GROUP 03

Real Estate Management System Project Report

1. Introduction

- ✧ The Real Estate Management System is a web-based application designed to simplify the process of managing properties, clients, and users for real estate businesses.
- ✧ It features role-based dashboards for different users, including Admins, Agents, and Clients, offering customized functionalities for each role.
- ✧ The system is built using React.js for the frontend and integrates with a backend that handles authentication, user management, property management, and more.

2. Objectives

- ✧ To create a user-friendly interface for managing real estate properties, clients, and users.
- ✧ To implement a secure and efficient authentication system for different user roles (Admin, Agent, Client).
- ✧ To facilitate property transactions and provide clients with easy access to property details.
- ✧ To provide an advanced and responsive design using CSS.

- ✧ To ensure robust data validation and secure data handling.

3. System Features

3.1. Home Page

- ✧ **Login and Register Buttons:** Provides entry points for users to log in or register based on their role.
- ✧ **Background Image:** A visually appealing background image to enhance the user experience.
- ✧ **Featured Properties Section:** Showcases selected properties to attract user interest.
- ✧ **Navigation Bar:** Contains items like Admin, Agent, Client, Help, and Notification for easy access.

3.2. User Authentication

- ✧ **Login:** Users (Admin, Agent, Client) can log in using their email and password.
- ✧ **Registration:** New users can register by providing their name, mobile number, email, password, and selecting their role (Admin, Agent, Client).
- ✧ **Role-Based Redirection:** After login, users are redirected to their respective dashboards based on their roles.

3.3. Admin Dashboard

- ✧ **User Management**: Admins can view, add, edit, and delete users (Agents and Clients).
- ✧ **Search Functionality:** A search bar to find user details by ID.
- ✧ **Count Display:** Displays updated counts of agents and clients.
- ✧ **Advanced Validation**: Forms have validation for name (10-15 characters), email (valid format), and phone number (exactly 10 digits).

3.4. Agent Dashboard

- ✧ **Property Management:** Agents can add, edit, and delete properties. Includes image handling, description, and bedroom type (1BHK, 2BHK, 3BHK).
- ✧ **Client Management:** Agents can add new clients.
- ✧ **Form Validation:** Advanced validation that shows errors only for the field currently being interacted with.
- ✧ **Property Display:** After adding a property, it appears as a card type on the Client Dashboard.

3.5. Client Dashboard

- ✧ **Property Display:** Properties are displayed as cards. Clients can filter properties using a toggle bar.
- ✧ **Property Details:** Clicking on a property card shows full information about the property.
- ✧ **'Buy Now' Feature:** Allows clients to purchase properties, displaying transaction details after purchase.
- ✧ **User Profile Management:** Clients can edit their profile, including name, email, phone number, and password. Advanced validation is applied for these fields.

3.6. Help Component

- ✧ Sidebar Chat Interface: Provides a chat interface with FAQs and an input field for user queries, responding like a conversation.

4. Technologies Used

- ✓ **Frontend:** React.js, HTML, CSS, JavaScript
- ✓ **CSS:** Advanced styling to enhance the user experience across different components.
- ✓ **React Components:** Modular components for Admin, Agent, Client, and other functionalities.
- ✓ **Backend:** Spring boot

  Combines both registration and login functionalities.

  Handles CRUD operations for users and properties.
- ✓ **Database:** Mysql

  Stores user details, property information, and transaction details.
- ✓ **API Integration:** Axios

Used to integrate the frontend with the backend for handling registration, login, and data retrieval.

5. System Architecture

- ✓ **Client-Server Architecture:** The application follows a client-server model where the React frontend interacts with the Node.js backend API.
- ✓ **Role-Based Access Control:** The system restricts access to certain functionalities based on the user's role (Admin, Agent, Client).

6. Team Contributions

TM 1: **Manjula C (**Frontend & Backend)

- ✓ Designed and implemented the Home Page, including the Login and Register components.
- ✓ Developed the Admin Dashboard, focusing on user management functionalities.

- ✓ Implemented advanced CSS for the Login and Register forms to enhance user experience.
- ✓ Integrated form validation rules across the application.

