PROJECT TITLE – Capstone Project

Banking Application

SUBMITTED BY:

KISHORE KUMAR R

TRAINEE (T&T EAD)

Kishorekr.ext@deloitte.com

Date: November 20, 2024

Introduction:

The Banking Application is an advanced, user-focused application made to handle and simplify essential banking services. This program provides a complete solution designed to satisfy the requirements of both users and administrators, including everything from account setup to transaction processing and balance management.

The application, which was created as a capstone project, uses contemporary technology to retain a highly scalable and secure backend architecture while offering a smooth and user-friendly experience. Its modular design guarantees strong integration capabilities, ease of maintenance, and flexibility for future improvements.

This application is a dependable tool for efficiently managing administrative and personal banking operations because it concentrates on essential banking features, showcasing the ideal fusion of state-of-the-art technology and useful financial solutions.

Objectives:

- To design and develop a scalable banking application using **Spring Boot** (backend), **ReactJS** (frontend).
- To provide a modular structure with **at least four core modules** including user authentication.
- To demonstrate backend and frontend integration using RESTful APIs.
- To ensure data integrity with **MySQL database** and to document the application's architecture, features, and endpoints.

Technologies Used:

- **Frontend:** ReactJS, Bootstrap, Axios
- **Backend:** Spring Boot, REST APIs
- **Database:** MySQL
- **Version Control:** GitHub

CASE STUDY

Title:

Features of the Application:

1. **User-Authentication**: It ensures that users can securely access their accounts. This module includes secure registration and login.
    - User Registration: Collects user details like name, email, password, and other relevant information. Implements validation to ensure the data is accurate and secure.
    - Login System: Authenticates users using email and password and uses hashing for password security.
    - Session Management: Allows users to stay logged in securely during their session.
    - Includes logout functionality.
    - Key Benefits:
        - Protects user data with robust security protocols.
        - Provides a seamless and user-friendly login experience.
        - Prevents unauthorized access with proper authentication mechanisms.

2. **Admin Management**: The Admin Management module allows administrators to oversee, manage, and maintain the system. It provides them with the tools to monitor users, transactions, and system activities.
    - User Monitoring: View and manage all registered users and deactivate accounts if necessary.
    - Transaction Oversight: Monitor all user transactions for security and compliance.
    - Key Benefits:
        - Ensures system integrity and smooth operation.
        - Allows proactive monitoring to prevent fraud or misuse.
        - Enhances control over user and transaction data.

3. **User Account Management**: The User Account Management module lets users view and manage their account details. It enhances user autonomy and ensures the accuracy of stored information.
    - Profile Management: View and update personal details like name, email, phone number, and address.
    - Account Summary: Display account details such as account number, balance, and account type.
    - Account Settings: Change password or configure preferences and view activity logs to track recent actions.
    - Key Benefits
        - Empowers users to maintain their account data.
        - Ensures accurate and up-to-date information for better service.
        - Provides transparency and control over personal data.

4. **Transaction Management**: The Transaction Management module enables users to manage their financial activities, including deposits, withdrawals, and transaction history.
    - Transaction Processing: Supports real-time deposits and withdrawals and categorizes transactions as credit or debit.
    - Transaction History: Displays a detailed history of all transactions, including date, amount, and type.
    - Includes search and filter options for easier navigation.
    - Balance Overview: Real-time display of account balance based on transaction activity.
    - Key Benefits
        - Provides a clear and concise view of financial activity.
        - Ensures accurate balance calculation for better financial planning.
        - Offers transparency and accountability for all transactions.
        - How These Modules Integrate

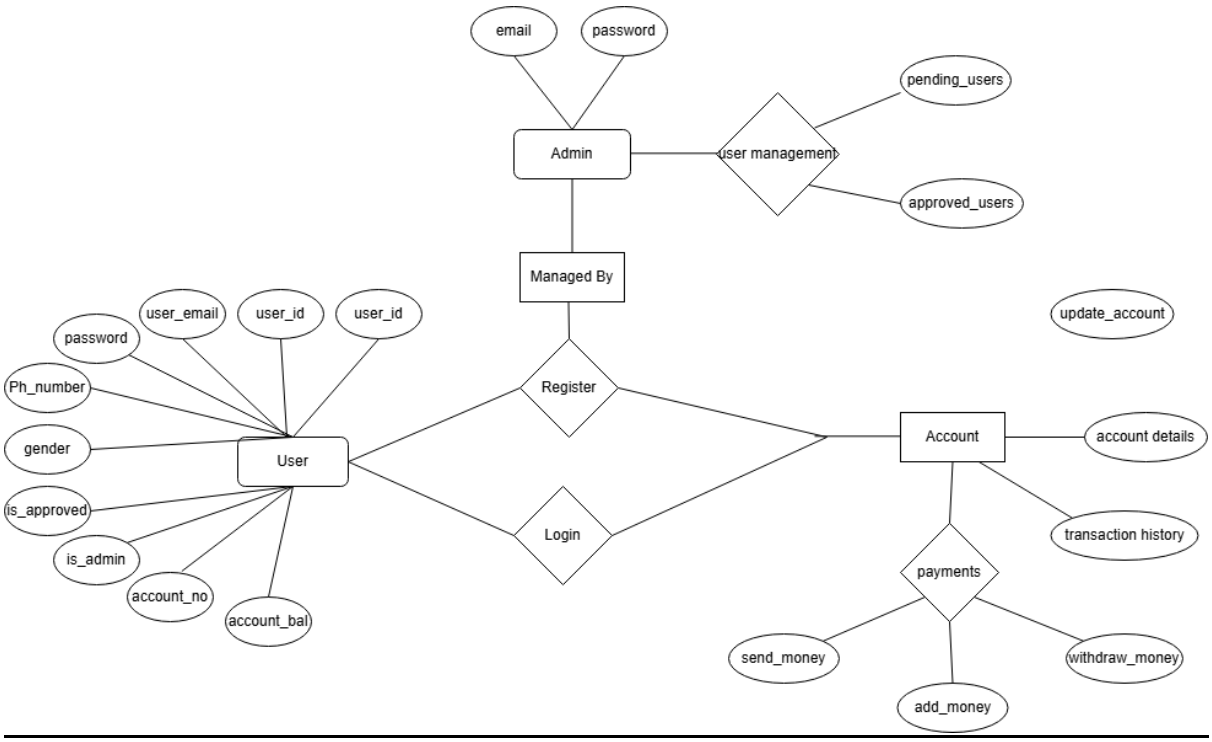
Implemented Backend Endpoints:

Endpoint URL	HTTP Method	Description
/user/register	POST	Registers a new user by accepting user details in the request body.
/user/login	POST	Logs in a user by verifying their credentials (email and password) and approval status.
/user/dashboard	GET	Fetches the dashboard details for a user (excluding admins), showing username and balance.
/user/account-details	GET	Fetches detailed account information of a user (email, name, phone, account balance).
/user/update	PUT	Updates the details of an existing user (name, phone number).
/user/delete	DELETE	Deletes the user based on their email.
/user/payments/add-money	POST	Adds money to the user's account, updates balance, and records the transaction.
/user/payments/send-money	POST	Sends money from one user to another, updates both users' balances, and records the transaction.
/user/payments/withdraw-money	POST	Allows a user to withdraw money from their account, updates the balance, and records the transaction.
/user/transactions/history	GET	Fetches the transaction history for a user, based on their email.

