##### APPENDIX 1

(A typical Specimen of Cover Page & Title Page)

<Font Style Times New Roman – Bold>

**TITLE OF PROJECT REPORT**

AGRICULTURE LOAN PROCESSING SYSTEM

## A PROJECT REPORT

### Submitted by

Amrith Menon S

# NAME OF THE CANDIDATE(S)

Amrith Menon S

### in partial fulfillment for the award of the degree of

BACHELORS OF TECHNOLOGY

# NAME OF THE DEGREE

BACHELORS OF TECHNOLOGY

**IN**

#### BRANCH OF STUDY

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

## SRI KRISHNA COLLEGE OF ENGINEERING AND TECHNOLOGY

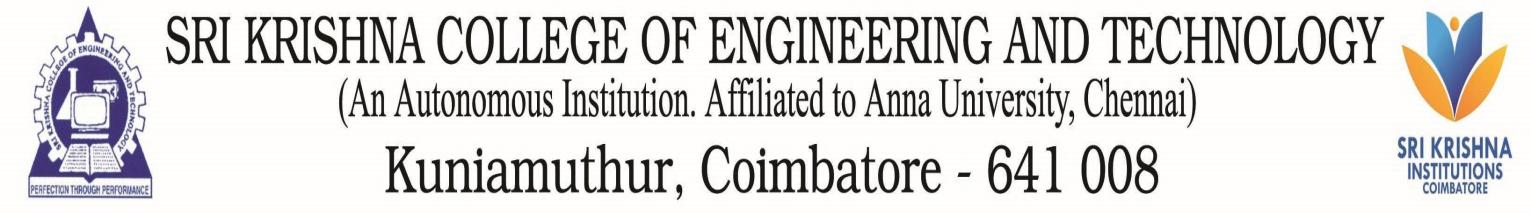
**(An Autonomous Institution, Affiliated to Anna University Chennai - 600 025)**

#### MONTH & YEAR

13/8/2024

##### APPENDIX 2

(A typical specimen of Bonafide Certificate)



# BONAFIDE CERTIFICATE

Certified that this project report **“AGRICULTURE LOAN PROCESSINNG SYSTEM ”**

is the bonafide work of “**…………..Amrith Menon S** **”**

#### who carried out the project work under my supervision.

##### SIGNATURE SIGNATURE

DR S VENKATA LAKSHIMI

##### HEAD OF THE DEPARTMENT SUPERVISOR

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

SRI KRISHAN COLLEGE OF ENGINEERING AND TECHNOLGY

KUNIYAMUTHUR,

COIMBATORE,TAMILNADU-641008,

INDIA

Submitted for the Project viva-voce examination held on 13/08/2024

INTERNAL EXAMINER EXTERNAL EXAMINER

##### APPENDIX 3

(A typical specimen of table of contents)

**TABLE OF CONTENTS**

##### CHAPTER NO. CONTENTS PAGE NO.

|  |  |  |
| --- | --- | --- |
| 1 | **Abstract** | 1 |
| 2 | **Introduction** | 2 |
|  | 2.1 General | 3 |
|  | 2.2 Objectives | 8 |
|  | 2.2.1 System Requirements | 12 |
|  | 2.2.2 Functional Requirements | 13 |
|  | 2.2.2.1 Frontend Development | 13 |
|  | 2.2.2.2 Backend Development | 14 |
|  | 2.2.2.3 Database Management | 16 |
|  | 2.3 Technologies Used | 17 |
|  | 2.4 System Architecture | 18 |
| 3 | **ER Diagram** | 19 |
| 4 | **Usecase Diagram** | 20 |
| 5 | **Literature Review** | 22 |
|  | 5.1 Previous Work | 22 |
|  | 5.2 Comparative Analysis | 22 |
|  | 5.3 Future Directions | 23 |
| 6 | **System Design and Implementation** | 24 |
|  | 6.1 Frontend Design | 25 |
|  | 6.2 Backend Design | 26 |
|  | 6.3 Database Design | 27 |
|  | 6.4 Integration | 28 |
| 7 | **Testing and Results** | 29 |
|  | 7.1 Test Cases | 30 |
|  | 7.2 Test Results | 31 |
|  | 7.3 Performance Analysis | 32 |
| 8 | **Results** | 33 |
| 9 | **Program** | 34 |
| 10 | **Outputs(ScreenShorts)** | 192 |
| 11 | **Conclusion and Future Enhancement** | 198 |
|  | 11.1 Conclusion | 199 |
|  | 11.2 Recommendations for Future Work | 200 |
|  |  |  |

### Abstract

The "Agriculture Loan Processing System" is an innovative platform designed to transform and modernize the traditional methods of loan processing within the agricultural sector. This system addresses the significant challenges faced by both farmers and financial institutions by offering a streamlined, efficient, and automated approach to loan management. Traditional loan processing methods are often marred by inefficiencies, delays, and a lack of transparency, which can hinder access to essential financial resources for farmers. By revolutionizing these processes, the system aims to empower farmers, improve financial institutions' operational efficiency, and foster economic growth in the agricultural sector. The platform is built using cutting-edge web technologies, with React.js providing a dynamic and responsive frontend that offers an intuitive user interface. Django, a robust and scalable web framework, serves as the backbone of the system, enabling the seamless integration of various components and functionalities. MySQL is utilized for backend database management, ensuring secure, efficient, and reliable storage and retrieval of data. Together, these technologies create a powerful, cohesive system that meets the diverse needs of its users. One of the key features of the system is its ability to automate the loan application, processing, and approval stages. This automation reduces the manual effort required from both farmers and loan officers, significantly speeding up the entire process. Farmers can submit loan applications, upload necessary documents, and monitor the status of their applications in real-time, all through a user-friendly interface. Financial institutions benefit from automated workflows that reduce administrative overhead and minimize the risk of human error.

### 2. Introduction

#### 2.1 General

The agriculture sector plays a crucial role in the economic stability and growth of many countries, particularly in developing nations where it serves as a major contributor to GDP and a primary source of employment. Despite its importance, the sector faces significant challenges, especially for small-scale and subsistence farmers who struggle with financial barriers. These barriers often limit their ability to adopt modern farming practices, purchase high-quality inputs like seeds and fertilizers, or invest in expanding their operations. Access to credit is a critical factor in overcoming these challenges, enabling farmers to enhance productivity, improve yields, and secure their livelihoods.

However, traditional loan processing methods in the agricultural sector are fraught with inefficiencies. The process is often cumbersome, involving multiple steps that can be time-consuming and prone to errors. Farmers may have to deal with lengthy paperwork, unclear application procedures, and extended waiting periods for loan approval, all of which can deter them from seeking the financial support they need. Additionally, the lack of transparency and inconsistent criteria used in traditional loan evaluations can further complicate the process, leading to delays and uncertainty.

The "Agriculture Loan Processing System" is designed to address these challenges by providing a comprehensive, automated solution that transforms the way agricultural loans are processed. This innovative platform simplifies the loan application process, making it more accessible and less intimidating for farmers. By using a user-friendly interface, the system guides farmers through each step of the application, allowing them to easily submit their information, upload necessary documents, and track the progress of their loan applications in real-time.

For financial institutions, the system offers significant advantages by streamlining the loan processing workflow. Automation reduces the manual effort required to evaluate applications, minimizing the risk of human error and speeding up the overall process. This efficiency not only allows institutions to handle a larger volume of loan applications but also reduces administrative burdens, freeing up resources that can be redirected to other critical tasks.

One of the standout features of the system is its integration of real-time data analytics, which enhances the decision-making process for loan officers.

By leveraging up-to-date information on market conditions, credit scores, and other relevant data, the system provides loan officers with the insights they need to make informed decisions quickly and accurately. This data-driven approach ensures that loans are approved based on objective criteria, reducing bias and improving the fairness of the loan distribution process.

Moreover, the system's ability to provide quick access to funds is a game-changer for farmers. With faster loan approvals, farmers can access the financial resources they need in a timely manner, enabling them to make critical investments during key agricultural seasons. Whether it's purchasing seeds before planting, acquiring machinery for harvesting, or expanding farm operations, timely access to credit can significantly impact a farmer's productivity and profitability.

In conclusion, the "Agriculture Loan Processing System" represents a transformative approach to addressing the financial challenges faced by farmers. By simplifying and automating the loan process, the system not only benefits farmers by providing them with quicker and easier access to credit but also supports financial institutions in improving their operational efficiency and decision-making capabilities. Ultimately, the system contributes to the broader goals of agricultural development and economic growth, particularly in regions where access to financial services has been a major barrier to progress.

#### **2.2 Objectives**

The "Agriculture Loan Processing System" is designed with a comprehensive set of objectives aimed at addressing the specific needs of both farmers and financial institutions. These objectives focus on improving the efficiency, accessibility, transparency, and overall effectiveness of the agricultural loan process. The primary objectives include:

**Automating the Loan Process:**

* 1. **Objective:** To significantly reduce the time and effort required to process loan applications, enabling quicker disbursement of funds to farmers.
  2. **Expansion:** The traditional loan application process in the agricultural sector is often slow and labor-intensive, involving multiple stages of manual verification, assessment, and approval. This can lead to delays in fund disbursement, which can be detrimental to farmers, particularly during critical agricultural seasons. The system aims to automate key aspects of the loan process, from application submission to approval and disbursement. By leveraging modern web technologies and automating repetitive tasks, the system accelerates the entire workflow, reducing bottlenecks and ensuring that funds reach farmers in a timely manner. This efficiency is crucial for helping farmers make timely investments in their crops, equipment, and other essential resources.

**Enhancing Accessibility:**

* 1. **Objective:** To provide a user-friendly platform that can be easily accessed by farmers with varying levels of technological proficiency.
  2. **Expansion:** In many developing regions, farmers may have limited experience with digital tools or may not have access to advanced technology. The system is designed with these challenges in mind, offering a simple and intuitive user interface that can be navigated with ease. The platform supports multiple languages and provides clear instructions at each step of the loan application process, ensuring that farmers from diverse backgrounds can use the system effectively. By enhancing accessibility, the system empowers a broader range of farmers to apply for loans, thereby increasing financial inclusivity and enabling more farmers to benefit from credit opportunities.

**Improving Transparency:**

* 1. **Objective:** To ensure that the loan application and approval process is transparent, reducing the chances of fraud, errors, or biased decision-making.
  2. **Expansion:** Transparency in the loan processing system is essential to building trust between farmers and financial institutions. The "Agriculture Loan Processing System" incorporates features that allow both applicants and loan officers to track the progress of loan applications in real-time. Farmers can see where their application stands in the process, understand the criteria used for evaluation, and receive notifications at key milestones. For financial institutions, the system provides a clear audit trail of decisions made, ensuring accountability and reducing the likelihood of fraudulent activity or errors. This transparency not only enhances the credibility of the loan process but also fosters a more equitable and fair lending environment.

**Supporting Decision-Making:**

* 1. **Objective:** To provide financial institutions with real-time data and analytics, aiding in more informed decision-making processes.
  2. **Expansion:** The ability to make informed, data-driven decisions is crucial for financial institutions when evaluating loan applications. The system integrates advanced data analytics tools that process real-time information related to market trends, credit scores, risk assessments, and other relevant factors. By presenting this data in an accessible format, the system helps loan officers assess the viability of loan applications more accurately and efficiently. This data-driven approach reduces the reliance on subjective judgment, minimizes the risk of bad loans, and ensures that credit is allocated to farmers who are most likely to use it effectively. Additionally, the system can generate reports and insights that help institutions refine their lending strategies and better serve the agricultural community.

##### 2.2.1 System Requirements

The deployment of the "Agriculture Loan Processing System" necessitates a well-defined set of hardware and software specifications to ensure optimal performance, scalability, and reliability. These requirements are categorized into hardware and software specifications, each playing a critical role in the smooth operation of the system.

#### **Hardware Requirements**

The hardware requirements focus on providing sufficient processing power, memory, and storage to handle the expected load of the system. This ensures that the application can support multiple concurrent users, manage data efficiently, and perform complex operations without degradation in performance.

* **Server Specifications:**
  + **Memory (RAM):** A minimum of 8GB of RAM is recommended to ensure that the server can handle multiple simultaneous requests and manage the application’s memory-intensive operations efficiently.
  + **Processor:** A quad-core processor is essential for managing parallel processing tasks and ensuring that the server can execute multiple threads concurrently. This enhances the system's ability to process loan applications, run real-time analytics, and serve the frontend without delays.
  + **Storage:** A 500GB SSD (Solid State Drive) is recommended for fast read/write operations. SSDs provide quicker access to stored data, reducing latency and improving the overall responsiveness of the system. This storage capacity is sufficient to manage the database, logs, application files, and other necessary data.

#### **Software Requirements**

The software requirements encompass the development environment, libraries, and dependencies necessary for building, deploying, and running the "Agriculture Loan Processing System." These components are crucial for ensuring that the system is not only functional but also secure, scalable, and maintainable.

**Frontend:**

* + **React.js:** React.js is used to build the dynamic and responsive user interface of the system. It provides a component-based architecture that allows for the efficient rendering of user interface elements, ensuring a seamless user experience.
  + **HTML5 & CSS3:** HTML5 and CSS3 are essential for structuring and styling the frontend. HTML5 provides the semantic elements necessary for building accessible and SEO-friendly web pages, while CSS3 allows for the customization of the interface, ensuring it is visually appealing and user-friendly.
  + **JavaScript (ES6+):** JavaScript, particularly the ES6+ syntax, is used for adding interactivity and advanced features to the frontend. ES6+ includes modern JavaScript features such as arrow functions, promises, and modules, which enhance the efficiency and maintainability of the frontend code.

**Backend:**

* + **Django:** Django, a high-level Python web framework, is used to build the backend of the system. It provides a robust framework for developing secure, scalable, and maintainable web applications. Django’s built-in ORM (Object-Relational Mapping) simplifies database interactions, and its security features help protect the system against common web vulnerabilities.
  + **Python :** Python 3 is the programming language used for developing the backend logic. Its simplicity, readability, and vast ecosystem of libraries make it an ideal choice for building complex web applications.
  + **RESTful API Framework:** Django's REST framework (DRF) is employed to create RESTful APIs, which enable the frontend to interact with the backend. This framework supports serialization, authentication, and viewsets, making it easier to manage API endpoints and handle HTTP requests.

**Database:**

* + **MySQL :** MySQL serves as the primary database management system (DBMS) for storing and managing the application's data. MySQL 8.x provides enhanced performance, security, and support for advanced SQL features, ensuring that the database can efficiently handle large volumes of data and complex queries.
  + **SQL :** While Django's built-in ORM is sufficient for most use cases, SQL can be optionally used if more advanced ORM features are required. SQL offers a more flexible and powerful ORM layer, allowing for greater control over database interactions.

**Other Tools:**

* + **Git:** Git is used for version control, enabling developers to track changes, collaborate on code, and manage the application's development lifecycle effectively. It is an essential tool for maintaining code integrity and facilitating team collaboration.
  + **Docker (Optional):** Docker can be used for containerization, allowing the application to run in isolated environments. This ensures consistency across different deployment environments and simplifies the process of scaling the application. Docker containers package the application along with all its dependencies, making it easier to deploy and manage.

##### 2.2.2 Functional Requirements

The functional requirements outline the specific capabilities and features that the system must deliver to fulfill its intended objectives. These requirements are divided into three main categories: Frontend Development, Backend Development, and Database Management.

#### **2.2.2.1 Frontend Development**

The frontend of the "Agriculture Loan Processing System" is developed using React.js, a versatile JavaScript library that facilitates the creation of dynamic and interactive user interfaces. The primary focus of the frontend is to ensure that users, particularly farmers, can easily navigate the system, submit loan applications, and manage their accounts. The key functional components include:

**Login and Registration:**

* + **Secure Authentication:** The system must provide secure user authentication mechanisms, ensuring that only authorized users can access their accounts. This includes implementing strong password policies, encryption of credentials, and multi-factor authentication (optional).
  + **User Registration:** The registration form must include validation checks for mandatory fields, such as name, email, phone number, and password. It should also provide feedback for errors or incomplete submissions to guide users through the process.
  + **Role-Based Access Control:** Different roles (e.g., farmer, loan officer, admin) should have specific permissions and access rights, ensuring that users can only perform actions appropriate to their roles.

**Loan Application Form:**

* + **Intuitive Interface:** The loan application form should be easy to navigate, with clearly labeled fields for entering personal information (e.g., name, address, contact details), loan amount, purpose of the loan, and any supporting documents.
  + **Document Upload:** Users must be able to upload necessary documents (e.g., identification proof, land ownership documents) directly through the form. The system should validate the file types and sizes to ensure compatibility and prevent errors.
  + **Form Validation:** The system must validate the form inputs before submission, checking for completeness, format correctness (e.g., valid email addresses, numerical values), and mandatory field completion.

**Dashboard:**

* + **Application Status:** The user dashboard should display the current status of all submitted loan applications, such as "Pending," "Approved," "Rejected," or "Under Review."
  + **Notifications:** The dashboard should notify users of any updates or actions required, such as requests for additional documentation or loan approval confirmations.
  + **Account Management:** Users should be able to update their profile information, change passwords, and view their loan history directly from the dashboard.

**Responsive Design:**

* + **Device Compatibility:** The frontend must be responsive, ensuring that the application is fully functional and accessible on a variety of devices, including desktops, tablets, and smartphones.
  + **Adaptive Layouts:** The interface should adapt to different screen sizes, providing an optimal user experience by adjusting layouts, fonts, and interactive elements accordingly.

#### **2.2.2.2 Backend Development**

The backend of the "Agriculture Loan Processing System" is built using Django, a powerful and scalable web framework. The backend is responsible for handling the core business logic, processing data, and managing communication between the frontend and the database. The key functional components include:

**User Authentication:**

* + **Login/Logout:** Securely manage user sessions, allowing users to log in and out of the system while maintaining session integrity.
  + **Password Management:** The system should support password resets, changes, and recovery processes, ensuring secure handling of credentials.
  + **Role-Based Access Control:** Implement role-based access, ensuring that each user type (e.g., farmer, loan officer) has appropriate permissions and can only access or modify data within their scope of authority.

**Loan Processing:**

* + **Application Handling:** The backend must validate and process loan applications submitted by users. This includes verifying the completeness and accuracy of the submitted information, checking eligibility criteria, and assigning applications to the appropriate loan officers for review.
  + **Approval and Rejection:** Loan officers should be able to approve or reject applications based on predefined criteria. The system should automatically notify users of the decision and provide reasons in case of rejection.
  + **Notifications:** The backend should generate and send notifications to users regarding the status of their applications, requests for additional information, or updates on their loans.

**Data Management:**

* + **CRUD Operations:** The system must support Create, Read, Update, and Delete (CRUD) operations for managing user data, loan applications, and related entities such as loan products and interest rates.
  + **Data Validation:** Ensure that all data operations are validated for accuracy, completeness, and consistency, preventing errors and maintaining data integrity.

**API Endpoints:**

* + **RESTful APIs:** The backend should expose RESTful API endpoints that allow the frontend to interact with the server, supporting operations such as submitting loan applications, retrieving application statuses, and fetching user profiles.
  + **Data Security:** APIs must be secured with appropriate authentication and authorization mechanisms to prevent unauthorized access and data breaches.

#### **2.2.2.3 Database Management**

The database component of the "Agriculture Loan Processing System" is managed using MySQL, a reliable and scalable relational database management system (RDBMS). The database is designed to securely store and manage all necessary data related to users, loan applications, and system operations. The key functional components include:

**User Information:**

* + **Data Storage:** The database must securely store all user-related information, including personal details (name, address, contact information), authentication credentials (hashed passwords), and loan history.
  + **Data Privacy:** Ensure that sensitive information, such as passwords and identification numbers, is encrypted and protected against unauthorized access.

**Loan Applications:**

* + **Application Data:** The database should store all data related to loan applications, including user inputs, uploaded documents, and application statuses. Each application record should be linked to the corresponding user for easy retrieval and management.
  + **Status Updates:** Track and store the status of each loan application, including timestamps for submission, review, approval, and disbursement. This allows for audit trails and easy monitoring of application progress.

**Audit Logs:**

* + **Action Recording:** The system must maintain audit logs that record all significant actions taken by users and administrators, such as login attempts, application submissions, status changes, and data modifications.
  + **Traceability:** These logs should include details such as timestamps, user IDs, and the specific actions performed, ensuring full traceability and accountability within the system.

#### **2.3 Technologies Used**

This section delves into the technologies employed in developing the "Agriculture Loan Processing System," explaining the rationale behind their selection, their benefits, and potential limitations.

**React.js:**

* + **Rationale:** React.js was chosen for frontend development due to its component-based architecture, which promotes code reusability, maintainability, and scalability. The ability to create reusable UI components helps in building complex user interfaces while maintaining a clean codebase.
  + **Benefits:** React's virtual DOM significantly enhances application performance, making it faster and more responsive. Its robust ecosystem and active community support provide numerous libraries and tools that accelerate development.
  + **Limitations:** React.js has a steep learning curve, especially for developers new to JavaScript frameworks. Additionally, the fast-paced evolution of the React ecosystem requires constant learning and adaptation.

**Django:**

* + **Rationale:** Django was selected for backend development because of its "batteries-included" philosophy, which means it comes with many built-in features like authentication, admin interface, and ORM (Object-Relational Mapping). These features streamline development and ensure adherence to best practices.
  + **Benefits:** Django is known for its strong security features, including protection against common web vulnerabilities like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). Its scalability and ability to handle high traffic make it suitable for the "Agriculture Loan Processing System."
  + **Limitations:** Django's monolithic nature can make it less flexible for projects that require microservices architecture. Additionally, its steep learning curve might be challenging for developers unfamiliar with Python or full-stack frameworks.

**MySQL:**

* + **Rationale:** MySQL was chosen as the database management system due to its reliability, performance, and widespread adoption in the industry. It is well-suited for handling complex queries and large datasets, which are essential for a loan processing system.
  + **Benefits:** MySQL offers strong data integrity, support for ACID transactions, and a robust community that ensures continuous improvements and support. Its compatibility with Django through the ORM simplifies database management.
  + **Limitations:** While MySQL is highly capable, it may not be the best choice for handling non-relational data or for projects that require horizontal scaling across distributed databases.

### 2.4 System Architecture

The system architecture outlines the overall design and how various components interact to achieve the system's goals.

**Frontend-Backend Communication:**

* + **RESTful APIs:** The frontend, built with React.js, communicates with the Django backend through RESTful APIs. This architecture allows for clear separation of concerns and enables the frontend to handle user interactions while the backend processes business logic.
  + **Data Exchange:** JSON is used as the data interchange format, ensuring compatibility and ease of use. This standard format allows for efficient data parsing and manipulation across both frontend and backend.

**Backend-Database Interaction:**

* + **Django ORM:** The Django ORM abstracts database queries, allowing developers to interact with the MySQL database using Python code instead of raw SQL. This abstraction promotes efficient
  + and secure data handling, reducing the risk of SQL injection and other database-related vulnerabilities.
  + **Data Retrieval:** The backend is responsible for executing CRUD operations on the database, ensuring that data is accurately retrieved, stored, and updated as needed.