T M2 : **Srividya** (Frontend & backend)

- ✓ Built the backend using springboot in eclipse
- ✓ Developed APIs for user authentication, registration, and role-based redirection.
- ✓ Implemented CRUD operations for managing users and properties.
- ✓ Integrated the backend with Mysql for data storage and retrieval.
- ✓ Added features for handling property images, descriptions, and bedroom types.

TM 3: **Ishika Kumari** (Agent Dashboard)

- ✓ Developed the Agent Dashboard, focusing on property and client management functionalities.
- ✓ Implemented advanced form validation that shows errors only for the currently interacted field.
- ✓ Worked on displaying added properties as cards on the Client Dashboard.

TM 4: **Gadapa chandrasekar**

- ✓ Client Dashboard and Help Component Developer
- ✓ Implemented the 'Buy Now' feature for property transactions, displaying transaction details post-purchase.
- ✓ Developed the User Profile Management section with advanced validation for profile updates.
- ✓ Created the Help component with a sidebar chat interface and FAQ functionality.

7. Implementation Details

7.1. Form Validation

1. Name: Required, 10-20 characters.
2. Email: Required, valid format, must contain '@'.
3. Phone Number: Required, exactly 10 digits.
4. Password: Required, 8-15 characters, must include at least one uppercase letter and one special character.

7.2. Property and Client Management

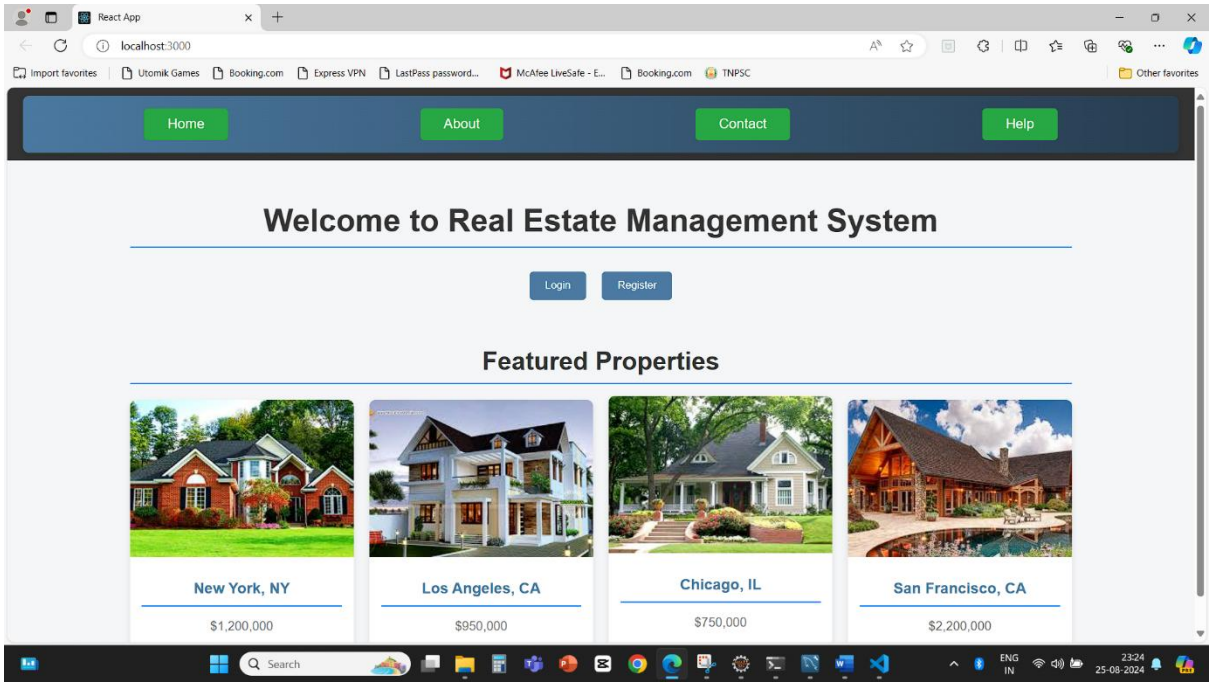
5. Property forms include fields for images, descriptions, and bedroom types.
6. Clients can view property cards with filters for easy navigation.
7. Property transactions are handled securely with transaction details displayed post-purchase.

