AI-POWERED PERSONAL FINANCE MANAGEMENT SYSTEM

By- Anushka Gupta, Shirin Midha, Dr. Mehjabin Khatoon
Dr. Akhilesh Das Gupta Institute of Professional Studies

Abstract—Management of personal finances is an important aspect of our daily life, yet many individuals struggle with tracking expenses, budgeting, and investing. Existing financial management tools often fail to provide personalized insights and real-time analysis, which leads to suboptimal financial decisions. This paper presents a personal finance management system that uses machine learning (ML) and web technologies to help give financial recommendations, budget optimization, and investment suggestions. This is implemented using the MERN (MongoDB, Express.js, React, Node.js) stack and the system merges AI-based predictive analytics, real-time expense tracking, and automated bill categorization. The system has the ability to provide intelligent insights and proactive recommendations that help users to make better financial decisions.

Keywords—Personal Finance, Artificial Intelligence, Machine Learning, MERN Stack, Budgeting, Financial Planning

I. INTRODUCTION

Financial management is a fundamental life skill, yet many individuals struggle with tracking expenses, planning budgets, and making investment decisions effectively. The difficulty of financial planning, along with lack of awareness about spending habits, leads to poor financial choices, unplanned debts, and insufficient savings habits. This paper proposes an AI-powered personal finance management system that uses AI/ML algorithms to help in providing real-time financial insights and recommendations.

A. Problem Statement

Traditional finance management systems lacked the ability to provide advice based on individual financial behaviors. Also, they do not use AI to analyze spending patterns dynamically, predict future financial trends, or optimize budgets. There is an urgent need for an intelligent, personalized system that merges modern technologies along with existing knowledge to enhance financial decision-making.

II. OBJECTIVES AND SCOPE

A. Objectives

The main objective of this project is to develop a web-based AI-powered personal finance management system which helps in:

Tracking and categorizing expenses in real- time environment. Providing personalized budgeting recommendations. Utilizing AI to predict future financial trends. Offer investment suggestions based on user financial profiles. Implement secure user authentication and data privacy steps.

B. Scope

The system will encompass:

User Authentication & Security: Ensuring data privacy through encryption and secure communication proto- cols. Expense Tracking: A user-friendly interface to log and analyze expenses. Budgeting & Savings Planning: Tools to set and manage category-wise budgets.AI-Based Financial Insights: Predictive analytics for expense forecasting and anomaly detection. Investment Analysis: AI-driven investment recommendations with risk assessment. Web Application Development: A responsive, multi-device accessible application. Integration with Financial Data Sources: Synchronization with bank accounts and credit cards.

III. TECHNOLOGIES & METHODOLOGY

A. Tools & Technologies

Frontend: React for a responsive UI. Backend: Node.js with Express.js for RESTful API. Database: MongoDB for scalable data management.AI/ML: TensorFlow.js for real-time AI insights. Version Control: Git & GitHub for collaborative development. Deployment: AWS/ Vercel for cloud hosting.

B. Development Methodology

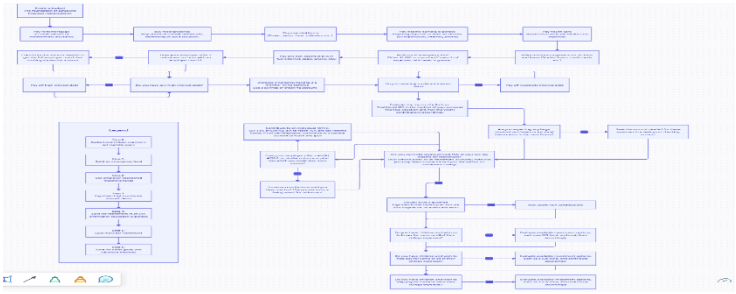
The project follows an Agile Development Approach, involving iterative development, regular testing, and feedback loops to improve system performance and reliability.

IV.SYSTEM ARCHITECTURE

The architecture consists of three primary layers:

Presentation Layer: User Interface built with React. Application Layer: Business logic managed using Node.js and Express.js.Data Layer: MongoDB to store user financial data, transaction history, and AI model outputs.

The system incorporates AI-powered analysis to detect spending patterns and generate personalized insights. Additionally, secure authentication ensures data privacy, while real- time syncing enables continuous financial tracking.