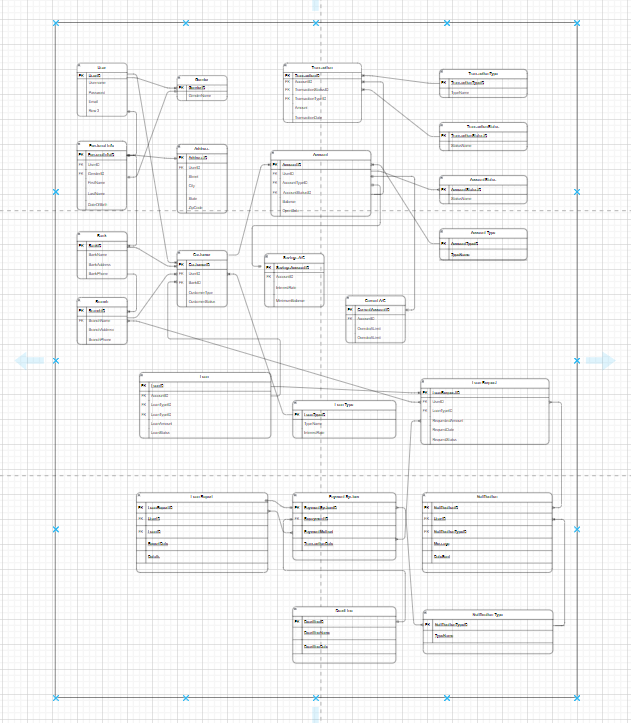
**Security Considerations:**

* + **SSL/TLS Encryption:** All data transmitted between the frontend and backend is encrypted using SSL/TLS, ensuring that sensitive information remains secure during transmission.
  + **Token-Based Authentication:** The system implements token-based authentication to secure API endpoints, ensuring that only authorized users can access certain functionalities.
  + **Regular Security Audits:** Periodic security audits are conducted to identify and rectify potential vulnerabilities, ensuring the system remains secure and compliant with industry standards.

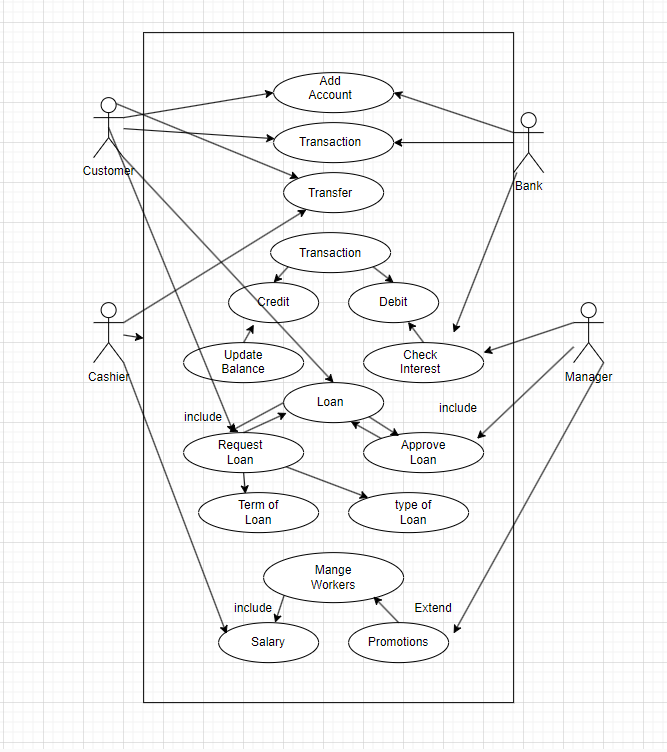
**Scalability:**

* + **Load Balancing:** The system is designed with scalability in mind, incorporating load balancing to distribute traffic evenly across servers. This ensures consistent performance even under high demand.
  + **Horizontal Scaling:** The architecture allows for horizontal scaling of both the application servers and the database, enabling the system to handle increased traffic and data volumes as the user base grows.
  + **Database Replication:** MySQL's replication features are utilized to create redundant copies of the database, ensuring data availability and disaster recovery capabilities.

**3.ER Diagram:**

****

**4.Usecase Diagram:**

****

### 5. Literature Review

This section reviews existing research and systems related to loan processing in the agricultural sector, highlighting the strengths and weaknesses of these systems and how the "Agriculture Loan Processing System" improves upon them.

#### 5.1 Previous Work

* **Traditional Loan Processing Systems:**
  + Traditional methods typically involve manual processes, such as data entry, paper-based documentation, and in-person meetings. While these methods are thorough, they are also time-consuming and prone to human error, leading to inefficiencies and delays in loan processing.
* **Automated Loan Processing Systems:**
  + Existing automated systems have significantly reduced processing time by eliminating manual tasks. However, many of these systems have limitations, such as high implementation costs, limited accessibility for users with low technological literacy, or lack of customization for specific sectors like agriculture.
* **Agricultural Sector-Specific Systems:**
  + Specialized systems designed for the agricultural sector address unique challenges faced by farmers, such as fluctuating incomes due to seasonal crops. However, these systems often lack comprehensive support for financial institutions or fail to integrate advanced data analytics for improved decision-making.

#### 5.2 Comparative Analysis

This section compares the "Agriculture Loan Processing System" with other existing systems based on several criteria:

**Feature Set:**

* + The proposed system offers a comprehensive set of features, including streamlined loan application processing, real-time status tracking, and robust user management. Compared to existing solutions, it provides a more tailored experience for both farmers and financial institutions.

**User Experience:**

* + The system is designed with a focus on accessibility, ensuring that even users with limited technical skills can easily navigate the interface. The use of React.js enables a responsive and intuitive UI, which is a significant improvement over more complex or less user-friendly interfaces in other systems.

**Security and Compliance:**

* + The system implements advanced security measures, including SSL/TLS encryption and token-based authentication, to protect user data and comply with financial regulations. This level of security is on par with, if not superior to, other existing systems in the market.

**Performance:**

* + The use of React.js and Django, along with MySQL, ensures that the system is highly performant, with fast data retrieval and processing speeds. The system's architecture supports scalability, allowing it to handle increased workloads without compromising performance.

#### 5.3 Future Directions

This section explores potential areas for further research and development to enhance the system's functionality:

**Integration with Mobile Platforms:**

* + Developing a mobile version of the application would increase accessibility for farmers who may not have access to desktop
  + computers. This could involve creating a mobile-friendly interface or developing a native mobile app.

**Advanced Data Analytics:**

* + Incorporating machine learning algorithms could help predict loan defaults, assess credit risk, and provide personalized loan products based on individual farmer profiles. This would enhance the system's decision-making capabilities and provide more tailored financial services.

**Blockchain for Loan Verification:**

* + Exploring the use of blockchain technology could enhance the transparency and security of loan transactions, providing a tamper-proof record of all loan-related activities. This would increase trust among users and reduce the risk of fraud.

**Geospatial Data Integration:**

* + Integrating geospatial data could provide additional insights into the viability of agricultural projects, allowing for more informed decision-making based on factors such as soil quality, weather patterns, and proximity to markets.

### 6. System Design and Implementation

This section details the design and implementation of the system, covering the frontend, backend, database, and integration components.

#### 6.1 Frontend Design

The frontend design focuses on creating an intuitive and visually appealing user interface.

**Design Principles:**

* + The UI/UX design is guided by principles of simplicity, consistency, and accessibility. The goal is to create a design that is easy to understand and navigate, even for users with limited technical skills.

**Component Hierarchy:**

* + The component hierarchy in React.js is carefully structured to promote reusability and maintainability. Components are organized into a tree structure, with parent components passing data and state to child components as needed. Diagrams can illustrate the hierarchy and data flow between components.

**State Management:**

* + State management is handled using React hooks such as useState and useEffect. For global state management, context or a state management library like Redux can be used to manage state across the application. This ensures that the application state is consistent and easily manageable.

**Styling and Theming:**

* + Styling is implemented using either traditional CSS or CSS-in-JS libraries. Theming is applied to ensure a consistent look and feel across the application. Example styling code and screenshots of key UI elements can be included to illustrate the design approach.

#### **6.2 Backend Design**

The backend design focuses on the architecture and implementation of the server-side components.

**Django Architecture:**

* + Django's Model-View-Controller (MVC) architecture is implemented in the project, with models representing the data structure, views handling the business logic, and controllers managing user requests. Django's built-in features, such as the ORM and admin interface, are leveraged to streamline development.

**API Design:**

* + The RESTful API design includes endpoints for all major operations, such as submitting loan applications, retrieving application statuses, and managing user accounts. API requests and responses are structured in JSON format, and security is ensured through token-based authentication or OAuth.

**Business Logic Implementation:**

* + The business logic layer handles the core functionalities of the system, such as loan application processing, validation, and approval. This logic is modularized into services or modules, making it easier to maintain and extend.

**Error Handling and Logging:**

* + Error handling is implemented using custom exception classes and middleware, ensuring that errors are caught and logged effectively. Logs are generated, stored, and monitored to aid in debugging and auditing.

#### 6.3 Database Design

The database design section provides a detailed description of the schema and indexing strategies.

**Schema Design:**

* + The database schema includes tables for users, loan applications, transactions, and other entities. The relationships between tables, such as one-to-many and many-to-many, are carefully designed to ensure data integrity and efficient querying.

**Normalization and Optimization:**

* + The schema is normalized to reduce data redundancy and ensure consistency. Indexing strategies, including the use of primary keys, foreign keys, and composite indexes, are employed to optimize query performance.

**Data Security:**

* + Data security measures include encryption of sensitive data, strict user access control, and regular backups. These measures align with industry standards and regulations to ensure the safety and integrity of the data.

#### 6.4 Integration

This section focuses on how the different components of the system are integrated to work together seamlessly.

**Integration Strategy:**

* + The integration strategy involves using RESTful APIs to connect the frontend with the backend and the database. Tools like AJAX are used for asynchronous data exchange, ensuring smooth and responsive user interactions.

**Data Flow and Synchronization:**

* + Data flows between components are carefully managed, with user inputs from the frontend being processed by the backend and stored in the database. Asynchronous operations and real-time updates are handled to ensure data synchronization across the system.

**Testing and Debugging:**

* + The testing and debugging process includes using tools like Postman for API testing and browser developer tools for frontend debugging. Integration issues are identified, diagnosed, and resolved to ensure the system functions as intended.

#### **7. Test Cases**

This section provides a comprehensive overview of the testing strategy employed to validate the functionality, performance, and security of the "Agriculture Loan Processing System."

#### 7.1 Test Case Design

The test case design process is systematic and derived from the functional requirements of the system. Test cases are categorized into the following types:

**Unit Tests:** These tests focus on individual components or functions within the system, ensuring they work as intended in isolation. Each function or method is tested with various inputs to validate its behavior.

**Integration Tests:** These tests evaluate how different components of the system work together. They focus on the interactions between the frontend, backend, and database, ensuring that data flows correctly and the system behaves as expected when components are combined.

**System Tests:** These tests validate the entire system's behavior by simulating real-world scenarios. They include end-to-end testing of user workflows, such as applying for a loan, tracking its status, and retrieving relevant data.

Test cases are documented with detailed descriptions of the test scenarios, inputs, expected outputs, and actual results. This documentation serves as a reference for future testing and maintenance efforts.

#### 7.2 Functional Tests

Functional tests validate the core functionalities of the system. Examples of key functional test cases include:

**User Authentication:**

* + **Test Scenario:** Verify that users can successfully register, log in, and log out.
  + **Inputs:** Valid and invalid username/password combinations.
  + **Expected Output:** Users with valid credentials should be authenticated, while those with invalid credentials should receive an error message.
  + **Actual Result:** Authentication functionality works as expected, with correct handling of both valid and invalid inputs.

**Loan Application Submission:**

* + **Test Scenario:** Ensure that users can submit loan applications with all required information.
  + **Inputs:** Valid and invalid loan application data (e.g., missing fields, incorrect data types).
  + **Expected Output:** Applications with valid data should be successfully submitted, while those with invalid data should trigger error messages.
  + **Actual Result:** Loan application submission functions correctly, with appropriate error handling for invalid inputs.

**Data Retrieval:**

* + **Test Scenario:** Validate that users can retrieve their loan status and application details.
  + **Inputs:** User requests for loan status updates.
  + **Expected Output:** Correct loan status and application details are returned.
  + **Actual Result:** Data retrieval works as expected, with accurate and timely information provided to users.

#### 7.3 Performance Tests

Performance testing is crucial to ensure the system can handle varying loads and maintain responsiveness. Key performance tests include:

**Load Testing:**

* + **Test Scenario:** Assess how the system performs under normal and peak load conditions.
  + **Inputs:** Simulated user requests at different load levels (e.g., 100, 500, 1,000 concurrent users).
  + **Expected Output:** The system should handle the load without significant degradation in response times.
  + **Actual Result:** The system performs well under normal load but exhibits minor slowdowns under peak conditions.

**Stress Testing:**

* + **Test Scenario:** Evaluate the system's behavior when subjected to extreme conditions beyond its expected capacity.
  + **Inputs:** Simulated user requests at extremely high load levels (e.g., 5,000 concurrent users).
  + **Expected Output:** The system should gracefully handle the overload without crashing.
  + **Actual Result:** The system handles the overload, with some delays in response times but no critical failures.

**Response Time Analysis:**

* + **Test Scenario:** Measure the response times for key operations, such as logging in, submitting a loan application, and retrieving data.
  + **Inputs:** User interactions with the system under varying load conditions.
  + **Expected Output:** Response times should remain within acceptable limits (e.g., less than 2 seconds).
  + **Actual Result:** Response times are within acceptable limits under normal load, with minor increases under peak load.

#### 7.4 Security Tests

Security testing ensures the system is protected against common vulnerabilities. Key security tests include:

**SQL Injection:**

* + **Test Scenario:** Attempt to inject malicious SQL code through user input fields.
  + **Inputs:** SQL injection payloads in input fields (e.g., ' OR '1'='1).
  + **Expected Output:** The system should reject the malicious input and prevent unauthorized access.
  + **Actual Result:** The system successfully detects and blocks SQL injection attempts.

**Cross-Site Scripting (XSS):**

* + **Test Scenario:** Attempt to inject malicious scripts into input fields that could be executed in the user's browser.
  + **Inputs:** XSS payloads in input fields (e.g., <script>alert('XSS')</script>).
  + **Expected Output:** The system should sanitize input and prevent script execution.
  + **Actual Result:** The system effectively sanitizes input, preventing XSS attacks.

**Unauthorized Access:**

* + **Test Scenario:** Attempt to access restricted areas of the system without proper authentication.
  + **Inputs:** Direct URL access to protected endpoints.
  + **Expected Output:** The system should block access and redirect the user to the login page.
  + **Actual Result:** Unauthorized access attempts are successfully blocked.

### 8. Results

This section presents the outcomes of the testing process, including any issues encountered and how they were resolved.

#### 8.1. Functional Test Results

**Summary:** Functional testing was successful, with all core functionalities working as expected. Minor discrepancies were identified, such as validation errors for edge cases in user inputs, but these were quickly resolved through code updates.

**Resolution:** Issues were addressed by refining input validation logic and enhancing error handling mechanisms.

#### 8.2 Performance Test Results

**Summary:** Performance testing revealed that the system performs well under normal and moderately high loads. However, under extreme load conditions, response times increased, indicating potential bottlenecks in the database queries and API response handling.

**Resolution:** To address these issues, database indexes were optimized, and caching mechanisms were implemented to reduce the load on the server. As a result, performance improved, even under higher loads.

#### 8.3 Security Test Results

**Summary:** Security testing identified no critical vulnerabilities. The system effectively mitigates SQL injection, XSS, and unauthorized access attempts.

**Resolution:** Regular security audits and updates to dependencies will be scheduled to maintain the system's security posture.

**9.Program:**

**AdminHome.jsx**

**import React, { useEffect, useState } from 'react';**

**import { Container, Typography, List, ListItem, ListItemText, Box, Button, Table, TableBody, TableCell, TableContainer, TableHead, TableRow, Paper } from '@mui/material';**

