:

**🏠 House Rent App - Project Report**

**📋 Project Overview**

The **House Rent App** is a full-stack web application developed using the **MERN stack** (MongoDB, Express.js, React.js, Node.js). It is designed to streamline the process of finding, managing, and renting properties such as apartments, houses, and rooms. The app serves three primary user roles: **Renters, Property Owners**, and **Administrators**.

**🎯 Key Features**

**🔍 Property Listings**

* Rich listings with detailed descriptions, photos, prices, and amenities.
* Location integration via Google Maps API (optional).

**🔎 Search Filters**

* Filter listings based on:
  + Location
  + Price range
  + Property type (house, apartment, room)
  + Number of bedrooms/bathrooms
  + Availability & amenities (Wi-Fi, parking, etc.)

**📞 Contact Property Owners**

* In-app messaging system or contact form.
* Email notifications for new inquiries and booking requests.

**👤 User Roles**

* **Renters**: Search and book properties.
* **Owners**: Add and manage property listings.
* **Admin**: Manage platform integrity and approve new owners.

**👩‍💻 Scenario-Based Case Study: Renting an Apartment**

**1. User Registration**

* Alice registers as a *Renter* by supplying her email and password.

**2. Property Browsing**

* Alice logs in and views vacant properties on her dashboard.
* She applies filters to narrow down options based on her criteria.

**3. Property Inquiry**

* She finds a suitable apartment and sends a booking inquiry via a contact form.

**4. Booking Confirmation**

* The property owner reviews Alice’s inquiry and approves the booking.
* Alice receives a notification and her dashboard updates accordingly.

**5. Admin Approval (Backend Process)**

* Admin verifies and approves new *Owner* registrations to ensure legitimacy.

**6. Owner Management**

* Bob registers as a *Property Owner*.
* After admin approval, Bob can manage listings and update availability.

**7. Platform Governance**

* Admin oversees the system, checks suspicious activity, and enforces policies.

**8. Lease Agreement and Payment**

* Alice and Bob finish rental terms using in-app messaging.
* The lease agreement overseen securely, ensuring transparency.

**9. Move-In**

* Alice moves in, successfully completing the process eased by the app.

**🏗️ Technical Architecture**

**📦 Tech Stack**

* **Frontend**: React.js (Bootstrap, Material UI, Axios)
* **Backend**: Node.js, Express.js
* **Database**: MongoDB (Mongoose ORM)
* **Authentication**: JWT (JSON Web Tokens)
* **APIs**: RESTful APIs for client-server communication
* **Hosting (optional)**:
  + Frontend: Vercel/Netlify
  + Backend: Heroku/Render
  + Database: MongoDB Atlas

**⚙️ Architecture Diagram**

SQL

Copy-edit

+-------------+ REST API +------------------+ +-----------------+

|Frontend| <-----------------> | Node.js/Express | <--> | MongoDB |

| React.js | | Backend | |(Database) |

+-------------+ +------------------+ +-----------------+

|

| Axios

|

↓

User Interactions (Forms, Filters, Bookings)

**🔐 Security**

* Role-based access control (RBAC)
* Data validation on both client and server
* Secure user authentication with JWT
* Admin moderation tools for user and content control

**🧪 Testing & QA**

* Manual testing for booking flow and admin dashboard.
* Unit and integration testing using Jest and React Testing Library
* Post man collection for API testing

**📈 Future Enhancements**

* Payment gateway integration (Stripe/PayPal)
* Real-time chat functionality
* Review and rating system for properties and owners
* Push notifications for bookings and confirmations
* Progressive Web App (PWA) capabilities

**✅ Conclusion**

This House Rent App offers a robust and scalable solution for property renting needs. Through intuitive UI, secure backend, and real-world features, it creates a seamless experience for renters, property owners, and administrators alike. By following the MERN architecture, it is modular, maintainable, and easily extendable for future needs.

Top of Form

Bottom of Form