REQUIREMENT ANALYSIS

1)Need stronger security
2)Scalability issues needs to be addressed
3)Implement Fraud Detection
4)Handle high volume transactions

Modules Needed :- 1) User module

2) Fraud Detection module

3) Credit module

4) Transaction module

Frontend module:- 1)User Authentication

i.Login- username and pwd

ii.Registration-

2)Transaction

i.Deposit

ii.Withdrawal

iii.Transaction History

3)Credit card details

i.View Details

ii.Credit card statements

iii. Issue credit

4)Fraud Detection

i.Fraud Alert

ii.Check Fraud

User Schema

1) Account Number (Primary key) (Auto Generated)
2) Account Holder Name
3) Account Type
4) Credit card number-
5) Username (Foreign Key)
6) Phone Number
7) pass

Transaction Schema

1. Transaction ID(Primary Key)(autogen)
2. Username
3. Amount to be transacted
4. Paid to (Account no)
5. Timestamp
6. Description
7. Transaction Type

Credit Card Schema

1. Credit Card Number
2. Credit ID(Foreign Key)(autogen)
3. Transaction Amount
4. Transaction ID(Primary key)(autogen)
5. Transaction timestamp
6. Available credit

Fraud Alert Schema

1. Account Number(foreign key)
2. Alert Timestamp
3. Alert ID(Primary Key)(autogen)
4. Alert Description

Relationships

1. User and Transactions- One to many
2. User and Credit Card- One to many
3. User and Fraud- One to many
4. Credit Card and transaction-many to many

Swiss Bank Application Frontend

Overview

The frontend of the Swiss Bank application is built using React, providing a dynamic and responsive user interface. The design focuses on a modern and user-friendly experience, with a clean, consistent look across all pages.

Key Features

1. Main Page (MainPage.js)

- **Navigation Hub**: Acts as the central navigation point for the application.
- **Header**: Features the Swiss Bank logo and an account access button.
- **Navigation Bar**: Provides dropdown menus for Credit and Deposit/Withdraw, as well as links to About Us and FAQs pages.
- **Content**: Includes welcome messages and highlights of key banking services.
- **Footer**: Contains additional links for benefits, fraud handling, rules and regulations, and feedback.

2. User Authentication

- **User Context**: Manages user state throughout the application.
- **Login Page**: Allows users to log in to their account.
- **Registration Page**: Enables new users to register for a new account.

3. User Details Page (UserDetails.js)

- **User Information**: Displays detailed user information including account holder name, account type, username, phone number, email, and address.
- **Dynamic Data Fetching**: Uses context and API calls to fetch and display user-specific data.

4. Transaction Management

- **Deposit Page (Deposit.js)**:

    - Provides a form for users to enter deposit details.
    - Utilizes a modern grid layout for a clean and organized form appearance.
- **Withdrawal Page (Withdrawal.js)**:
    - Provides a form for users to enter withdrawal details.
    - Designed with a modern grid layout for ease of use.
- **Transaction History Page (TransactionHistory.js)**:
    - Displays the user's transaction history in a tabular format.
    - Fetches data from the backend to provide up-to-date transaction details.

5. Credit Card Services

- **Issue Credit Card Page (IssueCreditLine.js)**:
    - Provides a form for users to enter credit card issuance details.
    - Includes fields for card number, holder name, expiration date, and CVV.
- **View Credit Card Details Page (ViewCreditCardDetails.js)**:
    - Displays credit card details in an ATM card-like UI and tabular format.
    - Fetches credit card information from the backend.

6. Informational Pages

- **About Us Page (AboutUs.js)**:
    - Provides detailed information about the bank's mission, vision, and values.
    - Neatly organized with headers and sections for easy reading.
- **FAQs Page (FAQs.js)**:
    - Lists frequently asked questions and answers about the bank's services.
    - Organized in a clean and accessible format.

7. Design and Usability

- **Consistent UI**: Maintains a consistent look and feel across all pages with a unified design language.
- **Modern Aesthetics**: Utilizes modern CSS techniques for layouts, hover effects, and animations.
- **Responsive Design**: Ensures the application is usable across various devices and screen sizes.

8. Technologies Used

- **React**: For building the user interface and handling client-side routing.
- **Axios**: For making HTTP requests to the backend services.
- **React Router DOM**: For handling navigation and routing.
- **CSS**: Custom stylesheets for each component to maintain a clean and modern UI.