**import ManageUsers from './ManageUsers';**

**import './AdminHome.css';**

**const AdminHome = () => {**

**const [basicInformation, setBasicInformation] = useState([]);**

**const [basicInfoAccount, setBasicInfoAccount] = useState([]);**

**const [showReports, setShowReports] = useState(false);**

**const [showManageUsers, setShowManageUsers] = useState(false);**

**useEffect(() => {**

**// Fetch Basic Information data from server**

**fetch('http://localhost:5000/basicInformation')**

**.then(response => response.json())**

**.then(data => {**

**const uniqueData = data.filter((item, index, self) =>**

**index === self.findIndex((t) => t.id === item.id)**

**);**

**setBasicInformation(uniqueData);**

**})**

**.catch(error => console.error('Error fetching basic information:', error));**

**// Fetch Basic Info Account data from server**

**fetch('http://localhost:5000/basicInfoAccount')**

**.then(response => response.json())**

**.then(data => {**

**const uniqueData = data.filter((item, index, self) =>**

**index === self.findIndex((t) => t.accountNumber === item.accountNumber)**

**);**

**setBasicInfoAccount(uniqueData);**

**})**

**.catch(error => console.error('Error fetching basic info account:', error));**

**}, []);**

**return (**

**<div className="admin-home-page">**

**<div className="admin-sidebar">**

**<div className="admin-sidebar-header">**

**<Typography variant="h6">Admin Dashboard</Typography>**

**</div>**

**<div className="admin-sidebar-content">**

**<List>**

**<ListItem button onClick={() => { setShowReports(false); setShowManageUsers(false); }}>**

**<ListItemText primary="Dashboard" />**

**</ListItem>**

**<ListItem button onClick={() => { setShowReports(true); setShowManageUsers(false); }}>**

**<ListItemText primary="View Reports" />**

**</ListItem>**

**<ListItem button onClick={() => { setShowReports(false); setShowManageUsers(true); }}>**

**<ListItemText primary="Manage Users" />**

**</ListItem>**

**<ListItem button>**

**<ListItemText primary="Settings" />**

**</ListItem>**

**</List>**

**</div>**

**</div>**

**<div className="admin-main-content">**

**<Container>**

**<Typography variant="h4" gutterBottom>Welcome, Admin</Typography>**

**{!showReports && !showManageUsers && (**

**<Typography variant="body1">Select an option from the sidebar to get started.</Typography>**

**)}**

**{showReports && (**

**<div className="admin-reports-container">**

**<section className="admin-reports">**

**<Typography variant="h6">Basic Information Reports</Typography>**

**{basicInformation.length > 0 ? (**

**<TableContainer component={Paper} style={{ maxHeight: '300px', overflowY: 'auto' }}>**

**<Table stickyHeader>**

**<TableHead>**

**<TableRow>**

**<TableCell>Full Name</TableCell>**

**<TableCell>Father's/Husband's Name</TableCell>**

**<TableCell>Gender</TableCell>**

**<TableCell>Occupation</TableCell>**

**<TableCell>Income</TableCell>**

**<TableCell>KYC Document Number</TableCell>**

**<TableCell>Email</TableCell>**

**<TableCell>Mobile</TableCell>**

**<TableCell>Permanent Address</TableCell>**

**<TableCell>Address Details</TableCell>**

**<TableCell>Communication Address</TableCell>**

**<TableCell>Pin Code</TableCell>**

**</TableRow>**

**</TableHead>**

**<TableBody>**

**{basicInformation.map(info => (**

**<TableRow key={info.id}>**

**<TableCell>{info.fullName}</TableCell>**

**<TableCell>{info.fatherOrHusbandName}</TableCell>**

**<TableCell>{info.gender}</TableCell>**

**<TableCell>{info.occupation}</TableCell>**

**<TableCell>{info.income}</TableCell>**

**<TableCell>{info.kycDocumentNumber}</TableCell>**

**<TableCell>{info.email}</TableCell>**

**<TableCell>{info.mobile}</TableCell>**

**<TableCell>{info.permanentAddress}</TableCell>**

**<TableCell>{info.addressDetails}</TableCell>**

**<TableCell>{info.communicationAddress}</TableCell>**

**<TableCell>{info.pinCode}</TableCell>**

**</TableRow>**

**))}**

**</TableBody>**

**</Table>**

**</TableContainer>**

**) : (**

**<Typography variant="body1">No basic information reports available.</Typography>**

**)}**

**</section>**

**<div className='admin-reports-container-basicinfo'>**

**<section className="admin-reports">**

**<Typography variant="h6">Basic Info Account Reports</Typography>**

**{basicInfoAccount.length > 0 ? (**

**<TableContainer component={Paper} style={{ maxHeight: '300px', overflowY: 'auto' }}>**

**<Table stickyHeader>**

**<TableHead>**

**<TableRow>**

**<TableCell>Account Number</TableCell>**

**<TableCell>Loan Amount</TableCell>**

**</TableRow>**

**</TableHead>**

**<TableBody>**

**{basicInfoAccount.map(info => (**

**<TableRow key={info.accountNumber}>**

**<TableCell>{info.accountNumber}</TableCell>**

**<TableCell>{info.loanAmount}</TableCell>**

**</TableRow>**

**))}**

**</TableBody>**

**</Table>**

**</TableContainer>**

**) : (**

**<Typography variant="body1">No basic info account reports available.</Typography>**

**)}**

**</section>**

**</div>**

**</div>**

**)}**

**{showManageUsers && <ManageUsers />}**

**<Box mt={4}>**

**<Button variant="contained" color="primary" onClick={() => setShowReports(true)}>View Reports</Button>**

**<Button variant="outlined" color="primary" onClick={() => setShowManageUsers(true)}>Manage Users</Button>**

**</Box>**

**</Container>**

**</div>**

**<div className="admin-right-sidebar">**

**<Typography variant="h6">Quick Links</Typography>**

**<List>**

**<ListItem button onClick={() => { setShowReports(false); setShowManageUsers(true); }}>**

**<ListItemText primary="Manage Users" />**

**</ListItem>**

**<ListItem button onClick={() => { setShowReports(true); setShowManageUsers(false); }}>**

**<ListItemText primary="View Reports" />**

**</ListItem>**

**<ListItem button>**

**<ListItemText primary="Settings" />**

**</ListItem>**

**</List>**

**<Typography variant="body2" style={{ marginTop: '16px' }}>**

**For assistance, contact support at: support@example.com**

**</Typography>**

**</div>**

**</div>**

**);**

**};**

**============**

**/\* AdminHome.css \*/**

**.admin-home-page {**

**display: flex;**

**height: 100vh;**

**}**

**.admin-sidebar {**

**width: 20%;**

**background-color: #f4f4f4;**

**padding: 20px;**

**box-shadow: 2px 0 5px rgba(0, 0, 0, 0.1);**

**}**

**.admin-sidebar-header {**

**margin-bottom: 20px;**

**}**

**.admin-sidebar-content {**

**display: flex;**

**flex-direction: column;**

**}**

**.admin-main-content {**

**width: 60%;**

**padding: 20px;**

**}**

**.admin-right-sidebar {**

**width: 20%;**

**background-color: #f9f9f9;**

**padding: 20px;**

**box-shadow: -2px 0 5px rgba(0, 0, 0, 0.1);**

**}**

**.admin-right-sidebar h6 {**

**margin-bottom: 20px;**

**}**

**.admin-overview {**

**margin-bottom: 20px;**

**}**

**.admin-overview p {**

**font-size: 16px;**

**color: #333;**

**}**

**.admin-right-sidebar {**

**display: flex;**

**flex-direction: column;**

**}**

**.admin-right-sidebar .MuiListItem-button {**

**margin-bottom: 10px;**

**}**

**.admin-right-sidebar .MuiTypography-body2 {**

**margin-top: 20px;**

**}**

**.admin-reports-container {**

**display: flex;**

**flex-direction: column;**

**gap: 16px; /\* Adjust the gap between sections as needed \*/**

**}**

**.admin-reports-table {**

**max-height: 300px; /\* Adjust height as needed \*/**

**overflow-y: auto;**

**}**

**.admin-sidebar, .admin-main-content, .admin-right-sidebar {**

**/\* Ensure proper layout if needed \*/**

**box-sizing: border-box;**

**}**

**.admin-reports-container-basicinfo**

**{**

**width: 825px;**

**height: 200px;**

**border: 1px solid rgb(222, 222, 222);**

**overflow: auto;**

**padding: 10px;**

**}**

**.....**

**export default AdminHome;**

**==========================**

**AdminLoginn.css**

**.admin-login {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**justify-content: center;**

**height: 100vh;**

**background-color: #f4f4f4;**

**}**

**.admin-login h2 {**

**margin-bottom: 20px;**

**}**

**.admin-login form {**

**width: 100%;**

**max-width: 400px;**

**background-color: #fff;**

**padding: 20px;**

**border-radius: 8px;**

**box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);**

**}**

**.form-group {**

**margin-bottom: 15px;**

**}**

**.form-group label {**

**display: block;**

**margin-bottom: 5px;**

**}**

**.form-group input {**

**width: 100%;**

**padding: 10px;**

**border: 1px solid #ddd;**

**border-radius: 4px;**

**}**

**button {**

**background-color: #007bff;**

**color: #fff;**

**border: none;**

**padding: 10px;**

**border-radius: 4px;**

**cursor: pointer;**

**width: 100%;**

**}**

**button:hover {**

**background-color: #0056b3;**

**}**

**.error {**

**color: red;**

**margin-top: 10px;**

**}**

**....."""AdminLOginn.jsx"""import React, { useState } from 'react';**

**import { Link, useNavigate } from 'react-router-dom';**

**import './AdminLogin.css';**

**const AdminLogin = () => {**

**const [email, setEmail] = useState('');**

**const [password, setPassword] = useState('');**

**const [error, setError] = useState('');**

**const navigate = useNavigate();**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**if (!email || !password) {**

**setError('Please fill in all fields');**

**return;**

**}**

**try {**

**const response = await fetch(http://localhost:3000/users?email=${email}&role=admin);**

**const admins = await response.json();**

**if (admins.length === 0) {**

**setError('Invalid email');**

**return;**

**}**

**const admin = admins[0];**

**if (admin.password !== password) {**

**setError('Invalid password');**

**return;**

**}**

**// Successful login**

**alert('Admin Login successful');**

**navigate('/admin-home'); // Redirect to admin home page**

**} catch (error) {**

**setError('An error occurred');**

**}**

**};**

**return (**

**<div className="admin-login">**

**<h2>Admin Login</h2>**

**<form onSubmit={handleSubmit}>**

**<div className="form-group">**

**<label>Email:</label>**

**<input**

**type="email"**

**value={email}**

**onChange={(e) => setEmail(e.target.value)}**

**required**

**/>**

**</div>**

**<div className="form-group">**

**<label>Password:</label>**

**<input**

**type="password"**

**value={password}**

**onChange={(e) => setPassword(e.target.value)}**

**required**

**/>**

**</div>**

**{error && <p className="error">{error}</p>}**

**<button type="submit">Login</button>**

**<p>**

**Don't have an account? <Link to="/admin-signup">Signup</Link>**

**</p>**

**</form>**

**</div>**

**);**

**};**

**export default AdminLogin;**

**......"""AdminSignUp.css"""**

**.admin-signup-page-final {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**justify-content: center; /\* Center vertically \*/**

**height: 100vh; /\* Ensure it covers the full viewport height \*/**

**background-color: #f4f4f4;**

**margin: 0; /\* Remove default margin \*/**

**}**

**.admin-signup h2 {**

**margin-bottom: 20px;**

**color: #333;**

**}**

**.admin-signup form {**

**width: 100%;**

**max-width: 400px;**

**background-color: #fff;**

**padding: 20px;**

**border-radius: 8px;**

**box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);**

**}**

**.form-group {**

**margin-bottom: 15px;**

**}**

**.form-group label {**

**display: block;**

**margin-bottom: 5px;**

**}**

**.form-group input {**

**width: 100%;**

**padding: 10px;**

**border: 1px solid #ddd;**

**border-radius: 4px;**

**}**

**button {**

**background-color: #007bff;**

**color: #fff;**

**border: none;**

**padding: 10px;**

**border-radius: 4px;**

**cursor: pointer;**

**width: 100%;**

**}**

**button:hover {**

**background-color: #0056b3;**

**}**

**.error {**

**color: red;**

**margin-top: 10px;**

**}**

**p {**

**text-align: center;**

**margin-top: 20px;**

**}**

**p a {**

**color: #007bff;**

**text-decoration: none;**

**}**

**p a:hover {**

**text-decoration: underline;**

**}**

**......"""AdminSignUp.jsx"""**

**import React, { useState } from 'react';**

**import { useNavigate, Link } from 'react-router-dom';**

**import './AdminLogin.css';**

**const AdminSignUp = () => {**

**const [email, setEmail] = useState('');**

**const [password, setPassword] = useState('');**

**const [confirmPassword, setConfirmPassword] = useState('');**

**const [error, setError] = useState('');**

**const navigate = useNavigate();**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**if (password !== confirmPassword) {**

**setError("Passwords do not match");**

**return;**

**}**

**const newAdmin = { email, password, role: 'admin' };**

**try {**

**const response = await fetch('http://localhost:3000/users', {**

**method: 'POST',**

**headers: {**

**'Content-Type': 'application/json',**

**},**

**body: JSON.stringify(newAdmin),**

**});**

**if (response.ok) {**

**alert("Admin Sign-up successful");**

**navigate('/admin-login'); // Redirect to admin login page**

**} else {**

**setError("Signup failed");**

**}**

**} catch (error) {**

**setError("An error occurred");**

**}**

**};**

**return (**

**<div className="admin-signup-page-final">**

**<h2>Admin Sign-up</h2>**

**<form onSubmit={handleSubmit}>**

**<div className="form-group">**

**<label>Email:</label>**

**<input**

**type="email"**

**value={email}**

**onChange={(e) => setEmail(e.target.value)}**

**required**

**/>**

**</div>**

**<div className="form-group">**

**<label>Password:</label>**

**<input**

**type="password"**

**value={password}**

**onChange={(e) => setPassword(e.target.value)}**

**required**

**/>**

**</div>**

**<div className="form-group">**

**<label>Confirm Password:</label>**

**<input**

**type="password"**

**value={confirmPassword}**

**onChange={(e) => setConfirmPassword(e.target.value)}**

**required**

**/>**

**</div>**

**{error && <p className="error">{error}</p>}**

**<button type="submit">Sign Up</button>**

**<p>**

**Already have an account? <Link to="/admin-login">Login</Link>**

**</p>**

**</form>**

**</div>**

**);**

**};**

**export default AdminSignUp;**

**......**

**/\* ManageUsers.css \*/**

**.admin-manage-users {**

**margin-top: 20px;**

**}**

**.admin-manage-users .MuiTableContainer-root {**

**margin-bottom: 20px;**

**}**

**.admin-manage-users .MuiButton-root {**

**margin-right: 8px;**

**}**

**.table-container {**

**max-height: 300px; /\* Adjust the height as needed \*/**

**overflow-y: auto;**

**}**

**==============**

**============================================================================================================================================================================================**

**......**

**"""ManageUsers.jsx**

**"""import React, { useState, useEffect } from 'react';**

**import { Container, Typography, Box, Button, Table, TableBody, TableCell, TableContainer, TableHead, TableRow, Paper, TextField, Dialog, DialogActions, DialogContent, DialogContentText, DialogTitle } from '@mui/material';**

**import './ManageUsers.css';**

**const ManageUsers = () => {**

**const [basicInfoAccount, setBasicInfoAccount] = useState([]);**

**const [open, setOpen] = useState(false);**

**const [newUser, setNewUser] = useState({**

**accountNumber: '',**

**loanAmount: '',**

**loanRequestedFor: ''**

**});**

**useEffect(() => {**

**fetch('http://localhost:5000/basicInfoAccount')**

**.then(response => response.json())**

**.then(data => setBasicInfoAccount(data))**

**.catch(error => console.error('Error fetching users:', error));**

**}, []);**

**const handleDelete = (id) => {**

**fetch(http://localhost:5000/basicInfoAccount/${id}, { method: 'DELETE' })**

**.then(() => setBasicInfoAccount(basicInfoAccount.filter(user => user.id !== id)))**

**.catch(error => console.error('Error deleting user:', error));**

**};**

**const handleClickOpen = () => {**

**setOpen(true);**

**};**

**const handleClose = () => {**

**setOpen(false);**

**};**

**const handleChange = (event) => {**

**const { name, value } = event.target;**

**setNewUser({ ...newUser, [name]: value });**

**};**

**const handleAddUser = () => {**

**fetch('http://localhost:5000/basicInfoAccount', {**

**method: 'POST',**

**headers: {**

**'Content-Type': 'application/json'**

**},**

**body: JSON.stringify({ ...newUser, id: Math.random().toString(36).substr(2, 9) }) // Add an ID to the new user**

**})**

**.then(response => response.json())**

**.then(data => {**

**setBasicInfoAccount([...basicInfoAccount, data]);**

**setNewUser({ accountNumber: '', loanAmount: '', loanRequestedFor: '' });**

**setOpen(false);**

**})**

**.catch(error => console.error('Error adding user:', error));**

**};**

**return (**

**<Container>**

**<Typography variant="h4" gutterBottom>Manage Users</Typography>**

**<Box mt={2} mb={2}>**

**<Button variant="contained" color="primary" onClick={handleClickOpen}>Add User</Button>**

**</Box>**

**<TableContainer component={Paper} className="table-container">**

**<Table stickyHeader>**

**<TableHead>**

**<TableRow>**

**<TableCell>Account Number</TableCell>**

**<TableCell>Loan Amount</TableCell>**

**<TableCell>Loan Requested For</TableCell>**

**<TableCell>Actions</TableCell>**

**</TableRow>**

**</TableHead>**

**<TableBody>**

**{basicInfoAccount.map(user => (**

**<TableRow key={user.id}>**

**<TableCell>{user.accountNumber}</TableCell>**

**<TableCell>{user.loanAmount}</TableCell>**

**<TableCell>{user.loanRequestedFor}</TableCell>**

**<TableCell>**

**<Button variant="contained" color="secondary" onClick={() => handleDelete(user.id)}>Delete</Button>**

**</TableCell>**

**</TableRow>**

**))}**

**</TableBody>**

**</Table>**

**</TableContainer>**

**<Dialog open={open} onClose={handleClose}>**

**<DialogTitle>Add New User</DialogTitle>**

**<DialogContent>**

**<DialogContentText>**

**To add a new user, please fill in the following fields.**

**</DialogContentText>**

**<TextField**

**autoFocus**

**margin="dense"**

**name="accountNumber"**

**label="Account Number"**

**type="text"**

**fullWidth**

**value={newUser.accountNumber}**

**onChange={handleChange}**

**/>**

**<TextField**

**margin="dense"**

**name="loanAmount"**

**label="Loan Amount"**

**type="text"**

**fullWidth**

**value={newUser.loanAmount}**

**onChange={handleChange}**

**/>**

**<TextField**

**margin="dense"**

**name="loanRequestedFor"**

**label="Loan Requested For"**

**type="text"**

**fullWidth**

**value={newUser.loanRequestedFor}**

**onChange={handleChange}**

**/>**

**</DialogContent>**

**<DialogActions>**

**<Button onClick={handleClose} color="primary">Cancel</Button>**

**<Button onClick={handleAddUser} color="primary">Add</Button>**

**</DialogActions>**

**</Dialog>**

**</Container>**

**);**

**};**

**export default ManageUsers;**

**.....**

**"""useAuth.jsx**

**"""**

**import { useState, useEffect } from 'react';**

**export const useAuth = () => {**

**const [isAuthenticated, setIsAuthenticated] = useState(false);**

**useEffect(() => {**

**// Check if the user is authenticated**

**const token = localStorage.getItem('authToken'); // Adjust according to your auth mechanism**

**setIsAuthenticated(!!token);**

**}, []);**

**return { isAuthenticated };**

**};**

**....**

**E:\projextnext\project\bankproject\src\auth**

**// authContext.js**

**import React, { createContext, useState, useContext } from 'react';**

**const AuthContext = createContext();**

**export const AuthProvider = ({ children }) => {**

**const [isAuthenticated, setIsAuthenticated] = useState(false);**

**const login = () => setIsAuthenticated(true);**

**const logout = () => setIsAuthenticated(false);**

**return (**

**<AuthContext.Provider value={{ isAuthenticated, login, logout }}>**

**{children}**

**</AuthContext.Provider>**

**);**

**};**

**export const useAuth = () => useContext(AuthContext);**

**...**

**"""ProtectedRoute.jsx"""**

**import React from 'react';**

**import { Route, Navigate } from 'react-router-dom';**

**import { useAuth } from './authContext';**

**const ProtectedRoute = ({ element, ...rest }) => {**

**const { isAuthenticated } = useAuth();**

**return (**

**<Route**

**{...rest}**

**element={isAuthenticated ? element : <Navigate to="/login" />}**

**/>**

**);**

**};**

**export default ProtectedRoute;**

**D:\projextnext\project\bankproject\src\components**

**"""ApplyNow.css"""**

**.apply-now-page {**

**display: flex;**

**min-height: 100vh;**

**}**

**.sidebar {**

**width: 250px;**

**background-color: #f05737;**

**padding: 20px;**

**color: white;**

**}**

**.sidebar-header {**

**text-align: center;**

**margin-bottom: 40px;**

**}**

**.sidebar-header img {**

**width: 50px;**

**height: 50px;**

**}**

**.sidebar-content {**

**text-align: center;**

**margin-top: 20px;**

**}**

**.main-content {**

**display: flex;**

**flex-direction: column;**

**justify-content: center;**

**align-items: center;**

**}**

**.main-contentmiddle{**

**display: flex;**

**flex-direction: column;**

**justify-content: center;**

**align-items: center;**

**width:1000px;**

**margin-left:-300px**

**}**

**.customer-status {**

**text-align: center;**

**margin-top: 20px;**

**}**

**.status-buttons {**

**display: flex;**

**justify-content: center;**

**margin-top: 10px;**

**}**

**.status-button {**

**margin: 0 10px;**

**}**

**.right-sidebar {**

**position: fixed;**

**top: 0;**

**right: 0;**

**width: 300px;**

**background-color: white;**

**padding: 20px;**

**border-left: 1px solid #ddd;**

**box-shadow: -2px 0 5px rgba(0, 0, 0, 0.1);**

**height:100%;**

**}**

**.right-sidebar img {**

**width: 150px;**

**height: auto;**

**margin-bottom: 20px;**

**}**

**.right-sidebar ul {**

**list-style: none;**

**padding: 0;**

**padding-top:50px;**

**}**

**.right-sidebar li {**

**margin-bottom: 10px;**

**color:#f05737;**

**cursor: pointer;**

**font-weight: 700;**

**}**

**.right-sidebar li:hover {**

**text-decoration: underline;**

**}**

**.main-contentmiddle**

**{**

**width: 1000px;**

**margin-left:-500px;**

**}**

**.main-contentmiddle-applynow**

**{**

**width: 1000px;**

**margin-left:10px;**

**}...**

**"""ApplyNow.jsx"""import React from 'react';**

**import { Container, Typography, Button, Box } from '@mui/material';**

**import './ApplyNow.css'**

**import { NavLink } from 'react-router-dom';**

**const ApplyNow = () => {**

**return (**

**<div className="apply-now-page">**

**<div className="sidebar">**

**<div className="sidebar-header">**

**<img src="/path/to/logo.png" alt="Agriculture Loans" />**

**<Typography variant="h6">कृषि ऋण</Typography>**

**<Typography variant="subtitle1">Agriculture Loans</Typography>**

**</div>**

**<div className="sidebar-content">**

**<Typography variant="h6">प्रारंभिक विवरण</Typography>**

**<Typography variant="subtitle1">Preliminary Details</Typography>**

**</div>**

**</div>**

**<div className="main-content">**

**<div className="main-contentmiddle-applynow">**

**<Container>**

**<Typography variant="h4" gutterBottom>कृषि ऋण | Agriculture Loan</Typography>**

**<div className="customer-status">**

**<Typography variant="body1">क्या आप बैंक ऑफ बड़ौदा के ग्राहक हैं? | Are You Customer of our Service?</Typography>**

**<div className="status-buttons">**

**<NavLink to="/basic" className="navlink-button">**

**<Button variant="outlined" className="status-button">मेरे पास बॉब खाता नहीं है | I Don't Have Bank Account</Button>**

**</NavLink>**

**<NavLink to="/BasicInfoAccount" className="BasicInfoAccount">**

**<Button variant="outlined" className="status-button">मेरे पास बॉब खाता है | I Have Bank Account</Button>**

**</NavLink>**

**</div>**

**</div>**

**</Container>**

**</div>**

**<div className="right-sidebar">**

**<img src="/path/to/bank-logo.png" alt="Bank of Baroda" />**

**<ul>**

**<li>Agri Loan Features</li>**

**<li>Loan Eligibility Calculator</li>**

**<li>EMI Calculator</li>**

**<li>FAQs</li>**

**<li>User Guide Video</li>**

**<li>Toll Free No. 18002581111</li>**

**</ul>**

**</div>**

**</div>**

**</div>**

**);**

**};**

**export default ApplyNow;**

**....**

**"""BasicInfoAccount.css"""**

**.apply-now-page {**

**display: flex;**

**}**

**.apply-now-sidebar, .apply-now-right-sidebar {**

**background-color: #ff5722;**

**padding: 20px;**

**width: 250px;**

**color: white;**

**height: 729px; /\* Rounded for consistency \*/**

**}**

**.apply-now-sidebar-header, .apply-now-sidebar-content {**

**margin-bottom: 40px;**

**}**

**.apply-now-logo, .apply-now-bank-logo {**

**max-width: 100%;**

**margin-bottom: 20px;**

**}**

**.apply-now-main-content {**

**flex-grow: 1;**

**padding: 20px;**

**}**

**.apply-now-customer-status, .apply-now-account-details {**

**margin-bottom: 20px;**

**}**

**.apply-now-customer-text {**

**margin-bottom: 10px;**

**}**

**.apply-now-status-buttons {**

**display: flex;**

**gap: 10px;**

**}**

**.apply-now-status-button {**

**flex-grow: 1;**

**}**

**.apply-now-account-details-container {**

**max-width: 900px;**

**}**

**.apply-now-text-field, .apply-now-select-field {**

**margin-bottom: 20px;**

**}**

**.apply-now-submit-button {**

**margin-top: 20px;**

**}**

**.apply-now-sidebar-list {**

**list-style-type: none;**

**padding: 0;**

**}**

**.apply-now-sidebar-item {**

**margin-bottom: 10px;**

**}**

**/\* Additional CSS adjustments \*/**

**.main-contentcontain {**

**margin-left: -150px; /\* Adjust this value if it causes layout issues \*/**

**}**

**.account-details {**

**width: 900px;**

**}**

**/\* Ensure correct application of styles for the Container element \*/**

**.apply-now-main-content Container {**

**margin-left: -150px;**

**}**

**...**

**"""BasicInfoAccountPresent.jsx"""**

**import React, { useState } from 'react';**

**import { Container, Typography, Button, Box, TextField, MenuItem } from '@mui/material';**

**import './BasicInfoAccount.css';**

**const BasicInfoAccountPresent = () => {**

**const [formData, setFormData] = useState({**

**accountNumber: '',**

**loanAmount: '',**

**loanRequestedFor: ''**

**});**

**const handleChange = (e) => {**

**const { name, value } = e.target;**

**setFormData({**

**...formData,**

**[name]: value**

**});**

**};**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**try {**

**await fetch('http://localhost:5000/basicInfoAccount', {**

**method: 'POST',**

**headers: {**

**'Content-Type': 'application/json',**

**},**

**body: JSON.stringify(formData),**

**});**

**alert('Loan request sent successfully!');**

**} catch (error) {**

**console.error('Error:', error);**

**}**

**};**

**return (**

**<div className="apply-now-page">**

**<div className="apply-now-sidebar">**

**<div className="apply-now-sidebar-header">**

**<img src="/path/to/logo.png" alt="Agriculture Loans" className="apply-now-logo" />**

**<Typography variant="h6">कृषि ऋण</Typography>**

**<Typography variant="subtitle1">Agriculture Loans</Typography>**

**</div>**

**<div className="apply-now-sidebar-content-basicinfo">**

**<Typography variant="h6">प्रारंभिक विवरण</Typography>**

**<Typography variant="subtitle1">Preliminary Details</Typography>**

**</div>**

**</div>**

**<div className="apply-now-main-content">**

**<Container>**

**<Typography variant="h4" gutterBottom>कृषि ऋण | Agriculture Loan</Typography>**

**<Container className="apply-now-account-details">**

**<div className="apply-now-account-details-container">**

**<Typography variant="h6" gutterBottom>बैंक ऑफ बड़ौदा खाते का विवरण | Bank of AMJ account Details</Typography>**

**<form onSubmit={handleSubmit}>**

**<TextField**

**fullWidth**

**name="accountNumber"**

**label="खाता संख्या | Account Number"**

**margin="normal"**

**variant="outlined"**

**required**

**onChange={handleChange}**

**/>**

**<TextField**

**fullWidth**

**name="loanAmount"**

**label="आवश्यक ऋण राशि | The Loan Amount Required"**

**margin="normal"**

**variant="outlined"**

**required**

**onChange={handleChange}**

**/>**

**<TextField**

**fullWidth**

**name="loanRequestedFor"**

**select**

**label="ऋण का अनुरोध किया गया | Loan Requested For"**

**margin="normal"**

**variant="outlined"**

**required**

**onChange={handleChange}**

**>**

**<MenuItem value="Financing Agriclinics & Agribusiness">Financing Agriclinics & Agribusiness</MenuItem>**

**<MenuItem value="Advances to Food Agro Based Units">Advances to Food Agro Based Units</MenuItem>**

**<MenuItem value="Animal Husbandry and Fisheries Kisan Credit Card Scheme">Animal Husbandry and Fisheries Kisan Credit Card Scheme</MenuItem>**

**<MenuItem value="Kisan Credit Card (KCC)">Kisan Credit Card (KCC)</MenuItem>**

**<MenuItem value="Kisan Tatkal Loan Scheme">Kisan Tatkal Loan Scheme</MenuItem>**

**<MenuItem value="Financing Agencies Providing Custom Services To Farmers">Financing Agencies Providing Custom Services To Farmers</MenuItem>**

**<MenuItem value="Financing Development of Dairy, Poultry, Fishery, etc.">Financing Development of Dairy, Poultry, Fishery, etc.</MenuItem>**

**<MenuItem value="Financing Tractors and Heavy Agriculture Machinery">Financing Tractors and Heavy Agriculture Machinery</MenuItem>**

