

Full Stack Development with MERN

1. Introduction

- **Project Title:** Banking Application Web App
- **Team Members:**

Sl No	Name	Register Number	Role
1	Kopparapu Shashank	21MID0028	Frontend Developer
2	Rangaiah Gari Harshavardhan Reddy	21MID0084	Project manager
3	Nimmala Sai Lahari	21MID0030	Quality Assurance
4	Tirumala Mukesh Goud	21MIS0102	Mongodb & backend developer

2. Project Overview

- **Purpose:**

The purpose of SIRI Bank, a banking application, is to provide a secure, user-friendly, and efficient digital banking platform for customers. This application aims to revolutionize the traditional banking system by offering various banking services online, reducing the need for physical visits to the bank, and saving customers' time and effort.

Features:

1. **User Registration and Login:** Customers can register and create their accounts using their email addresses and other necessary details. They can log in to their accounts securely using their credentials.
2. **Account Management:** Users can view their account details, including account number, balance, transaction history, and linked accounts. They can also manage their account settings and update their personal information.
3. **Fund Transfer:** Customers can transfer funds to other accounts within the bank or to external accounts using NEFT, RTGS, or IMPS. They can also schedule or set up recurring transfers for regular payments like bills or rent.
4. **Loan Applications:** Customers can apply for loans within the application, track their application status, and view their loan details.
5. **Notifications and Alerts:** SIRI Bank sends real-time notifications and alerts to customers for transactions, account updates, and other important information.
6. **Secure Banking:** The application ensures secure banking by implementing strong encryption, multi-factor authentication, and other security measures to protect customers' data and transactions.

7. **Customer Support:** SIRI Bank offers in-app customer support through chat, email, or phone to assist customers with any queries or issues they may have.

3. Architecture

Frontend:

- **Framework:** React.js
- **Structure:** Component-based for reusability and maintainability.
- **State Management:** React Context API or Redux.
- **Routing:** React Router for SPA experience.
- **UI Libraries:** Material-UI or Bootstrap.
- **Form Handling:** Formik and Yup.
- **API Integration:** Axios or Fetch API.

Backend:

- **Framework:** Node.js and Express.js
- **Structure:** Modular for specific functionalities (user management, transactions, loans).
- **API Endpoints:** RESTful APIs.
- **Middleware:** Authentication (JWT), logging, error handling.
- **Real-Time Communication:** WebSocket or Socket.io.
- **Service Layer:** Abstracts business logic.

Database:

- **Database:** MongoDB with Mongoose
- **Schema Design:**
 - **Users:** Personal details, credentials, settings.
 - **Transactions:** Deposits, withdrawals, transfers.
 - **Loans:** Applications, statuses.
 - **Notifications:** Real-time alerts.
- **Interactions:** CRUD operations, indexing for performance, references for data consistency, ACID transactions.

4. Setup Instructions

Prerequisites:

- Vs code
- MongoDB (version 4.x or later)
- Node.js (version 14.x or later)

- Git
- npm or yarn (package manager)

Installation:

1. Clone the Repository:

```
git clone https://github.com/yourusername/SIRI-bank.get  
cd SIRI-bank
```

2. Install Dependencies: For the frontend:

Command :

```
cd frontend  
npm install  
yarn install (if using yarn)
```

For the backend:

Command :

```
cd backend  
npm install  
yarn install (id using yarn)
```

3. Set Up Environment Variables:

Create a .env file in the backend directory with the following contents:

```
Port = 5000  
MONGODB_URI= mongodb://localhost:27017/SIRIbank  
JWT_SECRET=your_jwt_secret
```

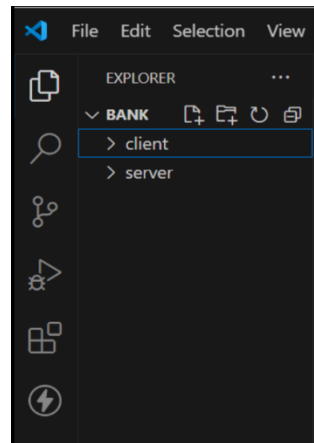
4. Run MongoDB: Make sure MongoDB is running on your system. You can start it with:

5. Start the Backend Server:

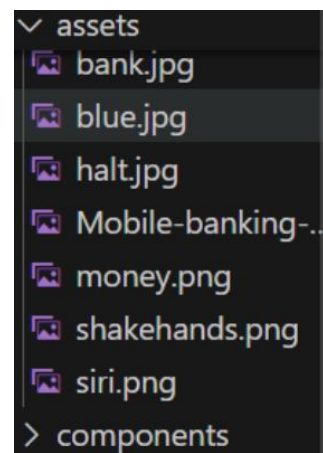
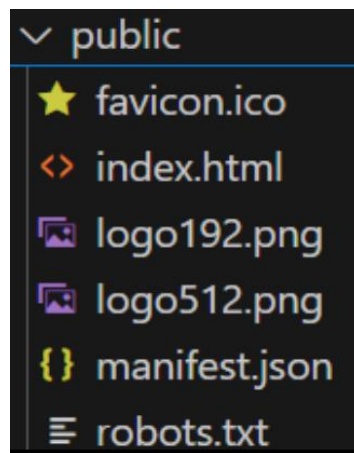
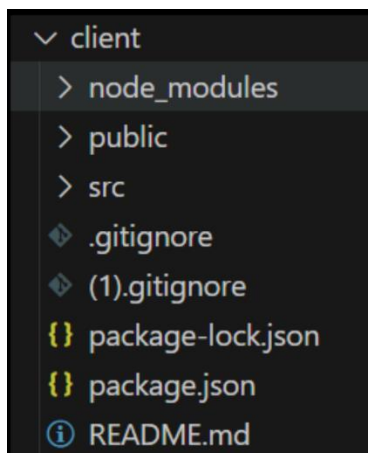
6. Start the Frontend Development Server:

7. Access the Application: Open your browser and navigate to <http://localhost:3000> to access the frontend. The backend API will be running on <http://localhost:5000>.

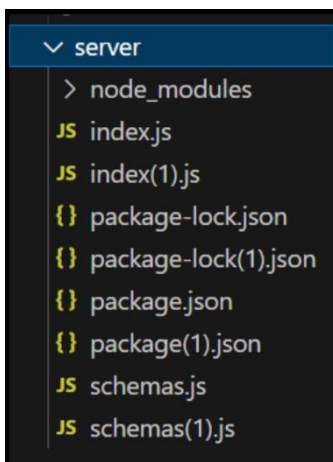
5. Folder Structure



- **Clientside:**



- **Serverside :**



6. Running the Application:

commands to start the frontend and backend servers locally.

- **Frontend:** npm start in the client directory.

1. Navigate to the frontend directory.

Command: cd frontend
npm start.

Backend: npm start in the server directory.

2. Navigate to the Backend directory.

Command: cd Backend
npm start.

7. API Documentation

Base URL:http://localhost:5000/api

1. User Registration and Authentication

1.1 Register User

- **Endpoint:**/api/auth/register
- **Method:**POST
- **Parameters:**
 - name (string) - Required
 - email (string) - Required
 - password (string) - Required

2. Login User

- **Endpoint:**/api/auth/login
- **Method:**POST
- **Parameters:**
 - email (string) - Required
 - password (string) – Required

3. Account Management

Get User Profile

- **Endpoint:**/api/users/me
- **Method:**GET
- **Headers:**
 - Authorization: Bearer <token>

4. Update User Profile

- **Endpoint:**/api/users/me
- **Method:**PUT
- **Headers:**
 - Authorization: Bearer <token>
- **Parameters:**
 - name (string) - Optional
 - email (string) - Optional
 - password (string) - Optional

5.. Loans

Apply for a Loan

- **Endpoint:**/api/loans
- **Method:**POST
- **Headers:**
 - Authorization: Bearer <token>
- **Parameters:**
 - amount (number) - Required
 - term (number) - Required (in months)
 - reason (string) - Required

Get Loan Status

- **Endpoint:**/api/loans/:id
- **Method:**GET
- **Headers:**
 - Authorization: Bearer <token>
- **Parameters:**
 - id (string) - Loan ID

8. Authentication :

In SIRI Bank, authentication and authorization are crucial to ensure the security of user data and access control.