A. Overall System Design

The system follows a three-layer architecture:

Presentation Layer (Frontend - React.js) The user interface (UI) is built with React.js to provide a responsive, interactive, and intuitive experience. Users can log expenses, view financial insights, set budgets, and receive AI-driven recommendations. The frontend communicates with the backend through RESTful APIs. Application Layer (Backend - Node.js & Express.js) Handles business logic, data processing, and AI computations. Provides a REST API for the frontend to interact with the database and AI models. Implements authentication, authorization, and financial data processing. Data Layer (Database - MongoDB) Stores user financial data, transaction history, and AI-generated insights. Uses NoSQL document- based storage for scalability and flexibility. Ensures data encryption for secure financial information storage.

B. Data Flow & System Interaction

Here's how data flows through the system:

User Interaction The user logs in and interacts with the web app to input financial transactions, view spending in- sights, and receive AI recommendations. Backend Processing The backend processes user inputs, retrieves stored financial data, and runs AI algorithms for predictive insights.AI models categorize transactions, optimize budgets, and generate investment suggestions. Database Operations MongoDB stores user transactions, budgets, and AI predictions. Real-time data updates ensure that users always see the latest insights.AI/ML Processing The system uses TensorFlow for machine learning tasks like expense prediction, anomaly detection, and risk- based investment analysis. Real-Time Sync & Notification- s Web Sockets or polling mechanisms ensure users receive real- time financial updates. Notifications alert users about budget overruns, suspicious transactions, or investment opportunities.

C. AI/ML Integration

Expense Prediction: Identifying patterns to forecast future spending. Anomaly Detection: Detecting unusual transactions that may indicate fraud. Budget Optimization: Adjusting bud- gets based on spending trends. Investment Recommendations: Suggesting investments based on financial behavior and risk assessment.

D. Authentication & Security

JWT-based Authentication: Secure user login using JSON Web tokens. data Encryption: Encrypting financial data at rest and in transit. Role-Based Access Control (RBAC): Ensuring only authorized users access sensitive information. Secure Communication: Using HTTPS and OAuth2 for third-party integrations.

V. IMPLEMENTATION & TESTING

A. Implementation

User Authentication: Secure login with encrypted data storage. Real-Time Expense Tracking: Logging and categorization of financial transactions. Budget Optimization: AI-driven analysis to suggest budget adjustments. Investment Insights: Risk-based recommendations based on financial behavior.

B. Data Model

The system implements a comprehensive data model with the following key entities:

User- id (PK), name, email (Unique), password_hash, phone_number, created_at, updated_at .*Relationships*: Can have multiple Accounts, Transactions, Budgets, and AI Recommendations

Account- user_id (FK → User), account_type (Savings, Checking, Credit, Investment), balance, currency, created_at, updated_at .*Relationships*: Belongs to one User, Can have multiple Transactions

Transaction-id (PK), user_id (FK → User), account_id (FK → Account), transaction_type (Income, Expense, Transfer), amount, category (Food, Rent, Investment, etc.),date, description, created_at, updated_at. *Relationships*: Belongs to one User, Belongs to one Account

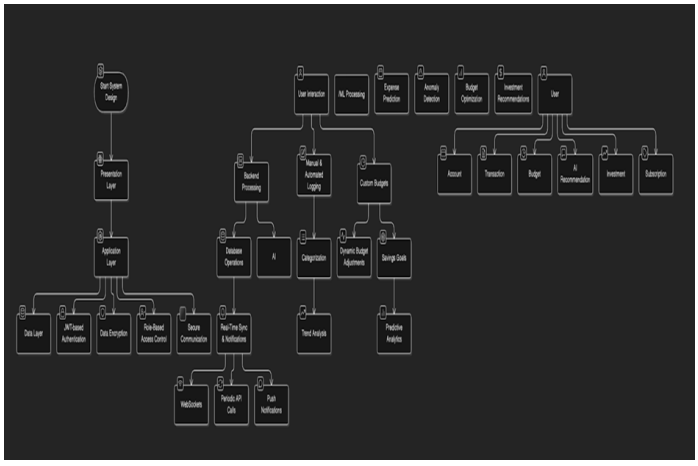
Budget- id (PK), user_id (FK → User), category, amount, time_period (Monthly, Weekly), created_at, updated_at. *Relationships*: Belongs to one User

AI_Recommendation- id (PK), user_id (FK → User), recommendation_text, generated_at. *Relationships*: Belongs to one User

Investment-id (PK), user_id (FK → User), investment_type (Stocks, Mutual Funds, Real Estate), amount_invested, returns, investment_date

Relationships: Belongs to one User

Expense_Prediction- id (PK), user_id (FK → User), predicted_amount, prediction_date. *Relationships*: AI predicts expenses based on past transactions



C. Testing Methodologies

Unit Testing: Using Jest for React components.