**<MenuItem value="Scheme for Purchase of Agriculture Land">Scheme for Purchase of Agriculture Land</MenuItem>**

**</TextField>**

**<Box mt={2}>**

**<Button type="submit" variant="contained" color="primary" fullWidth className="apply-now-submit-button">**

**प्रस्तुत करें | SUBMIT**

**</Button>**

**</Box>**

**</form>**

**</div>**

**</Container>**

**</Container>**

**</div>**

**<div className="apply-now-right-sidebar">**

**<img src="/path/to/bank-logo.png" alt="Bank of Baroda" className="apply-now-bank-logo" />**

**<ul className="apply-now-sidebar-list">**

**<li className="apply-now-sidebar-item">Agri Loan Features</li>**

**<li className="apply-now-sidebar-item">Loan Eligibility Calculator</li>**

**<li className="apply-now-sidebar-item">EMI Calculator</li>**

**<li className="apply-now-sidebar-item">FAQs</li>**

**<li className="apply-now-sidebar-item">User Guide Video</li>**

**<li className="apply-now-sidebar-item">Toll Free No. 18002581111</li>**

**</ul>**

**</div>**

**</div>**

**);**

**};**

**export default BasicInfoAccountPresent;**

**...**

**"""BasicInformationPage.css"""**

**.basicinfo{**

**width: 250px;**

**background-color: #f05737;**

**padding: 20px;**

**color: white;**

**height:800px;**

**}**

**.contentcontaininformation{**

**display: flex;**

**flex-direction: column;**

**justify-content: center;**

**align-items: center;**

**width:700px;**

**margin-left:300px;**

**margin-top:-750px;**

**}**

**.right-sidebarbasicinfo{**

**position: fixed;**

**top: 0;**

**right: 0;**

**width: 300px;**

**background-color: white;**

**padding: 20px;**

**border-left: 1px solid #ddd;**

**box-shadow: -2px 0 5px rgba(0, 0, 0, 0.1);**

**height:100%;**

**color: #f05737;**

**font-weight: 200;**

**}**

**...**

**"""BasicInformationPage.jsx"""**

**import React, { useState } from 'react';**

**import { Container, Typography, TextField, Button, Box, Grid } from '@mui/material';**

**import './BasicInformationPage.css';**

**const BasicInformationPage = () => {**

**const [formData, setFormData] = useState({**

**fullName: '',**

**fatherOrHusbandName: '',**

**gender: '',**

**occupation: '',**

**income: '',**

**kycDocumentNumber: '',**

**email: '',**

**mobile: '',**

**permanentAddress: '',**

**addressDetails: '',**

**communicationAddress: '',**

**pinCode: ''**

**});**

**const handleChange = (e) => {**

**const { name, value } = e.target;**

**setFormData({**

**...formData,**

**[name]: value**

**});**

**};**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**try {**

**await fetch('http://localhost:5000/basicInformation', {**

**method: 'POST',**

**headers: {**

**'Content-Type': 'application/json',**

**},**

**body: JSON.stringify(formData),**

**});**

**alert('Data saved successfully!');**

**} catch (error) {**

**console.error('Error:', error);**

**}**

**};**

**return (**

**<div className="basic-information-page">**

**<div className="basicinformation-final">**

**<div className='basicinfo'>**

**<div className="basicinfoheader">**

**<Typography variant="h6">कृषि ऋण</Typography>**

**<Typography variant="subtitle1">Agriculture Loans</Typography>**

**<Typography variant="h6">आधारभूत जानकारी</Typography>**

**<Typography variant="subtitle1">Basic Details</Typography>**

**<Typography variant="h6">ऋण जानकारी</Typography>**

**<Typography variant="subtitle1">Loan Details</Typography>**

**<Typography variant="h6">Submit</Typography>**

**</div>**

**</div>**

**</div>**

**<div className="contentinfofinalinformation">**

**<div className="contentcontaininformation">**

**<Container>**

**<Typography variant="h4" gutterBottom>आधारभूत जानकारी | Basic Information</Typography>**

**<form onSubmit={handleSubmit}>**

**<Grid container spacing={2}>**

**<Grid item xs={12} sm={6}>**

**<TextField name="fullName" label="Applicant's Full Name" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="fatherOrHusbandName" label="Father's/Husband's Name" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="gender" label="Gender" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="occupation" label="Occupation Type" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="income" label="Gross Annual Income" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="kycDocumentNumber" label="KYC Document Number" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="email" label="Email ID" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="mobile" label="Mobile No." fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="permanentAddress" label="Permanent Address" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="addressDetails" label="Address Details" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="communicationAddress" label="Communication Address" fullWidth required onChange={handleChange} />**

**</Grid>**

**<Grid item xs={12} sm={6}>**

**<TextField name="pinCode" label="Pin Code" fullWidth required onChange={handleChange} />**

**</Grid>**

**</Grid>**

**<Box mt={4}>**

**<Button type="button" variant="outlined" color="primary" className="SaveButtonbasic">SAVE AND CONTINUE LATER</Button>**

**<Button type="submit" variant="contained" color="primary" className="ContainedButtonbasic">CONTINUE</Button>**

**</Box>**

**</form>**

**</Container>**

**</div>**

**</div>**

**<div className="right-sidebarbasicinfo">**

**<img src="/path/to/bank-logo.png" alt="Bank of AMJ" />**

**<ul>**

**<li>Agri Loan Features</li>**

**<li>Loan Eligibility Calculator</li>**

**<li>EMI Calculator</li>**

**<li>FAQs</li>**

**<li>User Guide Video</li>**

**<li>Toll Free No. 18002581111</li>**

**</ul>**

**</div>**

**</div>**

**);**

**};**

**export default BasicInformationPage;**

**.....**

**"""CardDetails.css"""**

**.Carddetailscontainer{**

**box-shadow:0 4px 2px rgb(155, 130, 238);**

**width: 1000px;**

**padding: 100px;**

**display: flex;**

**align-items: center;**

**margin-left:255px;**

**}**

**h1{**

**font-size: 40px;**

**display: flex;**

**align-items: center;**

**padding-left: 55px;**

**padding-top: 55px;**

**}**

**/\* CardDetails.css \*/**

**.cardiApplyOnlineContainer {**

**display: flex;**

**align-items: flex-start; /\* Align items to the top \*/**

**justify-content: space-between;**

**padding: 20px;**

**background-color: #f9f9f9;**

**border-radius: 8px;**

**margin-bottom: 20px;**

**}**

**.Carousel {**

**flex: 3; /\* Make the carousel bigger \*/**

**margin-right: 20px;**

**}**

**.applyOnlineSection {**

**flex: 1; /\* Make the Apply Online section smaller \*/**

**text-align: center;**

**}**

**.cardiApplyOnlinepara {**

**margin-bottom: 10px;**

**font-size: 16px; /\* Slightly smaller font size \*/**

**box-shadow:0 4px 2px rgb(155, 130, 238);**

**}**

**.cardiApplyOnlinebutton {**

**padding: 10px 15px; /\* Slightly smaller padding \*/**

**background-color: #ff6600;**

**color: #fff;**

**border: none;**

**border-radius: 5px;**

**cursor: pointer;**

**}**

**.cardiApplyOnlinebutton:hover {**

**background-color: #e65c00;**

**}**

**.Assistancemaincontainer {**

**margin-top: 40px;**

**}**

**.Assistancephonenumnumbers {**

**display: flex;**

**justify-content: space-around;**

**}**

**.Assistancephonenumcontent {**

**text-align: center;**

**margin: 20px;**

**}**

**.Assistancephonenumcontent img {**

**width: 50px;**

**height: 50px;**

**margin-bottom: 10px;**

**}**

**/\***

**.Assistancephonenumnumbers**

**{**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**}**

**.Assistancephonenumcontent**

**{**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**flex-direction: row;**

**}**

**.Assistancemaincontainer**

**{**

**display: flex;**

**}\*/**

**.Assistancemaincontainer {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**background-color: #f8f9fa;**

**padding: 20px;**

**text-align: center;**

**}**

**.Assistancemaincontainer h2 {**

**font-size: 24px;**

**color: #666666;**

**margin-bottom: 20px;**

**}**

**.Assistancephonenumnumbers {**

**display: flex;**

**justify-content: center;**

**gap: 40px;**

**}**

**.Assistancephonenumcontent {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**background-color: white;**

**padding: 20px;**

**border-radius: 10px;**

**box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);**

**width: 200px;**

**}**

**.Assistancephonenumcontent img {**

**width: 30px;**

**height: 30px;**

**margin-bottom: 10px;**

**}**

**.Assistancephonenumcontent p {**

**margin: 5px 0;**

**font-size: 14px;**

**color: #333333;**

**}**

**.Assistancephonenumcontent.numbers p:nth-child(3),**

**.Assistancephonenumcontent.numbers p:nth-child(4) {**

**font-size: 18px;**

**font-weight: bold;**

**color: #e94e1b;**

**}**

**.Assistancephonenumcontent.tollfree p:nth-child(5) {**

**font-size: 18px;**

**font-weight: bold;**

**color: #e94e1b;**

**}**

**.Assistancephonenumcontent.asist p:nth-child(2),**

**.Assistancephonenumcontent.help p:nth-child(2) {**

**font-size: 18px;**

**font-weight: bold;**

**color: #1b72e9;**

**}**

**.Assistancephonenumcontent.asist p:nth-child(3),**

**.Assistancephonenumcontent.help p:nth-child(3) {**

**font-size: 18px;**

**font-weight: bold;**

**color: #1b72e9;**

**}**

**...**

**===================**

**============================================================================================================================================================================================**

**"""CardDetails.jsx"""**

**import React from 'react';**

**import { useParams, Link, Outlet } from 'react-router-dom';**

**import './CardDetails.css';**

**import location from './images/helpimages/location.png';**

**import locationaccount from './images/helpimages/locationaccount.png';**

**import phone from './images/helpimages/phone.png';**

**import img1 from "./images/carousel/image1.png";**

**import img2 from "./images/carousel/image2.png";**

**import img3 from "./images/carousel/image3.png";**

**import { NavLink } from 'react-router-dom';**

**const cardDetails = {**

**1: {**

**title: 'Advances to Food Agro Based Units Under Agriculture',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'New, existing Food and Agro based processing units can avail this facility',**

**'Individual, Proprietorship, Partnership concerns, Private, Public Limited companies and Limited Liability Partnership Concerns.',**

**'50% concession towards unified Processing upfront and Documentation Charges',**

**'Collection Charges on collection of cheques: 100% waiver of the Collection charges on collection of cheques issued in favour of the Borrower by Food Corporation of India, Cotton Corporation of India.',**

**'Up to Rs. 100 crores with our bank(however, the aggregate sanctioned limit per borrower from the Banking system not to exceed Rs. 100 crores)',**

**'Margin:',**

**'Stocks & Book Debts: 25 %',**

**'Plant & Machinery: 25%',**

**'Land & Building: 30%',**

**'ROI: Depending on Hard Security Coverage and Internal Credit Rating'**

**]**

**},**

**2: {**

**title: 'Animal Husbandry and Fisheries Kisan Credit Card (BAHFKCC) Scheme : Benefits',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'Nil Processing Charges for aggregate loan upto Rs 3.00 Lakhs',**

**'Nil Inspection Charges for aggregate loan up to Rs 3.00 Lakhs',**

**'Simple interest is applied till due date of repayment',**

**'Credit Balance will attract interest as in saving bank rate',**

**'No penal interest is charged up to limit of Rs. 25000.00',**

**'Free Personal Accidental Insurance of Rs. 50000.00',**

**'Loans up to Rs. 2.00 Lakhs will be charged @ 7%p.a, subject to Govt. of India providing Interest Subvention to Bank on such advances.',**

**'Prompt Repayment Incentive of 3% per annum for loans up to Rs 2.00 lakh for borrowers with timely repayment. However, the interest subvention and prompt repayment incentive benefit on working capital (i.e Crop Loan+ Working capital loan for animal husbandry and fisheries) will be available only on an overall limit of Rs 3.00 lakh per annum and subject to maximum limit of Rs 2.00 lakh per farmer involved in activities only related to animal husbandry and /or fisheries.',**

**'Credit Guarantee for loans upto Rs 10.00 lakhs is available under Credit Guarantee Fund for Micro Units (CGFMU)'**

**]**

**},**

**3: {**

**title: 'Kisan Credit Card (KCC)',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'Digital BKCC Renewal facility',**

**'All farmers including tenants and share croppers are eligible for loans',**

**'Minimum loan amount is Rs. 5,000/-',**

**'Credit facility is provided for Production Credit and Investment Credit',**

**'Loans up to Rs. 3.00 Lakhs will be charged @ 7%p.a, subject to Govt. of India providing Interest Subvention to Bank on such advances',**

**'Interest at SB rate on credit balance in account',**

**'Coverage available under Pradhan Mantri Fasal Bima Yojana',**

**'Concession available in ROI for existing regular borrowers',**

**'Limit is valid for 5 years',**

**'Eligible for free insurance cover of Rs. 50,000/ under Personal Accident Insurance Scheme',**

**'Nil Processing Charges for aggregate loan upto Rs 3.00 Lakhs',**

**'Nil Inspection Charges for aggregate loan upto Rs 3.00 Lakhs'**

**]**

**},**

**8: {**

**title: 'Kisan Tatkal Loan Scheme : Benefits',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'A scheme to meet the emergent funds requirements for agriculture and domestic purposes during off season.',**

**'Maximum Loan upto Rs 1.00 Lakhs',**

**'Individual Farmers/Joint borrowers who are existing BKCC holder of our bank for at least 2 years, with satisfactory repayment track record of all advance accounts.'**

**]**

**},**

**4: {**

**title: 'Financing Agriclinics & Agribusiness Centres by Agriculture Graduates',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'Provides financial aid for setting up of Agriclinics and Agribusiness Centers by Agricultural Graduates and other technically qualified entrepreneurs',**

**'Maximum Loan of Rs 25.Lakhs for Project by individual is allowed',**

**'Maximum Loan of Rs.20 Lakhs per trained person and overall ceiling of Rs.100 Lakhs for Project by group of individuals is allowed',**

**'No collateral security for Loan upto Rs 5 Lakhs',**

**'Nil margin for Loan upto Rs 5 Lakhs',**

**'Can avail subsidy through NABARD'**

**]**

**},**

**5: {**

**title: 'Financing Development of Dairy, Poultry, Fishery, etc.',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'Processing charges Nil for Aggregate Loan upto Rs 3.00 Lakhs',**

**'Inspection charges Nil for Aggregate Loan upto Rs 3.00 Lakh',**

**'Margin Nil for loans upto Rs 1.00 Lakhs',**

**'Meets the capital expenditure and working capital requirements of units engaged Agri Allied activities'**

**]**

**},**

**9: {**

**title: 'Financing Tractors and Heavy Agriculture Machinery',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'Processing charges Nil for Aggregate Loan upto Rs 3.00 Lakhs',**

**'Inspection charges Nil for Aggregate Loan upto Rs 3.00 Lakh',**

**'Low Land Requirement.',**

**'Low Margin Requirement',**

**'Repayment as per the income flow of borrower'**

**]**

**},**

**6: {**

**title: 'Pradhan Mantri Kisan Urja Suraksha Evam Uthaan Mahabhiyaan (PM-KUSUM Scheme)',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'Small renewable energy project of 2 MW',**

**'Installation of standalone solar pumps and',**

**'Solarization of existing grid connected agriculture pumps.',**

**'Inspection Charges:',**

**'Agriculture exposure upto Rs. 3.00 Lakhs- Nil',**

**'Above Rs 3 lakhs to Rs 10 Lakhs - Rs 250',**

**'Above Rs 10 Lakhs to Rs 1 Crore - Rs 1000',**

**'Above Rs 1 Crore - Rs 5000'**

**]**

**},**

**7: {**

**title: 'Scheme for Purchase of Agriculture Land',**

**imgSrc: './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png',**

**content: [**

**'Providing term finance to eligible farmers for purchase, development and cultivation of agricultural as well as fallow and Waste Lands.',**

