International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Finance Expense Tracker

K. Devi Priyanka[1], A.P.S. Mounika[2], G.Vineetha[3], K. Narmada[4], K. mariyamma[5], K.D.S.Manikant[6]

[1]Assistant Professor, Department of Electronics and Communication Engineering, VSM College of Engineering, Ramachandrapuram, Andhra Pradesh, India

[2,3,4,5,6]UG Students, Department of Electronics and Communication Engineering, VSM College of Engineering, Ramachandrapuram, Andhra Pradesh, India

ABSTRACT

The **Finance Expense Tracker** is a web-based solution designed to simplify and enhance personal financial management. This platform addresses common challenges such as unorganized expense tracking, lack of real-time insights, and the absence of automated reminders for overspending. Users can record their income and expenses, set spending limits, and receive timely notifications when limits are exceeded. Built with modern technologies such as React.js, Node.js, MongoDB, and Express.js, the platform ensures a seamless user experience and robust backend support. Secure authentication with password hashing and OTP verification protects sensitive user data. The platform's intuitive design offers real-time insights into financial activities, promoting better budgeting habits. The Expense Tracker aims to make financial planning accessible, efficient, and secure for individuals and families, empowering them to make informed decisions and achieve financial stability.

.

Keywords: Finance Tracker, Budget Limits, OTP Authentication, Income, Expense Management, Email Notifications, MERN Stack.

INTRODUCTION

The Finance Expense Tracker System presented in this project is a vital innovation aimed at enhancing financial management and budgeting efficiency in personal and business environments. Financial interfaces, where income intersects with expenditure, are inherently prone to potential missteps and budgetary concerns. Therefore, implementing a robust tracking system becomes imperative to mitigate risks and ensure smooth financial operations. The system integrates various components built on the MERN stack including MongoDB for database management, Express.js and Node.js for backend operations, React for the frontend interface, and Nodemailer for email notifications to create a comprehensive finance management ecosystem.

This system is designed to detect budget limit violations, track recurring expenses, and issue timely email alerts to users. The implementation of budget limits enables precise financial control, allowing the system to accurately identify when spending approaches or exceeds predefined thresholds. Upon detection, the system triggers appropriate responses, including sending email notifications, displaying visual alerts on the dashboard, or generating detailed reports. Additionally, OTP verification enhances security during user authentication, ensuring that only authorized users can access sensitive financial data..

The incorporation of Nodemailer facilitates real-time communication by sending alerts or updates via email to users when they approach or exceed their budget limits. MongoDB serves as the central database, orchestrating the functionality of the system, processing financial data, and executing programmed actions based on detected patterns. Its document-oriented structure makes it an ideal choice for coordinating the intricate operations of the Finance Expense Tracker System. Through the seamless integration of these components, the system offers a robust solution to enhance financial awareness and operational efficiency in personal and business environments. By promptly detecting and alerting users to potential budget issues, the system helps mitigate financial risks, prevent overspending, and ensure the smooth flow of financial management.

LITERATURE REVIEW

Throughout the development of our project, we were able to review some projects, journals, articles, and books which are related to the title of our project. We believe that all the reviewed materials have been a good asset for the overall design and development of the project we have chosen. In this section, some of the related projects we have reviewed are discussed.

Modern Web-Based Financial Management Systems

One of the key areas where expense tracking systems find application is in personal finance management, where they serve as the interface between daily spending activities and long-term financial planning.

The MERN stack (MongoDB, Express.js, React, Node.js) has emerged as a powerful technological foundation for developing financial applications. MongoDB's document- oriented database structure allows for flexible schema design, making it ideal for storing varying financial data types. Express.js provides a robust framework for building RESTful APIs that handle financial transactions securely. React offers an interactive and responsive user interface for visualizing financial data, while Node.js enables efficient server-side operations and real-time data processing.

Budget Limit and Financial Notification Systems

Notification systems are essential for alerting users about potential financial dangers, particularly when approaching or exceeding budget limits. Email notification systems like Nodemailer have become instrumental in delivering timely financial alerts to users. These notifications provide warnings to financial managers and account holders about approaching budget limits or unusual expenses. The implementation of budget thresholds allows for proactive financial management, enabling users to setspecific spending limits for various categories and receive alerts when these limits are approached or exceeded. Modern authentication systems, particularly those employing One-Time Passwords (OTP), have significantly enhanced the security of financial applications. By requiring users to verify their identity through a time-sensitive code sent to their registered email or mobile device, these systems add an extra layer of protection against unauthorized access to sensitive financial data. This multi-factor authentication approach is particularly crucial for applications handling personal financial information.