- **Authentication:**

Authentication is the process of verifying the identity of a user. In SIRI Bank, we use JSON Web Tokens (JWT) for authentication. When a user logs in with their credentials (username/email and password), the following steps occur:

- a. The login request is sent to the backend, where the user's credentials are validated against the records stored in the MongoDB database.
- b. If the credentials are valid, the backend generates a JWT, which contains a payload with user information (e.g., user ID, username, or email) and is signed using a secret key.
- c. The JWT is sent back to the frontend and stored in the browser's local storage or as an HttpOnly cookie to prevent access by malicious scripts.
- d. For subsequent requests that require authentication, the JWT is sent along with the request in the Authorization header as a Bearer token.

Authorization:

Authorization is the process of granting or denying access to specific resources or functionalities based on the user's role or permissions. In SIRI Bank, once a user is authenticated, their JWT is used to authorize their access to protected routes and resources.

- a. When a request with a JWT is received at the backend, the token is verified using the secret key to ensure its authenticity and validity.
 - b. If the token is valid, the user information in the payload is extracted, and the user's role or permissions are checked to determine whether they have access to the requested resource or functionality.
 - c. If the user is authorized, the request is processed, and the appropriate response is sent back to the frontend. Otherwise, an error message or a 401/403 status code is returned to indicate that the user is unauthorized or forbidden.
- To enhance security and minimize the impact of stolen or compromised tokens, JWTs are issued with an expiration time. In SIRI Bank, we can implement token refresh functionality using refresh tokens.
 - a. When a user logs in, the backend generates and sends both an access token (short-lived JWT) and a refresh token (long-lived JWT) to the frontend.

- b. The access token is used for authentication and authorization in subsequent requests, while the refresh token is securely stored in the browser's local storage or as an HttpOnly cookie.
- c. When the access token expires, the frontend sends a request to the backend with the refresh token to obtain a new access token.
- d. The backend verifies the refresh token, and if it's valid, a new access token is issued and sent back to the frontend.

```
const jwt = require('jsonwebtoken');
const config = require('config');

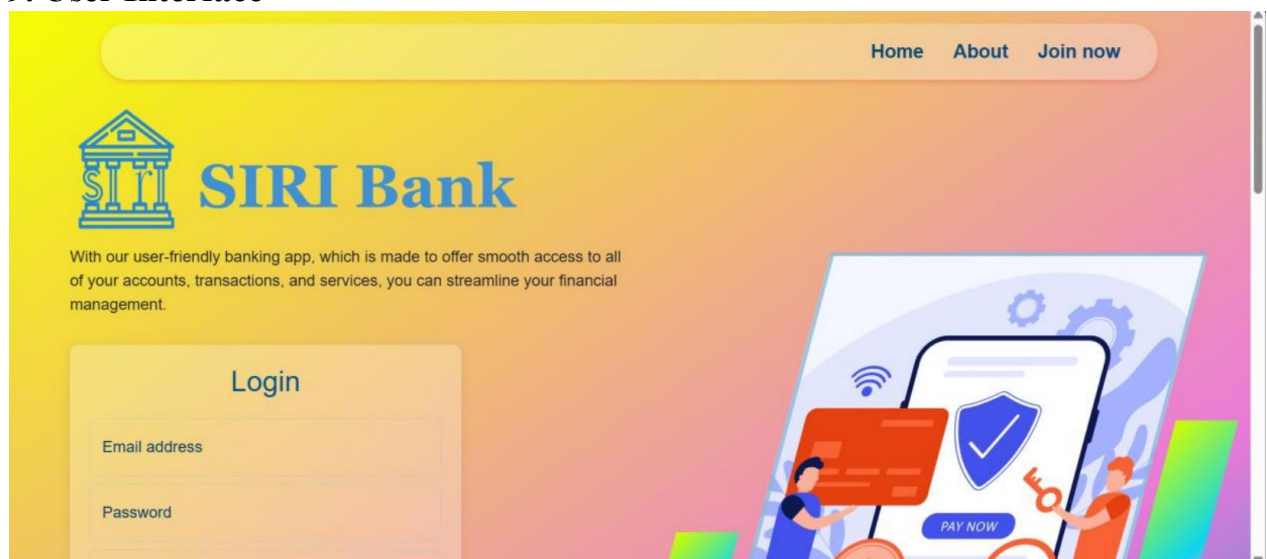
function auth(req, res, next) {
  const token = req.header('Authorization').split(' ')[1];
  if (!token) return res.status(401).send('Access denied. No token provided.');
```

```
  try {
    const decoded = jwt.verify(token, config.get('jwtSecret'));
    req.user = decoded;
    next();
  } catch (ex) {
    res.status(400).send('Invalid token.');
```

```
  }
}

module.exports = auth;
```