**'Quantum of loan will depend on the area of the land to be purchased and its valuation and also development cost',**

**'Helps Small and Marginal farmers for buying more land'**

**]**

**}**

**};**

**function CardDetails() {**

**const { id } = useParams();**

**const card = cardDetails[id];**

**if (!card) {**

**return <div>Card not found</div>;**

**}**

**return (**

**<div>**

**<h1>{card.title}</h1>**

**<div className='cardiApplyOnlineContainer'>**

**<div className="Carousel">**

**<div id="carouselExample" className="carousel slide" data-bs-ride="carousel">**

**<div className="carousel-inner">**

**<div className="carousel-item active">**

**<img src={img1} className="d-block w-100" alt="..." />**

**</div>**

**<div className="carousel-item">**

**<img src={img2} className="d-block w-100" alt="..." />**

**</div>**

**<div className="carousel-item">**

**<img src={img3} className="d-block w-100" alt="..." />**

**</div>**

**</div>**

**<button className="carousel-control-prev" type="button" data-bs-target="#carouselExample" data-bs-slide="prev">**

**<span className="carousel-control-prev-icon" aria-hidden="true"></span>**

**<span className="visually-hidden">Previous</span>**

**</button>**

**<button className="carousel-control-next" type="button" data-bs-target="#carouselExample" data-bs-slide="next">**

**<span className="carousel-control-next-icon" aria-hidden="true"></span>**

**<span className="visually-hidden">Next</span>**

**</button>**

**</div>**

**</div>**

**<div className='applyOnlineSection'>**

**<p className='cardiApplyOnlinepara'>A package of abundances and value for your food business.</p>**

**<NavLink to="/apply-now" className='cardiApplyOnlinebutton'>Apply Online</NavLink>**

**</div>**

**</div>**

**<div className='Carddetailscontainer'>**

**<ul className='Carddetailslist'>**

**{card.content && card.content.map((item, index) => (**

**<li key={index}>{item}</li>**

**))}**

**</ul>**

**</div>**

**<div className='Assistancemaincontainer'>**

**<h2>Need Assistance?</h2>**

**<div className='Assistancephonenumnumbers'>**

**<div className='Assistancephonenumcontent numbers'>**

**<img src={phone} alt="phone"/>**

**<p>Toll Free Number (Domestic):</p>**

**<p>1800 5700</p>**

**<p>1800 5000</p>**

**</div>**

**<div className='Assistancephonenumcontent tollfree'>**

**<img src={phone} alt="phone"/>**

**<p>Toll Free Number</p>**

**<p>For Domestic Customers</p>**

**<p>Calling From Abroad (24X7):</p>**

**<p>+91 79-66296009</p>**

**</div>**

**<div className='Assistancephonenumcontent asist'>**

**<img src={location} alt="location"/>**

**<p>Locate</p>**

**<p>Our Branch</p>**

**</div>**

**<div className='Assistancephonenumcontent help'>**

**<img src={locationaccount} alt="location account"/>**

**<p>Find</p>**

**<p>Our BC Agent</p>**

**</div>**

**</div>**

**</div>**

**</div>**

**);**

**}**

**export default CardDetails;**

**.....**

**"""Footer.css"""**

**.footer-container {**

**padding: 40px 20px;**

**background-color: #fff;**

**color: #333;**

**}**

**.footer-columns {**

**display: flex;**

**flex-wrap: wrap;**

**justify-content: space-between;**

**}**

**.footer-column {**

**flex: 1;**

**margin: 10px;**

**min-width: 150px;**

**}**

**.footer-column h6 {**

**margin-bottom: 15px;**

**font-weight: bold;**

**}**

**.footer-column ul {**

**list-style: none;**

**padding: 0;**

**}**

**.footer-column li {**

**margin-bottom: 10px;**

**}**

**.footer-column a {**

**color: #333;**

**text-decoration: none;**

**}**

**.footer-column a:hover {**

**text-decoration: underline;**

**}**

**.footer-bottom {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**margin-top: 40px;**

**}**

**.footer-social {**

**display: flex;**

**justify-content: center;**

**margin-bottom: 20px;**

**}**

**.social-links a {**

**margin: 0 10px;**

**font-size: 24px;**

**color: #333;**

**text-decoration: none;**

**}**

**.social-links a:hover {**

**color: #0056b3;**

**}**

**.footer-bottom-links {**

**display: flex;**

**justify-content: center;**

**flex-wrap: wrap;**

**margin-bottom: 20px;**

**}**

**.footer-bottom-links a {**

**margin: 0 10px;**

**color: #333;**

**text-decoration: none;**

**}**

**.footer-bottom-links a:hover {**

**text-decoration: underline;**

**}**

**.footer-copyright {**

**text-align: center;**

**color:#0056b3;**

**}**

**@media (max-width: 768px) {**

**.footer-columns {**

**flex-direction: column;**

**}**

**.footer-column {**

**margin: 20px 0;**

**}**

**}**

**....**

**"""Footer.jsx"""import React from 'react';**

**import './Footer.css';**

**const Footer = () => {**

**return (**

**<footer className="footer-container">**

**<div className="footer-columns">**

**<div className="footer-column">**

**<h6>About Us</h6>**

**<ul>**

**<li><a href="#">Overview</a></li>**

**<li><a href="#">History</a></li>**

**<li><a href="#">Initiative</a></li>**

**<li><a href="#">Compliance</a></li>**

**<li><a href="#">Board of Directors</a></li>**

**<li><a href="#">Amalgamation</a></li>**

**<li><a href="#">Offices & Branches</a></li>**

**<li><a href="#">Social Commitment</a></li>**

**<li><a href="#">Awards</a></li>**

**<li><a href="#">Green Initiatives by Bank</a></li>**

**<li><a href="#">Subsidiaries and Joint Ventures</a></li>**

**<li><a href="#">AMJ Alok Chandra Bravery Award</a></li>**

**<li><a href="#">Human Resources</a></li>**

**<li><a href="#">Careers</a></li>**

**</ul>**

**</div>**

**<div className="footer-column">**

**<h6>Shareholder’s Corner</h6>**

**<ul>**

**<li><a href="#">Overview</a></li>**

**<li><a href="#">Shareholders Meeting AGM/EGM</a></li>**

**<li><a href="#">Stock Exchanges Notifications under SEBI (LODR) Regulations, 2015</a></li>**

**<li><a href="#">Contact Points - Shares & Debentures</a></li>**

**<li><a href="#">Share Transfer/Transmission Services</a></li>**

**<li><a href="#">Unpaid Dividend</a></li>**

**<li><a href="#">Unclaimed Interest/Principal on Bank of AMJ Bonds</a></li>**

**<li><a href="#">Sustainability Disclosures</a></li>**

**<li><a href="#">Annual Reports</a></li>**

**<li><a href="#">Financial Reports</a></li>**

**<li><a href="#">Presentation made to Analyst</a></li>**

**<li><a href="#">Shareholding Pattern</a></li>**

**<li><a href="#">Policies Documents</a></li>**

**<li><a href="#">Disclosures under Basel III</a></li>**

**</ul>**

**</div>**

**<div className="footer-column">**

**<h6>Customer Corner</h6>**

**<ul>**

**<li><a href="#">Find Important Customer Information</a></li>**

**<li><a href="#">RBI Kehta Hai</a></li>**

**<li><a href="#">Doorstep Banking</a></li>**

**<li><a href="#">Customer Care</a></li>**

**<li><a href="#">Online Complaints/Track Status/ Feedback</a></li>**

**<li><a href="#">Holidays in India</a></li>**

**<li><a href="#">Central Pension Processing Centre (CPPC)</a></li>**

**<li><a href="#">AMJ Academy</a></li>**

**<li><a href="#">Right to Information Act</a></li>**

**<li><a href="#">Periodic Updation of KYC (Re-KYC)</a></li>**

**<li><a href="#">Online OTS</a></li>**

**<li><a href="#">Central Know Your Customer (CKYC)</a></li>**

**</ul>**

**</div>**

**<div className="footer-column">**

**<h6>Media Centre</h6>**

**<ul>**

**<li><a href="#">Events</a></li>**

**<li><a href="#">News Coverage</a></li>**

**<li><a href="#">Podcast</a></li>**

**<li><a href="#">Press Releases</a></li>**

**<li><a href="#">Announcements</a></li>**

**<li><a href="#">Webcast</a></li>**

**<li><a href="#">Azadi Ka Amrit Mahotsav</a></li>**

**</ul>**

**</div>**

**<div className="footer-column">**

**<h6>Resources</h6>**

**<ul>**

**<li><a href="#">Banking Mantra</a></li>**

**<li><a href="#">E-Auction</a></li>**

**<li><a href="#">Economic Scenario</a></li>**

**<li><a href="#">Interest Rate & Service Charges</a></li>**

**<li><a href="#">Tenders</a></li>**

**<li><a href="#">Download Forms</a></li>**

**<li><a href="#">FAQs</a></li>**

**<li><a href="#">शब्दनाद</a></li>**

**</ul>**

**</div>**

**<div className="footer-column">**

**<h6>Other Links</h6>**

**<ul>**

**<li><a href="#">Bharat Aadhaar Seeding Enabler</a></li>**

**<li><a href="#">UDGAM Portal</a></li>**

**<li><a href="#">Online Deceased Claim Portal</a></li>**

**<li><a href="#">Bank of AMJ Employees Pension Regulation, 1995</a></li>**

**<li><a href="#">Code of Banks commitment to Micro & Small Enterprises</a></li>**

**<li><a href="#">Code of Bank’s Commitment to Customers</a></li>**

**<li><a href="#">Central Scheme of Interest Subsidy for Education Loan</a></li>**

**<li><a href="#">List of Inoperative Accounts</a></li>**

**<li><a href="#">List of Inoperative Accounts (e-Dena)</a></li>**

**<li><a href="#">List of Inoperative Accounts (e-Vijaya)</a></li>**

**<li><a href="#">DEA FUND Claim Procedure</a></li>**

**<li><a href="#">Whistle Blower guidelines</a></li>**

**<li><a href="#">Report suspicious/fraud transaction</a></li>**

**</ul>**

**</div>**

**</div>**

**<div className="footer-bottom">**

**<div className="footer-social">**

**<h6>Connect with Us</h6>**

**<div className="social-links">**

**<a href="#"><i className="fab fa-facebook"></i></a>**

**<a href="#"><i className="fab fa-instagram"></i></a>**

**<a href="#"><i className="fab fa-linkedin"></i></a>**

**<a href="#"><i className="fab fa-youtube"></i></a>**

**<a href="#"><i className="fab fa-quora"></i></a>**

**</div>**

**</div>**

**<div className="footer-bottom-links">**

**<a href="#">Cookie Policy</a>**

**<a href="#">Disclaimer</a>**

**<a href="#">Privacy Policy</a>**

**<a href="#">Terms of Use</a>**

**</div>**

**<div className="footer-copyright">**

**<p>&copy; 2023 Bank of AMJ. All Rights Reserved.</p>**

**</div>**

**</div>**

**</footer>**

**);**

**};**

**export default Footer;**

**...**

**"""Header.css"""**

**.login-btn {**

**background-color: #ffffff;**

**color: #ff6f00;**

**border: none;**

**padding: 8px 12px;**

**margin-left: 15px;**

**cursor: pointer;**

**border-radius: 25px;**

**box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);**

**transition: background-color 0.3s ease, transform 0.3s ease;**

**}**

**.login-btn:hover {**

**background-color: #ff9900;**

**color: white;**

**transform: translateY(-2px);**

**}**

**/\* Navbar Styles \*/**

**.navbar {**

**width: 100%;**

**background-color: #ff6f00;**

**color: white;**

**display: flex;**

**justify-content: left;**

**padding: 10px 0;**

**box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);**

**}**

**.navbar-top {**

**display: flex;**

**justify-content: center;**

**padding: 10px 0;**

**border-bottom: 1px solid #e5e5e5;**

**}**

**.navbar-top span {**

**margin: 0 20px;**

**cursor: pointer;**

**transition: color 0.3s ease;**

**}**

**.navbar-top span:hover {**

**color: #ff9900;**

**}**

**.navbar-bottom {**

**display: flex;**

**justify-content: space-around;**

**padding: 10px 0;**

**}**

**/\* Dropdown Styles \*/**

**.dropdown {**

**position: relative;**

**display: inline-block;**

**}**

**.dropbtn {**

**background-color: #ff6f00;**

**color: white;**

**padding: 8px 16px;**

**font-size: 14px;**

**border: none;**

**cursor: pointer;**

**transition: background-color 0.3s ease, transform 0.3s ease;**

**border-radius: 5px;**

**}**

**.dropbtn:hover {**

**background-color: #ff9900;**

**transform: translateY(-2px);**

**}**

**.dropdown-content {**

**display: none;**

**position: absolute;**

**background-color: white;**

**color: black;**

**min-width: 200px;**

**box-shadow: 0px 8px 16px 0px rgba(0, 0, 0, 0.2);**

**z-index: 1;**

**border-radius: 8px;**

**overflow: hidden;**

**transition: all 0.3s ease;**

**}**

**.dropdown-content a {**

**color: black;**

**padding: 12px 16px;**

**text-decoration: none;**

**display: block;**

**transition: background-color 0.3s ease, padding 0.3s ease;**

**}**

**.dropdown-content a:hover {**

**background-color: #f1f1f1;**

**padding-left: 20px;**

**}**

**.dropdown:hover .dropdown-content {**

**display: block;**

**animation: fadeIn 0.3s ease;**

**}**

**/\* Header Styles \*/**

**/\* Keyframes for fade-in effect \*/**

**@keyframes fadeIn {**

**from {**

**opacity: 0;**

**transform: translateY(-10px);**

**}**

**to {**

**opacity: 1;**

**transform: translateY(0);**

**}**

**}**

**.name**

**{**

**display:flex;**

**align-items: center;**

**}**

**.nameimg{**

**height:100px;**

**width:100px;**

**}**

**.Name**

**{**

**margin-left:-50px;**

**}**

**.....**

**======**

**============================================================================================================================================================================================**

**"""Header.jsx"""**

**import React from 'react';**

**import { useNavigate } from 'react-router-dom';**

**import './Header.css'; // Ensure this CSS file exists and is properly linked**

**import logo from './images/assistance/logo.png';**

**const Header = () => {**

**const navigate = useNavigate();**

**const handleLogout = () => {**

**// Clear user session or token here (if using authentication)**

**// For example, remove a token from localStorage**

**localStorage.removeItem('authToken');**

**// Redirect to login page**

**navigate('/logindupe');**

**};**

**return (**

**<div>**

**<div className="headerpart">**

**<div className='name'>**

**<img src={logo} className='nameimg' alt="Bank of Baroda Logo" />**

**<h1 className='Name'>AMJ</h1>**

**</div>**

**</div>**

**<div className="navbar">**

**<Dropdown title="Accounts">**

**<a href="#">Savings Accounts</a>**

**<a href="#">Salary Accounts</a>**

**<a href="#">Current Account</a>**

**<a href="#">Term Deposit</a>**

**<a href="#">bob LITE Savings Account</a>**

**<a href="#">bob BRO Saving Account</a>**

**<a href="#">bob Family Savings Accounts Segment</a>**

**<a href="#">bob Advantage Saving Account</a>**

**<a href="#">bob Champ Account</a>**

**<a href="#">bob Super Savings Account</a>**

**<a href="#">bob Jeevan Suraksha Saving Account</a>**

**<a href="#">bob SB Self Help Group</a>**

**<a href="#">bob Mahila Shakti Saving Account</a>**

**<a href="#">bob Pensioners Savings Bank Account</a>**

**<a href="#">bob Platinum Saving Bank Account</a>**

**<a href="#">bob Professional Saving Bank Account</a>**

**<a href="#">bob Senior Citizen Privilege Scheme</a>**

**<a href="#">bob Bank Mitra Bachat Khata</a>**

**<a href="#">Government Savings Account (bob Govt. Bodies SB Account)</a>**

**<a href="#">View All Saving Accounts</a>**

**<a href="#">Open a Savings Account Digitally</a>**

**<a href="#">Application Form</a>**

**<a href="#">FAQs</a>**

**<a href="#">Contact Us</a>**

**<a href="#">View All Accounts</a>**

**</Dropdown>**

**<Dropdown title="Loans">**

**<a href="#">Home Loan</a>**

**<a href="#">Vehicle Loan</a>**

**<a href="#">Personal Loans</a>**

**<a href="#">AMJ Yoddha Loans for Defence Personnel</a>**

**<a href="#">Education Loan</a>**

**<a href="#">Other Loans</a>**

**<a href="#">Gold Loan</a>**

**<a href="#">Mortgage Loan</a>**

**<a href="#">Fintech</a>**

**<a href="#">AMJ Yoddha Home Loan</a>**

**<a href="#">AMJ Yoddha Auto Loan</a>**

**<a href="#">AMJ Yoddha Two-Wheeler Loan</a>**

**<a href="#">AMJ Yoddha Loan to Defence Pensioners</a>**

**<a href="#">AMJ Yoddha Education Loans</a>**

**<a href="#">JanSamarth Portal</a>**

**<a href="#">Application Form</a>**

**<a href="#">FAQs</a>**

**<a href="#">Contact Us</a>**

**<a href="#">Sachet Portal</a>**

**<a href="#">Privacy Policy</a>**

**<a href="#">Grievance Redressal</a>**

**<a href="#">List of DLAs</a>**

**<a href="#">List of LSPs</a>**

**<a href="#">List of Recovery Agents</a>**

**<a href="#">View All Loans</a>**

**</Dropdown>**

**<Dropdown title="Investments">**

**<a href="#">Govt. Deposit Schemes/Bonds</a>**

**<a href="#">Investment Products</a>**

**<a href="#">Atal Pension Yojana</a>**

**<a href="#">Sovereign Gold Bonds</a>**

**<a href="#">E-Kisan Vikas Patra Scheme, 2014</a>**

**<a href="#">Floating Rate Savings Bonds (RBI bonds)</a>**

**<a href="#">Gold Monetization Scheme</a>**

**<a href="#">Senior Citizen Savings Deposit Scheme</a>**

**<a href="#">Public Provident Fund</a>**

**<a href="#">Sukanya Samriddhi Yojana</a>**

**<a href="#">National Pension Scheme</a>**

**<a href="#">Mahila Samman Savings Certificate (MSSC), 2023</a>**

**<a href="#">Open AMJ 3 in 1 Demat & Trading Account Digitally</a>**

**<a href="#">Online Closure of CDSL Demat Account</a>**

**<a href="#">Application Form</a>**

**<a href="#">FAQs</a>**

**<a href="#">Contact Us</a>**

**<a href="#">View All Investments</a>**

**</Dropdown>**

**<Dropdown title="Digital Products">**

**<a href="#">Digital Payment</a>**

**<a href="#">Cards</a>**

**<a href="#">Instant Banking</a>**

**<a href="#">Merchant Payment Solutions</a>**

**<a href="#">bob World (Mobile Banking)</a>**

**<a href="#">bob World Internet Banking</a>**

**<a href="#">bob World UPI</a>**

**<a href="#">bobWorld Digital Rupee</a>**

**<a href="#">bob World Wave - A Wearable Contactless Payment Smart Watch</a>**

**<a href="#">bob World Merchant (POS)</a>**

**<a href="#">bob World Internet</a>**

**<a href="#">Application Form</a>**

**<a href="#">FAQs</a>**

**<a href="#">Contact Us</a>**

**<a href="#">View All Digital Products</a>**

**</Dropdown>**

**<Dropdown title="Other Services">**

**<a href="#">Miscellaneous</a>**

**<a href="#">Aadhar Seva Kendra</a>**

**<a href="#">ASBA Facility</a>**

**<a href="#">Collection Services</a>**

**<a href="#">Custom Duty Payment</a>**

**<a href="#">ECS</a>**

**<a href="#">e-stamping</a>**

**<a href="#">Fintech Alliance</a>**

**<a href="#">Get your Form 16A</a>**

**<a href="#">Government Business Schemes</a>**

**<a href="#">One Time Settlement</a>**

**<a href="#">Pension Saarthi</a>**

**<a href="#">Preventive Vigilance Initiative</a>**

**<a href="#">Settlement of Death Claim</a>**

**<a href="#">Video based Life Certificate</a>**

**<a href="#">Statewise Bank Goods and Services Tax Identification Number (GSTIN)</a>**

**<a href="#">View All Miscellaneous</a>**

**<a href="#">Application Form</a>**

**<a href="#">FAQs</a>**

**<a href="#">Contact Us</a>**

**<a href="#">View All Other Services</a>**

**</Dropdown>**

**<button className="logout-btn" onClick={handleLogout}>Logout</button>**

**</div>**

**</div>**

**);**

**};**

**const Dropdown = ({ title, children }) => {**

**return (**

**<div className="dropdown">**

**<button className="dropbtn">{title}</button>**

**<div className="dropdown-content">**

**{children}**

**</div>**

**</div>**

**);**

**};**

**export default Header;**

**....**

**"""Home.css"""**

**\* {**

**box-sizing: border-box;**

**}**

**body {**

**font-family: Arial, sans-serif;**

**margin: 0;**

**padding: 0;**

**overflow-x: hidden; /\* Prevent horizontal scroll \*/**

**}**

**body {**

**font-family: Arial, sans-serif;**

**}**

**.Waystobank{**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**}**

**.Waystobankcards {**

**display: grid;**

**grid-template-rows: repeat(2, 1fr);**

**grid-template-columns: repeat(3, 1fr);**

**gap: 2px;**

**align-items: center;**

**padding:25px;**

**width: 1000px;**

**box-shadow:0 4px 30px rgb(155, 130, 238);**

**border-radius:30px 0px 30px 30px;**

**}**

**.Waystobankh1{**

**font-size: 60px;**

**padding: 10px;**

**margin-left: -600px;**

**position: relative;**

**padding-bottom:20px;**

**color: #7b26a6;**

**margin-top:100px;**

**}**

**.searchboxandbutton**

**{**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**padding:50px;**

**}**

**.searchbutton{**

**width: 100px;**

**border-radius: 30px;**

**height: 60px;**

**}**

**.micbutton**

**{**

**border-radius: 30px;**

**width: 100px;**

**height: 60px;**

**}**

**.searchbox{**

**border-radius: 30px;**

**width: 600px;**

**padding:10px;**

**height: 60px;**

**}**

**.searchboxotherbuttonscontainer {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**padding: 60px;**

**background-color: #c5d7ff;**

**height: 250px;**

**margin:-80px;**

**}**

**.searchboxotherbuttons1,**

**.searchboxotherbuttons2 {**

**display: flex;**

**gap: 10px;**

**justify-content: center;**

**}**

**button {**

**background-color: #a06cd7;**

**color: #ffffff;**

**padding: 10px 20px;**

**border: none;**

**border-radius: 40px;**

**cursor: pointer;**

**transition: background-color 0.3s ease;**

**}**

**button:hover {**

**background-color: #c8c4e9;**

**}**

**button:active {**

**background-color: #b0a7d6;**

**}**

**.cardwithvalues**

**{**

**display: flex;**

**align-items: center;**

**justify-content: center;**

**flex-direction:column;**

**box-shadow:0 4px 2px rgb(155, 130, 238);**

**width: 370px;**

**height: 600px;**

**border-radius: 10px;**

**padding:10px;**

**background-color: rgb(247, 249, 250);**

**overflow: hidden;**

**margin: 10px;**

**margin-bottom:50px**

**}**

**.cardwithvalues1img{**

**width: 110%;**

**height: 120px;**

**border-radius: 10px 10px 0px 0px;**

**display: flex;**

**position: relative;**

**margin-top:-8px;**

**padding-bottom: 10px;**

**}**

**.cardfeatures**

**{**

**display: grid;**

**grid-template-rows: repeat(2, 1fr);**

**grid-template-columns: repeat(3, 1fr);**

**align-items: center;**

**justify-content: center;**

**padding:10px;**

**padding-top: 150px;**

**padding-left: 600px;**

**padding-right: 100px;**

**width: 1000px;**

**}**

**.cardtitle**

**{**

**font-size: 33px;**

**position: relative;**

**color: #7b26a6;**

**}**

**.cardpara**

**{**

**padding-top: 10px;**

**font-size: 18px;**

**position: relative;**

**color: #585c82;**

**}**

**.cardbutton**

**{**

**background-color: #ffffff;**

**color: #cc80ff;**

**font-size: 20px;**

**padding:10px;**

**border:solid rgb(164, 97, 247);**

**margin-top: 100px;**

**position: relative;**

**bottom: 20px;**

**}**

**.cardwithvalues:hover {**

**background-color: hsl(248, 40%, 96%);**

**box-shadow:0 4px 20px rgb(155, 130, 238);**

**transform: translateY(-5px);**

**transition: background-color 0.3s ease, transform 0.3s ease;**

**cursor: pointer;**

**}**

**.cardwithvalues:active {**

**box-shadow:0 4px 8px rgb(131, 38, 244);**

**}**

**.AgriLoansandAdvancesContainer{**

**display: flex;**

**align-items: center;**

**padding: 30px;**

**margin-bottom: -100px;**

**}**

**.AgriLoansandAdvancesTitle**

**{**

**width: 1500px;**

**padding: 100px;**

**}**

**.AgriLoansandAdvancesPara**

**{**

**margin-left:-20px;**

**padding-top:10px;**

**border-left: solid rgb(155, 130, 238);**

**text-align: left;**

**}**

**.assistancecontainer**

**{**

**box-shadow:0 4px 30px rgb(155, 130, 238);**

**border-radius:30px 0px 30px 30px;**

**display:flex;**

**align-items: center;**

**width:1000px;**

**margin-left:300px;**

**height: 1000px;**

**position: relative;**

**}**

**.assistancetext**

**{**

**display: flex;**

**flex-direction: column;**

**align-items:center;**

**}**

**.assistanceh2{**

**font-size:70px;**

**width: 600px;**

**}**

**.assistancep{**

**font-size:40px;**

**width: 300px;**

**position: relative;**

**color: #7b26a6;**

**}**

**.assistanceimage-section**

**{**

**background-image:url('C:/webdev college/projects/project/bankproject/src/components/images/assistance/bankimg.png');**

**background-size: cover;**

**background-position: center;**

**margin: auto;**

**overflow: hidden;**

**position: relative;**

**color: aliceblue;**

**padding-top:400px;**

**padding-bottom:300px;**

**border-radius: 30px 0px 0px 30px;**

**}**

**.assistancebutton**

**{**

**position: relative;**

**align-items: center;**

**bottom:0;**

**left:0;**

**filter: brightness(100%);**

**padding:10px;**

**width:200px;**

**margin-left: 95px;**

**margin-top: 95px;**

**}**

**.carousel-item {**

**transition: transform 0.5s ease-in-out;**

**}**

**.carousel-indicators [data-bs-target] {**

**background-color: #ff6600;**

**}**

**.carousel-control-prev-icon,**

**.carousel-control-next-icon {**

**filter: invert(100%);**

**.marquee-container {**

**width: 100%;**

**overflow: hidden;**

**background-color: #f2f2f2;**

**}**

**.marquee {**

**display: inline-block;**

**white-space: nowrap;**

**padding: 10px 0;**

**animation: marquee 10s linear infinite;**

**font-size: 1rem;**

**color: #ff6600;**

**position: relative;**

**margin-top:10px;**

**margin-left: 120px;**

**}**

**@keyframes marquee {**

**0% {**

**transform: translateX(100%);**

**}**

**100% {**

**transform: translateX(-100%);**

**}**

**}}**

**.marquee-container {**

**overflow: hidden;**

**white-space: nowrap;**

**}**

**.marquee {**

**display: inline-block;**

**animation: marquee 15s linear infinite;**

**}**

**@keyframes marquee {**

**from {**

**transform: translateX(100%);**

**}**

**to {**

**transform: translateX(-100%);**

**}**

**}**

**.logout-btn {**

**width: 70px;**

**background-color: #ffffff;**

**color: #ff6f00;**

**display: flex;**

**align-items: center;**

**}**

**....**

**"""Home.jsx"""**

**import React from 'react';**

**import 'bootstrap/dist/css/bootstrap.min.css';**

**import 'bootstrap/dist/js/bootstrap.bundle.min.js';**

**import './Home.css';**

**import { Link } from 'react-router-dom';**

**import digiaccounts from './images/waystobank/DigitalAccounts.png';**

**import bobworld from './images/waystobank/bobWorld.png';**

**import Applock from './images/waystobank/AppLockerOnline.png';**

**import digiHome from './images/waystobank/DigitalHomeLoans.png';**

**import intbanking from './images/waystobank/internetBanking.png';**

**import Whatsapp from './images/waystobank/WhatsAppBanking.png';**

**import barodaanimal from './images/cardimages/BarodaAnimalHusbandryandFisheries.png';**

**import barodakisan from './images/cardimages/BarodaKisanCreditCard.png';**

**import barodakisantatkal from './images/cardimages/BarodaKisanTatkal.png';**

**import financingagriclinic from './images/cardimages/FinancingAgriclinics&AgribusinessCentresbyAgricultureGraduates.png';**

**import financingdevelopment from './images/cardimages/FinancingDevelopmentofDairyPoultryandFishery.png';**

**import fianancingtractor from './images/cardimages/Financingtractorsandheavyagriculture.png';**

**import pradhanmantri from './images/cardimages/PradhanMantriKisanUrjaSuraksha.png';**

**import scemeforpurchaseof from './images/cardimages/Schemeforpurchaseofagricultureland.png';**

**import foodagrobased from './images/cardimages/toFoodAgroBasedUnitsUnderAgriculture.png';**

**import appstore from './images/assistance/appstore.png';**

**import googleplay from './images/assistance/googleplay.png';**

**import img1 from './images/carousel/image1.png';**

**import img2 from './images/carousel/image2.png';**

**import img3 from './images/carousel/image3.png';**

**import bankimg from './images/assistance/bankimg.png';**

**function Home() {**

**return (**

**<>**

**<div className="Carousel">**

**<div id="carouselExampleInterval" class="carousel slide" data-bs-ride="carousel">**

**<div class="carousel-inner">**

**<div class="carousel-item active" data-bs-interval="10000">**

**<img src={img1} class="d-block w-100" alt="..."/>**

**</div>**

**<div class="carousel-item" data-bs-interval="2000">**

**<img src={img2} class="d-block w-100" alt="..."/>**

**</div>**

**<div class="carousel-item">**

**<img src={img3} class="d-block w-100" alt="..."/>**

**</div>**

**</div>**

**<button class="carousel-control-prev" type="button" data-bs-target="#carouselExampleInterval" data-bs-slide="prev">**

**<span class="carousel-control-prev-icon" aria-hidden="true"></span>**

**<span class="visually-hidden">Previous</span>**

**</button>**

**<button class="carousel-control-next" type="button" data-bs-target="#carouselExampleInterval" data-bs-slide="next">**

**<span class="carousel-control-next-icon" aria-hidden="true"></span>**

**<span class="visually-hidden">Next</span>**

**</button>**

**</div>**

**</div>**

**<div className="Marquee">**

**<div className="marquee-container">**

**<div className="marquee">**

**Closure of Trading Window – Q1 (FY2024-25) Financial Results • bob World Digital Rupee application services will be unavailable for 15 minutes from 09.00 PM to 09.15 PM on Friday**