Swiss Bank Application Backend

Overview

This document provides an overview of the backend services developed for the Swiss Bank application using Spring Boot. The backend comprises three main modules: User Authentication, Transaction Management, and Credit Card Services. MySQL is used as the database for storing and retrieving information.

Modules

1. User Authentication

Purpose: To manage user accounts, including registration and login, with account numbers being auto-generated.

Key Features:

- **User Registration**: Allows new users to create an account.
  - Account number is auto-generated.
  - Input fields include username, password, email, phone number, and address.
- **User Login**: Allows existing users to log in using their username and password.

2. Transaction Management

Purpose: To handle money transactions, including deposits and withdrawals.

Key Features:

- **Deposit Money**: Users can deposit money into their account.
- **Withdraw Money**: Users can withdraw money from their account.

Endpoints:

- POST /api/transactions/deposit: Deposits money into the user's account.
- POST /api/transactions/withdraw: Withdraws money from the user's account.
- GET /api/transactions/history: Retrieves the transaction history for the user.

Database:

- **Transactions Table**: Stores transaction details including transaction ID, account number, type (deposit/withdraw), amount, and date.

Credit Card Services

Purpose: To manage credit card issuance and display credit card details.

Key Features:

- **Issue Credit Card**: Users can apply for a new credit card.
- **View Credit Card Details**: Users can view their credit card details.

Endpoints:

- POST /api/credit-cards/issue: Issues a new credit card to the user.
- GET /api/credit-cards/details: Retrieves the credit card details for the user.

Database:

- **CreditCards Table**: Stores credit card information including card number, account number, card holder name, expiration date, and CVV.

Database Configuration

MySQL Database: MySQL is used to store all user, transaction, and credit card data.

Security

JWT Authentication: The application uses JWT tokens for securing endpoints and managing user sessions.

Conclusion

The backend of the Swiss Bank application is designed to handle user authentication, manage transactions, and provide credit card services efficiently. It leverages Spring Boot for building the RESTful services and MySQL for the persistent storage of data.

Swiss Bank Application: Individual Contributions Documentation

Aayush Krishna

Role: Backend Developer **Module**: User Authentication

- **Responsibilities**:
  - Designed and implemented the user authentication module.
  - Developed endpoints for user registration and login.
  - Implemented auto-generation of account numbers during user registration.
  - Ensured secure password hashing and storage.
  - Backend endpoints security
- **Key Achievements**:
  - Successfully established a secure and efficient user authentication process.
  - Enhanced the user experience by streamlining the registration and login procedures.

Sandeep Sahoo

Role: Backend Developer **Module**: Transaction Management

- **Responsibilities**:
  - Developed the transaction management module.
  - Created endpoints for depositing and withdrawing money.
  - Implemented functionality to log transaction details in the database.
  - Developed endpoints for retrieving transaction history.
  - Ensured the security and reliability of transactions.
- **Key Achievements**:
  - Effectively managed the deposit and withdrawal processes with clear, concise endpoints.
  - Provided transparency for users by implementing a comprehensive transaction history feature.

Sai Nikhil Kanchumarthi

Role: Backend Developer **Module**: Credit Card Services

- **Responsibilities**:
  - Developed the credit card services module.
  - Created endpoints for issuing new credit cards.
  - Implemented functionality to store and retrieve credit card details.
  - Ensured secure storage and handling of credit card information.
- **Key Achievements**:
  - Successfully set up the credit card issuance process with robust security measures.
  - Enhanced user experience by providing detailed credit card information on demand.

Pragyan Das

Role: Frontend Developer **Module**: React Modules

- **Responsibilities**:
  - Designed and implemented the frontend using React.
  - Developed components for the main page, user details, transaction management, and credit card services.
  - Integrated React Router for navigation between different pages.
  - Utilized Context API for state management across the application.
  - Ensured a consistent and modern UI/UX design across all pages.
- **Key Achievements**:
  - Created a user-friendly and responsive interface for the application.
  - Successfully integrated frontend with backend services, ensuring seamless data flow and interaction.

Conclusion

Each team member significantly contributed to the development of the Swiss Bank application. Their individual efforts in user authentication, transaction management, credit card services, and frontend development ensured a robust, secure, and user-friendly banking application.

Github Link: https://github.com/sandeepsahoo542/CapstoneProjectBackend