8. Security Measures

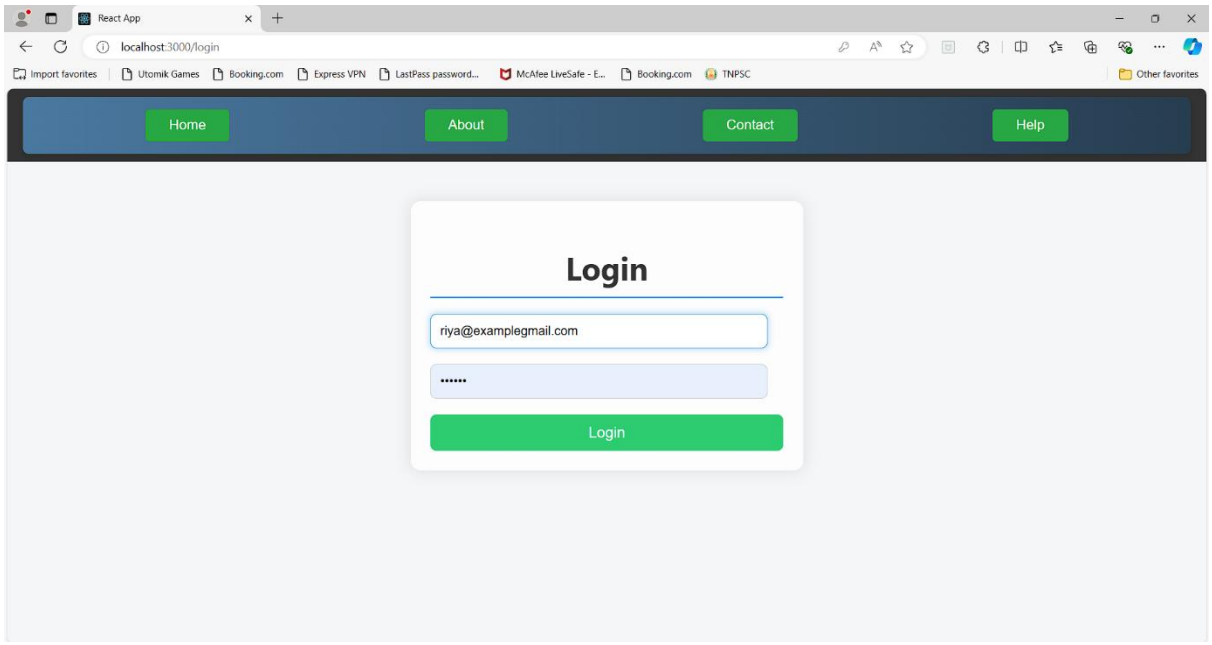
8. Data Validation: Ensures only valid data is entered in forms.
9. Authentication: Secure login and registration with encrypted passwords.
10. Role-Based Access: Controls access to different parts of the system based on user roles.

Screen Shots :