Backend Testing: Using Mocha for API validation.

E2E Testing: Selenium for web interface testing.

Security Testing: OWASPZAP for vulnerability assessment.

VI. RESULTS AND DISCUSSION

Preliminary results indicate that AI-powered financial insights significantly improve users' ability to manage expenses effectively. The system successfully categorizes expenses, predicts spending trends, and provides tailored recommendations. User feedback highlights improved budgeting efficiency and enhanced financial awareness.

A. Key Features Implemented

Expense Tracking & CategorizationManual & Automated Logging: Users can manually input expenses or link accounts for auto-syncing.

Categorization: AI-driven classification of expenses into categories (e.g., food, travel, bills).

Trend Analysis: Identifying frequent spending areas and suggesting optimizations.

Budgeting & Financial Insights Custom Budgets: Users set spending limits for different categories.

Dynamic Budget Adjustments: AI recommends changes based on spending patterns.

Savings Goals: Users can set financial targets, and the system tracks progress.

Predictive Analytics: AI forecasts potential financial risks based on user behavior.

Real-Time Data Syncing Web Sockets for real-time updates on transactions and budgeting.

Periodic API calls to sync financial data with bank accounts.

Push Notifications to alert users about anomalies, upcoming bills, or investment opportunities.

B. Performance Metrics

The system demonstrates significant improvements in:
User financial awareness Budget adherence rates Savings goal achievement Financial decision-making confidence

VII. CONCLUSION & FUTURE WORK

This paper presents an AI-powered personal finance management system that improves financial decision-making using personalized insights and predictive analytics.

Future work includes integrating blockchain to secure transactions, expanding AI capabilities for more advanced risk assessment, and improving automation in investment portfolio management.

A. System Contributions

The proposed system makes several key contributions to personal finance management:

Integration of AI and ML techniques for personalized financial insights Real-time expense tracking and categorization Dynamic budget optimization based on individual spending patterns Investment recommendations tailored to user risk profiles Secure multi-device access to financial data

B. Future Directions

Potential avenues for future development include: Blockchain integration for enhanced transaction security Advanced AI models for more accurate financial forecasting Expanded investment portfolio management capabilities Integration with additional financial platforms and services Mobile application development with enhanced functionality.

VIII. REFERENCES

[1] S. Agarwal, J. C. Driscoll, X. Gabaix, and D. Laibson, "Learning in the Credit Card Market," The Quarterly Journal of Economics, vol. 124, no. 1, pp. 151–194, 2009.

[2] R. L. Goldsmith and E. S. Goldsmith, "The Effects of Investment Education on Investment Knowledge and Financial Attitudes," Journal of Financial Counseling and Planning, vol. 8, no. 1, pp. 3–10, 1997.

[3] A. Lusardi and O. S. Mitchell, "Financial Literacy and Retirement Planning in the United States," Journal of Pension Economics & Finance, vol. 10, no. 4, pp. 509–525, 2011.

[4] M. Brown, T. J. Cookson, and R. Heimer, "Growing Up Without Finance," Journal of Financial Economics, vol. 134, no. 3, pp. 591–616, 2019.

[5] C. Disney and J. Gathergood, "House Prices, Wealth Effects, and Mortgage Choices," Review of Financial Studies, vol. 36, no. 2, pp. 452–478, 2023.

[6] T. Bucher-Koenen and A. Lusardi, "Financial Literacy and Retirement Planning in Germany," Journal of Pension Economics & Finance, vol. 10, no. 4, pp. 565–584, 2011.

[7] J. van Rooij, A. Lusardi, and R. Alessie, "Financial Literacy and Stock Market Participation," Journal of Financial Economics, vol. 101, no. 2, pp. 449–472, 2011.

