

Real-time/Field-Based Research Project Report On FinSight

A dissertation submitted to the Jawaharlal Nehru Technological University, Hyderabad in partial fulfillment of the requirement for the award of a degree of

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted by

K.Jalaja(23B81A0517)

Rajeevi Madhireddy(23B81A0540)

Sowmith Bachu(23B81A0548)



Department of Computer Science and Engineering

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH, Accredited by NBA, and NAAC)

Vastunagar, Mangalpalli (V),

Ibrahimpattam (M),

Ranga Reddy (Dist.) - 501510, Telangana
State.

2024-25



CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institution, Affiliated to JNTUH,
Accredited by NBA, and NAAC)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M), Ranga
Reddy (Dist.) - 501510, Telangana State.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled “**FinSight**” is being submitted by
K.Jalaja (23B81A0517), Rajeevi Madhireddy (23B81A0540), and Sowmith Bachu
(23B81A0548) in partial fulfillment of the requirement for the award of the degree of
Bachelor of Technology in Computer Science and Engineering, during the academic year
2024-2025

Professor-in-charge RFP

Professor and Head, CSE
(Dr. A. Vani Vasthala)

DECLARATION

I hereby declare that this project report titled “**FinSight**” submitted to the Department of Computer Science and Engineering, CVR College of Engineering, is a record of original work done by me. The information and data given in the report are authentic to the best of my knowledge. This RFP report is not submitted to any other university or institution for the award of any degree or diploma or published at any time before.

K.Jalaja(23B81A0517)

Rajeevi Madhireddy(23B81A0540)

Sowmith Bachu(23B81A0548)

Date:

Place:

ACKNOWLEDGEMENT

We would like to express our sincere gratitude and indebtedness to our project guide **Dr. S.Srinivasulu** for his valuable suggestions and interest throughout this project.

We are also thankful to our principal **Prof. K. Ramamohan Reddy** and Prof **A. Vani Vasthala**, Professor and Head, Department of Computer Science and Engineering CVR College of Engineering, Hyderabad for providing us with supporting infrastructure, environment, and ambiance for completing this project successfully as a part of our B.Tech (CSE) Degree. We would like to thank our project coordinator **Mrs. M.Sunitha** for her constant monitoring, guidance, and support.

We convey our heartfelt thanks to the lab staff for allowing me to use the required equipment whenever needed. Finally, we would like to take this opportunity to thank our families for their support through the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support in the completion of this work.

TABLE OF CONTENTS

			Page No.
		Abbreviations	
		Symbols	
1		INTRODUCTION	1
	1.1	Motivation	
	1.2	Problem Statement	
	1.3	Project Objectives	
	1.4	Project Report Organization	
2		LITERATURE REVIEW	
	2.1	Existing Work	
	2.2	Limitations of Existing Work	
3		REQUIREMENT ANALYSIS	
	3.1	Software requirements	
	3.2	Hardware requirements	
	3.3	User requirements	
4		SYSTEM DESIGN	
	4.0	Proposed System architecture	
	4.1	Proposed Methods/ Algorithms	
	4.2	Class / Use Case / Activity/ Sequence Diagrams	
	4.3	Datasets and Technology stack	
5		IMPLEMENTATION	
	5.1	Frontpage Screenshot	

	5.2	Results and Discussions	
	5.3	Testing	
	5.4	Validation	
6		CONCLUSION	
	6.1	Conclusion	
	6.2	Future scope	
		REFERENCES	

Abstract

We are building a modern banking app to help people manage their money easily. Without such a tool, it's hard to keep track of finances, with information spread across different accounts and no clear idea of where the money goes. This app gives a simple view of account balances, recent transactions, and spending habits, making it easy to see trends. Users can get real-time updates, set savings goals, track their progress, and adjust their budgets. With a focus on simplicity, this app helps users take charge of their finances and plan better for the future.

1. Introduction

Managing personal finances can be challenging, especially when financial information is scattered across multiple accounts with no clear insights into spending patterns. Our modern banking app aims to simplify money management by providing users with a unified and intuitive platform to track their finances effortlessly. With real-time updates on account balances, recent transactions, and spending habits, users can easily monitor their financial health and identify trends. The app also offers features such as savings goal tracking, budget adjustments, and personalized financial insights, empowering users to make informed decisions. Designed with simplicity and user convenience in mind, this app serves as a comprehensive tool to help individuals take control of their finances and plan effectively for the future.