9. User Interface



management.

Login

Email address

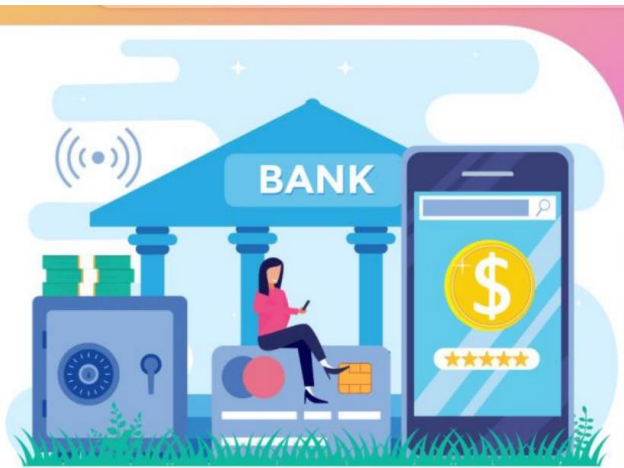
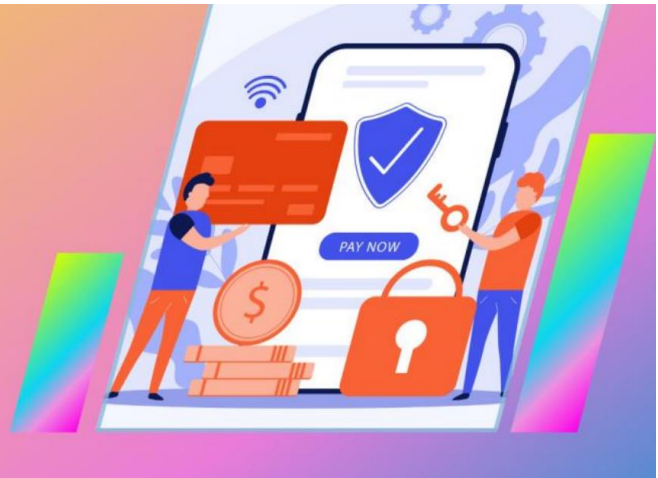
Password

User type



Sign in

Not registered? Register



Safe Deposit Boxes & Guaranteed Maturity

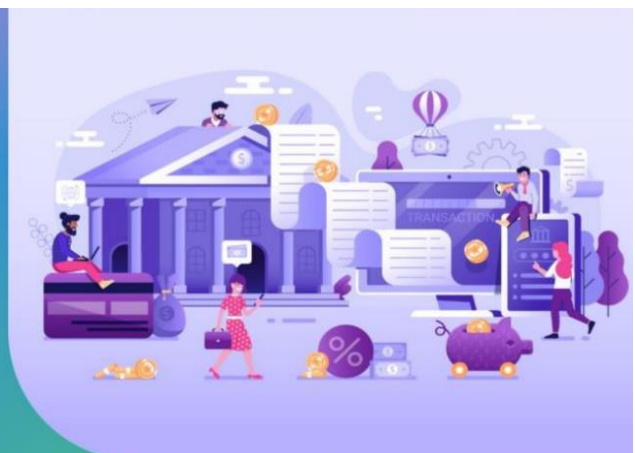
There has never been an easier or more secure way to deposit money. Feel secure in the knowledge that your money is protected by the best protection available. You can use our app to put money into our cutting-edge safety deposit boxes and take full management of your financial assets.