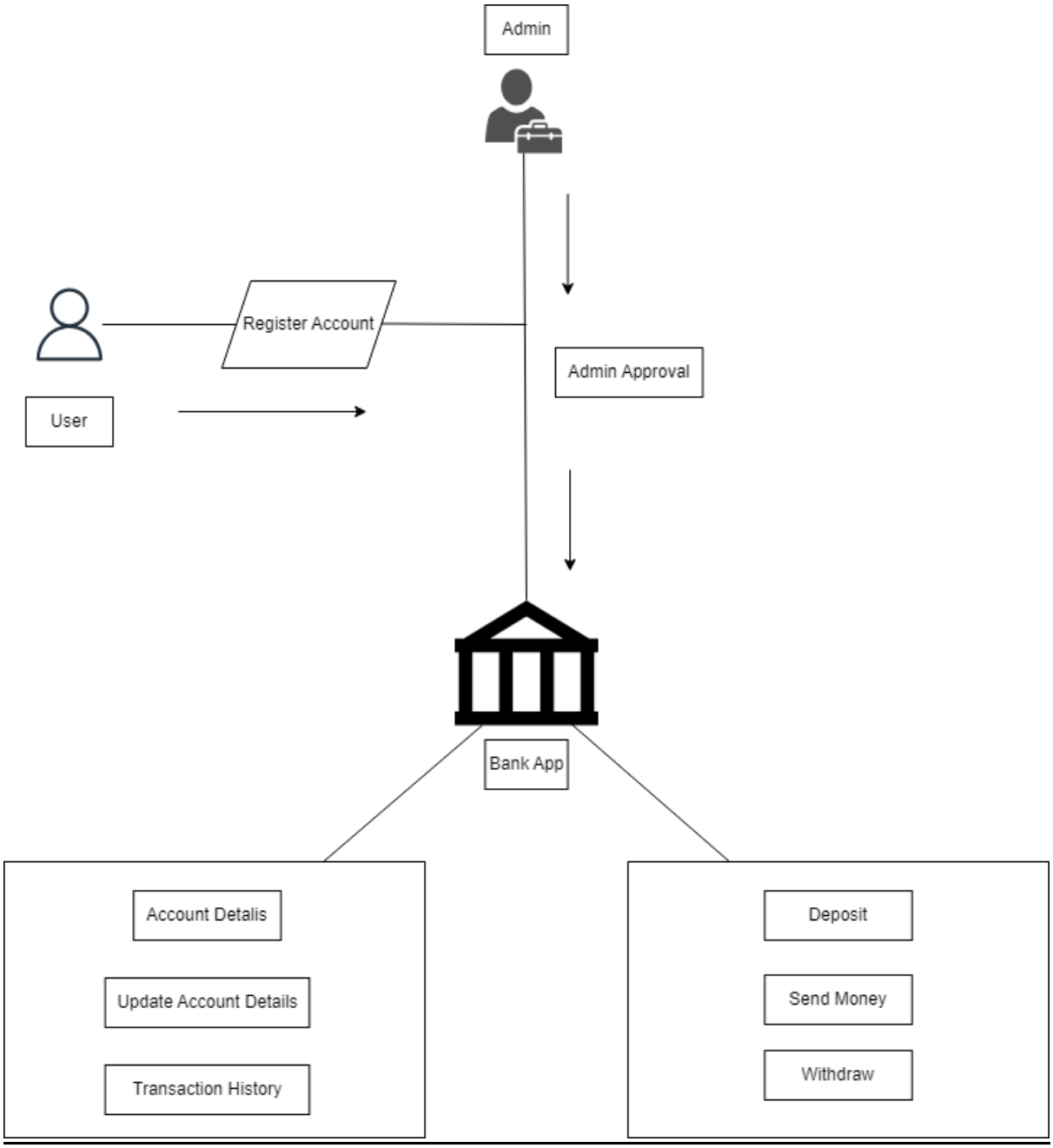
Frontend URLs and Pages:

Action	Frontend URL	Description
Home	/	Homepage for all users
Sign In	/user/login	User login page
Admin Login	/admin/login	Admin login page
Approved Users (Admin)	/admin/dashboard	Admin dashboard for managing approved users
Approved Users (Admin)	/admin/user-management	Admin user management page for viewing approved users
Pending Users (Admin)	/admin/dashboard	Admin dashboard showing pending users when clicked
Sign Out	/	Logs the admin or user out and redirects to homepage

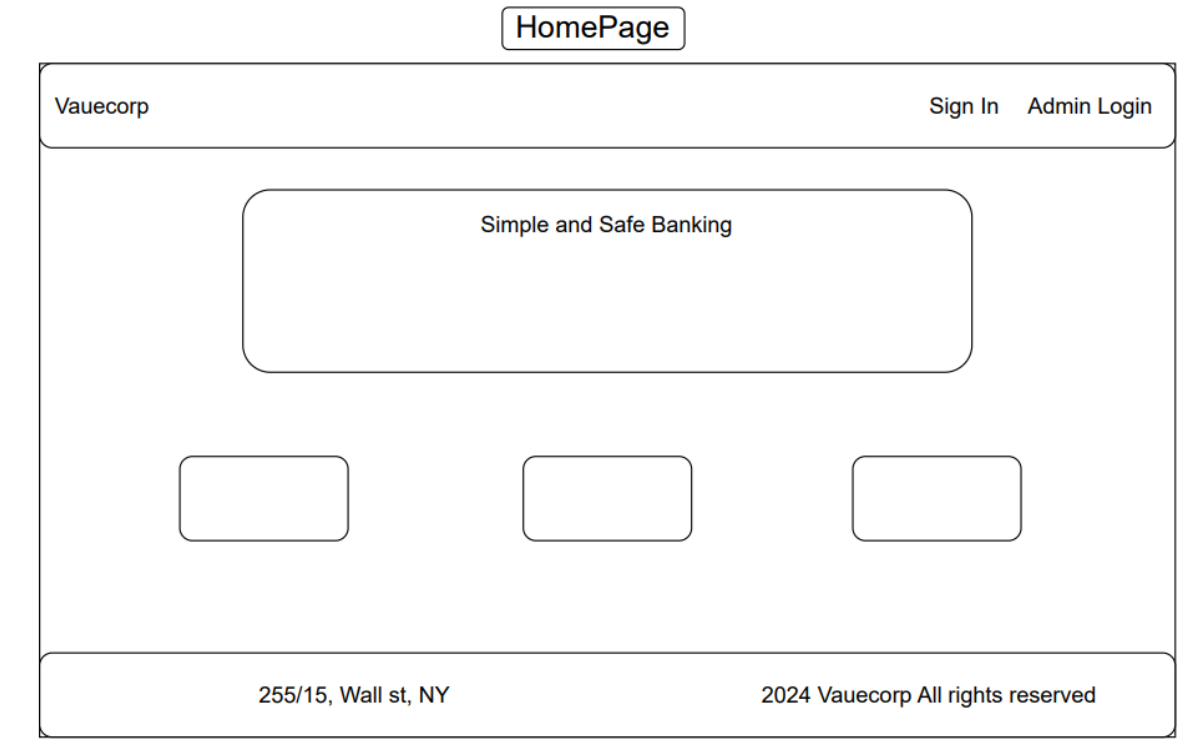
ER Diagram:

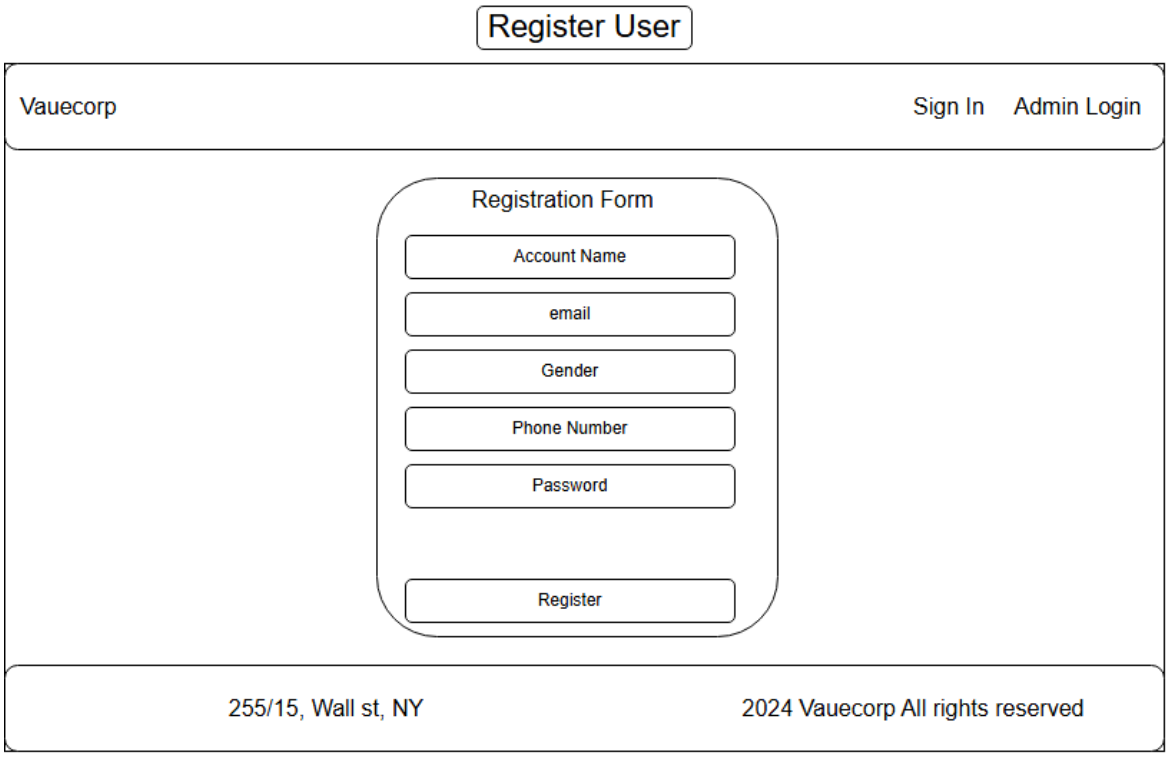


Flow Diagram:

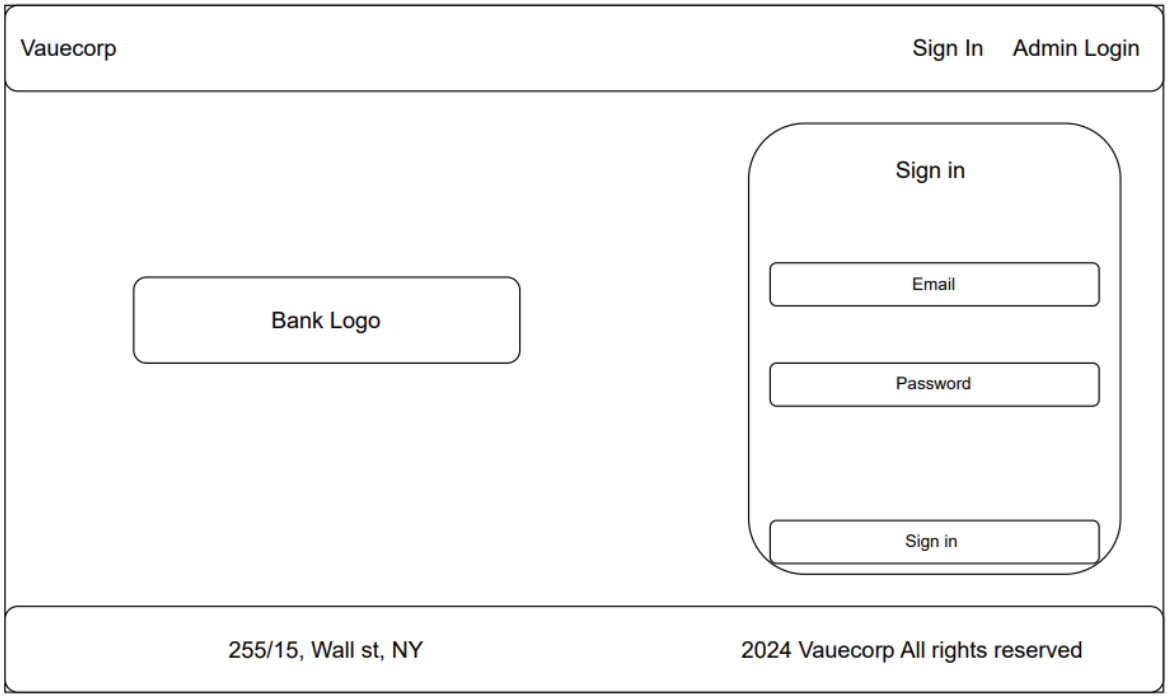


Wire framing:

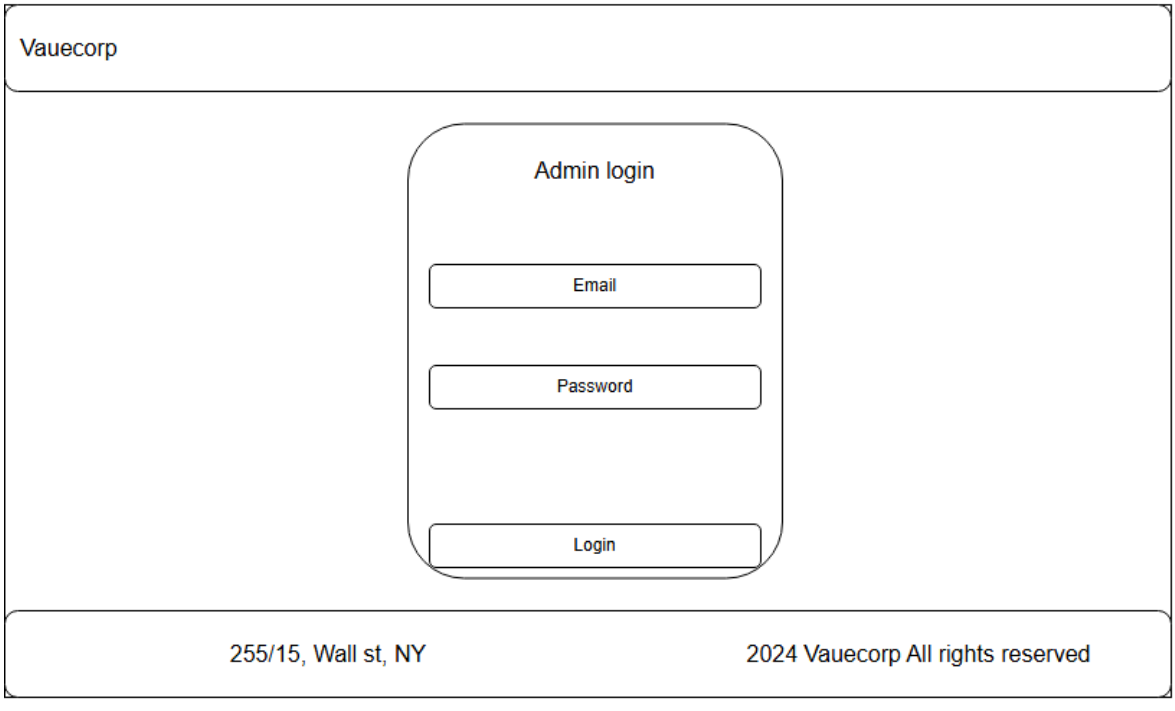




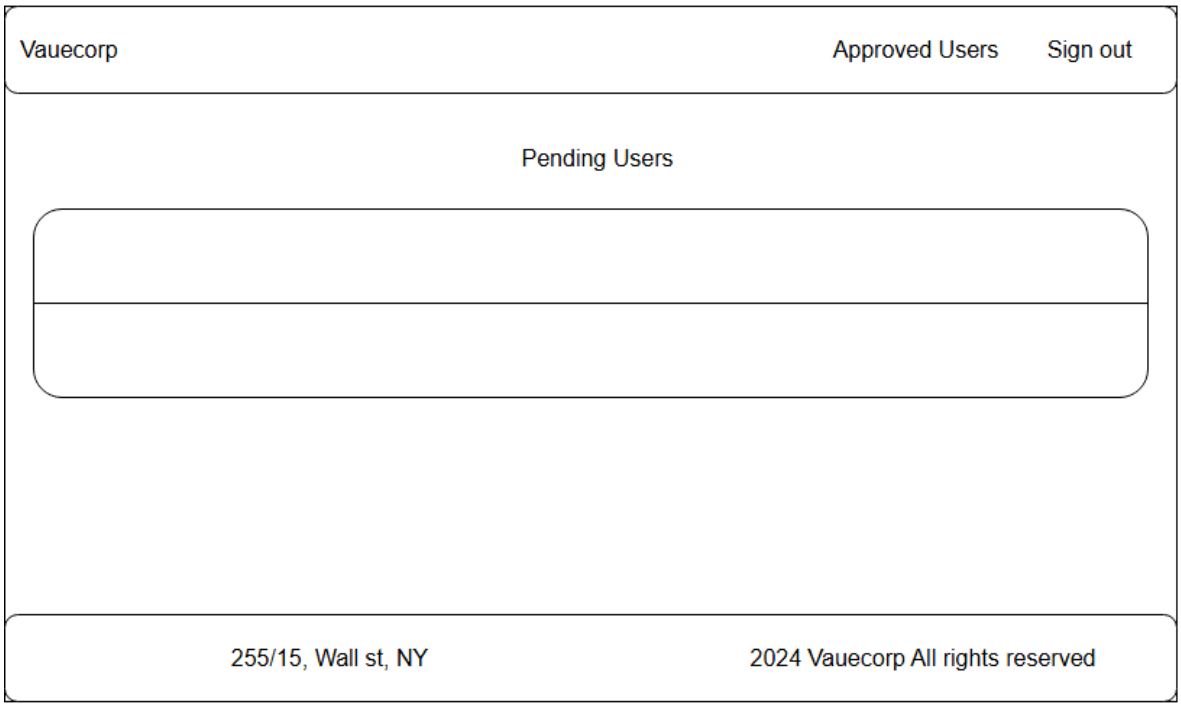
User Sign in



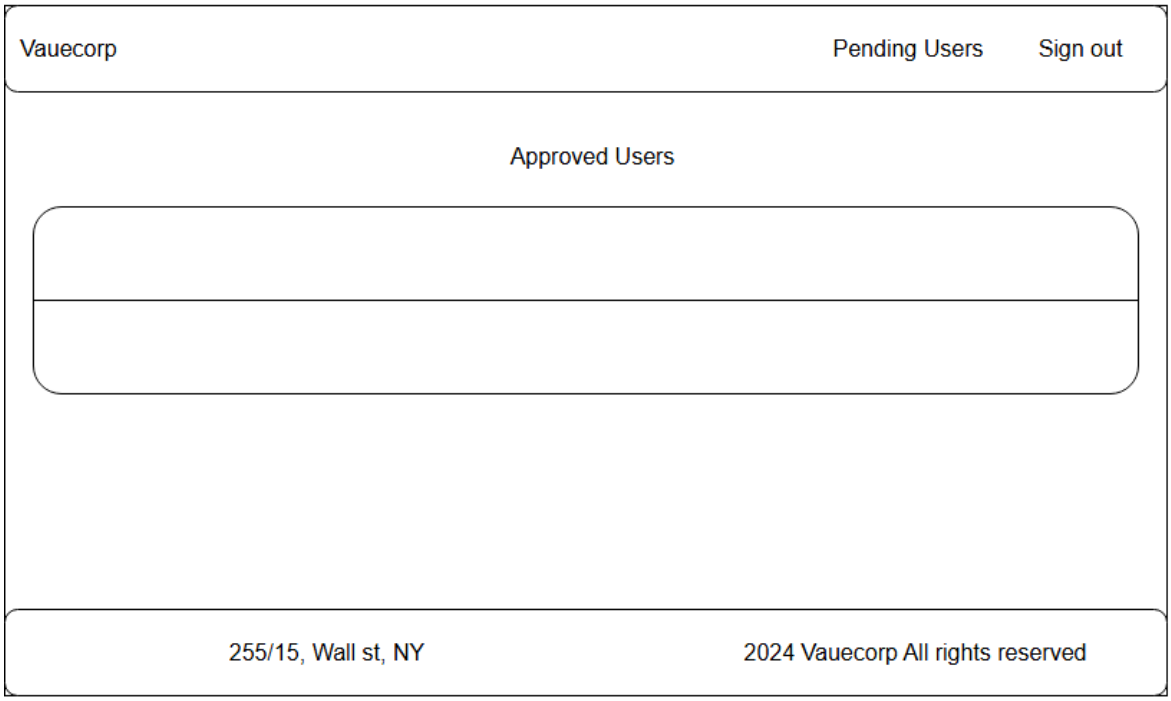
Admin Login



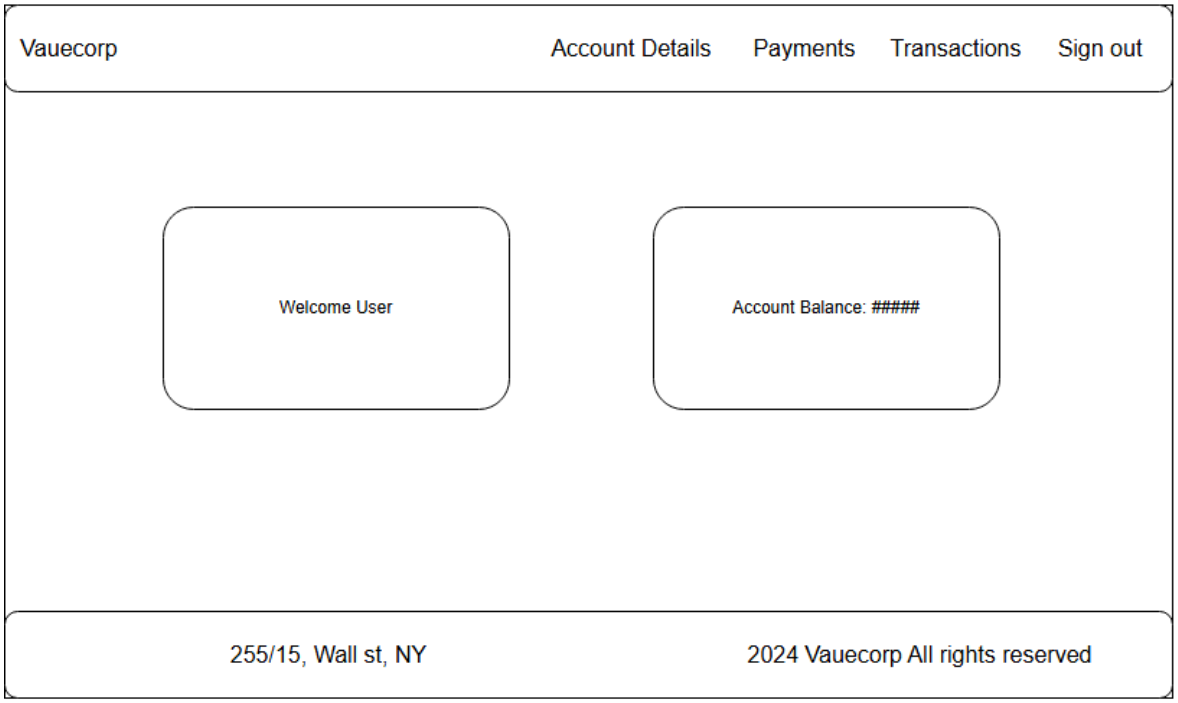
Admin Dashboard



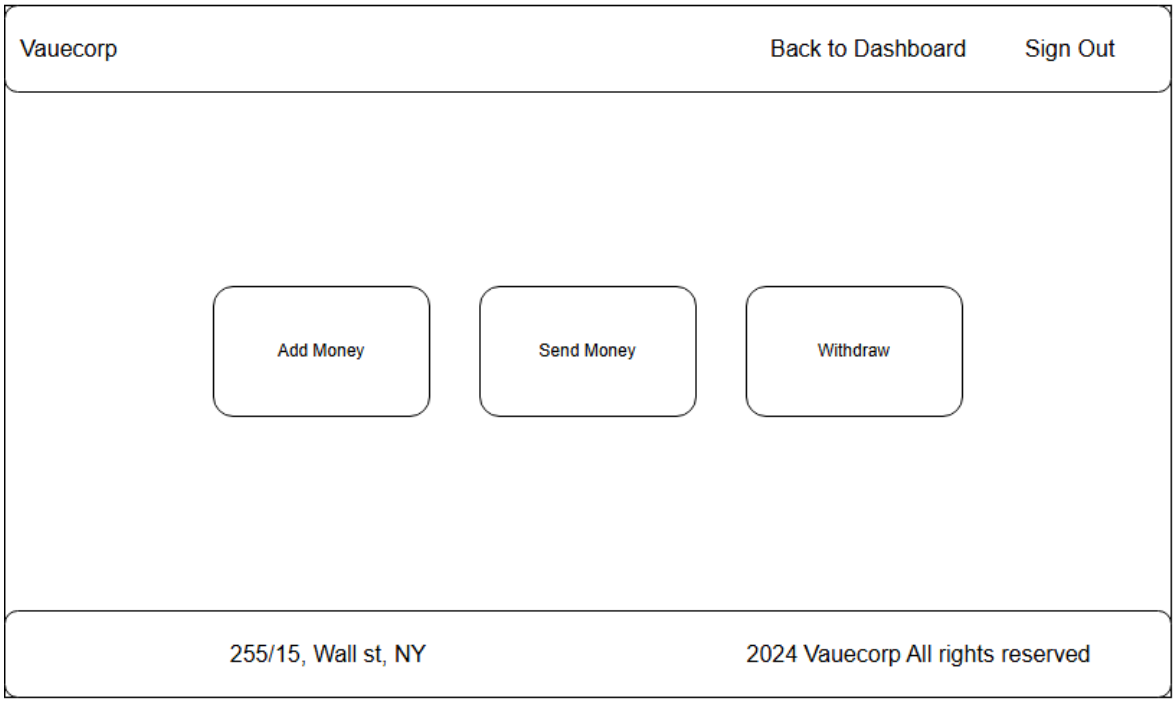
Admin Dashboard