1.1 Motivation

- Managing finances can be overwhelming without a dedicated tool. People often struggle with scattered financial information, unclear spending patterns, and difficulty in tracking expenses. This app simplifies money management by offering a clear and organized view of finances, empowering users to make informed decisions.

Lack of a Centralized Financial Overview

- Many individuals have multiple bank accounts, credit cards, and expenses, making it hard to track finances.
- The app consolidates all financial information into one simple dashboard, providing a clear picture of a user's money.

Need for Budgeting & Financial Planning

- Many struggle with setting and sticking to a budget.
- The app allows users to set spending limits and savings goals, helping them manage finances effectively.

Improving Financial Awareness and Decision-Making

- Without proper financial awareness, users may overspend or make poor investment choices.
- The app provides insights into spending trends, helping users make informed financial decisions.

1.2 Problem Statement

Managing personal finances effectively is a challenge for many individuals. With multiple bank accounts, credit cards, and digital wallets, financial information is often scattered across different platforms, making it difficult to track expenses, savings, and overall financial health. Many existing solutions are either too complex or lack real-time insights, leading to poor financial planning and budgeting.

People struggle to understand their spending habits due to the absence of a centralized platform that provides a clear, consolidated view of their transactions. Without proper tracking, users may overspend, miss bill payments, or fail to meet savings goals. The lack of automated insights and budgeting tools makes it hard to adjust financial plans based on real-time data.

A modern banking app is needed to simplify financial management by offering a user-friendly interface that consolidates all account balances, recent transactions, and spending trends in one place. The app should provide real-time updates on financial activities, allowing users to set savings goals, track their progress, and make informed decisions.

By providing a streamlined and intuitive experience, this banking app empowers users to take control of their finances, make better financial decisions, and plan for a secure future.

1.3 Project Objectives

- **Centralized Financial Management** – Provide users with a single platform to view and manage all their bank accounts and financial transactions in one place.
- **Real-Time Account Updates** – Ensure users receive instant updates on their account balances, deposits, withdrawals, and spending activities.
- **Simplified Transaction Tracking** – Display recent transactions in a clear and organized manner to help users understand their financial activity.
- **Spending Analysis and Insights** – Offer visual representations of spending habits and trends to help users make informed financial decisions.
- **Budgeting and Expense Control** – Enable users to set monthly budgets, categorize expenses, and receive alerts when they exceed spending limits.
- **Savings Goal Management** – Allow users to set financial goals, track progress, and get recommendations to achieve their targets faster.
- **User-Friendly Interface** – Design a simple, intuitive, and visually appealing UI that enhances user experience and accessibility.
- **Secure and Private Transactions** – Implement robust security features like encryption, multi-factor authentication, and biometric login to protect user data.
- **Smart Notifications and Alerts** – Send timely reminders for bill payments, savings milestones, and unusual account activities.
- **Financial Planning Assistance** – Provide insights and recommendations to help users optimize their financial plans and prepare for future expenses.

1.4 Project Report Organization

The project report is organized as follows:

- Chapter 1: Introduction - Provides an overview of the project, including motivation, problem statement, objectives, and report organization.
- Chapter 2: Literature Review - Reviews existing work and discusses the limitations of current task management solutions.
- Chapter 3: Requirement Analysis - Identifies software, hardware, and user requirements for the task management application.

- Chapter 4: Design and Implementation - Describes the design and implementation of the task management application, including system architecture, user interface design, and development process.
- Chapter 5: Testing and Evaluation - Presents the testing methodologies and results, as well as an evaluation of the application's performance and usability.
- Chapter 6: Conclusion and Future Work - Summarizes the findings of the project, discusses limitations, and proposes recommendations for future enhancements.