EXISTING METHOD

In existing Finance Expense Tracker Systems, the approach to financial management and efficiency has been largely limited to basic tracking without advanced budget control mechanisms. Typically, traditional systems only allow users to record income and expenses without implementing budget limits or notification systems. However, this method is prone to errors and may not provide timely responses to

emerging financial problems. Basic tracking systems such as spreadsheets or simple applications are often employed to record expenses, but these systems have limitations in their effectiveness for proactive financial management. Communication during financial anomalies or potential budget overruns is usually handled through manual checks, requiring users to regularly review their financial status. This can result in delays in recognizing critical information, thereby compromising financial discipline. Additionally, existing systems often lack comprehensive authentication mechanisms, potentially exposing sensitive financial data to security risks. Moreover, the notification capabilities of traditional expense systems are limited, making it challenging to provide timely alerts about financial issues. This limitation can lead to delayed awareness of emerging financial dangers, increasing the risk of overspending or misallocation of resources. Furthermore, the integration capabilities of existing systems are often restricted, making it difficult to implement features like third-party budget setting or collaborative financial management.

Limited Budget Control Mechanisms

Traditional expense tracking systems often lacked advanced budget control features, making it difficult for users to set and maintain spending limits. This could lead to uncontrolled spending and financial discipline issues..

Inadequate Authentication Security

Existing systems typically relied on basic username/password authentication without additional security layers such as OTP verification, increasing vulnerability to unauthorized access to sensitive financial data.

Absence of Notification Systems

Communication during financial anomalies or budget violations was often non-existent, with users having to manually check their financial status. This could result in delayed awareness of critical financial situations, compromising financial stability.

Lack of Collaborative Features

Existing systems typically lacked features for collaborative financial management, such as allowing trusted individuals to set budget limits for others, limiting their effectiveness for family or team financial planning.

PROPOSED SOLUTION

The proposed Finance Expense Tracker System aims to enhance financial management and efficiency by integrating advanced web technologies and financial control mechanisms. Key elements include MongoDB for flexible data storage, Express.js and Node.js for robust backend operations, React for interactive frontend interfaces, Nodemailer for email notifications, OTP verification for secure authentication, comprehensive budget limit setting, and collaborative financial management capabilities. This system offers improved financial control through customizable budget limits, ensuring timely email alerts when thresholds are approached or exceeded. Real-time communication via Nodemailer enables immediate awareness of financial situations. MongoDB serves as the central database, storing all financial transactions, user profiles, and budget settings. The Finance Expense Tracker System is designed to be both secure and user-friendly, with OTP verification ensuring that only authorized users can access sensitive financial information. Overall, the proposed system provides a comprehensive solution to enhance financial discipline and proactive management.

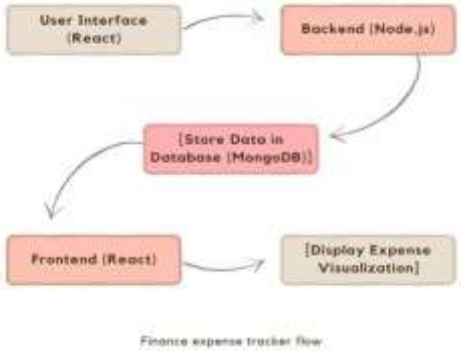


Fig.4.1.1: Block Diagram

SYSTEM COMPONENTS

User Interface (UI) - Frontend (React.js)

The frontend is developed using **React.js**, which allows the application to be highly interactive and responsive. React components handle the various views and pages of the website, such as:

Login/Registration Pages: For user authentication.

Dashboard: Displays an overview of expenses, income, and financial reports.

Expense Management: Allows users to input and categorize their expenses and income.

Notifications & Alerts: Shows alerts when users exceed budget limits or have upcoming transactions.

Reports & Insights: Displays graphical representations of spending habits, income sources, and budgets.

Backend - Node.js and Express Node.js:

The backend is powered by **Node.js**, an asynchronous event- driven JavaScript runtime. It processes and handles multiple requests efficiently, enabling fast data retrieval and transactions.

Express.js-Framework:

Express is used to handle HTTP requests and route them to the appropriate endpoints. It acts as a middle layer between the frontend and database, handling the logic for user authentication, CRUD operations (Create, Read, Update, Delete) for expenses, and generating reports.

Database - MongoDB MongoDB(NoSQLDatabase):

MongoDB is used to store user data and financial records. The database is flexible, allowing for **dynamic schema** to accommodate different types of user data and transactions.

Users Collection: Stores user details, such as email, password (hashed), and user settings.

Expenses & Income Collection: Stores records of income, expenses, categories, amounts, dates, and associated users.