**</div>**

**</div>**

**</div>**

**<div className="Waystobank">**

**<h2 className="Waystobankh1">Ways to Bank</h2>**

**<div className="Waystobankcards">**

**<div className="Waystobankcard">**

**<img src={digiaccounts} alt="Digital accounts" />**

**<h3>Digital accounts</h3>**

**</div>**

**<div className="Waystobankcard">**

**<img src={digiHome} alt="Digital Home Loans" />**

**<h3>Digital Home Loans</h3>**

**</div>**

**<div className="Waystobankcard">**

**<img src={bobworld} alt="bob World(Mobile Banking)" />**

**<h3>bob World(Mobile Banking)</h3>**

**</div>**

**<div className="Waystobankcard">**

**<img src={intbanking} alt="Internet Banking" />**

**<h3>Internet Banking</h3>**

**</div>**

**<div className="Waystobankcard">**

**<img src={Whatsapp} alt="WhatsApp Banking" />**

**<h3>WhatsApp Banking</h3>**

**</div>**

**<div className="Waystobankcard">**

**<img src={Applock} alt="Apply Locker Online" />**

**<h3>Apply Locker Online</h3>**

**</div>**

**</div>**

**</div>**

**<div className="AgriLoansandAdvances">**

**<div className="AgriLoansandAdvancesContainer">**

**<div className="AgriLoansandAdvancesTitle">**

**<h2 className="AgriLoansandAdvancesTitleh2">Agriculture Loans & Advances</h2>**

**</div>**

**<div className="AgriLoansandAdvancesPara">**

**<p className="AgriLoansandAdvancesp">**

**List of bank's authorised venders for Overdue Collection Activities the Bank is a pioneer in introducing various customer centric initiatives in the Indian Banking industry. It is a holistic, ecosystem driven platform that helps in digitizing the agronomic journey by offering new-age solutions related to agri-inputs and implements, financing, and advisory among other functionalities at the user’s fingertips.**

**</p>**

**</div>**

**</div>**

**</div>**

**<div className="Box"> </div>**

**<div className="search">**

**<div className="searchboxandbutton">**

**<input type="text" className="searchbox" placeholder="Looking for something specific?" />**

**<button className="searchbutton">Search</button>**

**<button className="micbutton">Mic</button>**

**</div>**

**<div className="searchboxotherbuttonscontainer">**

**<div className="searchboxotherbuttons1">**

**<button className="buttonsearchshareholderscorner">Shareholder's Corner</button>**

**<button className="buttonsearchbobworldstartup">bob World Start-Up</button>**

**<button className="buttonsearchpositivepaysystem">Positive Pay System</button>**

**<button className="buttonsearchrighttoinformationact">Right to Information Act</button>**

**<button className="buttonsearchdoorstepbanking">Doorstep Banking</button>**

**<button className="buttonsearchforlawenforcementagencies">For Law Enforcement Agencies</button>**

**</div>**

**<br />**

**<div className="searchboxotherbuttons2">**

**<button className="buttonsearchchiefvigilanceofficer">Chief Vigilance Officer</button>**

**<button className="buttonsearchlifeinsurance">Life Insurance</button>**

**<button className="buttonsearchonlinedeceasedclaimportal">Online Deceased Claim Portal</button>**

**<button className="buttonsearchudgamportal">UDGAM Portal</button>**

**<button className="buttonsearchonlineots">Online OTS</button>**

**</div>**

**</div>**

**</div>**

**<div className="cards">**

**<div className="cardfeatures">**

**<div className="cardwithvalues">**

**<img src={foodagrobased} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle">Advances to Food Agro Based Units Under Agriculture</h2>**

**<p className="cardpara">**

**Financing the new projects, working capital and non fund based facilities to the units engaged in activities of oil mill/solvent extraction, agro/food**

**</p>**

**<Link to="/details/1">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={barodaanimal} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle">Animal Husbandry and Fisheries Kisan Credit Card</h2>**

**<p className="cardpara">**

**Provide adequate and timely credit support under a single window to the farmers for activities related to Animal Husbandry and Fisheries.**

**</p>**

**<Link to="/details/2">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={barodakisan} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle"> Kisan Credit Card (BKCC)</h2>**

**<p className="cardpara">**

**Aims at providing adequate and timely credit support from the banking system under a single window to the farmers for their cultivation & other needs.**

**</p>**

**<Link to="/details/3">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={financingagriclinic} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle">Financing Agriclinics & Agribusiness Centres by Agriculture Graduates</h2>**

**<p className="cardpara">**

**A financial aid for setting up of Agriclinics and Agribusiness centers in order to supplement efforts of public extension**

**</p>**

**<Link to="/details/4">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={financingdevelopment} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle">Financing Development of Dairy, Poultry, Fishery, etc.</h2>**

**<p className="cardpara">**

**To fulfil capital and working capital requirement of units engaged in dairy, piggery, poultry, equipment/machinery/transport vehicle for purchase of feed and to meet other expenditure**

**</p>**

**<Link to="/details/5">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={pradhanmantri} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle">Pradhan Mantri Kisan Urja Suraksha Evam Uthaan Mahabhiyaan (PM-KUSUM Scheme)</h2>**

**<p className="cardpara">**

**PM-KUSUM scheme is a financing installation of small renewable energy project of 2 MW. Installation of standalone solar pumps and solarization of existing grid connected agriculture pumps.**

**</p>**

**<Link to="/details/6">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={scemeforpurchaseof} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle">Scheme for Purchase of Agriculture Land</h2>**

**<p className="cardpara">A financial aid for farmers to purchase</p>**

**<Link to="/details/7">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={barodakisantatkal} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle"> Kisan Tatkal Loan Scheme</h2>**

**<p className="cardpara">**

**A scheme to meet the emergent funds requirements for agriculture and domestic purposes during off season. Farmers who are existing Kisan Card Holders are eligible.**

**</p>**

**<Link to="/details/8">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**<div className="cardwithvalues">**

**<img src={fianancingtractor} className="cardwithvalues1img" alt="..." />**

**<h2 className="cardtitle">Financing Tractors and Heavy Agriculture Machinery</h2>**

**<p className="cardpara">**

**Helps with the purchase of new tractor, tractor drawn implements and power tiller and other agriculture machines etc.**

**</p>**

**<Link to="/details/9">**

**<button className="cardbutton">Know more 🔜</button>**

**</Link>**

**</div>**

**</div>**

**</div>**

**<div className="assistance">**

**<div className="assistancecontainer">**

**<div className="assistanceimage-section">**

**<h2 className="assistancetitle">Gain access to many exciting offers & trending deals on various bob World Debit & Credit Cards.</h2>**

**<button className="assistancebutton">View All offers</button>**

**</div>**

**<div className="assistancetext">**

**<h2 className="assistanceh2">Experience a new Digital World. Welcome to bob World</h2>**

**<p className="assistancep">**

**Our official mobile banking application with new and exciting features and easy user interface.**

**</p>**

**<div className="assistancebadges">**

**<img src={appstore} alt="App Store" />**

**<img src={googleplay} alt="Google Play" />**

**</div>**

**</div>**

**</div>**

**</div>**

**</>**

**);**

**}**

**export default Home;**

**...**

**==============**

**============================================================================================================================================================================================**

**"""Login.css"""**

**.container {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**justify-content: center;**

**height: 100vh;**

**background-color: #f4f4f4;**

**}**

**h2 {**

**margin-bottom: 20px;**

**color: #333;**

**}**

**form {**

**width: 100%;**

**max-width: 400px;**

**background-color: #fff;**

**padding: 20px;**

**border-radius: 8px;**

**box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);**

**}**

**.form-group {**

**margin-bottom: 15px;**

**}**

**.form-group label {**

**display: block;**

**margin-bottom: 5px;**

**}**

**.form-group input {**

**width: 100%;**

**padding: 10px;**

**border: 1px solid #ddd;**

**border-radius: 4px;**

**}**

**button {**

**background-color: #007bff;**

**color: #fff;**

**border: none;**

**padding: 10px;**

**border-radius: 4px;**

**cursor: pointer;**

**width: 100%;**

**}**

**button:hover {**

**background-color: #0056b3;**

**}**

**.error {**

**color: red;**

**margin-top: 10px;**

**}**

**p {**

**text-align: center;**

**margin-top: 20px;**

**}**

**p a {**

**color: #007bff;**

**text-decoration: none;**

**}**

**p a:hover {**

**text-decoration: underline;**

**}**

**....**

**"""Login.jsx"""**

**import React, { useState } from 'react';**

**import { Link, useNavigate } from 'react-router-dom';**

**import './Login.css';**

**const Login = () => {**

**const [email, setEmail] = useState('');**

**const [password, setPassword] = useState('');**

**const [error, setError] = useState('');**

**const navigate = useNavigate();**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**if (!email || !password) {**

**setError('Please fill in all fields');**

**return;**

**}**

**try {**

**const response = await fetch(http://localhost:3000/users?email=${email});**

**const users = await response.json();**

**if (users.length === 0) {**

**setError('Invalid email');**

**return;**

**}**

**const user = users[0];**

**if (user.password !== password) {**

**setError('Invalid password');**

**return;**

**}**

**// Successful login**

**alert('Login successful');**

**navigate('/home');**

**} catch (error) {**

**setError('An error occurred');**

**}**

**};**

**return (**

**<div className="container">**

**<h2>Login</h2>**

**<form onSubmit={handleSubmit}>**

**<div className="form-group">**

**<label>Email:</label>**

**<input**

**type="email"**

**value={email}**

**onChange={(e) => setEmail(e.target.value)}**

**required**

**/>**

**</div>**

**<div className="form-group">**

**<label>Password:</label>**

**<input**

**type="password"**

**value={password}**

**onChange={(e) => setPassword(e.target.value)}**

**required**

**/>**

**</div>**

**{error && <p className="error">{error}</p>}**

**<button type="submit">Login</button>**

**</form>**

**<p>**

**Don't have an account? <Link to="/signup">Signup</Link>**

**</p>**

**</div>**

**);**

**};**

**export default Login;**

**.....**

**"""Signup.css"""**

**/\* AdminSignIn.css \*/**

**html, body {**

**height: 100%; /\* Ensure the body takes up the full height \*/**

**margin: 0;**

**font-family: Arial, sans-serif; /\* Optional: Add a font-family for consistency \*/**

**}**

**.container {**

**max-width: 400px;**

**margin: 0 auto;**

**padding: 2rem;**

**background-color: #f8f9fa;**

**border-radius: 8px;**

**box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);**

**position: absolute; /\* Position it absolutely within the viewport \*/**

**top: 50%; /\* Center it vertically \*/**

**left: 50%; /\* Center it horizontally \*/**

**transform: translate(-50%, -50%); /\* Adjust to truly center the element \*/**

**}**

**h2 {**

**text-align: center;**

**margin-bottom: 1.5rem;**

**color: #343a40;**

**}**

**.form-group {**

**margin-bottom: 1.5rem;**

**}**

**label {**

**display: block;**

**margin-bottom: 0.5rem;**

**font-weight: 600;**

**color: #495057;**

**}**

**input[type="email"],**

**input[type="password"] {**

**width: 100%;**

**padding: 0.75rem;**

**border: 1px solid #ced4da;**

**border-radius: 4px;**

**box-sizing: border-box;**

**}**

**input[type="email"]:focus,**

**input[type="password"]:focus {**

**border-color: #80bdff;**

**outline: none;**

**box-shadow: 0 0 0 0.2rem rgba(38, 143, 255, 0.25);**

**}**

**button {**

**width: 100%;**

**padding: 0.75rem;**

**border: none;**

**border-radius: 4px;**

**background-color: #007bff;**

**color: white;**

**font-size: 1rem;**

**cursor: pointer;**

**}**

**button:hover {**

**background-color: #0056b3;**

**}**

**p {**

**text-align: center;**

**margin-top: 1.5rem;**

**}**

**a {**

**color: #007bff;**

**text-decoration: none;**

**}**

**a:hover {**

**text-decoration: underline;**

**}**

**......**

**"""Signup.jsx"""**

**import React, { useState } from 'react';**

**import { Link, useNavigate } from 'react-router-dom';**

**import './Signup.css';**

**const Signup = () => {**

**const [email, setEmail] = useState('');**

**const [password, setPassword] = useState('');**

**const [confirmPassword, setConfirmPassword] = useState('');**

**const navigate = useNavigate();**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**if (password !== confirmPassword) {**

**alert("Passwords do not match");**

**return;**

**}**

**const newUser = { email, password };**

**const response = await fetch('http://localhost:3000/users', {**

**method: 'POST',**

**headers: {**

**'Content-Type': 'application/json',**

**},**

**body: JSON.stringify(newUser),**

**});**

**if (response.ok) {**

**alert("Signup successful");**

**navigate('/login');**

**} else {**

**alert("Signup failed");**

**}**

**};**

**return (**

**<div className="container">**

**<h2>Signup</h2>**

**<form onSubmit={handleSubmit}>**

**<div className="form-group">**

**<label>Email:</label>**

**<input**

**type="email"**

**value={email}**

**onChange={(e) => setEmail(e.target.value)}**

**required**

**/>**

**</div>**

**<div className="form-group">**

**<label>Password:</label>**

**<input**

**type="password"**

**value={password}**

**onChange={(e) => setPassword(e.target.value)}**

**required**

**/>**

**</div>**

**<div className="form-group">**

**<label>Confirm Password:</label>**

**<input**

**type="password"**

**value={confirmPassword}**

**onChange={(e) => setConfirmPassword(e.target.value)}**

**required**

**/>**

**</div>**

**<button type="submit">Signup</button>**

**</form>**

**<p>**

**Already have an account? <Link to="/login">Login</Link>**

**</p>**

**</div>**

**);**

**};**

**export default Signup;**

**.....**

**D:\projextnext\project\bankproject\src**

**"""App.css"""**

**/\*#root {**

**max-width: 1280px;**

**margin: 0 auto;**

**padding: 2rem;**

**text-align: center;**

**}**

**.logo {**

**height: 6em;**

**padding: 1.5em;**

**will-change: filter;**

**transition: filter 300ms;**

**}**

**.logo:hover {**

**filter: drop-shadow(0 0 2em #646cffaa);**

**}**

**.logo.react:hover {**

**filter: drop-shadow(0 0 2em #61dafbaa);**

**}**

**@keyframes logo-spin {**

**from {**

**transform: rotate(0deg);**

**}**

**to {**

**transform: rotate(360deg);**

**}**

**}**

**@media (prefers-reduced-motion: no-preference) {**

**a:nth-of-type(2) .logo {**

**animation: logo-spin infinite 20s linear;**

**}**

**}**

**.card {**

**padding: 2em;**

**}**

**.read-the-docs {**

**color: #888;**

**}**

**\*/**

**...**

**"""App.jsx"""**

**import React from 'react';**

**import { BrowserRouter as Router, Routes, Route, Navigate, useLocation } from 'react-router-dom';**

**import Login from './components/Login';**

**import Signup from './components/Signup';**

**import Home from './components/Home';**

**import Header from './components/Header';**

**import CardDetails from './components/CardDetails';**

**import Footer from './components/Footer';**

**import ApplyNow from './components/ApplyNow';**

**import BasicInformation from './components/BasicInformationPage';**

**import BasicInfoAccount from './components/BasicInfoAccountPresent';**

**import LoginPage from './LoginPage';**

**import AdminHome from './adminaccess/AdminHome';**

**import AdminLogin from './adminaccess/AdminLogin';**

**import AdminSignUp from './adminaccess/AdminSignUp'; // Import AdminSignUp**

**import ManageUsers from './adminaccess/ManageUsers'**

**const AppContent = () => {**

**const location = useLocation();**

**const hideHeaderFooter = ['/apply-now', '/basic', '/BasicInfoAccount','/admin-login','/admin-signup','/login','/signup','/logindupe'].includes(location.pathname);**

**return (**

**<>**

**{!hideHeaderFooter && <Header />}**

**<Routes>**

**<Route path="/logindupe" element={<LoginPage />} />**

**<Route path="/home" element={<Home />} />**

**<Route path="/admin-home" element={<AdminHome />} />**

**<Route path="/admin-login" element={<AdminLogin />} />**

**<Route path="/admin-signup" element={<AdminSignUp />} />**

**<Route path="/login" element={<Login />} />**

**<Route path="/signup" element={<Signup />} />**

**<Route path="/details/:id" element={<CardDetails />} />**

**<Route path="/apply-now" element={<ApplyNow />} />**

**<Route path="/basic" element={<BasicInformation />} />**

**<Route path="/BasicInfoAccount" element={<BasicInfoAccount />} />**

**<Route path="/manage-users" component={ManageUsers} />**

**<Route path="/" element={<Navigate to="/home" />} />**

**<Route path="\*" element={<Navigate to="/login" />} />**

**</Routes>**

**{!hideHeaderFooter && <Footer />}**

**</>**

**);**

**};**

**function App() {**

**return (**

**<Router>**

**<AppContent />**

**</Router>**

**);**

**}**

**export default App;**

**=============**

**========================================================================================================================================================================================**

**.....**

**"""LoginPage.css"""**

**/\* LoginPage Styles \*/**

**.login-page {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**justify-content: center;**

**min-height: 100vh;**

**background-color: #f4f4f4;**

**padding: 20px;**

**}**

**.login-page .logintitle {**

**margin-bottom: 30px;**

**font-size: 2em;**

**color: #333;**

**}**

**.login-options {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**}**

**.login-option {**

**margin-bottom: 20px;**

**}**

**.login-option a {**

**margin: 0 10px;**

**}**

**.login-option button {**

**background-color: #007bff;**

**color: white;**

**border: none;**

**padding: 10px 20px;**

**border-radius: 5px;**

**cursor: pointer;**

**transition: background-color 0.3s ease, transform 0.3s ease;**

**font-size: 1em;**

**box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);**

**}**

**.login-option button:hover {**

**background-color: #0056b3;**

**transform: translateY(-2px);**

**}**

**.signup-btn {**

**background-color: #28a745;**

**}**

**.signup-btn:hover {**

**background-color: #218838;**

**}**

**...**

**"""LoginPage.jsx"""**

**import React from "react";**

**import { Link } from "react-router-dom";**

**import "./LoginPage.css";**

**const LoginPage = () => {**

**return (**

**<div className="login-page">**

**<h1 className="logintitle">Welcome to AMJ bank</h1>**

**<div className="login-options">**

**<div className="login-option">**

**<Link to="/login">**

**<button>User Login</button>**

**</Link>**

**<Link to="/signup">**

**<button className="signup-btn">User Sign-up</button>**

**</Link>**

**</div>**

**<div className="login-option">**

**<Link to="/admin-login">**

**<button>Admin Login</button>**

**</Link>**

**<Link to="/admin-signup">**

**<button className="signup-btn">Admin Sign-up</button>**

**</Link>**

**</div>**

**</div>**

**</div>**

**);**

**};**

**export default LoginPage;**

**/\*import React from "react";**

**import { Link } from "react-router-dom";**

**import "./LoginPage.css";**

**const LoginPage = () => {**

**return (**

**<div className="login-page">**

**<h1>Welcome to the Bank</h1>**

**<div className="login-options">**

**<div className="login-option">**

**<Link to="/login">**

**<button>User Login</button>**

**</Link>**

**</div>**

**<div className="login-option">**

**<Link to="/admin-login">**

**<button>Admin Login</button>**

**</Link>**

**</div>**

**</div>**

**</div>**

**);**

**};**

**export default LoginPage;\*/**

**/\*import React from "react";**

**import { Link } from "react-router-dom";**

**import "./LoginPage.css";**

**const LoginPage = () => {**

**return (**

**<div className="login-page">**

**<h1>Welcome to the Bank</h1>**

**<div className="login-options">**

**<div className="login-option">**

**<Link to="/login">**

**<button>User Login</button>**

**</Link>**

**<Link to="/signup">**

**<button>User Sign-up</button>**

**</Link>**

**</div>**

**<div className="login-option">**

**<Link to="/admin-login">**

**<button>Admin Login</button>**

**</Link>**

**<Link to="/admin-signup">**

**<button>Admin Sign-up</button>**

**</Link>**

**</div>**

**</div>**

**</div>**

**);**

**};**

**export default LoginPage;**

**\*/**

**....**

**"""main.jsx"""**

**import React from 'react'**

**import ReactDOM from 'react-dom/client'**

**import App from './App.jsx'**

**import './index.css'**

**ReactDOM.createRoot(document.getElementById('root')).render(**

**<React.StrictMode>**

**<App />**

**</React.StrictMode>,**

**)**

**.........**

**Django:**

**D:\djangoprojectscratch\bankproject**

**"""admin.py"""**

**from django.contrib import admin**

**# Register your models here.**

**...**

**"""apps.py"""**

**from django.apps import AppConfig**

**class BankappConfig(AppConfig):**

**default\_auto\_field = 'django.db.models.BigAutoField'**

**name = 'bankapp'**

**....**

**"""modules.py"""from django.db import models**

**# Create your models here.**

**class BasicInformation(models.Model):**

**id = models.CharField(max\_length=50, primary\_key=True)**

**full\_name = models.CharField(max\_length=255)**

**father\_or\_husband\_name = models.CharField(max\_length=255)**

**gender = models.CharField(max\_length=10)**

**occupation = models.CharField(max\_length=255)**

**income = models.CharField(max\_length=255)**

**kyc\_document\_number = models.CharField(max\_length=255)**

**email = models.EmailField()**

**mobile = models.CharField(max\_length=20)**

**permanent\_address = models.TextField()**

**address\_details = models.TextField()**

**communication\_address = models.TextField()**

**pin\_code = models.CharField(max\_length=10)**

**# def \_\_str\_\_(self):**

**# return self.full\_name**

**class BasicInfoAccount(models.Model):**

**account\_number = models.CharField(max\_length=50)**

**loan\_amount = models.CharField(max\_length=50)**

**loan\_requested\_for = models.CharField(max\_length=255)**

**# def \_\_str\_\_(self):**

**# return self.account\_number**

**class User(models.Model):**

**id = models.CharField(max\_length=50, primary\_key=True)**

**email = models.EmailField()**

**password = models.CharField(max\_length=255)**

**role = models.CharField(max\_length=20, blank=True, null=True)**

**# def \_\_str\_\_(self):**

**# return self.email**

**class Admin(models.Model):**

**id = models.CharField(max\_length=50, primary\_key=True)**

**email = models.EmailField()**

**password = models.CharField(max\_length=255)**

**role = models.CharField(max\_length=20)**

**# def \_\_str\_\_(self):**

**# return self.email**

**....**

**"""serializer.py"""**

**from rest\_framework import serializers**

**from .models import \***

**class BasicInformationSerializer(serializers.ModelSerializer):**

**class Meta:**

**model = BasicInformation**

**fields = '\_\_all\_\_'**

**class BasicInfoAccountSerializer(serializers.ModelSerializer):**

**class Meta:**

**model = BasicInfoAccount**

**fields = '\_\_all\_\_'**

**class UserSerializer(serializers.ModelSerializer):**

**class Meta:**

**model = User**

**fields = '\_\_all\_\_'**

**class AdminSerializer(serializers.ModelSerializer):**

**class Meta:**

**model = Admin**

**fields = '\_\_all\_\_'**

**.....**

**"""tests.py"""from django.test import TestCase**

**# Create your tests here.**

**...**

**"""views.py"""**

**from rest\_framework.views import APIView**

**from rest\_framework.response import Response**

**from rest\_framework import status**

**from .models import BasicInformation, BasicInfoAccount, User, Admin**

**from .serializer import BasicInformationSerializer, BasicInfoAccountSerializer, UserSerializer, AdminSerializer**

**class BasicInformationView(APIView):**

**def get(self, request, pk=None):**

**if pk:**

**try:**

**info = BasicInformation.objects.get(pk=pk)**

**serializer = BasicInformationSerializer(info)**

**return Response(serializer.data)**

**except BasicInformation.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**else:**

**basic\_info\_objects = BasicInformation.objects.all()**

**serializer = BasicInformationSerializer(basic\_info\_objects, many=True)**

**return Response(serializer.data)**

**def post(self, request):**

**serializer = BasicInformationSerializer(data=request.data)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data, status=status.HTTP\_201\_CREATED)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def put(self, request, pk=None):**