2. Literature Review

2.1 Existing Work

Many financial management tools have been developed to help users track their expenses, monitor their accounts, and plan their budgets. Traditional banking apps provide basic services like balance inquiries, fund transfers, and transaction history. However, they often lack in-depth insight into spending habits and savings goals. Personal finance software has evolved to offer features such as categorizing transactions, generating financial reports, and providing budget recommendations. Some tools integrate with multiple bank accounts, allowing users to view all their finances in one place. Despite these advancements, many existing solutions have complex interfaces, making them difficult for users who prefer simplicity. Additionally, real-time updates and interactive budgeting features are not always available in traditional systems. This creates a need for a modern, user-friendly app that simplifies money management while offering powerful financial insights.

2.2 Limitations of Existing Work

1. **Fragmented Financial Data** – Most existing apps fail to integrate multiple bank accounts, credit cards, and investment portfolios into a single, seamless view. Users often have to switch between different apps or manually enter data, making financial tracking inconvenient and time-consuming.
2. **Lack of Real-Time Insights** – Many banking apps provide transaction updates with delays, making it difficult for users to track expenses accurately. Without instant notifications and real-time spending insights, users may overspend before realizing their financial status.
3. **Complex Interfaces and Overwhelming Features** – Some financial apps overwhelm users with too many complex features, charts, and financial jargon, making them difficult to navigate. This complexity discourages casual users from engaging with financial planning, leaving them without a clear path to better money management.

3. Requirement analysis

3.1 Software Requirements

The software requirements for Finsight include:

- Operating system: Windows 11
- Text Editor: vs code
- Frontend : React js , appwrite
- Backend : Node js , Express js

3.2 Hardware Requirements

The hardware requirements for Finsight include:

- RAM: 16 GB (preferred), 8 GB (minimum)
- Storage: 256 GB
- CPU: intel core i5 8th Gen (minimum)
- GPU: Dedicate RTX 3070 8 GB VRAM, NVIDIA, etc.
- Keyboard: Standard Windows Keyboard
- Audio device: Speaker/Headphones

3.3 User Requirements

The user requirements for FinSight include:

- **Intuitive Design** – The app should have a clean and simple interface with easy navigation, minimal clutter, and clear financial insights at a glance. Users should be able to access key features like account balances, recent transactions, and budgeting tools effortlessly.
- **Customization Options** – Users should be able to personalize their experience by choosing themes, setting custom spending categories, and enabling widgets for quick access to financial summaries and goals. The dashboard should be configurable to display the most relevant financial information

3.3.1 Functional Requirements:

- **User Authentication & Security** – Users must be able to sign up, log in securely, and recover their accounts. Multi-factor authentication should be available.
- **Account Management** – The app should display all linked bank accounts, their balances, and recent transactions in one place.
- **Transaction Tracking & Categorization** – Transactions should be automatically categorized (e.g., food, shopping, bills), and users should have the option to edit them.
- **Budgeting & Savings Goals** – Users should be able to set savings goals, monitor their progress, and adjust budgets based on spending trends.
-

3.3.2 Non-Functional Requirements:

- **Performance & Scalability** – The app should load data quickly and support a growing user base without slowdowns.
- **Security & Compliance** – All transactions and user data should be encrypted, following banking regulations.
- **User-Friendly Interface** – The design should be simple and intuitive, ensuring ease of use for all age groups.

- **Availability & Reliability** – The app should have a 99.9% uptime, ensuring users can access their finances anytime.

4. System Design

4.0 Proposed System Architecture

The “FinSight” is classified into several components:

1. Client Layer (Frontend)

Technologies : React (Web), React Native (Mobile), Flutter, or Swift/Kotlin (for native apps)

Features :

- User authentication (Login, Signup, Two-Factor Authentication)
- Dashboard (Account balances, recent transactions, spending insights)
- Budgeting & Goal tracking
- Notifications & Alerts
- Fund Transfers & Payments

2. API Gateway & Load Balancer

Technologies: Nginx, AWS API Gateway, Kong, or Traefik

Role:

- Manages requests from frontend to backend
- Ensures load distribution across multiple backend instances
- Implements security features (rate limiting, authentication)

3. Backend Layer (Microservices)

Technologies: Node.js (Express.js), Spring Boot (Java), or Django/FastAPI (Python)

Microservices Architecture:

- **User Service:** Handles user registration, authentication (OAuth, JWT), and profile management
- **Account Service:** Manages user accounts, retrieves balances, and transaction history
- **Transaction Service:** Handles deposits, withdrawals, fund transfers, and transaction categorization
- **Budgeting & Insights Service:** Tracks spending trends, goal setting, and financial analysis
- **Notification Service:** Sends alerts via email, push notifications, and SMS

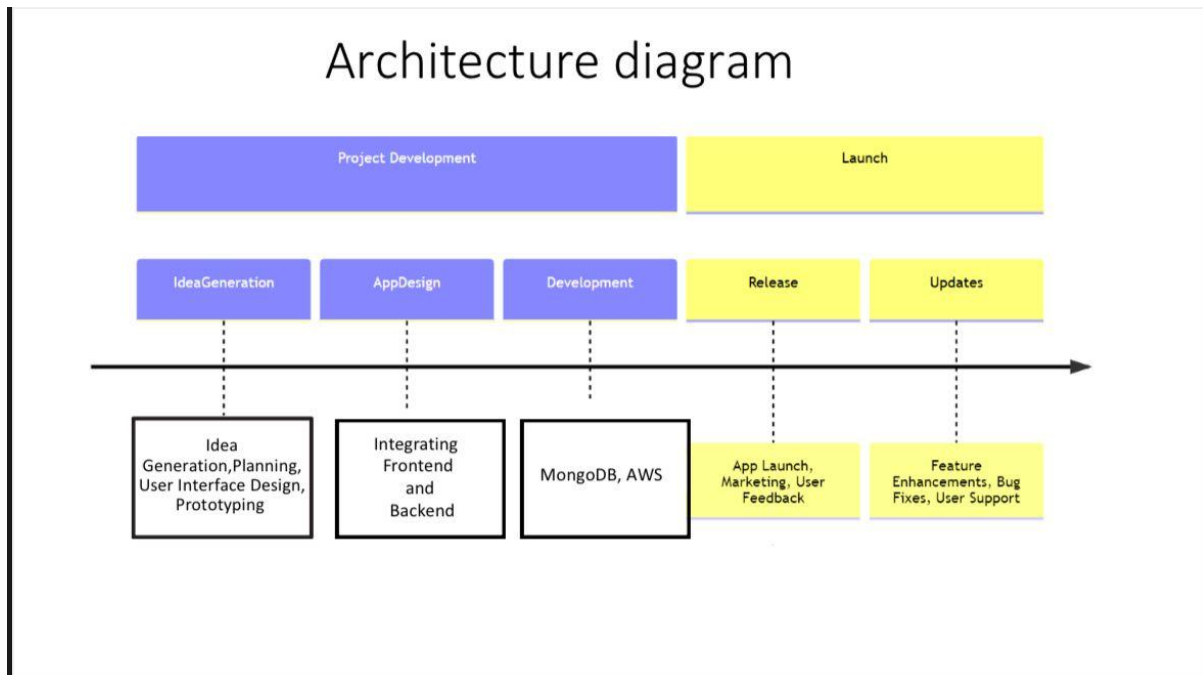
- **Security & Fraud Detection Service:** Monitors transactions for suspicious activities
- 4. Database Layer**
- **Relational Database (RDBMS):** PostgreSQL or MySQL (for structured data like accounts, transactions)
 - **NoSQL Database:** MongoDB or Redis (for fast retrieval of user activity, logs, and caching)

4.1 Proposed Methods

Here, let's get into the details of how all the methods work in the project.

- 1) **User Authentication & Security Layer** – Ensures secure access through encryption, multi-factor authentication (MFA), and token-based authentication to protect user accounts.
2. **Transaction Processing Engine** – Handles all financial transactions, including deposits, withdrawals, transfers, and payments, ensuring accuracy and consistency.
3. **Real-Time Notification Service** – Sends instant alerts for transactions, balance updates, budget limits, and goal progress via push notifications or emails.
4. **Budgeting & Analytics Module** – Tracks spending patterns, categorizes expenses, and provides insights into financial habits to help users make informed decisions.
5. **Third-Party API Integration Layer** – Connects with external financial institutions, payment gateways, and other services to fetch account details and facilitate seamless transactions.