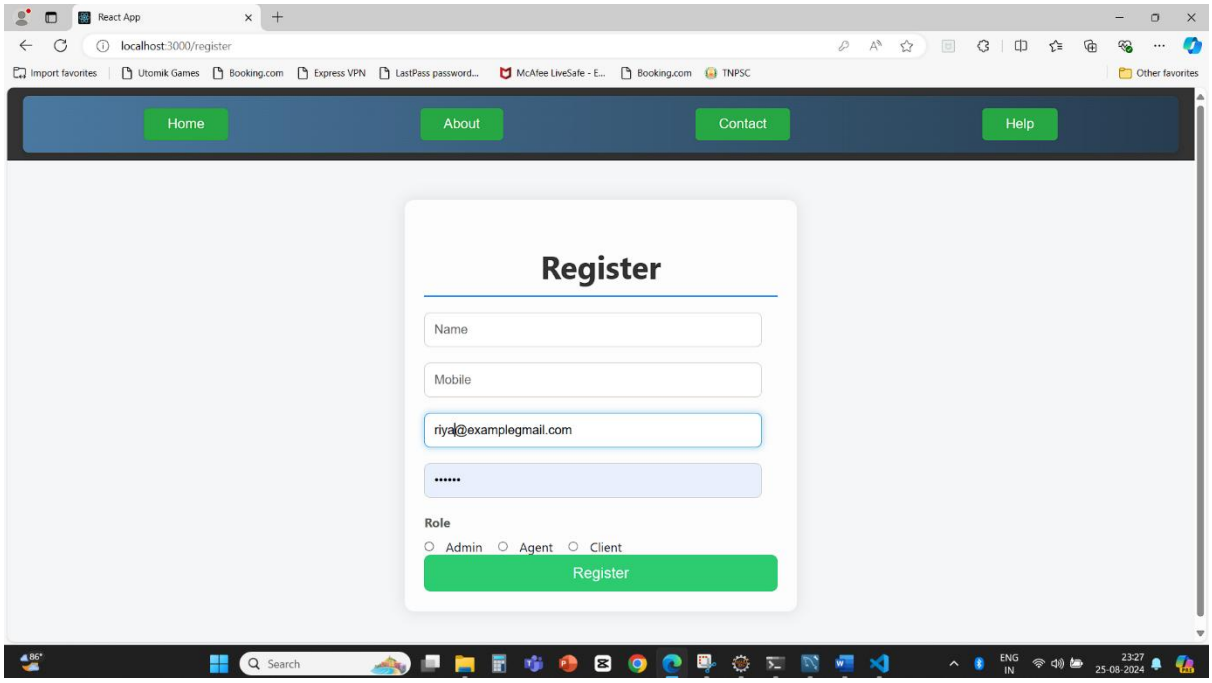
Home page :



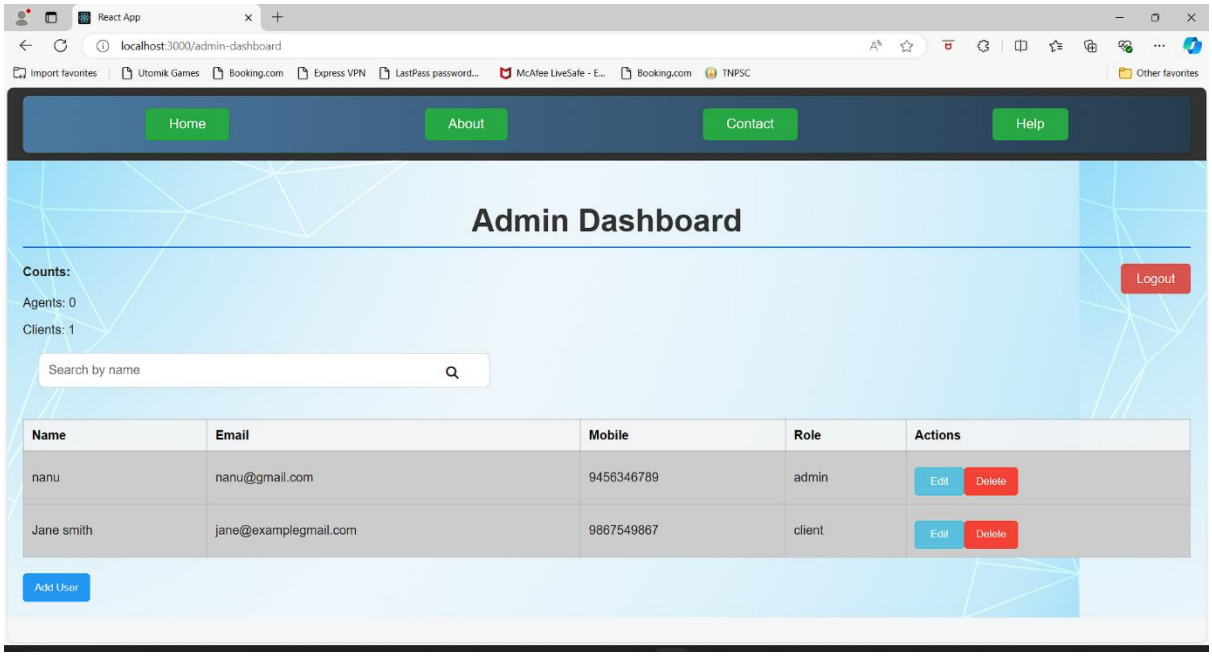
Admin Login :



