1. Introduction

- **Project Title: House Rent App using MERN**
- **Team Members:**
  - **Chandrashekar Deverchetty - Project Lead / Full Stack Developer**
  - **Delli Ganesh - Frontend Developer**
  - **Chandra Soodan - Backend Developer**
  - **Chappidi Rohan Reddy - Database Specialist**

2. Project Overview

- **Purpose**: The House Rent App is designed to simplify the rental process for property owners and tenants by offering a comprehensive digital platform. It aims to streamline communication, facilitate easy property management, and enhance user experience in the rental market. The application serves as a centralized system where property owners can manage listings, and tenants can browse and book properties based on their preferences.
- **Features**:
  - **User Management**: Role-based user management, enabling distinct functionalities for admins, property owners, and tenants.
  - **Property Listings**: Property owners can list, update, and manage their rental properties, including images, pricing, and detailed descriptions.
  - **Search Functionality**: Advanced search filters based on location, price range, property type, and amenities.
  - **Booking Management**: Booking requests, status updates, and tracking features for tenants, with notification systems in place.
  - **Admin Oversight**: Admins manage platform policies, approve property listings, handle disputes, and ensure data integrity.
  - **Virtual Tours**: Incorporates virtual tour capability, allowing tenants to explore properties remotely before visiting.
  - **Automated Payment Reminders**: Notification system to alert tenants about payment deadlines and avoid late fees.

- **Dynamic Pricing Model**: Allows property owners to adjust rental prices based on market trends and demand.
- **Community Forums**: A dedicated section for tenants and owners to share insights, experiences, and advice.

3. Architecture

- **Frontend**:
  - Developed using **React.js**, the frontend leverages state management (e.g., Redux) for seamless data flow and interactivity.
  - Integration of **React Router** for smooth navigation between pages without reloading.
  - Responsive and adaptive design using **CSS3** and **Bootstrap**, ensuring a consistent user experience across various devices.
  - Utilizes component-based architecture for better code maintainability and scalability.
- **Backend**:
  - Built with **Node.js** and **Express.js**, providing a robust REST API framework for handling client-server communication.
  - Modular structure with controllers, services, and middleware to separate concerns and improve code maintainability.
  - Utilizes **Express.js middleware** for error handling, logging, authentication, and validation.
  - Implements secure server-side operations, such as input validation and sanitization, to prevent vulnerabilities.
- **Database**:
  - Utilizes **MongoDB** as the primary database for storing data, including user profiles, property listings, booking details, and transactions.
  - Designed with a flexible schema to accommodate various property types, user roles, and dynamic content.

- Implements efficient indexing strategies to optimize search queries and data retrieval.
- Uses **Mongoose** for object modeling and data validation, enforcing schema consistency and facilitating database interactions.

4. Setup Instructions

- **Prerequisites**:
  - **Node.js** (version 16.x or higher)
  - **MongoDB** (version 5.x or higher)
  - **npm** (Node package manager)
  - Optional: **Docker** for containerization.
- **Installation**:
  1. Clone the repository:

     bash

     git clone https://github.com/DeverchettyChandrashekar-18/House-rent-app-using-MERN.git
  2. Navigate to the project directory:

     bash

     cd House-rent-app-using-MERN
  3. Install dependencies for both client and server:
     - **Client**:

       bash

       cd client

       npm install
     - **Server**:

       bash

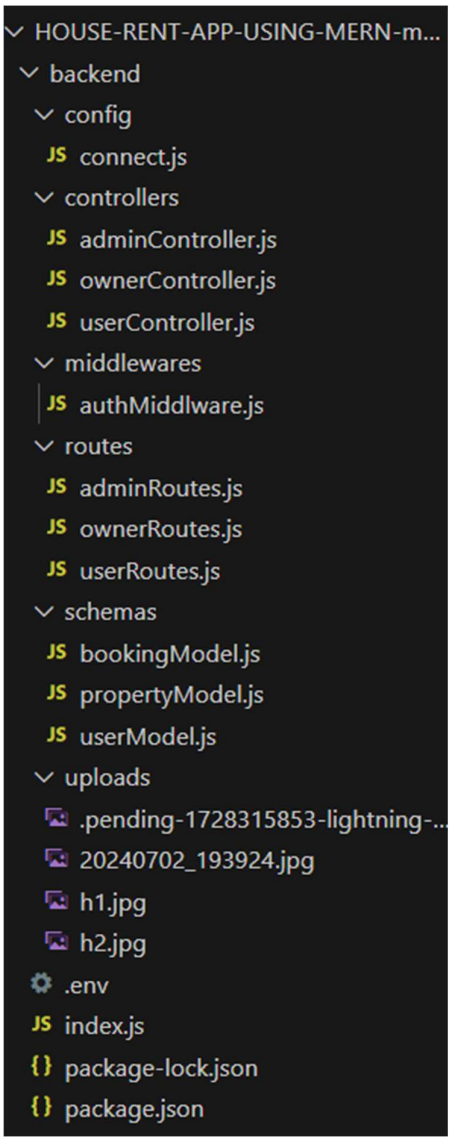
       cd server

npm install

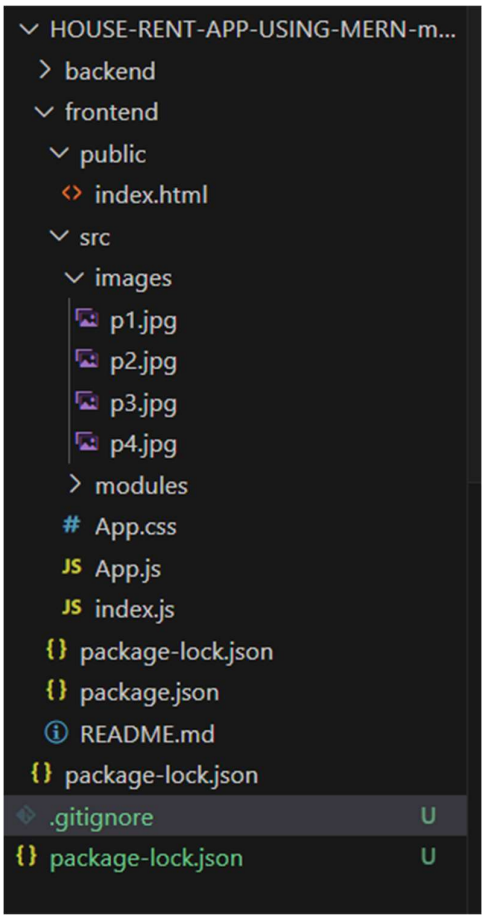
4. Set up environment variables in a .env file in the server directory (e.g., database URI, JWT secret, API keys).

5. Folder Structure

- **Backend**



Frontend



6. Running the Application

- **Frontend**:
  - Navigate to the client directory:

bash

npm start

- **Backend**:
  - Navigate to the server directory:

bash

npm start

7. API Documentation

- **Authentication**:
  - **POST** /api/users/login: User login.
  - **POST** /api/users/register: New user registration.
- **Properties**:
  - **GET** /api/properties: Retrieve all property listings.
  - **POST** /api/properties: Create a new property.
  - **PUT** /api/properties/:id: Update an existing property.
  - **DELETE** /api/properties/:id: Remove a property listing.
- **Bookings**:
  - **GET** /api/bookings: Get all bookings for a user.
  - **POST** /api/bookings: Create a new booking.
  - **PATCH** /api/bookings/:id: Update booking status.
- **Example Response**:

json


```
{
  "success": true,
  "message": "Booking created successfully",
  "data": {
```

```
    "bookingId": "5f8f8f8f8f8f8f8f8f8f8f8",
    "propertyId": "5f8f8f8f8f8f8f8f8f8f8f8",
    "tenant": "John Doe",
    "status": "pending"
  }
}
```


8. Authentication

- Utilizes **JWT (JSON Web Tokens)** for secure user authentication.
- Encrypted passwords using **bcrypt**.
- Role-based access control (RBAC) to limit permissions based on user roles.
- Secure token storage using **HTTP-only cookies**.