4.2 Use-Case Diagram



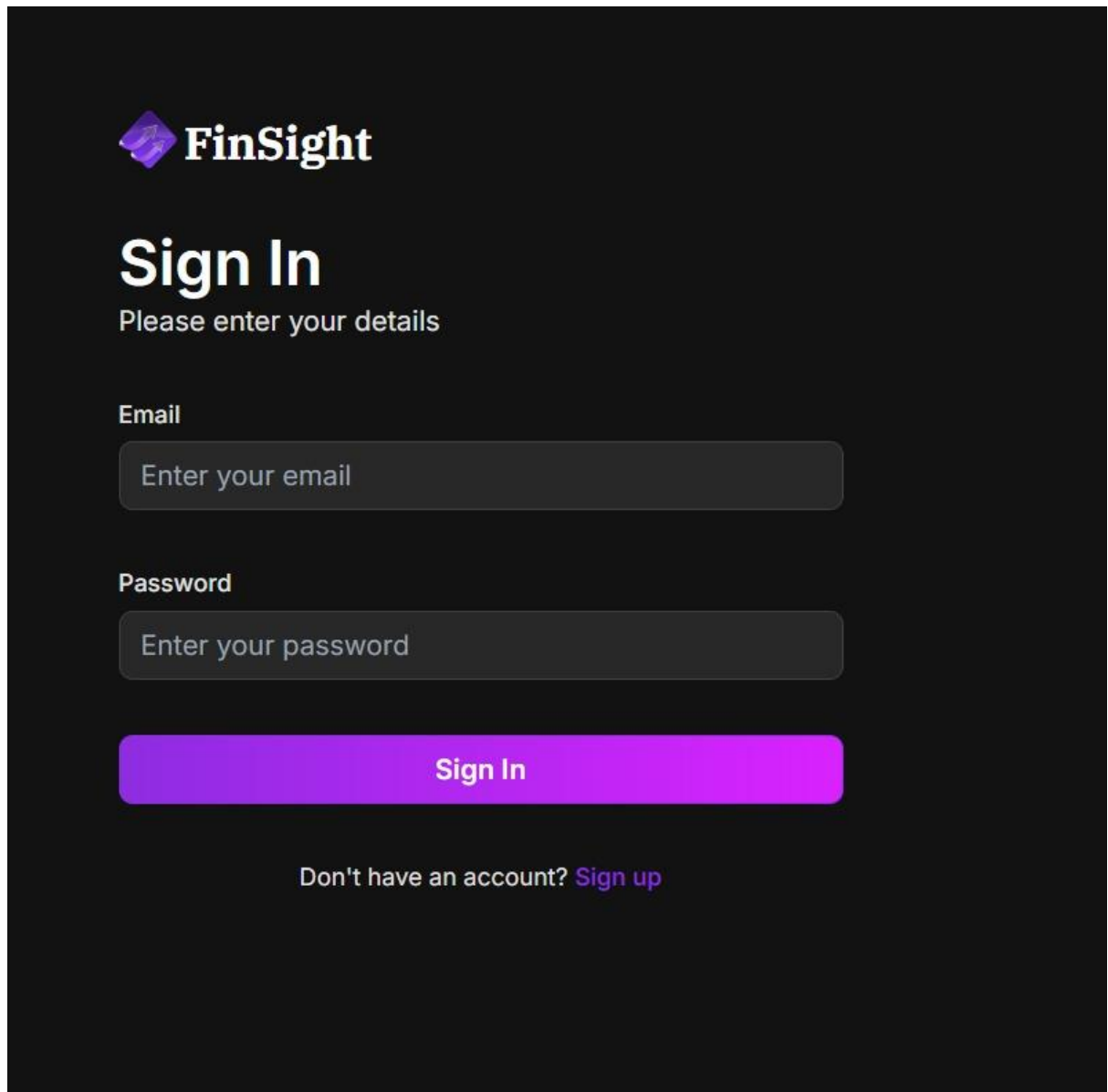
4.3 Datasets and Technology Stack

- **Frontend:** Next.js (React framework for a modern, fast UI)
- **Backend & Database:** SQL (Relational database for structured financial data)
- **Visualization & Other Tech:** Chart.js (for spending insights), authentication (OAuth, JWT), and cloud hosting (Vercel, AWS)