**try:**

**info = BasicInformation.objects.get(pk=pk)**

**except BasicInformation.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**serializer = BasicInformationSerializer(info, data=request.data, partial=True)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def delete(self, request, pk=None):**

**try:**

**info = BasicInformation.objects.get(pk=pk)**

**except BasicInformation.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**info.delete()**

**return Response(status=status.HTTP\_204\_NO\_CONTENT)**

**class BasicInfoAccountView(APIView):**

**def get(self, request, pk=None):**

**if pk:**

**try:**

**account = BasicInfoAccount.objects.get(pk=pk)**

**serializer = BasicInfoAccountSerializer(account)**

**return Response(serializer.data)**

**except BasicInfoAccount.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**else:**

**account\_objects = BasicInfoAccount.objects.all()**

**serializer = BasicInfoAccountSerializer(account\_objects, many=True)**

**return Response(serializer.data)**

**def post(self, request):**

**serializer = BasicInfoAccountSerializer(data=request.data)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data, status=status.HTTP\_201\_CREATED)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def put(self, request, pk=None):**

**try:**

**account = BasicInfoAccount.objects.get(pk=pk)**

**except BasicInfoAccount.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**serializer = BasicInfoAccountSerializer(account, data=request.data, partial=True)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def delete(self, request, pk=None):**

**try:**

**account = BasicInfoAccount.objects.get(pk=pk)**

**except BasicInfoAccount.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**account.delete()**

**return Response(status=status.HTTP\_204\_NO\_CONTENT)**

**class UserView(APIView):**

**def get(self, request, pk=None):**

**if pk:**

**try:**

**user = User.objects.get(pk=pk)**

**serializer = UserSerializer(user)**

**return Response(serializer.data)**

**except User.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**else:**

**user\_objects = User.objects.all()**

**serializer = UserSerializer(user\_objects, many=True)**

**return Response(serializer.data)**

**def post(self, request):**

**serializer = UserSerializer(data=request.data)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data, status=status.HTTP\_201\_CREATED)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def put(self, request, pk=None):**

**try:**

**user = User.objects.get(pk=pk)**

**except User.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**serializer = UserSerializer(user, data=request.data, partial=True)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def delete(self, request, pk=None):**

**try:**

**user = User.objects.get(pk=pk)**

**except User.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**user.delete()**

**return Response(status=status.HTTP\_204\_NO\_CONTENT)**

**class AdminView(APIView):**

**def get(self, request, pk=None):**

**if pk:**

**try:**

**admin = Admin.objects.get(pk=pk)**

**serializer = AdminSerializer(admin)**

**return Response(serializer.data)**

**except Admin.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**else:**

**admin\_objects = Admin.objects.all()**

**serializer = AdminSerializer(admin\_objects, many=True)**

**return Response(serializer.data)**

**def post(self, request):**

**serializer = AdminSerializer(data=request.data)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data, status=status.HTTP\_201\_CREATED)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def put(self, request, pk=None):**

**try:**

**admin = Admin.objects.get(pk=pk)**

**except Admin.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**serializer = AdminSerializer(admin, data=request.data, partial=True)**

**if serializer.is\_valid():**

**serializer.save()**

**return Response(serializer.data)**

**return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)**

**def delete(self, request, pk=None):**

**try:**

**admin = Admin.objects.get(pk=pk)**

**except Admin.DoesNotExist:**

**return Response(status=status.HTTP\_404\_NOT\_FOUND)**

**admin.delete()**

**return Response(status=status.HTTP\_204\_NO\_CONTENT)**

**.......**

**D:\djangoprojectscratch\bankproject\bankproject**

**"""asgi.py"""**

**import os**

**from django.core.asgi import get\_asgi\_application**

**os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'bankproject.settings')**

**application = get\_asgi\_application()**

**......**

**"""urls.py"""**

**from django.contrib import admin**

**from django.urls import path, re\_path,include**

**from django.urls import path**

**from bankapp.views import BasicInformationView, BasicInfoAccountView, UserView, AdminView**

**urlpatterns = [**

**path('admin/', admin.site.urls),**

**path('basic-information/', BasicInformationView.as\_view(), name='basic\_information\_list\_create'),**

**path('basic-information/<int:pk>/', BasicInformationView.as\_view(), name='basic\_information\_detail'),**

**path('basic-info-account/', BasicInfoAccountView.as\_view(), name='basic\_info\_account\_list\_create'),**

**path('basic-info-account/<int:pk>/', BasicInfoAccountView.as\_view(), name='basic\_info\_account\_detail'),**

**path('users/', UserView.as\_view(), name='user\_list\_create'),**

**path('users/<int:pk>/', UserView.as\_view(), name='user\_detail'),**

**path('admins/', AdminView.as\_view(), name='admin\_list\_create'),**

**path('admins/<int:pk>/', AdminView.as\_view(), name='admin\_detail'),**

**]**

**.......**

**"""wsgi.py"""**

**import os**

**from django.core.wsgi import get\_wsgi\_application**

**os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'bankproject.settings')**

**application = get\_wsgi\_application()**

**......**

**"""manage.py"""**

**#!/usr/bin/env python**

**"""Django's command-line utility for administrative tasks."""**

**import os**

**import sys**

**def main():**

**"""Run administrative tasks."""**

**os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'bankproject.settings')**

**try:**

**from django.core.management import execute\_from\_command\_line**

**except ImportError as exc:**

**raise ImportError(**

**"Couldn't import Django. Are you sure it's installed and "**

**"available on your PYTHONPATH environment variable? Did you "**

**"forget to activate a virtual environment?"**

**) from exc**

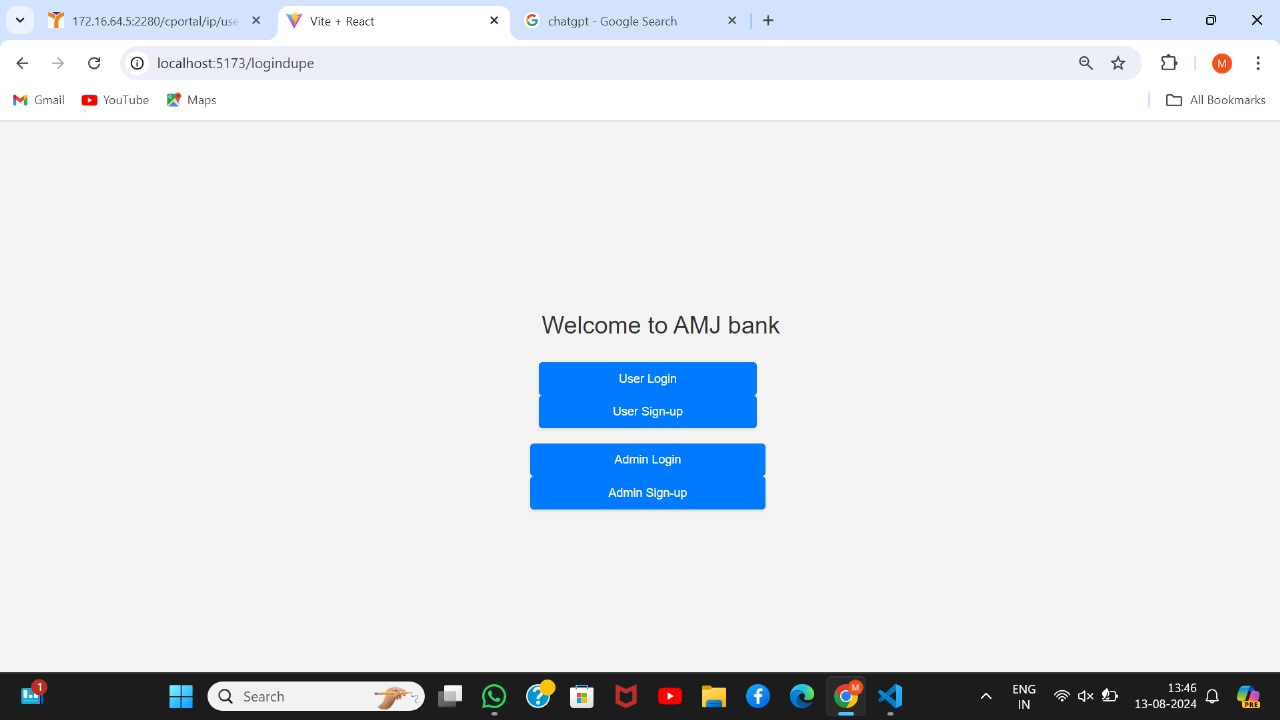
**execute\_from\_command\_line(sys.argv)**