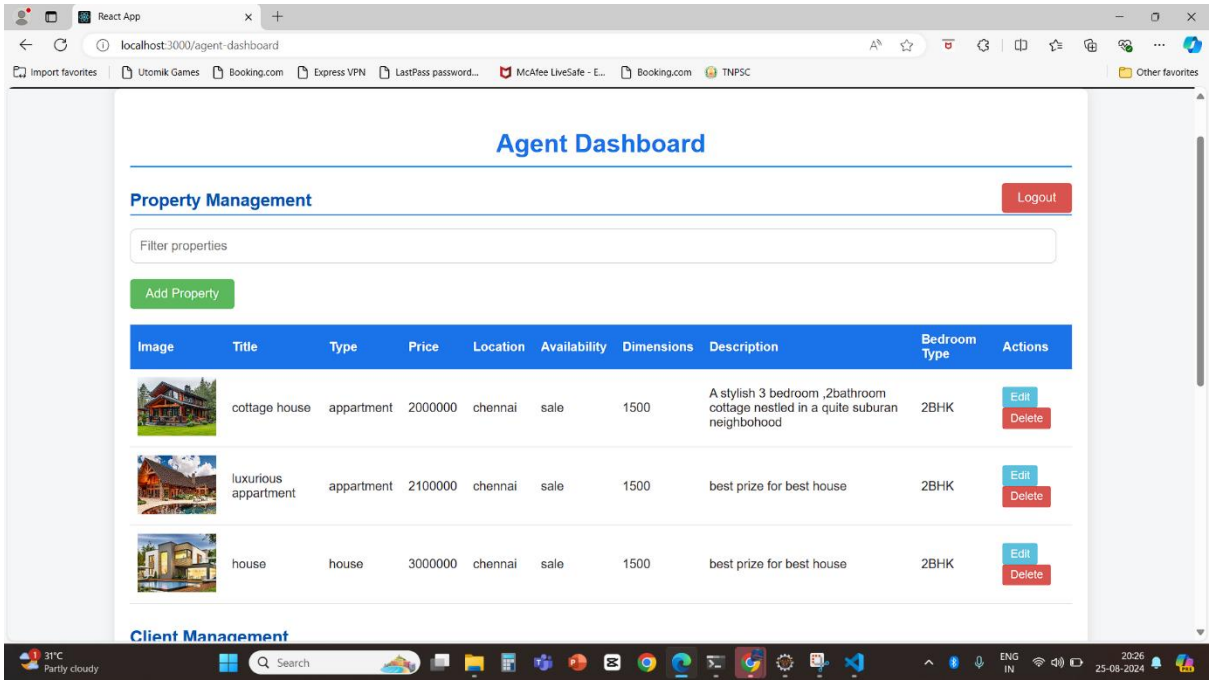
Admin Register:



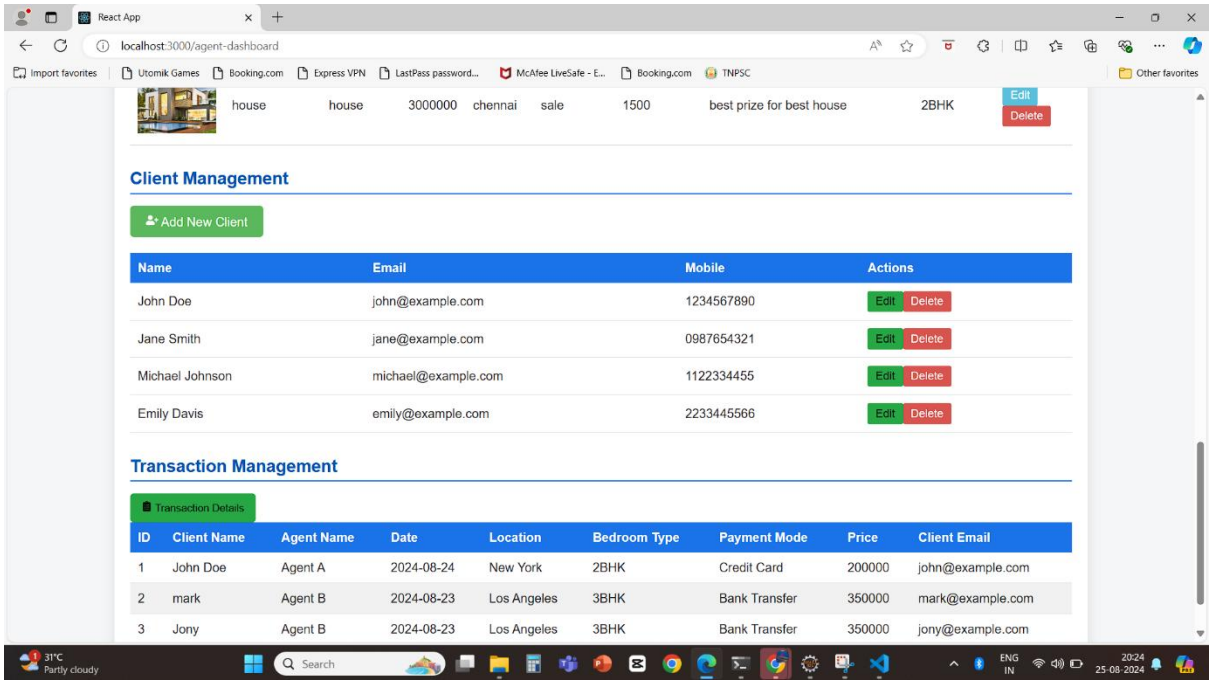
Admin Dashboard :



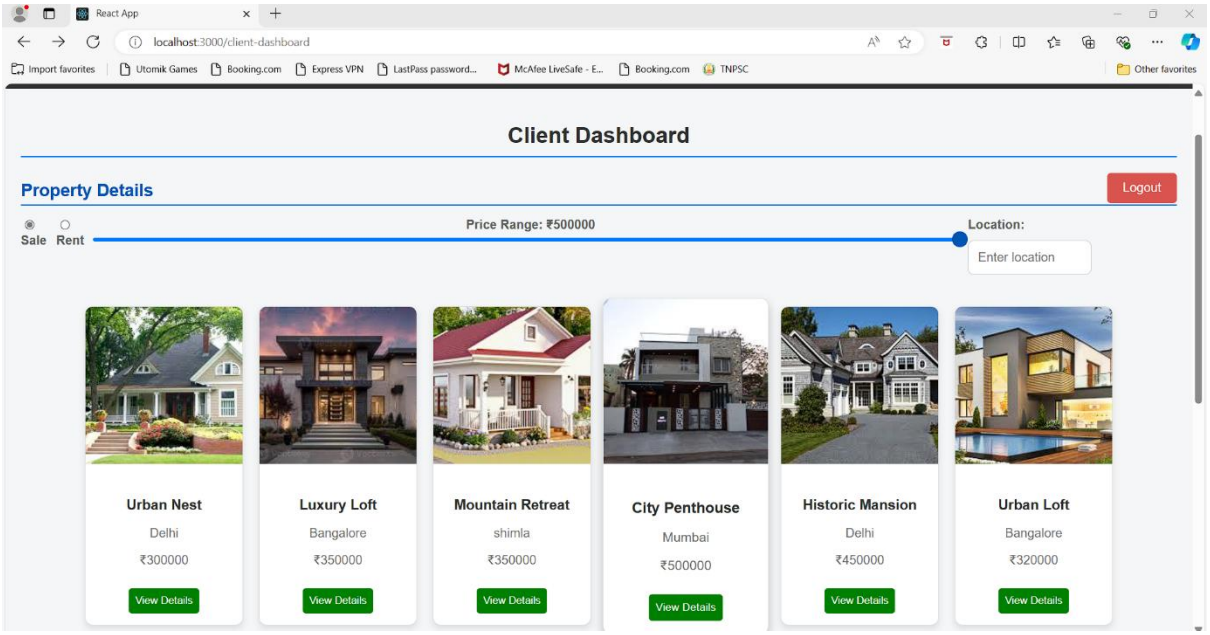
Agent Dashboard :