Implementation

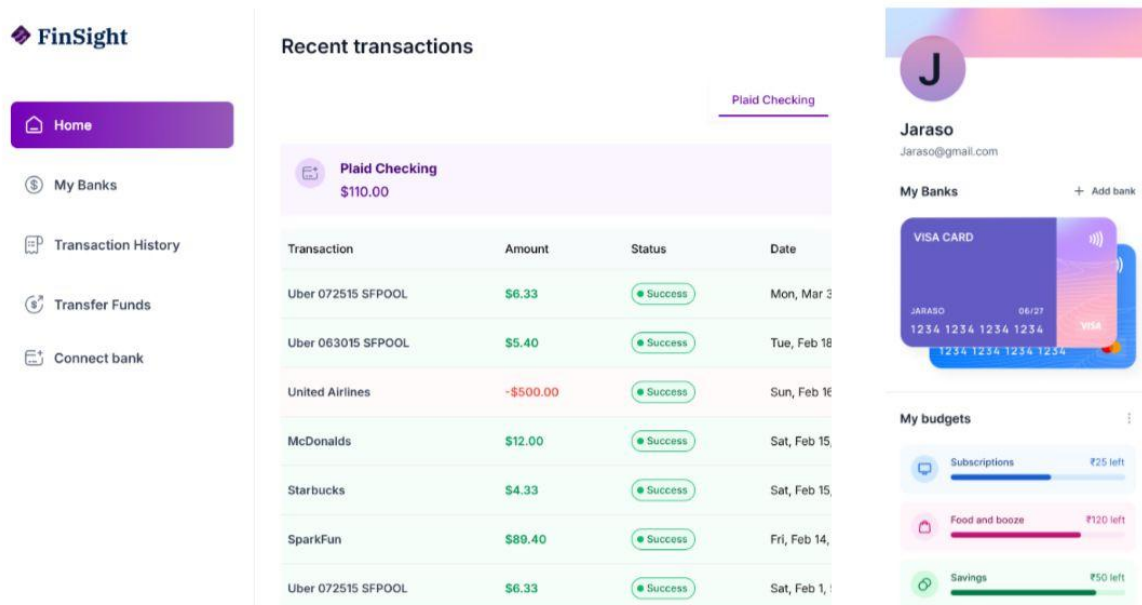
5.1 Sign in Page Screen Shot

Before entering the email and password :



The screenshot shows the FinSight Sign In page. At the top left is the FinSight logo, which consists of a blue diamond icon with a white 'F' and the text 'FinSight' in a bold, sans-serif font. Below the logo, the text 'Sign In' is displayed in a large, bold, white font. Underneath 'Sign In' is the instruction 'Please enter your details' in a smaller, white font. There are two input fields: the first is labeled 'Email' and contains the placeholder text 'Enter your email'; the second is labeled 'Password' and contains the placeholder text 'Enter your password'. Both labels are in a small, white font. Below the input fields is a large, blue, rounded rectangular button with the text 'Sign In' in white. At the bottom of the page, there is a link that says 'Don't have an account? Sign up', where 'Sign up' is in blue and the rest is in white.

After Signing in : Home page of Finsight



5.2 Results and Discussions

5.2.1 Implementation

```
export default function RootLayout({
  children,
}): Readonly<{
  children: React.ReactNode;
}> {
  return (
    <main className="flex min-h-screen w-full justify-between font-inter">
      {children}
      <div className="auth-asset">
        <div>
          <Image
```

```

        src="/icons/auth-image.svg"
        alt "Auth image"
        width={500}
        height={500}
        className="rounded-l-xl object-contain"
      />
    </div>
  </div>
</main>
);
}

```

```

    {
      "$schema": "https://ui.shadcn.com/schema.json",
      "style": "default",
      "rsc": true,
      "tsx": true,
      "tailwind": {
        "config": "tailwind.config.ts",
        "css": "app/globals.css",
        "baseColor": "slate",
        "cssVariables": true,
        "prefix": ""
      },
      "aliases": {
        "components": "@components",
        "utils": "@lib/Utils"
      }
    }
  }