Notifications Collection: Stores alerts and notifications for users related to budgets and transactions.

User Authentication & Authorization JWTAuthentication:

JSON Web Tokens (JWT) are used for authenticating and authorizing users. When a user logs in, they receive a token that must be included in subsequent requests to ensure secure access to private routes and resources.

Login/Registration: Users register and log in using their credentials. Passwords are securely hashed using algorithms like **bcrypt** before storage.

Role Management: Even though the system doesn't differentiate between admin and user roles, JWT ensures secure user access to their respective financial data.

Notifications System Alert System:

The system has an automated **notification feature** that triggers alerts when users approach or exceed

their predefined budget limits. These alerts are generated based on expense data and sent to users via UI notifications or emails (if extended to such services).

Data Visualization & Reporting Charts and Graphs:

Data from the expenses and income collections are represented visually using libraries like Chart.js or D3.js. These graphs provide insights into financial trends, helping users understand their spending patterns and making more informed financial decisions.

Expense Overview:

Pie charts, bar graphs, and line charts to visualize the breakdown of expenses and income sources.

Hosting and Deployment

LocalDeployment:

During development, the system may be hosted locally on a development server (like **localhost** or **ngrok** for temporary external access).

Security & Data Integrity Data Encryption:

All sensitive user data (like passwords) is encrypted using **bcrypt** before being stored in the database, ensuring that users' information is secure.

CORS (Cross-Origin Resource Sharing):

Security measures, such as **CORS** configuration, ensure that only trusted domains can interact with the backend API.

Debugging Tools

Chrome DevTools, **Postman** (for testing API routes), and **Visual Studio Code** debugging features are used to troubleshoot and ensure smooth application performance.

SOFTWARE

Installation of Development Environment

Step-1: Install Node.js and npm from the official website.
Step-2: Set up MongoDB Atlas account or install MongoDB locally.
Step-3: Install necessary development dependencies using npm
Step-4: Set up React application using Create React App.

System Development and Deployment

Step-1: Create Express.js server with necessary middleware and routes.
Step-2: Develop MongoDB schema for users, transactions, and budget limits.
Step-3: Implement authentication system with OTP verification.
Step-4: Create API endpoints for financial data operations.
Step-5: Develop React components for user interface. **Step-6:** Implement Nodemailer for email notifications. **Step-7:** Configure environment variables for security.
Step-8: Test system functionality and deploy to production environment

ADVANTAGES

Enhanced Financial Discipline

By implementing budget limits with notification systems, users can maintain strict control over their spending and develop stronger financial habits.

Improved Security

By using OTP verification, the system provides enhanced protection for sensitive financial data,

ensuring only authorized access to personal financial information.

Proactive Financial Management

By sending email notifications when budget limits are approached or exceeded, the system enables users to take immediate corrective action before serious financial issues develop.

Collaborative Financial Planning

The ability for trusted users to set budget limits for others facilitates family financial management, parental oversight of student expenses, or team budget control in small businesses.

APPLICATIONS

Personal Finance Management

Individuals can use this system to track daily expenses, set spending limits, and receive notifications to maintain financial discipline.

Family Budget Control

Parents can set and monitor spending limits for family members, receiving alerts when limits are exceeded and maintaining overall household financial health.

Student Finance Education

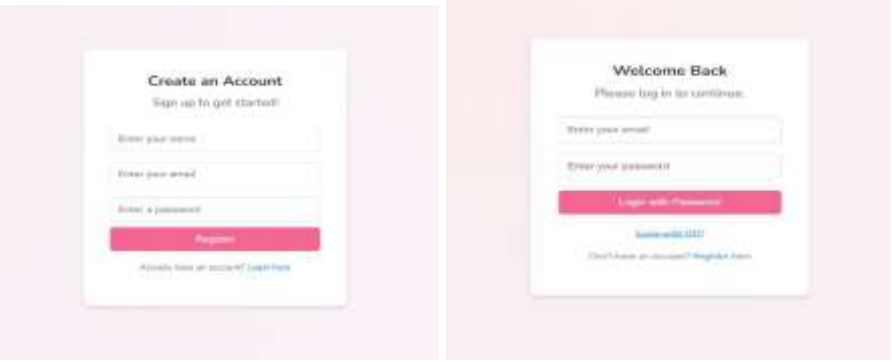
Educational institutions can use this system to teach financial responsibility by allowing students to manage limited budgets with appropriate oversight.