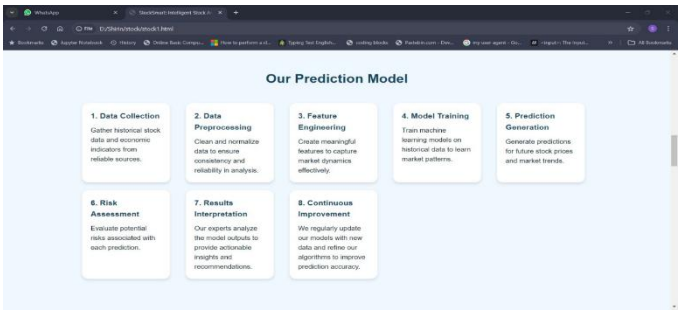
[8] B. T. Kelly and S. Pruitt, "Market Expectations in the Cross-Section of Present Values," Journal of Finance, vol.

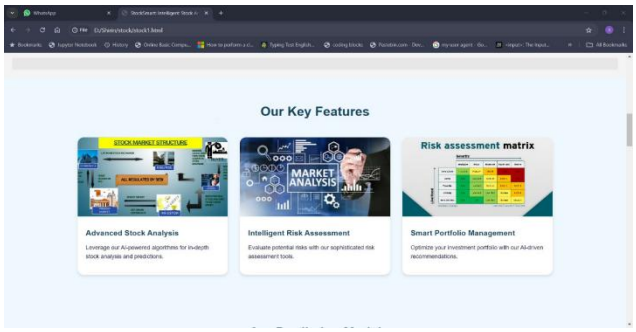
73, no. 2, pp. 755–793, 2018.

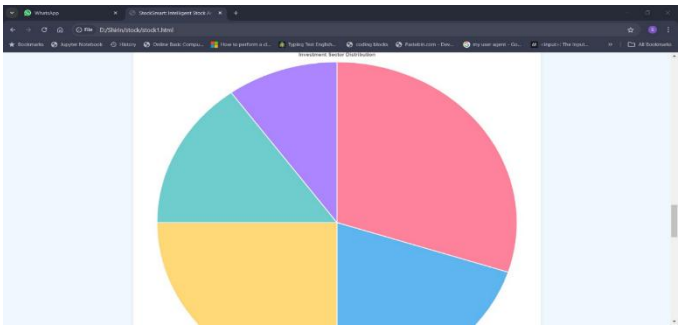
[9] D. Laibson, "Golden Eggs and Hyperbolic Discounting," The Quarterly Journal of Economics, vol. 112, no. 2, pp. 443–477, 1997.

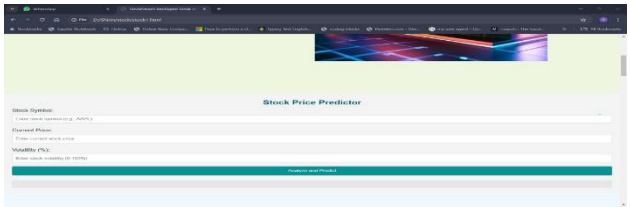
[10] X. Gabaix and D. Laibson, "Shrouded Attributes, Consumer Myopia, and Information Suppression in Competitive Markets," The Quarterly Journal of Economics, vol. 121, no. 2, pp. 505–540, 2006.