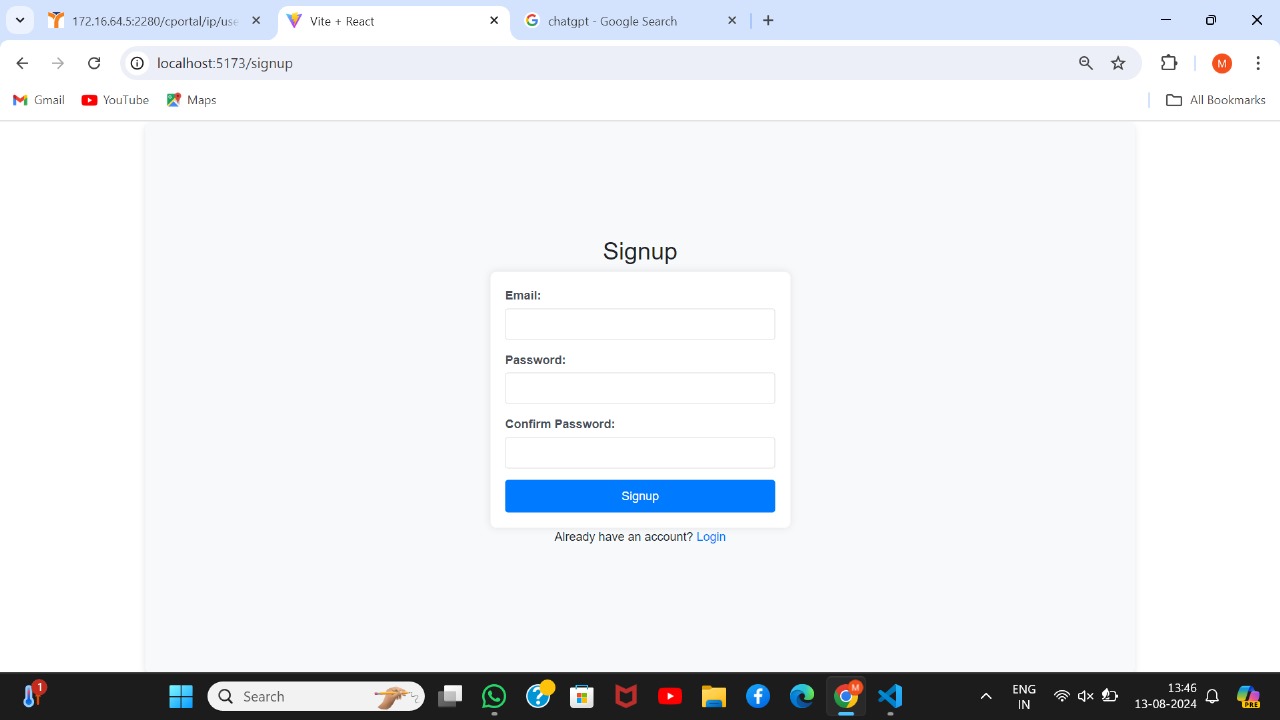
**if \_\_name\_\_ == '\_\_main\_\_':**

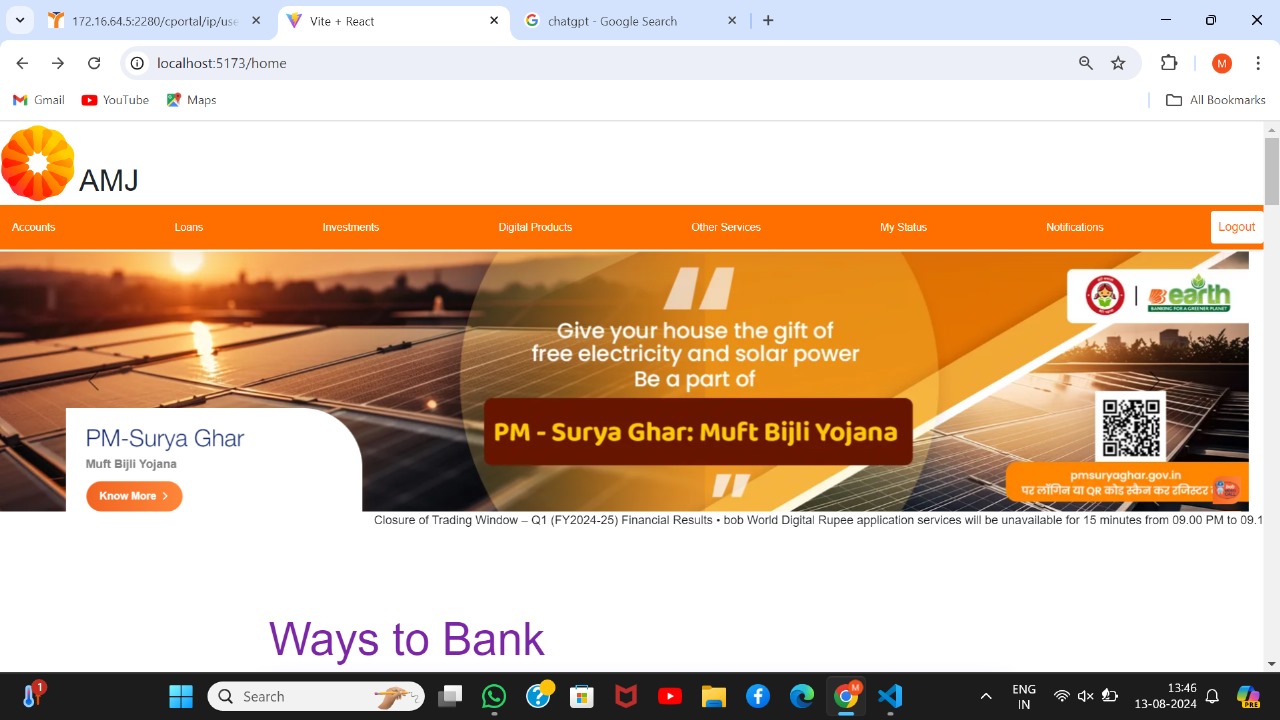
**main()**

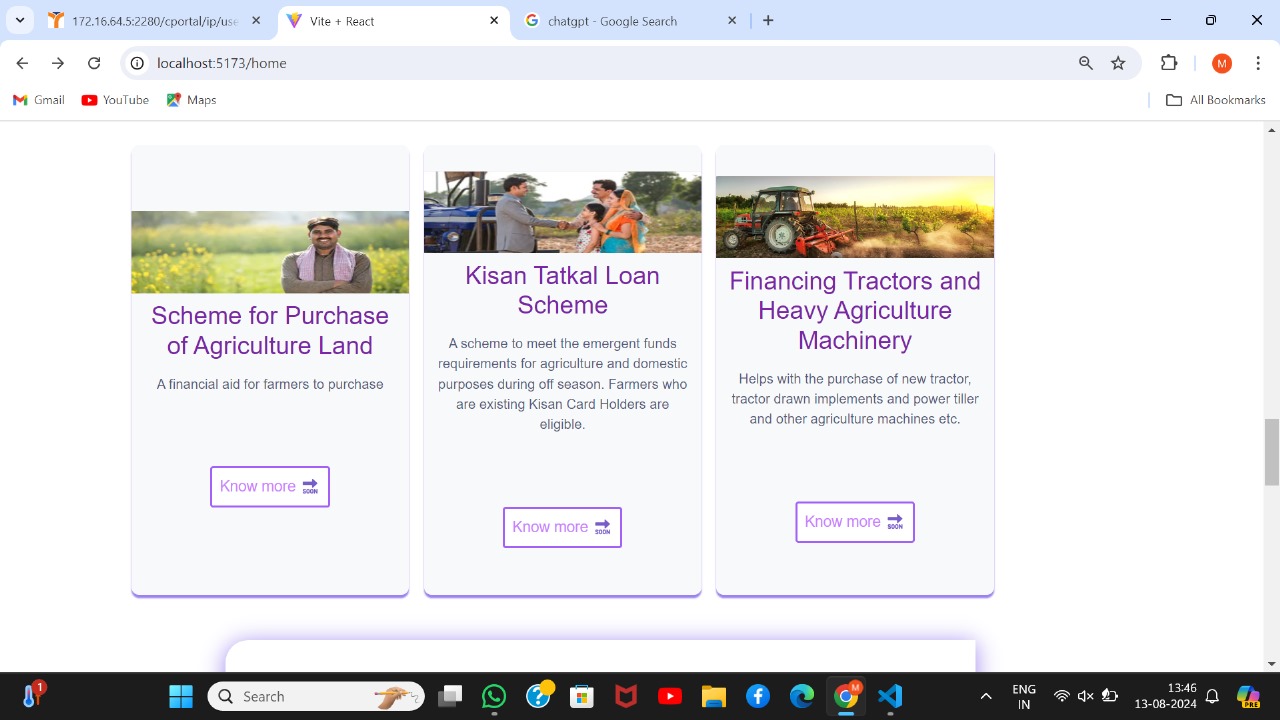
**.....**

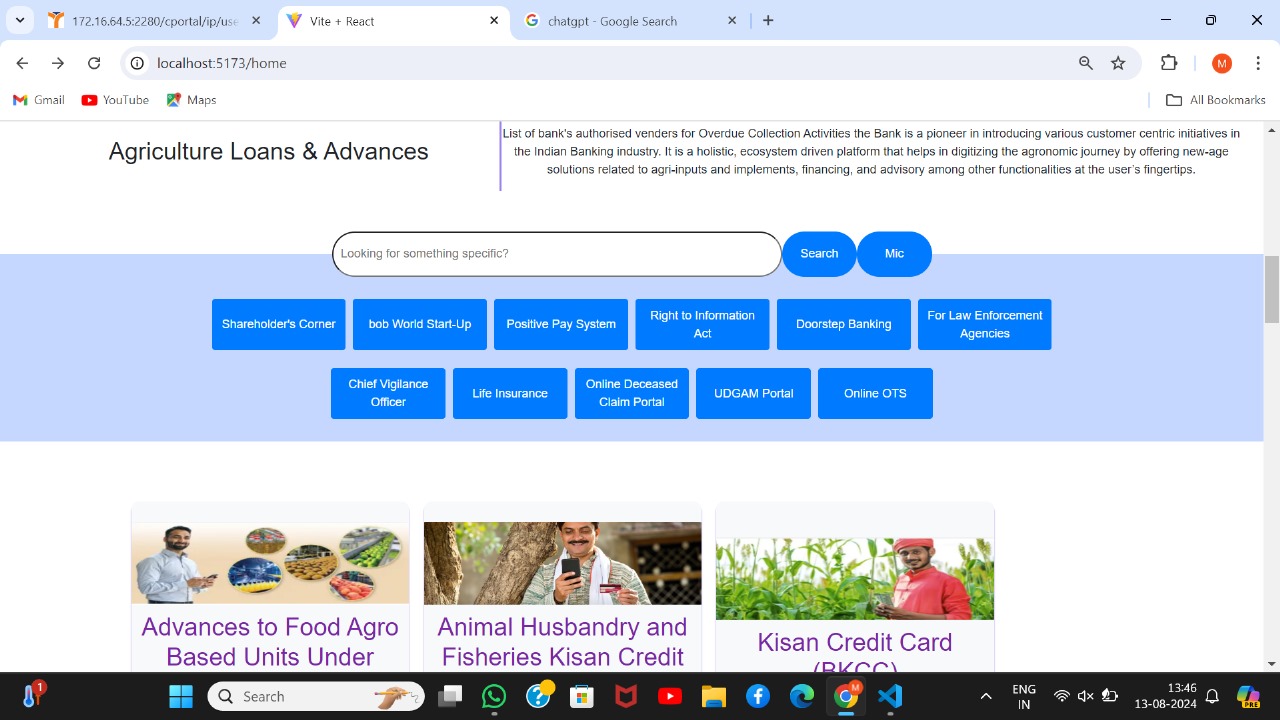
1. **Outputs(ScreenShorts)**

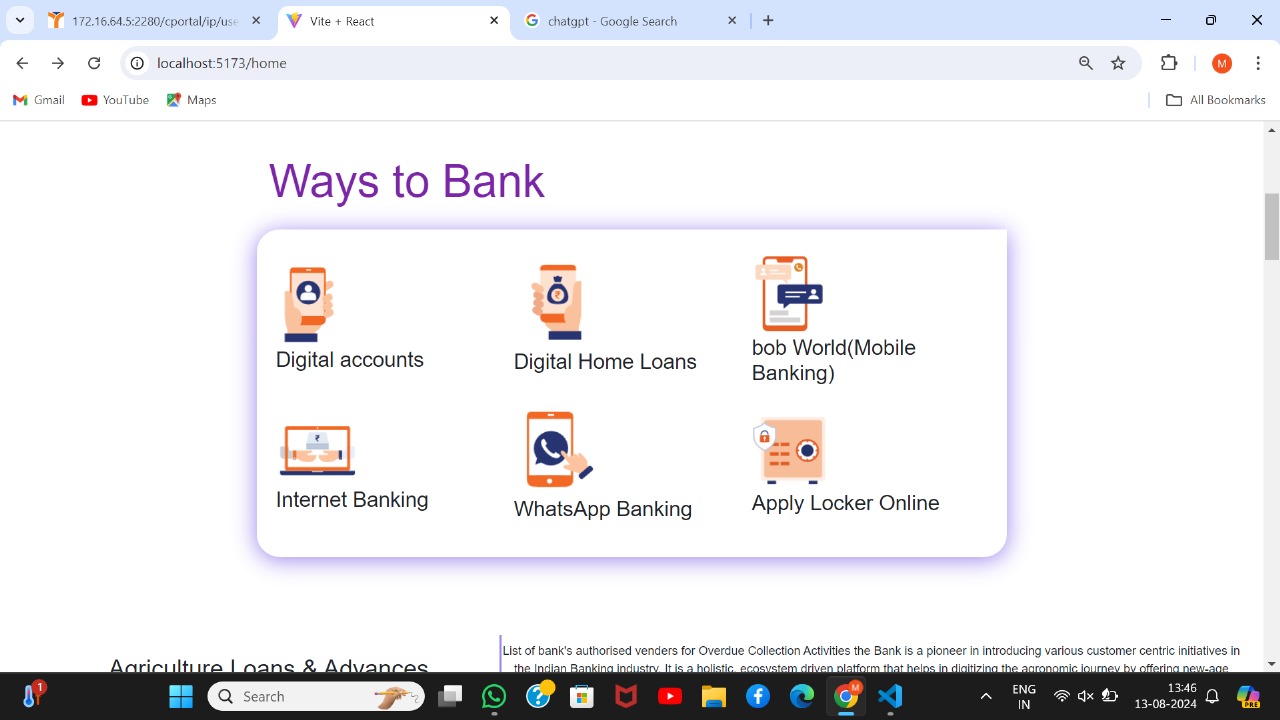
****

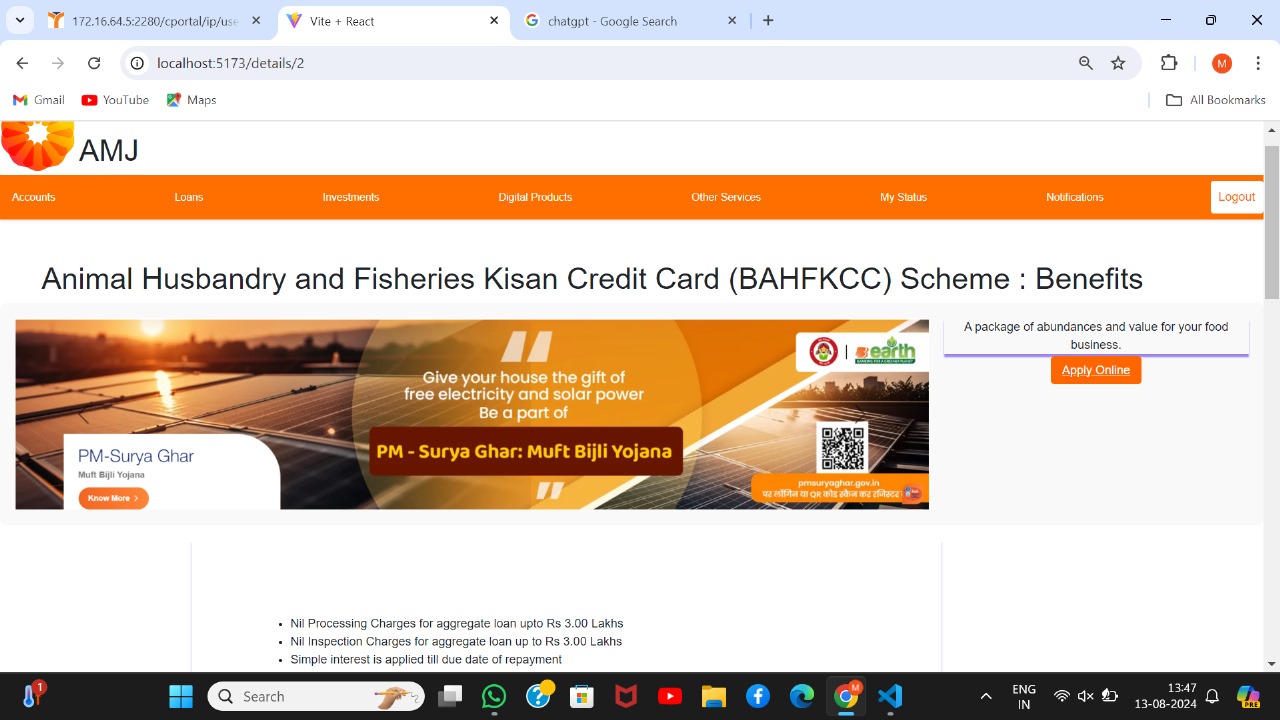
****

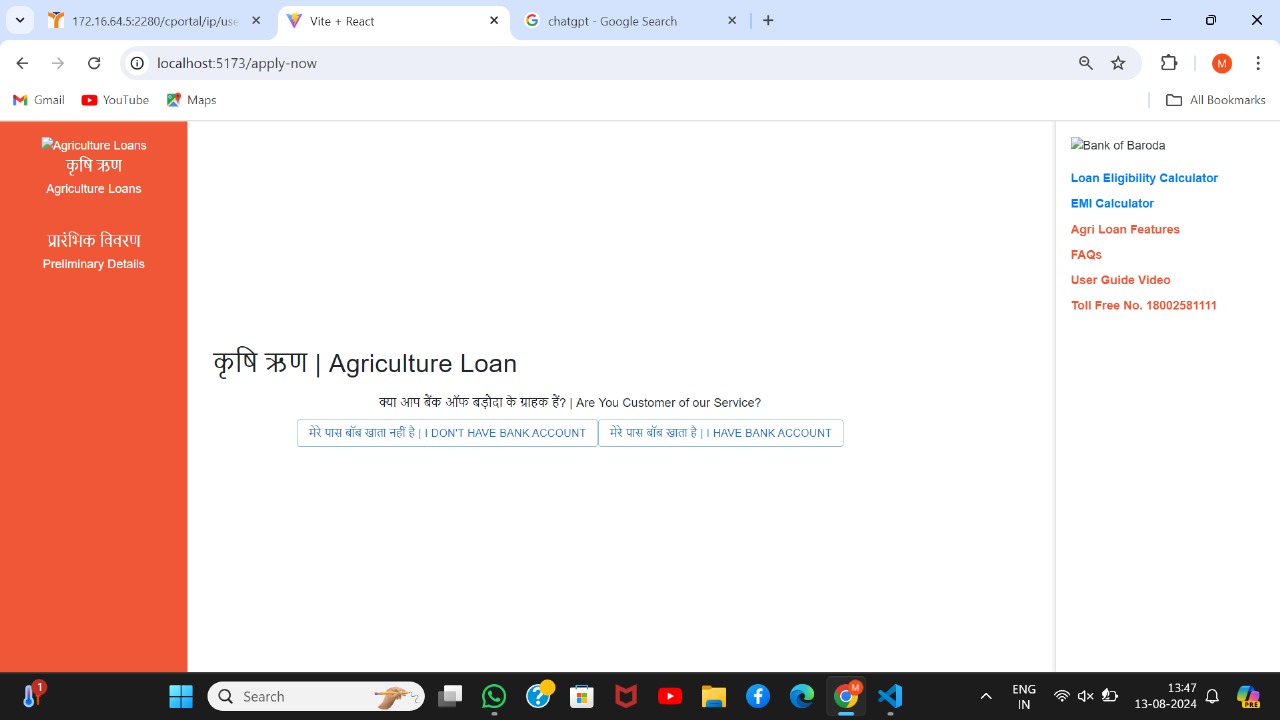
**s**

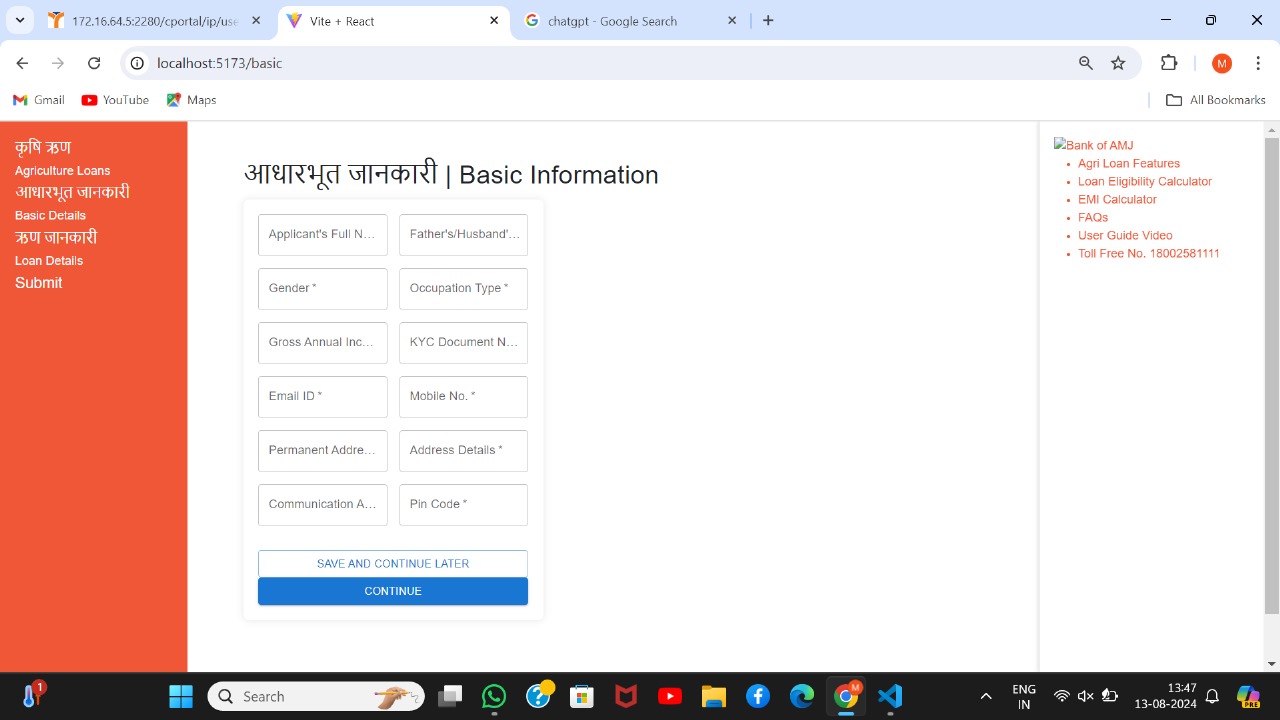
****

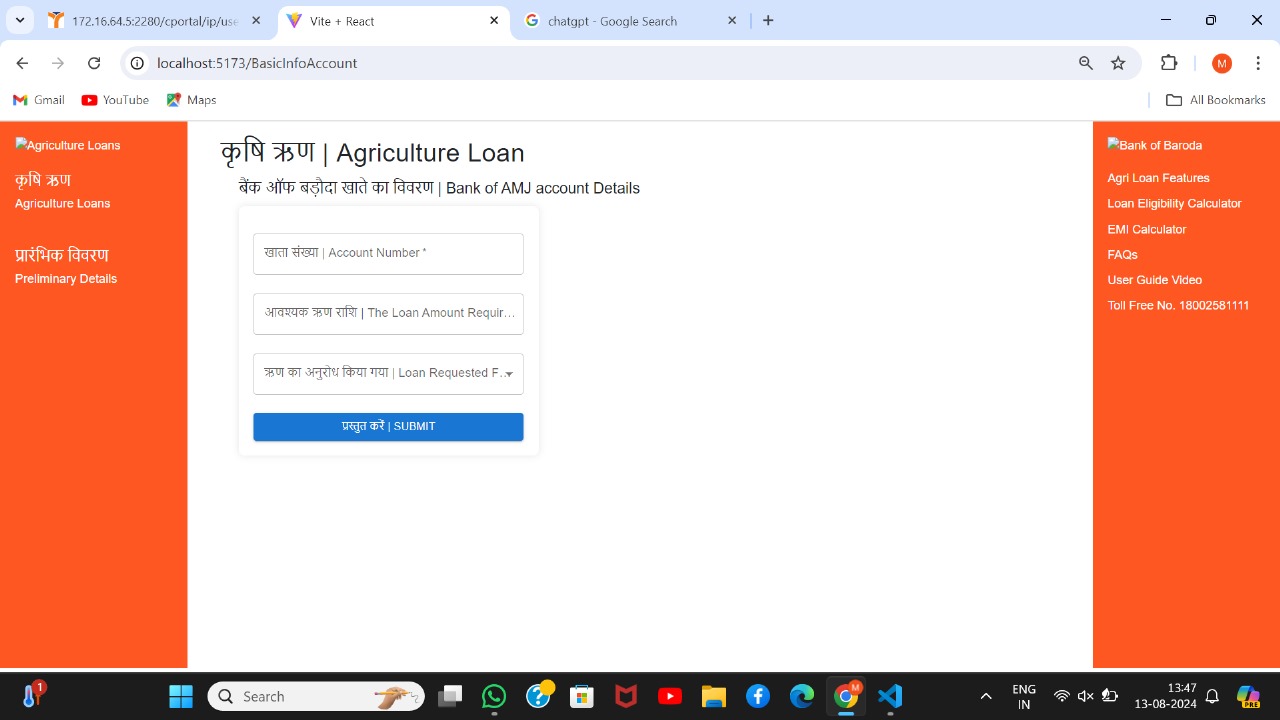
****

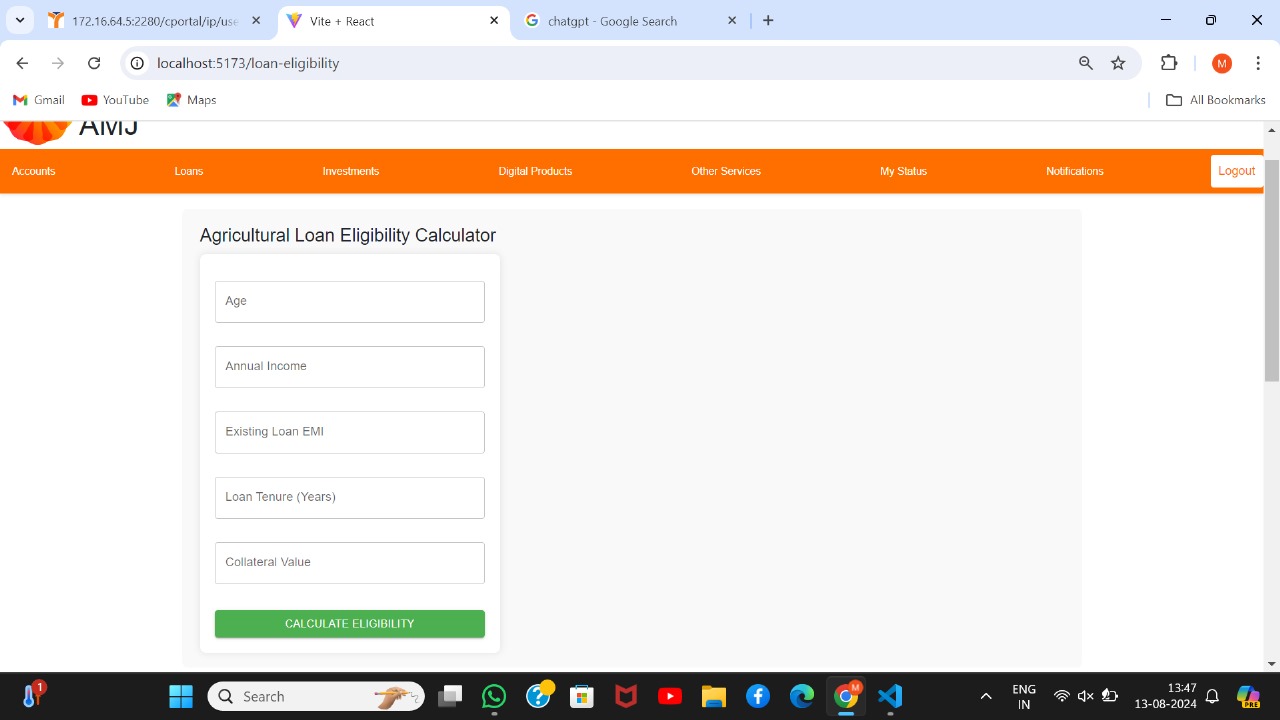
****

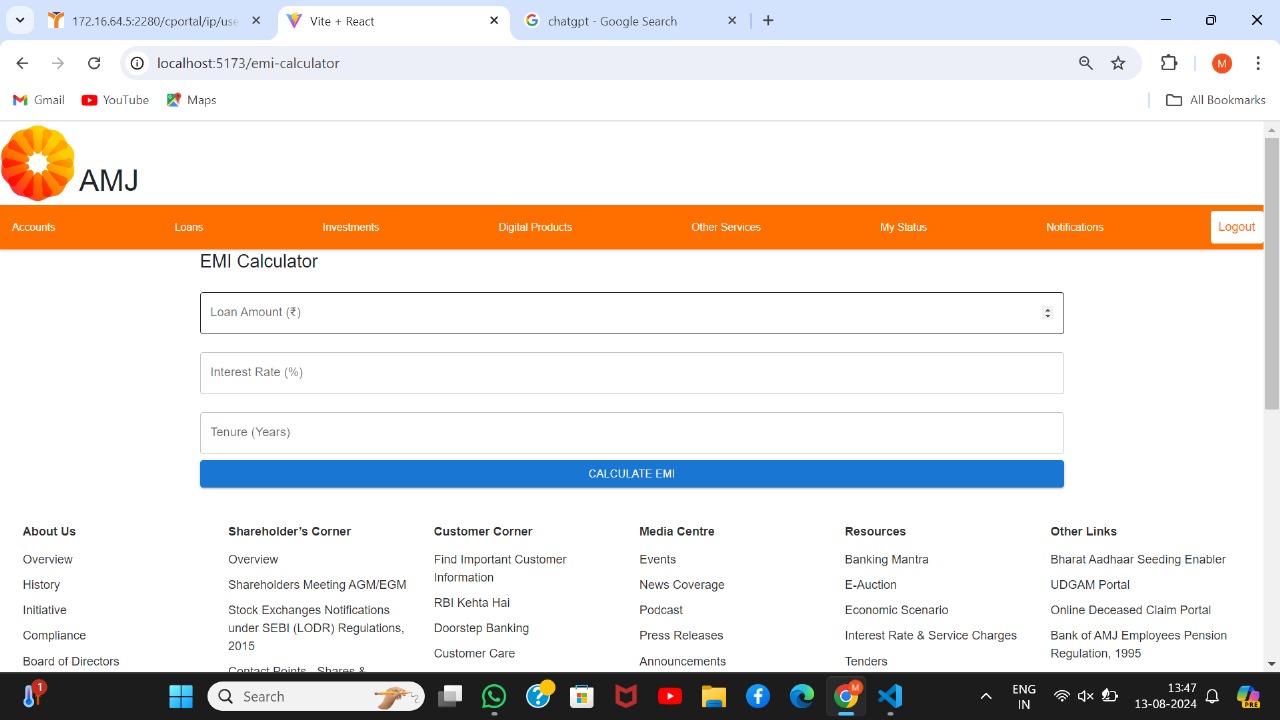
****

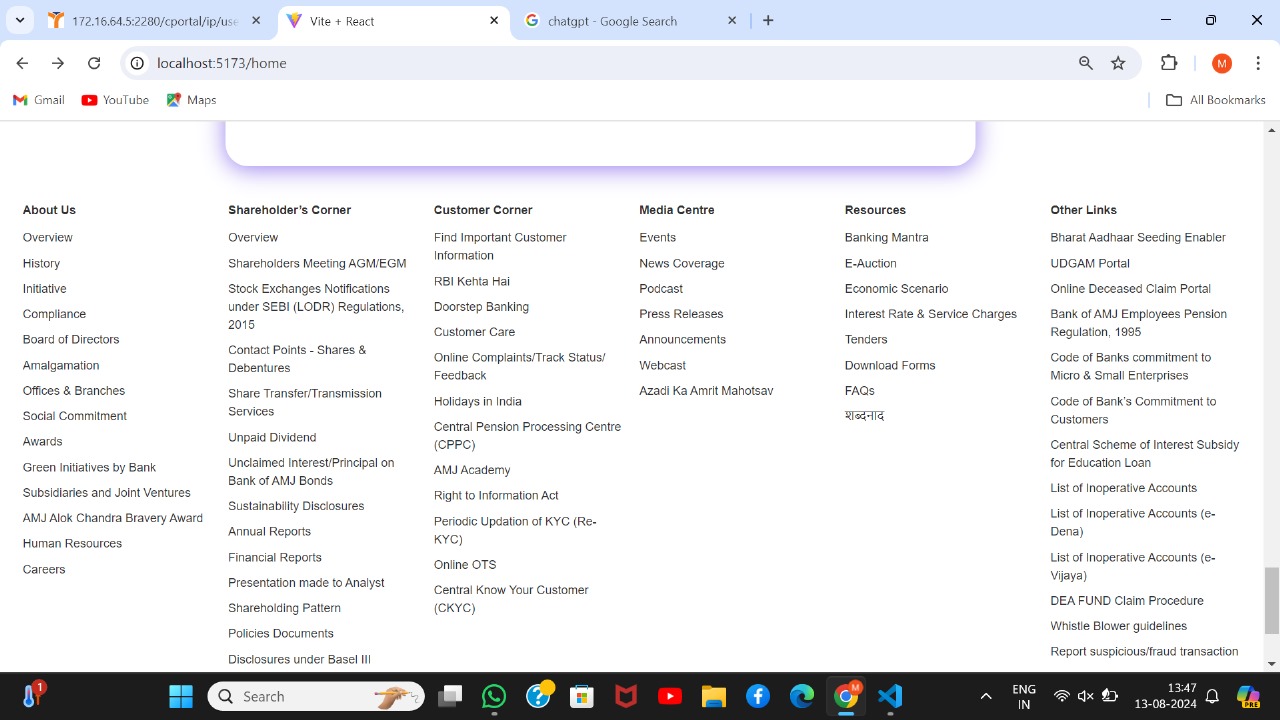
****

****

****

****

****

****

### 11.Conclusion and Future Enhancements

### 11.1 Conclusion

The "Agriculture Loan Processing System" project successfully met its objectives, providing a robust and secure platform for loan processing in the agricultural sector. The system addresses the inefficiencies of traditional loan processing methods and offers a scalable solution that can grow with the needs of users.

### 11.2 Future Enhancements

This section outlines potential enhancements and future work to further improve the "Agriculture Loan Processing System."

#### 11.2.1 Feature Enhancements

**Advanced Analytics:** Incorporate data analytics to provide insights into loan performance, risk assessment, and personalized loan products for farmers.

**Machine Learning:** Implement machine learning models to improve credit scoring, predict loan defaults, and optimize loan approval processes.

**Mobile Integration:** Develop a mobile app or responsive design to increase accessibility for users in rural areas with limited access to desktop computers.

#### 11.2.2 Technology Upgrades

**React and Django Upgrades:** Keep the system updated with the latest versions of React and Django to take advantage of new features and performance improvements.

**Cloud Infrastructure:** Migrate to a cloud-based infrastructure to improve scalability, availability, and disaster recovery capabilities.

**Blockchain Integration:** Explore the integration of blockchain technology for secure and transparent loan transactions, reducing the risk of fraud.

#### 11.2.3 Scalability Considerations

**Database Optimization:** Continue optimizing database queries and indexing strategies to handle larger datasets as the user base grows.

**Caching Mechanisms:** Implement advanced caching mechanisms to reduce server load and improve response times.

**Distributed Architecture:** Consider deploying the system in a distributed architecture to enhance scalability and fault tolerance.