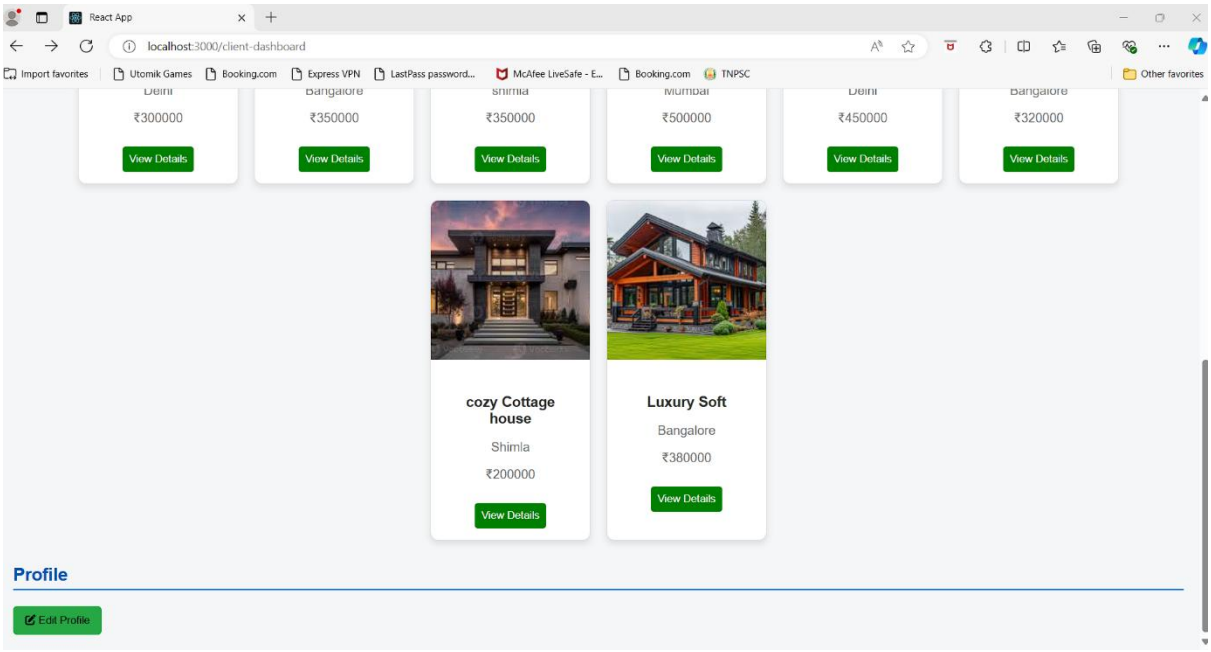


Client Management :