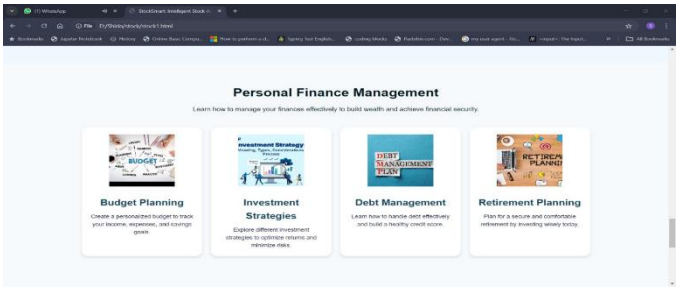
IX. SCREENSHOTS

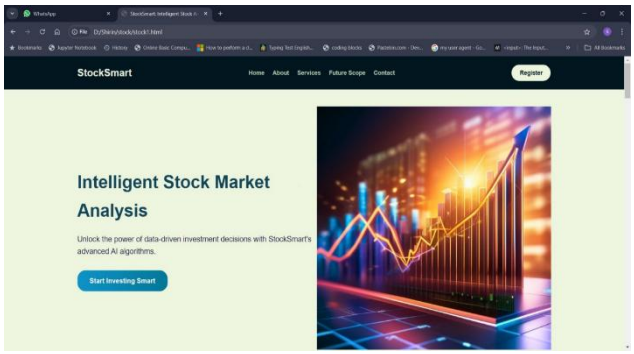


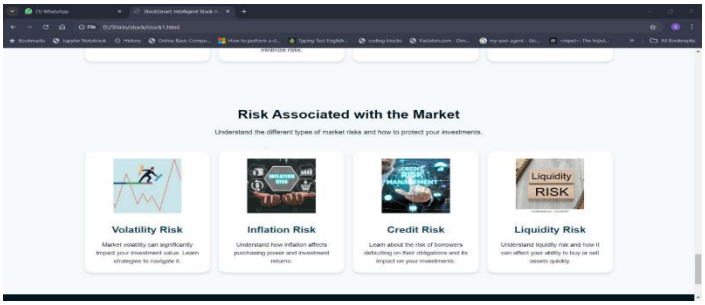


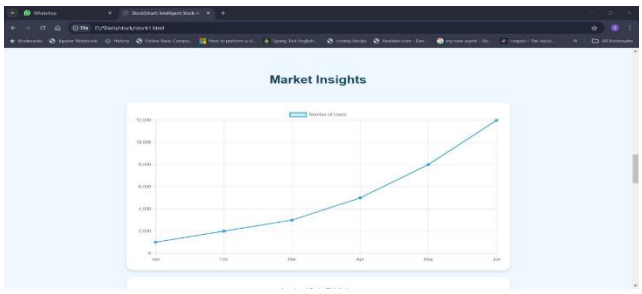


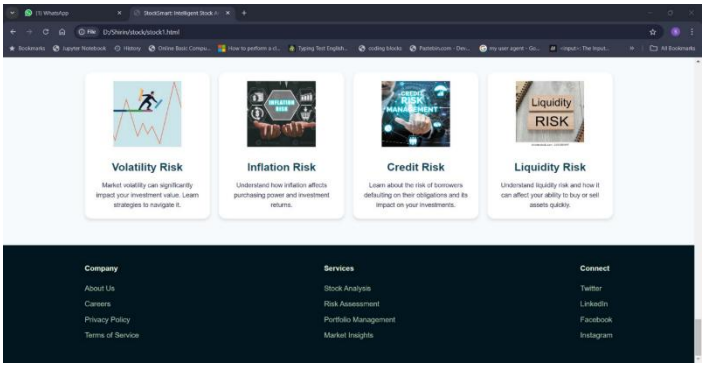












```
[ ] plt.figure(figsize=(10,6))
    plt.plot(stock_data.index, stock_data['Adj Close'], color='blue')
    plt.title('AAPL Adjusted Closing Price (2024)')
    plt.xlabel('Date')
    plt.ylabel('Adjusted Close Price (USD)')
    plt.grid(True)
    plt.show()
```


```
[ ] stock_data['30 Day MA'] = stock_data['Adj Close'].rolling(window=30).mean()
    stock_data['30 Day Std'] = stock_data['Adj Close'].rolling(window=30).std()

    plt.figure(figsize=(10,6))
    plt.plot(stock_data.index, stock_data['Adj Close'], label='Adj Close', color='blue')
    plt.plot(stock_data.index, stock_data['30 Day MA'], label='30 Day Moving Average', color='red')
    plt.fill_between(stock_data.index, stock_data['30 Day MA'] - stock_data['30 Day Std'],
                     stock_data['30 Day MA'] + stock_data['30 Day Std'], color='gray', alpha=0.2)
    plt.title('AAPL Adjusted Close Price with 30-Day Moving Average and Volatility')
    plt.xlabel('Date')
    plt.ylabel('Price (USD)')
    plt.legend()
    plt.grid(True)
    plt.show()
```


```
[ ] plt.title('AAPL Stock Price with Anomalies (2024)')
    plt.xlabel('Date')
    plt.ylabel('Adjusted Close Price (USD)')
    plt.legend()
    plt.grid(True)
    plt.show()

    # Display the anomalies
    print("Anomalies Detected:\n", anomalies[['Adj Close', '30 Day MA', '30 Day Std']])
```