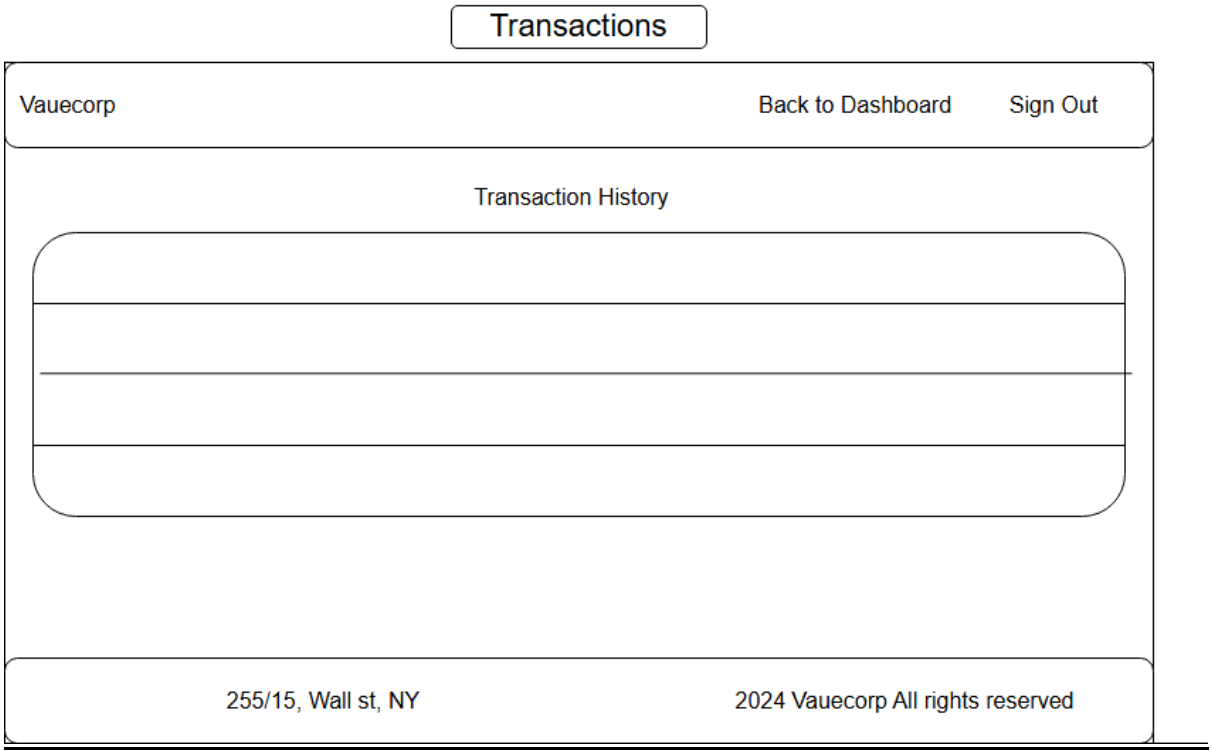


User Dashboard



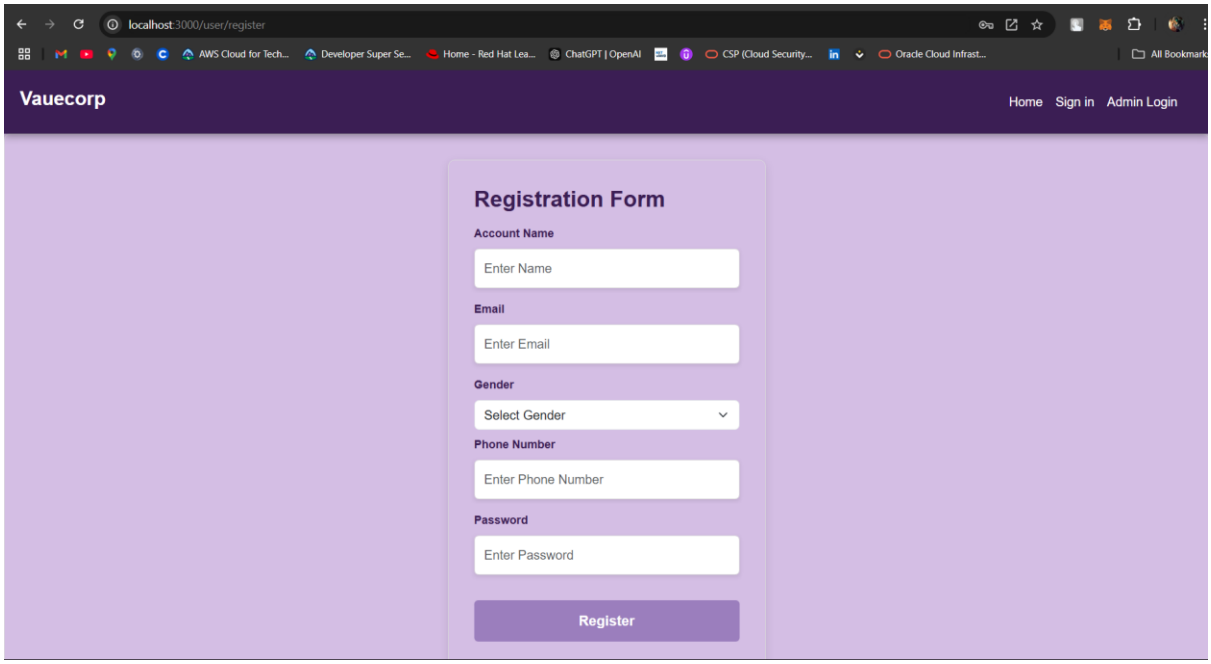
Payments



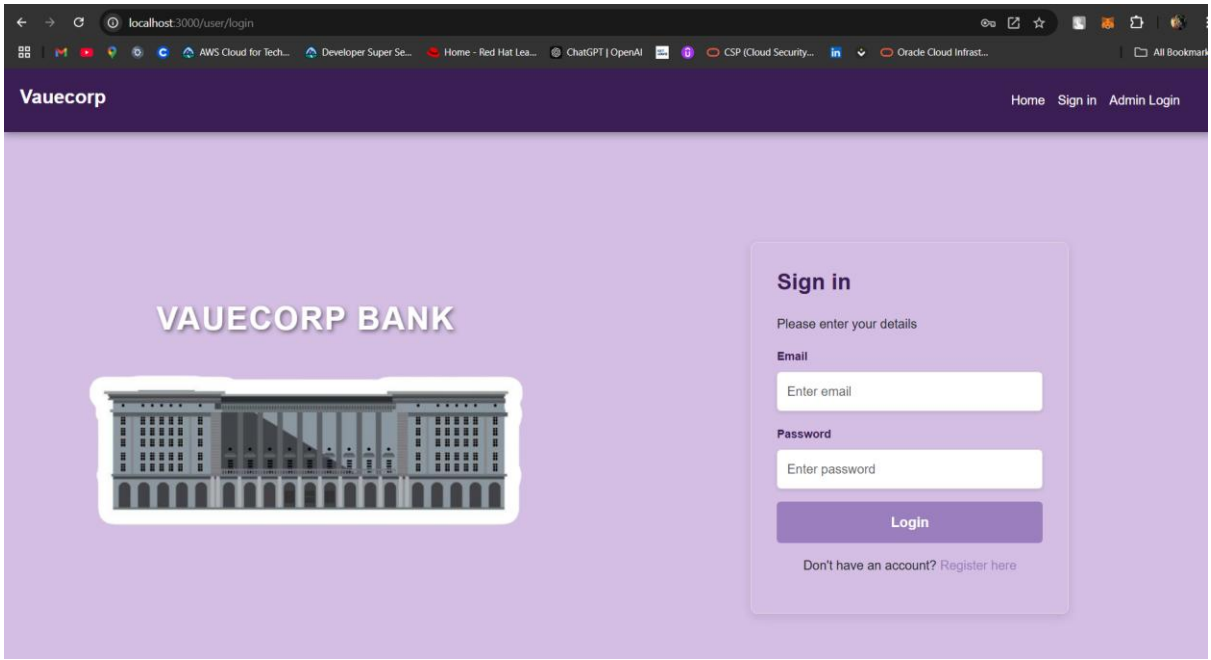

Transactions
Vauecorp
Back to Dashboard
Sign Out
Transaction History
255/15, Wall st, NY
2024 Vauecorp All rights reserved

Screenshots:

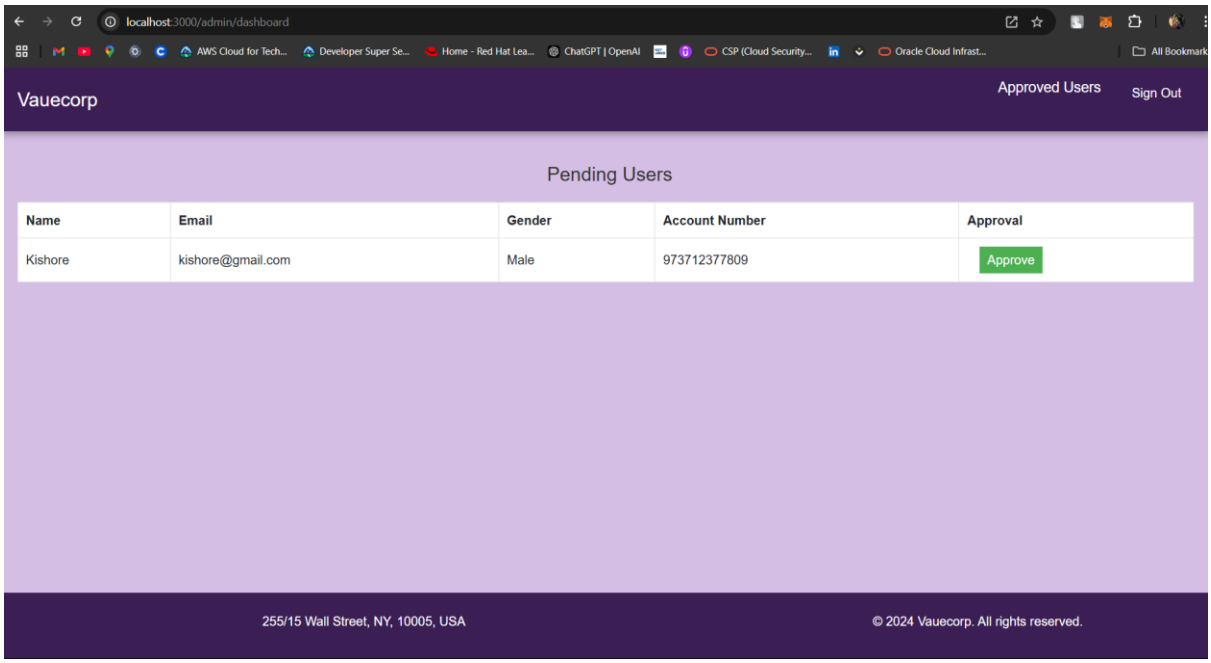
/user/register



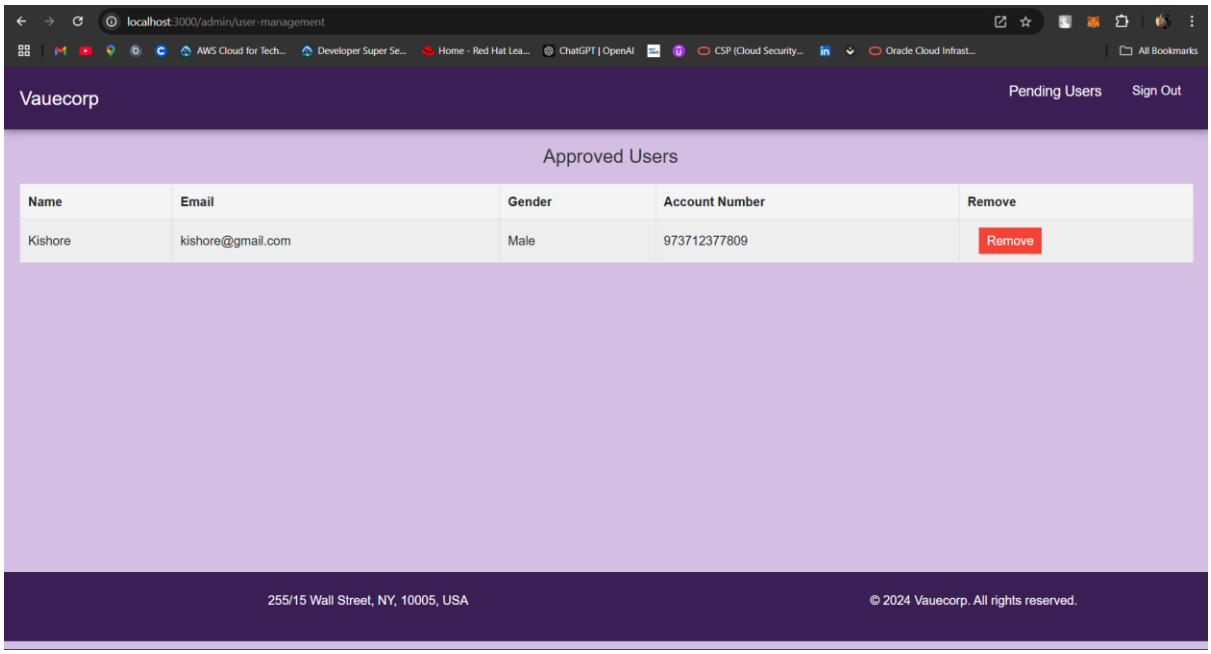
/user/login



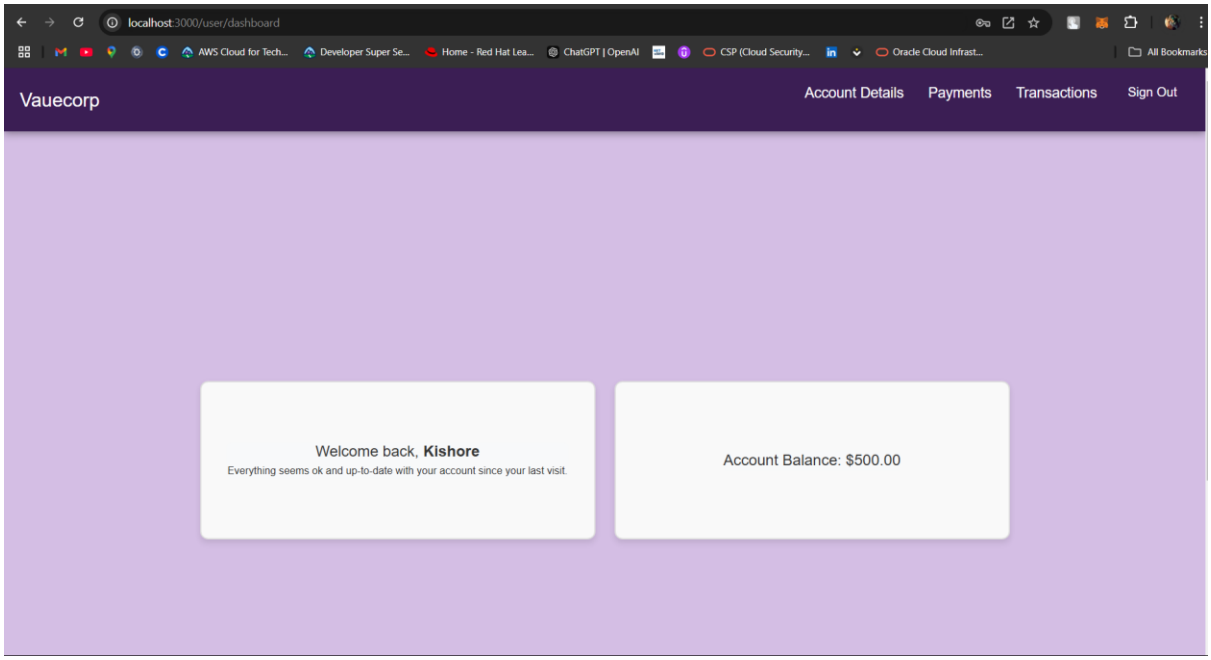
/admin/dashboard (Newly registered users will be approved by admin)



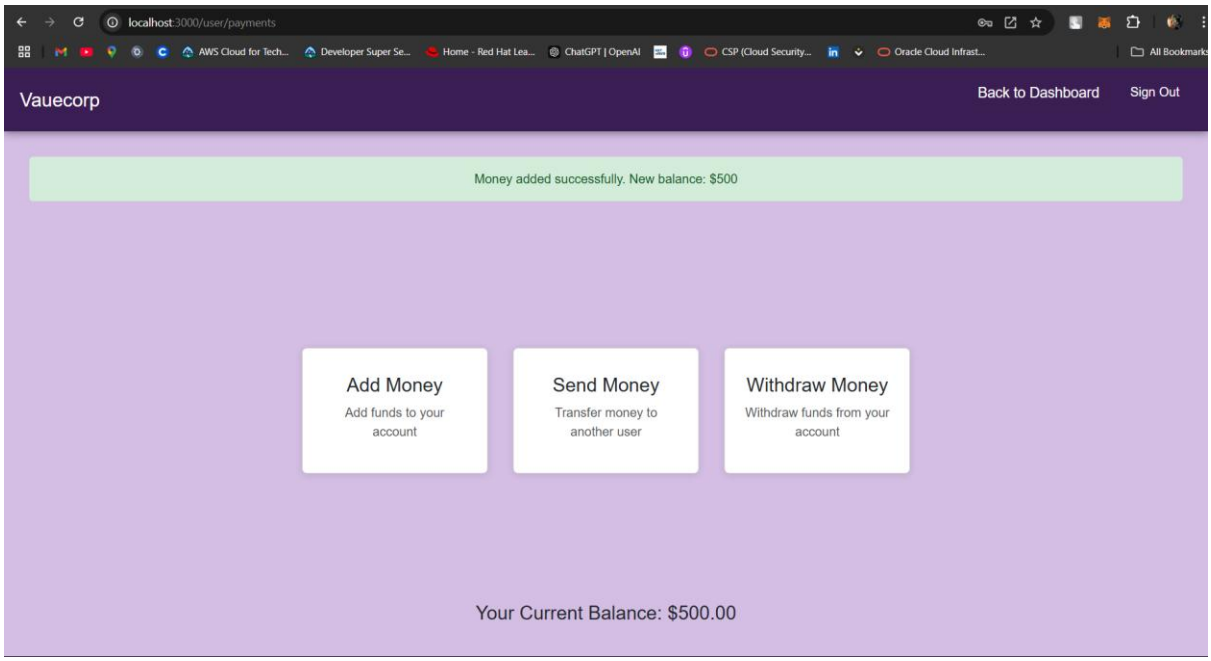
/admin/user-management (Admin have the right to delete the user)