Join now!!

Boost Your Goals Regarding Money

Reaching your financial objectives and opening up new opportunities. LoanLift offers a variety of customised loan options to meet your demands, whether you want to launch a business, go back to school, or remodel your house. Our easy-to-use application procedure makes getting a loan simple, and our reasonable interest rates guarantee timely repayments.

Join now!!



Login

User type

▼

Sign in

Not registered? Register


Register

User type

▼

Sign up

Already registered? Login

 **SIRI Bank_Admin**

HomeUsersDepositsLoansTransactionsLogout

Users9View all

Transactions12View all

Deposits3View all

Loans4View all



All Deposits

Deposit name: Sasi Amount: 500 Duration: 18 months	Nominee name: Harsha Customer name: Venky Start Date: 2024-07-11	Nominee age: 210 Customer A/c id: 668ff55c093d1dea372380e1 Mature Date: 11-6-2025
Deposit name: Mukesh Amount: 50 Duration: 12 months	Nominee name: Sasi Customer name: Venky Start Date: 2024-07-19	Nominee age: 21 Customer A/c id: 668ff55c093d1dea372380e1 Mature Date: 19-6-2025
Deposit name: Mukesh Amount: 100 Duration: 12 months	Nominee name: Sasi Customer name: Mern Start Date: 2024-07-19	Nominee age: 20 Customer A/c id: 669a9b6010c0dc302bc7f5f3 Mature Date: 19-6-2025



All loans

Loan type: home-loan Balance: 1000 Duration: 12 months	Nominee name: Mukesh Total amount: 1000 Start Date: 2024-07-11	Customer name: Venky Customer A/c id: 668ff55c093d1dea372380e1 End Date: 11-6-2025
Loan type: vehicle-loan Balance: 100 Duration: 6 months	Nominee name: Harsha Total amount: 100 Start Date: 2024-07-18	Customer name: Venky Customer A/c id: 668ff55c093d1dea372380e1 End Date: 18-6-2024
Loan type: personal-loan Balance: 40 Duration: 6 months	Nominee name: Sasi Total amount: 100 Start Date: 2024-07-19	Customer name: Venky Customer A/c id: 668ff55c093d1dea372380e1 End Date: 19-6-2024
Loan type: vehicle-loan Balance: 50 Duration: 12 months	Nominee name: Harsha Total amount: 100 Start Date: 2024-07-19	Customer name: Mern Customer A/c id: 669a9b6010c0dc302bc7f5f3 End Date: 19-6-2025

Pending Applications

Loan type: vehicle-loan
Customer name: Mern
Customer A/c id: 669a9b6010c0dc302bc7f5f3
Amount: 300
Duration: 6 months

[Approve](#)[Decline](#)

All users

Username	A/c id	IFSC	Email	Balance
Lahari	668ff352e9af8d6cf1b0a8e5	SB007BLR30	Lahari@gmail.com	2150
Username Shashank	A/c id 668ff35e1922d59ec5349d5b	IFSC	Email Shashank@gmail.com	Balance 0
Username	A/c id	IFSC	Email	Balance
Venky	668ff55c093d1dea372380e1	SB007HYD25	Venky@gmail.com	2390
Username	A/c id	IFSC	Email	Balance
Mukesh	668ff5a3093d1dea372380e8	SB007VZG229	Mukesh@gmail.com	2050
Username	A/c id	IFSC	Email	Balance
Sasi	668ff5ef093d1dea372380ef	SB007TPTY05	Sasi@gmail.com	2000
Username	A/c id	IFSC	Email	Balance



SIRI Bank

[Home](#) [Deposits](#) [Loans](#) [Transactions](#) [Logout](#)

User: Mern

Account Id: 669a9b6010c0dc302bc7f5f3

IFSC code: SB007MBI12

Home Branch: mumbai

Account balance

₹ 2250

Send money

Receiver's a/c id

IFSC

Amount
0

Recent Transactions

Amount: ₹ 50

Loan name: vehicle-loan

Remarks: Loan re-payment

Time: Jul 19 2024 22:41:09

Amount: ₹ 200

Receiver a/c id: 668f94f1738a141eb3a6e6d

Receiver: Charan

Receiver IFSC: SB007CNI99

Payment Method: NEFT

Time: 2024-07-19T17:06:38.508Z

Remarks:

Amount: ₹ 100

Deposit name: Mukesh

Remarks: Deposit payment

Time: Jul 19 2024 22:34:31



SIRI Bank

[Home](#) [Deposits](#) [Loans](#) [Transactions](#) [Logout](#)

All Transactions

Amount: ₹ 50

Loan name: vehicle-loan

Remarks: Loan re-payment

Time: Jul 19 2024 22:41:09

Amount: ₹ 200

Receiver a/c id: 668f94f1738a141eb3a6e6d

Receiver: Charan

Receiver IFSC: SB007CNI99

Payment Method: NEFT

Time: 2024-07-19T17:06:38.508Z

Remarks:

Amount: ₹ 100

Deposit name: Mukesh

Remarks: Deposit payment

Time: Jul 19 2024 22:34:31



SIRI Bank

[Home](#) [Deposits](#) [Loans](#) [Transactions](#) [Logout](#)

Fixed Deposits

Deposit name: Mukesh

Nominee name: Sasi

Nominee age: 20

Amount: 100

Duration: 12 months

Maturity Date: 19-6-2025

New Fixed deposit

Deposit name

Nominee name

Age
0

Deposit amount
0

Duration (in months)
0

* I hereby agree to all the terms & conditions to make this deposit. I agree to gain interest with the dynamic interests depending up on the market conditions. I agree that at any case I cannot break deposit before maturity date.

deposit

SIRI Bank Home Deposits Loans Transactions Logout

Loans

Loan type: vehicle-loan Nominee name: Shashank Loan amount: 300
Balance: 300 Duration: 6 months Loan status: pending

Loan type: vehicle-loan Nominee name: Harsha Loan amount: 100
Balance: 50 Duration: 12 months End Date: 19-6-2025
[Make payment](#)

Apply for New Loan

Choose loan type ▼

Nominee name Age 0

Loan amount 0 Duration (in months) 0

* I here by agree to all the terms & conditions to make this loan. I agree to pay interest with the dynamic interests depending up on the market conditions. I agree to repay the loan before the deadline.

[Apply](#)

10. Testing

Testing Strategy

In SIRI Bank, ensuring the reliability and functionality of the application is paramount. The testing strategy encompasses both manual and automated testing approaches to validate different aspects of the system.

1. Unit Testing

Verify individual units or components of the backend and frontend.

- **Tools:**
 - **Backend:** Jest, Mocha, Chai for Node.js
 - **Frontend:** Jest, React Testing Library for React components
- **Process:** Developers write unit tests for functions, modules, and components to ensure they work as expected. Mocking and stubbing techniques are used to isolate units for testing.

2. Integration Testing

Test interactions between various modules and components.

- **Tools:**
 - **Backend:** Supertest, Postman for API testing
 - **Frontend:** Cypress, Selenium for UI testing

Integration tests validate how different parts of the application work together.

API endpoints are tested for correct responses, data flow, and error handling.

3. Security Testing

Identify and fix security vulnerabilities.

- **Tools:** OWASP ZAP, Burp Suite, Security Headers checkers
- **Process:** Security testing involves penetration testing, vulnerability scanning, and code reviews to mitigate risks like SQL injection, XSS attacks, and data breaches.

4. Performance Testing

Evaluate application performance under various conditions.

- **Tools:** Apache JMeter, LoadRunner, New Relic
- **Process:** Performance tests measure response times, throughput, and resource usage to ensure the application handles expected loads without degradation.

5. User Acceptance Testing (UAT)

Validate the application meets business requirements and user expectations.

- **Process:** Conducted by stakeholders or end-users in a production-like environment to confirm usability, functionality, and overall satisfaction.

6. Monitoring and Logging

Monitor application performance and detect issues in real-time.