```

```

{
  "name": "jsm_banking",
  "version": "0.1.0",

```

```

"private": true,
"scripts": {
  "dev": "next dev",
  "build": "next build",
  "start": "next start",
  "lint": "next lint"
},
"dependencies": {
  "@hookform/resolvers": "^3.3.4",
  "@radix-ui/react-dialog": "^1.0.5",
  "@radix-ui/react-label": "^2.0.2",
  "@radix-ui/react-progress": "^1.0.3",
  "@radix-ui/react-select": "^2.0.0",
  "@radix-ui/react-slot": "^1.0.2",
  "@radix-ui/react-tabs": "^1.0.4",
  "@sentry/nextjs": "^7.112.2",
  "chart.js": "^4.4.2",
  "class-variance-authority": "^0.7.0",
  "clsx": "^2.1.1",
  "dwolla-v2": "^3.4.0",
  "lucide-react": "^0.374.0",
  "next": "^14.2.24",
  "node-appwrite": "^12.0.1",
  "plaid": "^23.0.0",
  "query-string": "^9.0.0",
  "react": "^18",
  "react-chartjs-2": "^5.2.0",
  "react-countup": "^6.5.3",
  "react-dom": "^18",
  "react-hook-form": "^7.51.3",
  "react-plaid-link": "^3.5.1",
  "recharts": "^2.15.1",
  "tailwind-merge": "^2.3.0",

```

```

    "tailwindcss-animate": "^1.0.7",
    "zod": "^3.23.4"
  },
  "devDependencies": {
    "@types/node": "^20",
    "@types/react": "^18",
    "@types/react-dom": "^18",
    "eslint": "^8",
    "eslint-config-next": "14.2.3",
    "postcss": "^8",
    "tailwindcss": "^3.4.1",
    "typescript": "^5"
  }
}

{
  "compilerOptions": {
    "lib": ["dom", "dom.iterable", "esnext"],
    "allowJs": true,
    "skipLibCheck": true,
    "strict": true,
    "noEmit": true,
    "esModuleInterop": true,
    "module": "esnext",
    "moduleResolution": "bundler",
    "resolveJsonModule": true,
    "isolatedModules": true,
    "jsx": "preserve",
    "incremental": true,
    "plugins": [
      {
        "name": "next"
      }
    ]
  }
}

```

```

    ],
    "paths": {
      "@/*": ["./*"]
    }
  },
  "include": ["next-env.d.ts", "**/*.ts", "**/*.tsx", ".next/types/**/*.ts"],
  "exclude": ["node_modules"]
}

```

Here's how the Specified Code implements the specified functionalities:

1. Viewing Account Balances

- After logging in, users will see a dashboard with all their connected bank accounts.
- Each account shows the **current balance** clearly.
- Users can scroll through if they have multiple accounts linked.

2. Checking Recent Transactions

- On the dashboard or in the "Transactions" tab, users can view a list of their **most recent transactions**.
- Each transaction entry shows the **date**, **amount**, **merchant**, and **type** (debit/credit).
- Clicking on a transaction gives more detailed information.

3. Analyzing Spending Habits

- Users can access the "Spending Overview" section.
- The app displays **graphs and charts** showing where money is being spent (e.g., groceries, bills, shopping).
- Users can filter these trends by week, month, or custom date ranges.

4. Setting and Tracking Savings Goals

- In the "Goals" tab, users can **set up a new savings goal** (e.g., "Save \$1000 for a vacation").
- They specify the target amount and timeline.

- The app shows a **progress bar** updating automatically as money is saved toward the goal.

5. Budget Management

- Users can create **monthly or custom budgets** for categories like food, entertainment, and bills.
- The app tracks how much has been spent in each category.
- Users receive **notifications** if they are close to or over their budget.

6. Real-time Updates

- Bank balances and transactions **update automatically** in near real-time.
- Users get **alerts** for big transactions or low balances if they enable notifications.