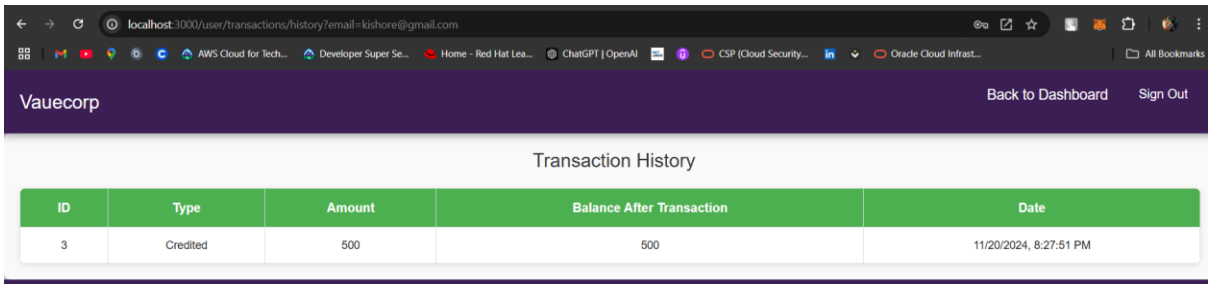
/user/dashboard



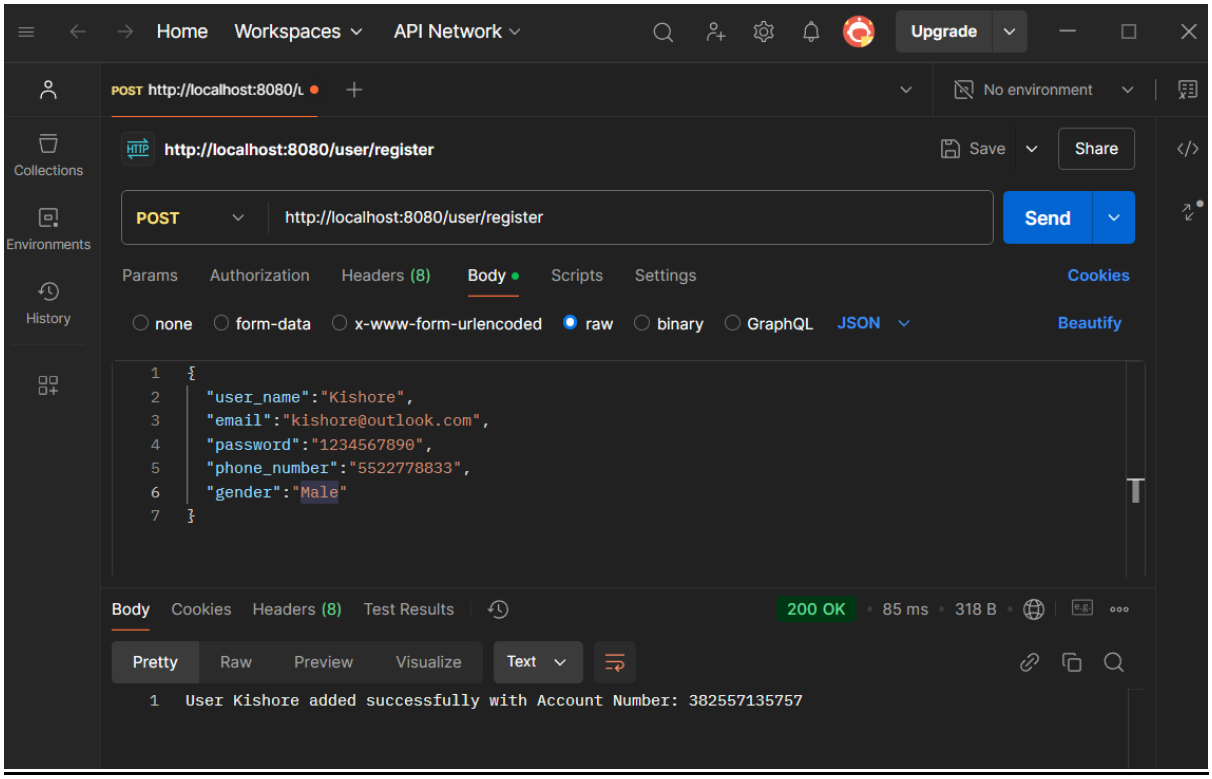
/user/payments

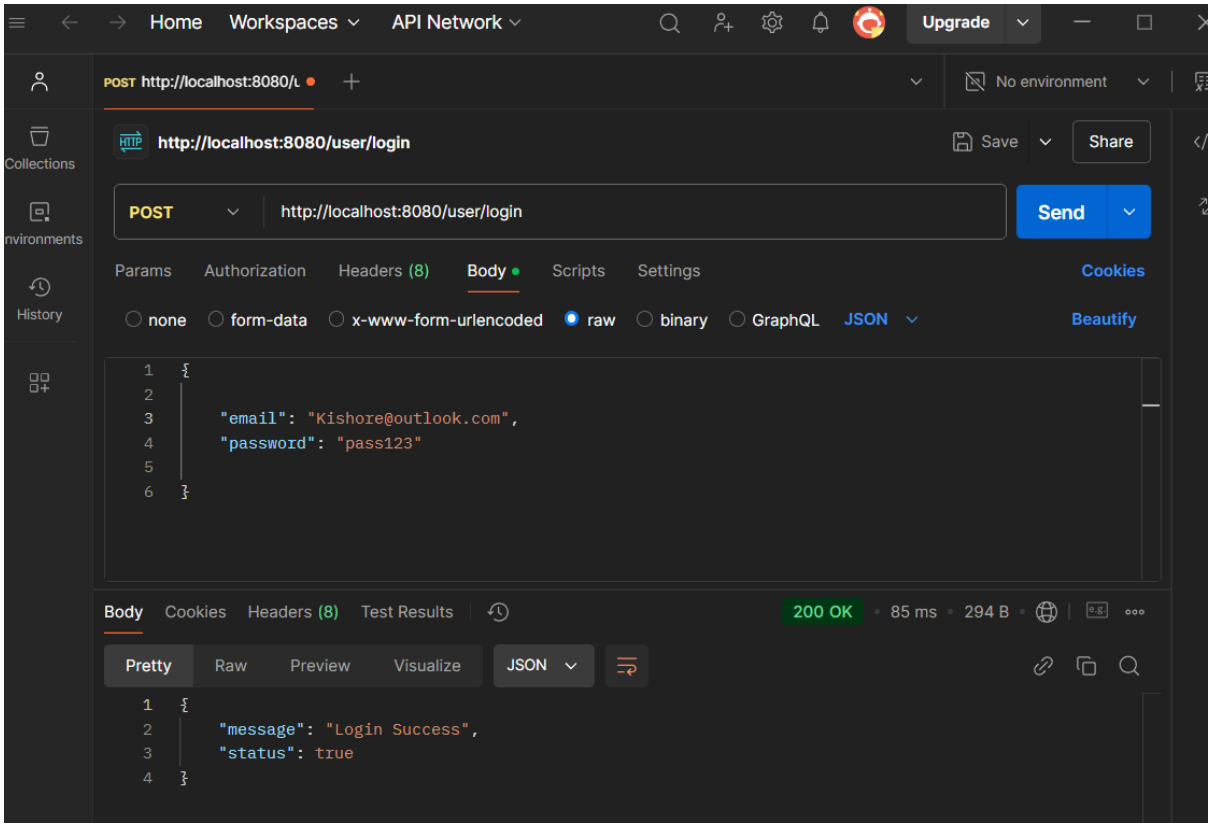


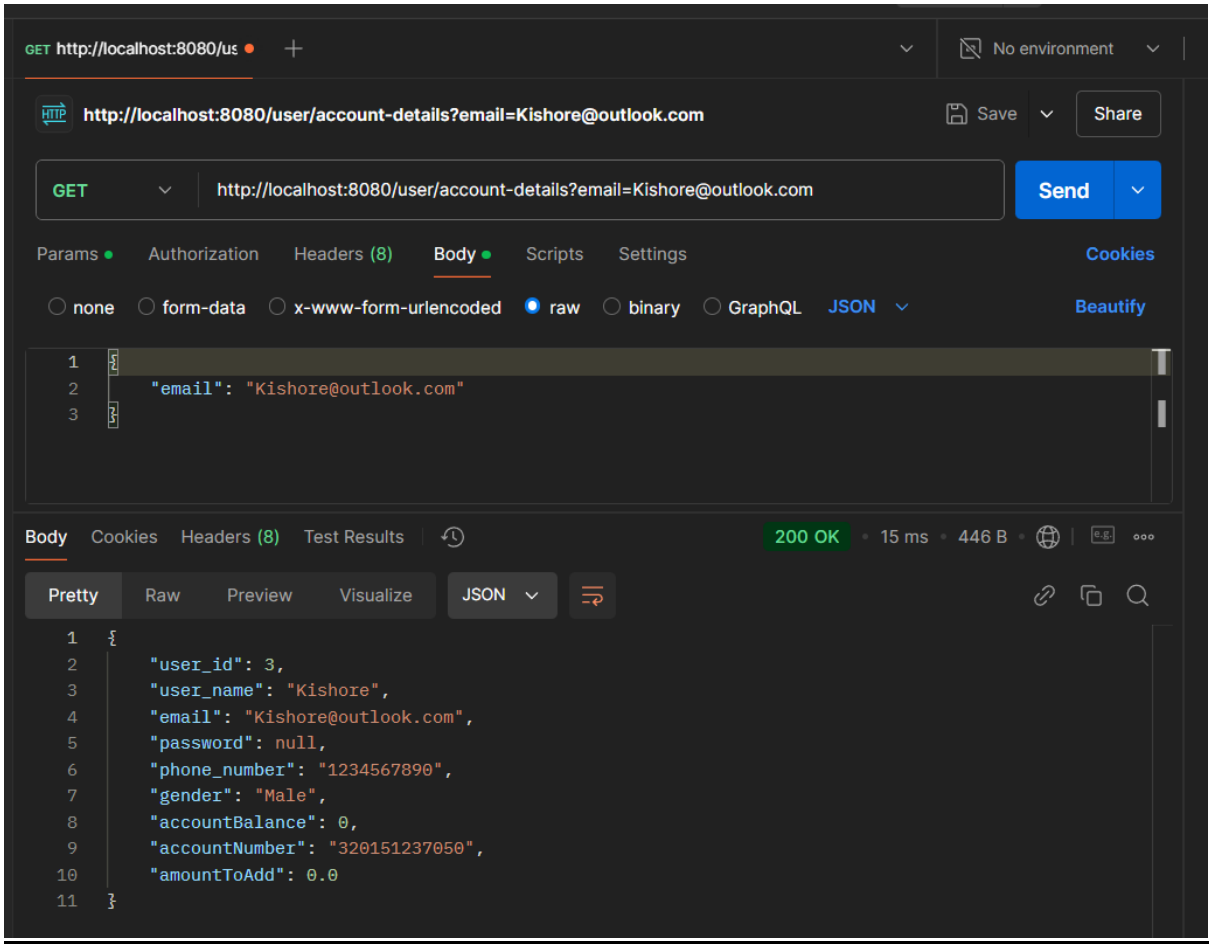
/user/transactions/history

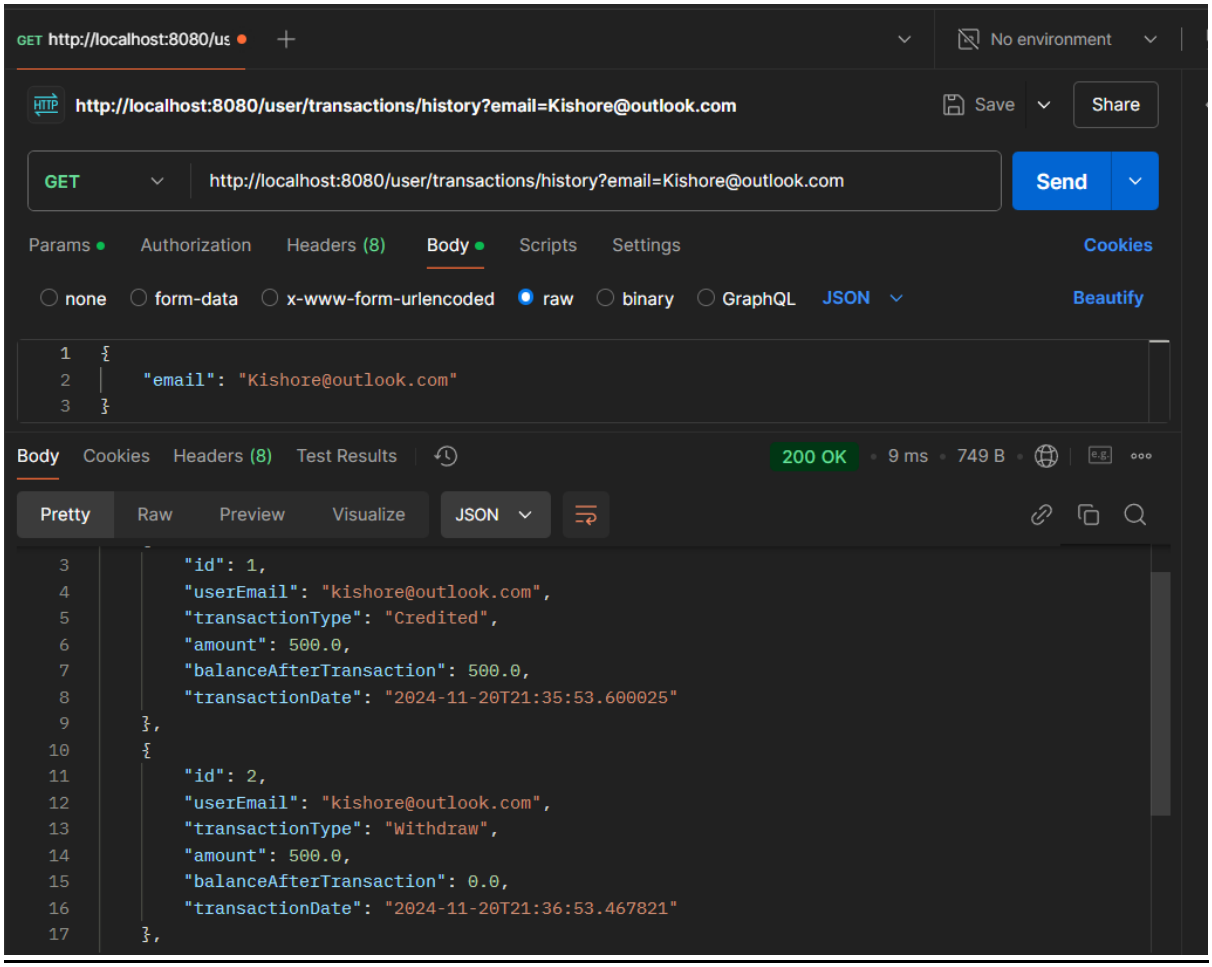


Endpoints:




Home
Workspaces
API Network
Upgrade
POST http://localhost:8080/l
No environment
Collections
Environments
History
http://localhost:8080/user/login
Save
Share
POST
http://localhost:8080/user/login
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "Kishore@outlook.com",
"password": "pass123"
}
Body
Cookies
Headers (8)
Test Results
200 OK
85 ms
294 B
Pretty
Raw
Preview
Visualize
JSON
{
"message": "Login Success",
"status": true
}


GET http://localhost:8080/us
No environment
http://localhost:8080/user/account-details?email=Kishore@outlook.com
Save
Share
GET
http://localhost:8080/user/account-details?email=Kishore@outlook.com
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
"email": "Kishore@outlook.com"
Body
Cookies
Headers (8)
Test Results
200 OK
15 ms
446 B
Pretty
Raw
Preview
Visualize
JSON
{
"user_id": 3,
"user_name": "Kishore",
"email": "Kishore@outlook.com",
"password": null,
"phone_number": "1234567890",
"gender": "Male",
"accountBalance": 0,
"accountNumber": "320151237050",
"amountToAdd": 0.0
}


GET http://localhost:8080/us
No environment
http://localhost:8080/user/transactions/history?email=Kishore@outlook.com
Save
Share
GET
http://localhost:8080/user/transactions/history?email=Kishore@outlook.com
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"email": "Kishore@outlook.com"
}
Body
Cookies
Headers (8)
Test Results
200 OK
9 ms
749 B
Pretty
Raw
Preview
Visualize
JSON
"id": 1,
"userEmail": "kishore@outlook.com",
"transactionType": "Credited",
"amount": 500.0,
"balanceAfterTransaction": 500.0,
"transactionDate": "2024-11-20T21:35:53.600025"
},
{
"id": 2,
"userEmail": "kishore@outlook.com",
"transactionType": "Withdraw",
"amount": 500.0,
"balanceAfterTransaction": 0.0,
"transactionDate": "2024-11-20T21:36:53.467821"
},