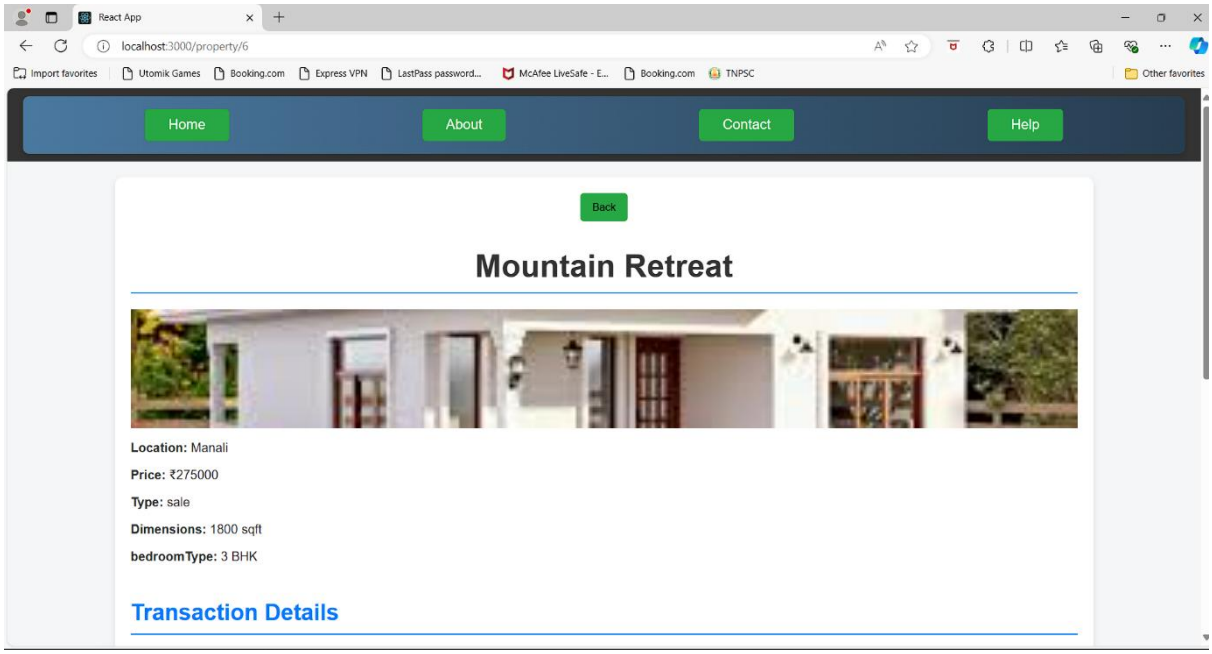


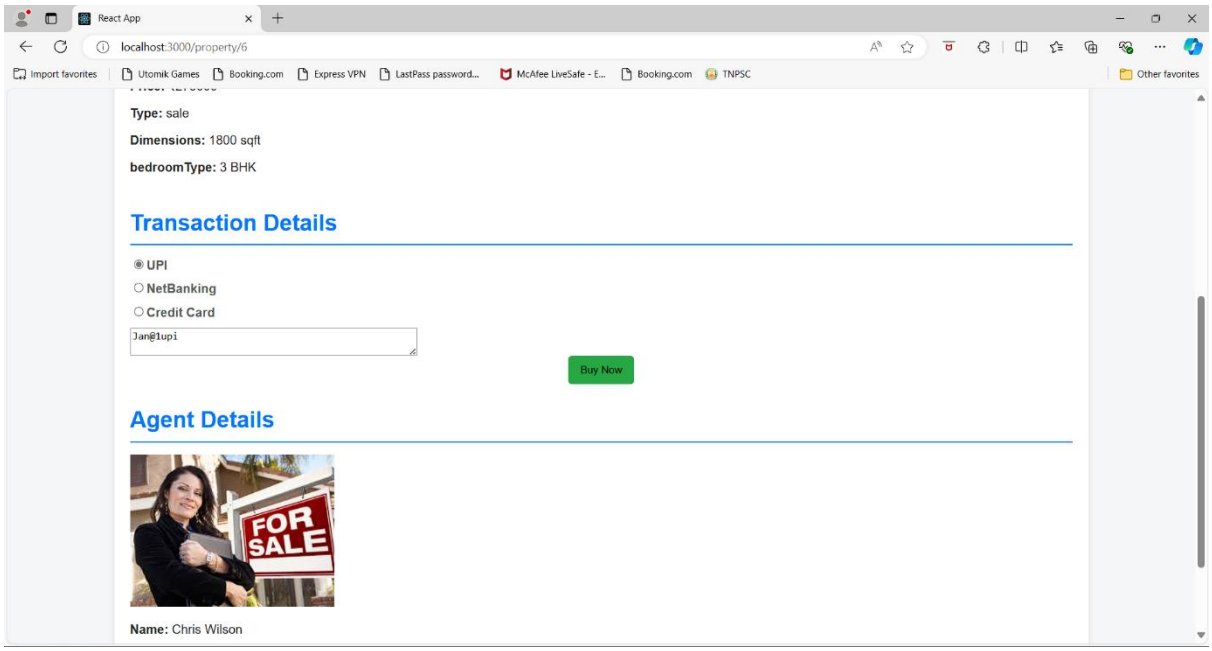
Client Dashboard :





View Details Of card Property





GithubLink:

https://github.com/2709Manjula/Full_Stack_Project/tree/master

Demo video link:

https://youtu.be/wSPgEgQTpH4?si=wcixodv-DBNfdV4F

10. Conclusion

The Real Estate Management System successfully provides a comprehensive solution for managing real estate properties, clients, and users. It offers a seamless experience for different user roles and ensures secure and efficient operations.