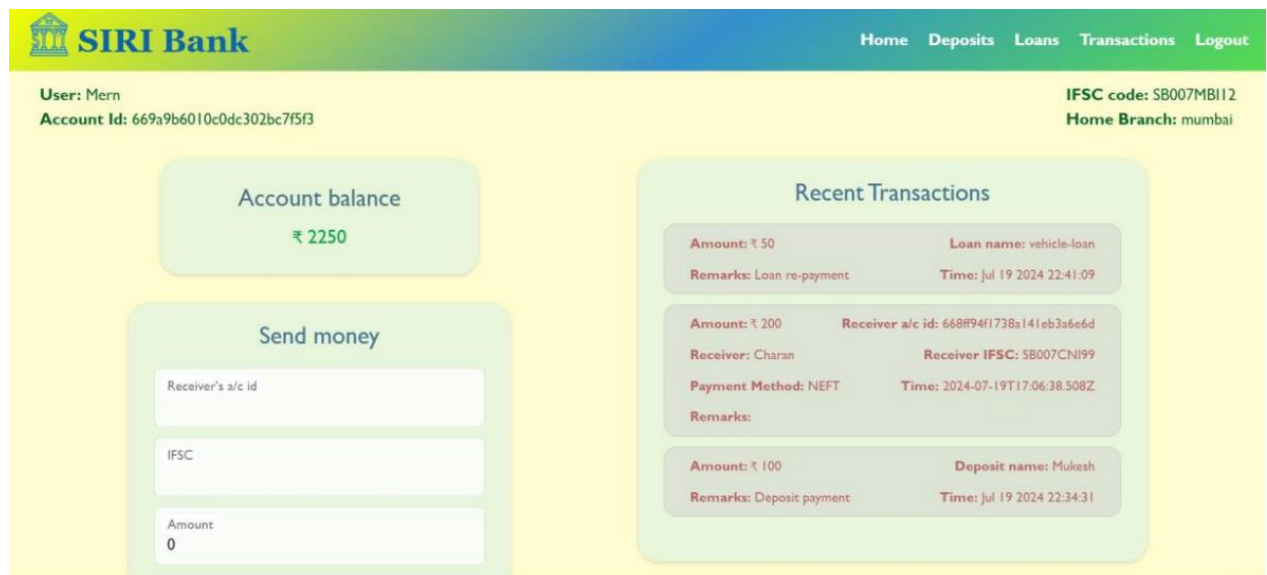
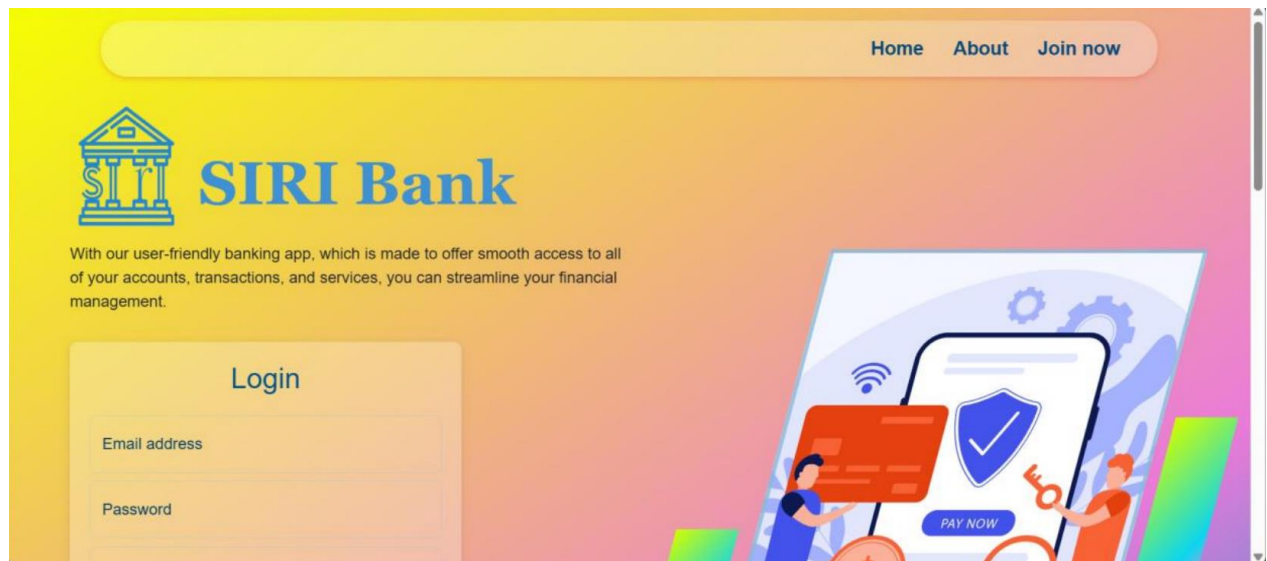
- **Tools:** ELK Stack (Elasticsearch, Logstash, Kibana), Prometheus, Grafana
- **Process:** Logs and metrics are collected to track errors, performance metrics, and user activities, aiding in proactive troubleshooting and optimization.