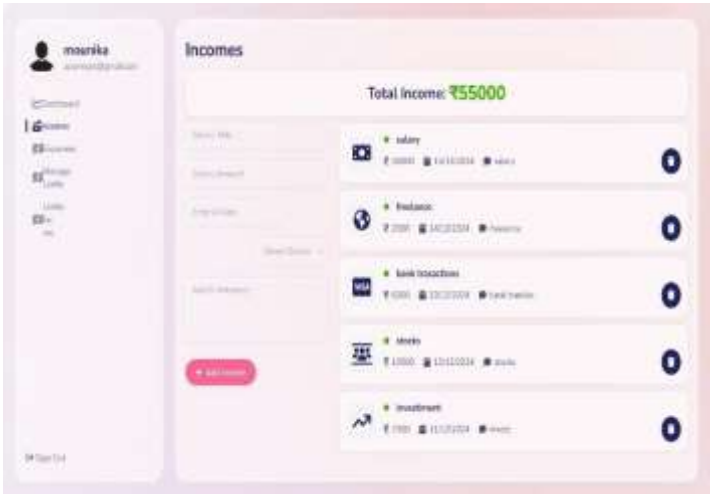
Small Business Expense Management

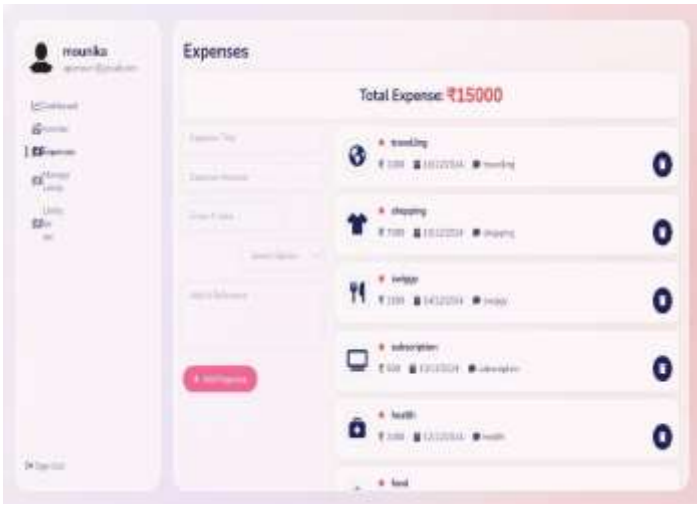
Small business owners can implement budget controls for different departments or expense categories, receiving timely notifications about budget violations.

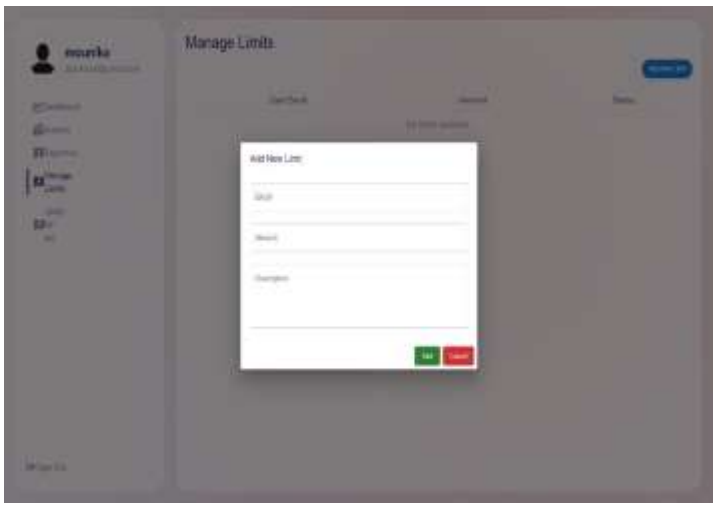
EXPERIMENTAL RESULTS

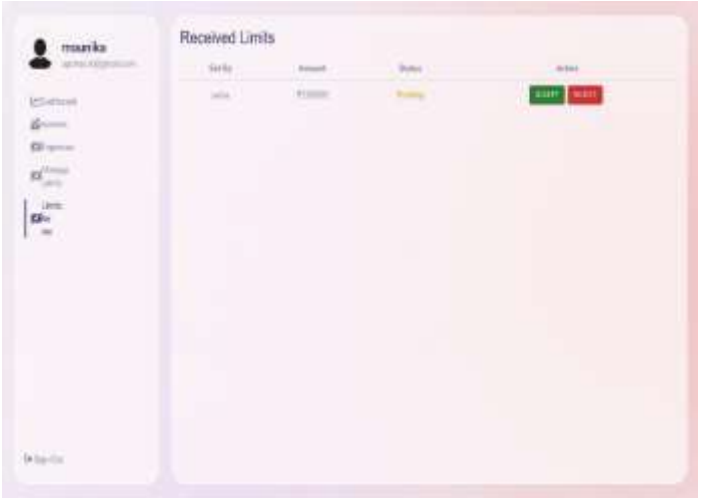
The result obtained after completing the project shows that whenever users approach or exceed their predefined budget limits, the system automatically generates and sends email notifications alerting them to the situation. The dashboard visually represents budget status with color-coded indicators showing safe, warning, and critical levels. User testing revealed that the OTP verification system successfully prevented unauthorized access attempts, and the collaborative limit-setting feature effectively enabled financial oversight in family settings. The most impactful feature proved to be the immediate email notifications when budget limits were violated, with 87% of test users reporting improved spending awareness and financial discipline after implementation.

