The React front-end communicates with the Express API to retrieve and store data in the MongoDB database. This communication is facilitated through API calls using the Axios library.

### React and Node.js Server: The React front-end is hosted on a Node.js server, which provides a stable and secure environment for the application to run. The React and Node.js server communicate through the use of environment variables and web socket connections.

### User Interface: The user interface is the main communication interface for the end-user. It provides an intuitive and user-friendly interface for managing expenses and generating reports. The user interface communicates with the backend through API calls and web socket connections.

### These communication interfaces allow for efficient data exchange between the different components of the system, ensuring that the application runs smoothly and efficiently.

**Project Architecture**



### Deployment: The application can be deployed on a cloud platform, such as Heroku , Render or AWS, or it can be hosted on a local server. The deployment process involves configuring the server, installing dependencies, and setting up the database.

**JWT**

### JWT, or JSON Web Tokens, play an important role in securing the expense tracker application built using the MERN stack. JWT is a standard for representing claims securely between two parties. In the context of the expense tracker application, JWT can be used to authenticate users and ensure that only authorized users have access to sensitive information, such as their expenses and spending history.

### Here's how JWT works in the expense tracker application:

### Login: When a user logs into the application, the server generates a JWT and sends it back to the client. The JWT contains information about the user's identity, such as their username, and is signed using a secret key.

### Requests: For subsequent requests, the client sends the JWT along with each request to the server. The server uses the information in the JWT to authenticate the user and determine their access level.

### Authorization: Based on the information in the JWT, the server can either allow or deny access to the requested resource. This ensures that only authorized users can view or manipulate their own expenses and spending history.

**Structure of JWT**



**CONCLUSION**

### In conclusion, the expense tracker application built using the MERN stack provides a simple and effective solution for managing daily expenses. The use of React, Node.js, Express, and MongoDB allowed for the development of a full-stack web application that is both user-friendly and efficient. The addition of Redux Toolkit for state management and lodash for utility functions enhanced the overall performance and functionality of the application.

### The project demonstrated the power of the MERN stack in building full-stack web applications, and the ease of incorporating other technologies to add additional functionality. The expense tracker application can be used by individuals, families, or small businesses to keep track of their daily expenses, categorize their spending, and generate reports for their spending habits.

### Overall, the expense tracker application is a successful example of how the MERN stack can be used to build practical and useful web applications, and provides a foundation for future expansion and improvement.

**References:**

* <https://restfulapi.net>
* [www.ibm.com](http://www.ibm.com)
* [www.geekforgeeeks.com](http://www.geekforgeeeks.com)
* [www.w3schools.com](http://www.w3schools.com)
* <https://reactjs.com>
* <https://mongodb.com>
* <https://github.com>
* <https://nodejs.com>
* [www.javapoint.com](http://www.javapoint.com)