11. Screenshots or Demo

Video Link :

https://drive.google.com/file/d/13RDq4rccV8kad9lA6lqQOyz2GYZKwbvj/view?usp=drive_link

Screenshots:




SIRI Bank

[Home](#)
[Deposits](#)
[Loans](#)
[Transactions](#)
[Logout](#)


Fixed Deposits

Deposit name: Mukesh Nominee name: Sasi Nominee age: 20
 Amount: 100 Duration: 12 months Maturity Date: 19-6-2025

New Fixed deposit

* I here by agree to all the terms & conditions to make this deposit. I agree to gain interest with the dynamic interests depending up on the market conditions. I agree that at any case I cannot break deposit before maturity date.

deposit


SIRI Bank

[Home](#)
[Deposits](#)
[Loans](#)
[Transactions](#)
[Logout](#)

Loans

Loan type: vehicle-loan Nominee name: Shashank Loan amount: 300
 Balance: 300 Duration: 6 months Loan status: pending

Loan type: vehicle-loan Nominee name: Harsha Loan amount: 100
 Balance: 50 Duration: 12 months End Date: 19-6-2025

Make payment

Apply for New Loan

* I here by agree to all the terms & conditions to make this loan. I agree to pay interest with the dynamic interests depending up on the market conditions. I agree to repay the loan before the deadline.

Apply

12. Known Issues:

- Performance Optimization:** As the application grows, there may be performance issues due to increased data and complex components. Optimizing the codebase, implementing code splitting, and lazy loading components can help improve performance.
- Cross-Browser Compatibility:** The application may not work consistently across different browsers and versions. Ensuring cross-browser compatibility through testing and using polyfills can help resolve these issues.

- **Responsive Design:** While the application is designed to be responsive, there may be layout and styling issues on certain devices or screen sizes. Addressing these issues and implementing a mobile-first design approach can improve the user experience.
- **State Management:** As the application grows, managing state can become complex and error-prone. Implementing a robust state management solution, such as Redux or MobX, can help simplify state management and improve the application's overall maintainability.

13. Future Enhancements:

- **Biometric Authentication:** Implementing biometric authentication, such as fingerprint or facial recognition, can provide an additional layer of security and improve the user experience.
- **Multi-Language Support:** Adding multi-language support can help cater to a global audience and improve the application's accessibility.
- **Financial Planning Tools:** Adding financial planning tools, such as budgeting, expense tracking, and investment planning, can help users better manage their finances and make informed decisions.
- **Offline Support:** Implementing offline support using service workers and caching strategies can help ensure the application remains functional even when the user's internet connection is unstable or unavailable.
- **Machine Learning and AI:** Incorporating machine learning and AI algorithms can help provide personalized recommendations, detect fraudulent activities, and improve the overall efficiency of the application.
- **Voice User Interface (VUI):** Integrating voice commands using technologies like Google Assistant, Amazon Alexa, or Siri can make the application more accessible and convenient for users, especially those with visual impairments or mobility issues.
- **Goal-Based Savings:** Introducing goal-based savings features can help users set financial goals, track their progress, and automatically save money towards those goals, encouraging better financial habits.
- **Currency Conversion and International Transfers:** Adding support for currency conversion and international transfers can help users manage their finances across different currencies and make international payments with ease.