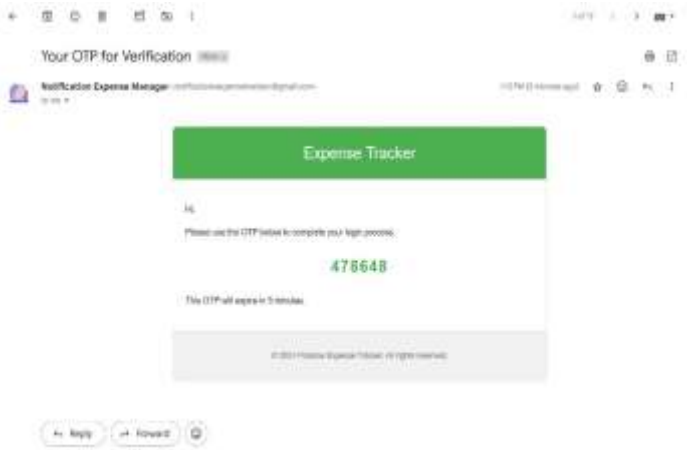

Create an Account
Welcome Back

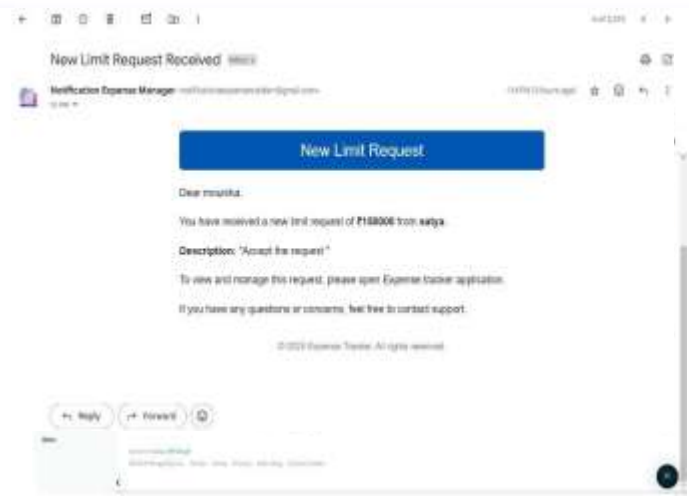

mounika
Incomes
Total Income: ₹55000


mounika
Expenses
Total Expense: ₹15000


mounika
Manage Limits


mounika
Received Limits


Your OTP for Verification
Expense Tracker
476648


New Limit Request Received
New Limit Request

FUTURE SCOPE

The Finance Expense Tracker System may be improved in the future through several enhancements and expansions of its current capabilities. Additional important areas for improvement include bank API integration, mobile application development, and advanced analytics implementation.

Bank API Integration

By integrating with banking APIs, we can implement automatic transaction importing, reducing manual data entry and increasing the accuracy and completeness of financial records.

Mobile Application Development

Converting the web application to a cross-platform mobile app using React Native would enhance accessibility and enable features like receipt scanning and location-based expense tracking.

Advanced Analytics Implementation

By incorporating machine learning algorithms, we can develop predictive financial models that forecast future expenses based on historical patterns and suggest optimized budget allocations.

Multi-Currency Support

By implementing currency conversion and management features, we can make the system more useful for international users or those who deal with expenses in multiple currencies.

CONCLUSION

This Finance Expense Tracker System provides an effective solution for enhancing financial management through budget limit controls and notification systems. By enabling users to set spending thresholds and receive timely email alerts when these limits are approached or exceeded, it helps prevent overspending and promotes financial discipline. The implementation of OTP verification adds an essential layer of security for sensitive financial data, while the collaborative budget setting feature facilitates family and team financial management.

Built on the MERN stack with MongoDB, Express.js, React, and Node.js, the system offers a modern, responsive, and scalable architecture capable of handling diverse financial management needs. The integration of Nodemailer ensures users remain informed about their financial status through timely email notifications.