7. Simple, User-Friendly Navigation

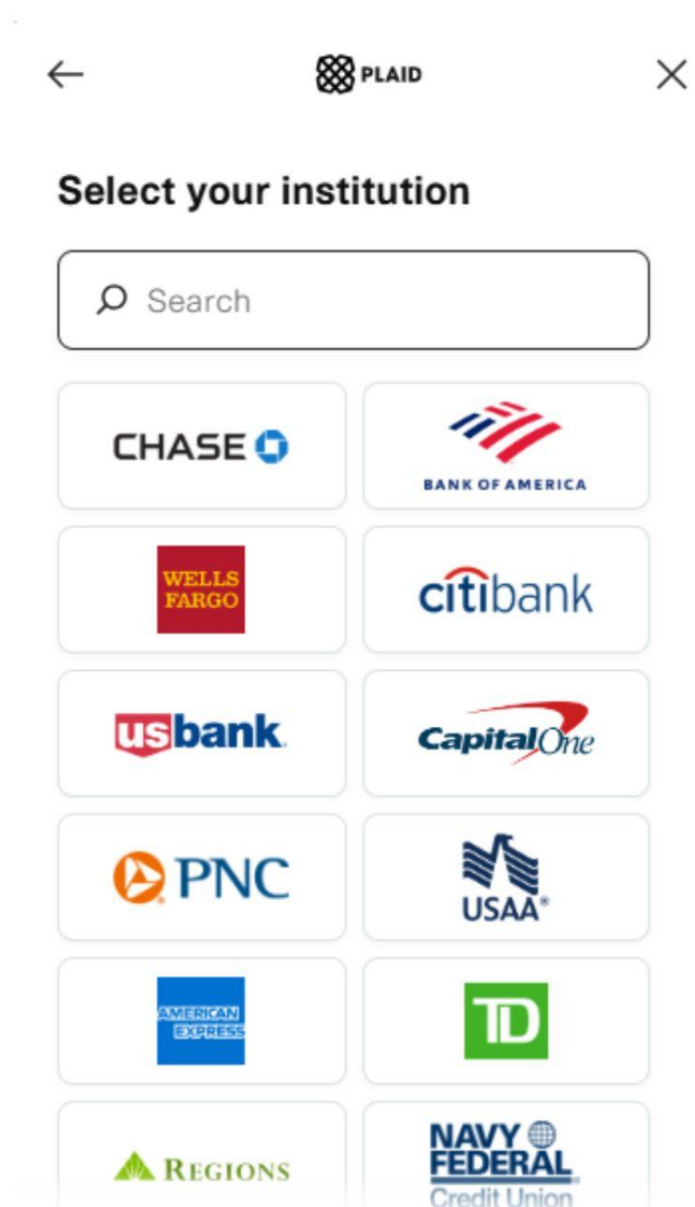
- Clear tabs for "Accounts," "Transactions," "Spending," "Goals," and "Budgets."
- A consistent, minimalist design ensures everything is easy to find.
- Users can customize the dashboard to prioritize the information they care about most.

8. Secure Experience

- Users' bank information is securely linked using trusted platforms (e.g., Plaid, Dwolla).
- The app uses encryption and requires authentication at login.

5.3 Testing:

1. Connect Your Bank Account :




5.4 Validation

Quick Payment Transfer

Payment Transfer

Please provide any specific details or notes related to the payment transfer

Select Source Bank
Select the bank account you want to transfer funds from

 Plaid Checking

Transfer Note (Optional)
Please provide any additional information or instructions related to the transfer

Write a short note here

Bank account details

Enter the bank account details of the recipient

Recipient's Email Address

ex: johndoe@gmail.com

Receiver's Plaid Sharable Id

Enter the public account number

Amount

ex: 5.00

Transfer Funds

6.Conclusion

6.1 Conclusion

By offering a clear, unified view of finances and empowering users with real-time updates, goal tracking, and budgeting tools, our modern banking app makes managing money simpler and smarter. It brings all financial information into one place, helping users stay organized, build better habits, and take full control of their financial future with confidence and ease.

6.2 Future Scope

As the app grows, we aim to integrate advanced features such as AI-driven financial advice, personalized investment recommendations, and automatic bill payments. Expanding multi-currency support, seamless international transactions, and partnerships with more financial institutions will further enhance user experience. In addition, implementing stronger security measures like biometric authentication and real-time fraud alerts will ensure user trust. Over time, the app can evolve into a complete personal finance ecosystem, helping users not just manage but also grow their wealth effortlessly.