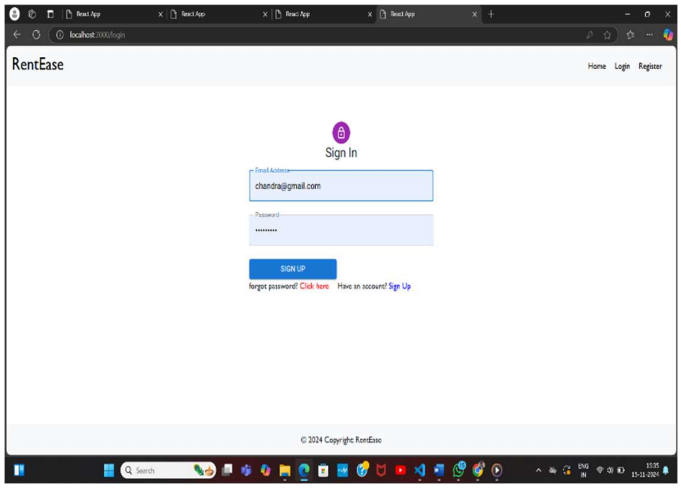
9. User Interface

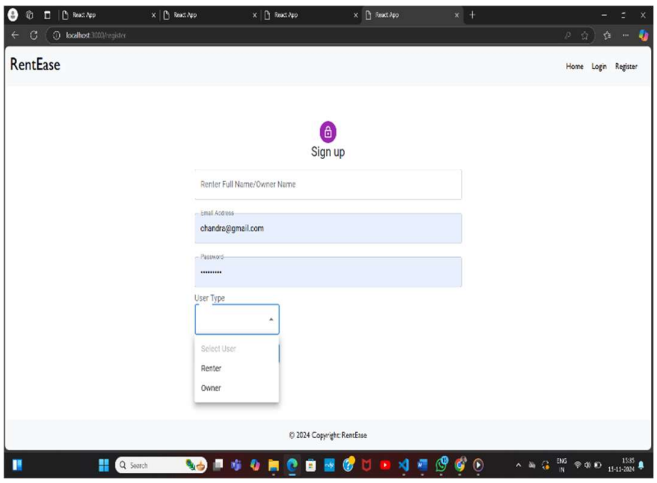
- Focuses on a clean, intuitive UI/UX design, ensuring a seamless browsing experience.
- Features a **dashboard** for property owners to manage listings and bookings.
- Tenants can easily filter search results using a sidebar with adjustable criteria.
- Mobile-first design with **responsive elements** for varying screen sizes.

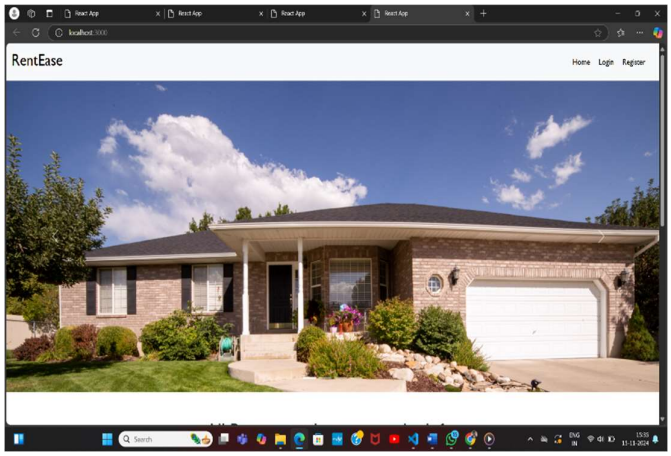
10. Testing

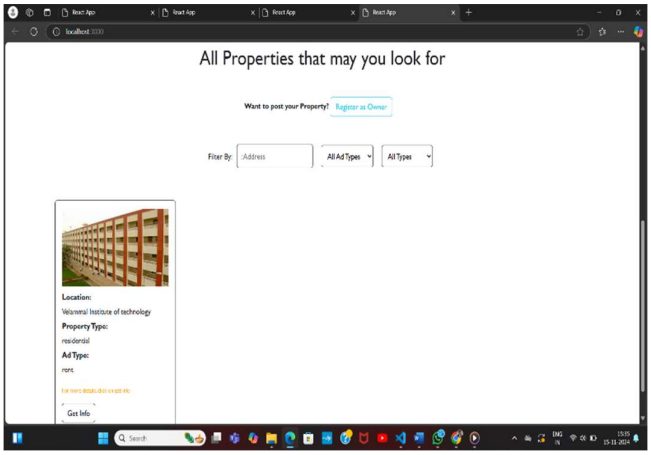
- **Frontend Testing**:
    - Uses **Jest** and **React Testing Library** for unit and integration tests.
- **Backend Testing**:
    - Implements **Mocha** and **Chai** for API testing.
    - **Postman** collections for manual testing of endpoints.

11. Screenshots or Demo









Demo Link : https://github.com/DeverchettyChandrashekar-18/House-rent-app-using-MERN/blob/main/Demo%20video.mp4

12. Known Issues

- Occasionally slow image uploads due to server limits.
- Lack of real-time updates for booking statuses; requires manual refresh.
- Some search filters may not be fully optimized for large datasets.

13. Future Enhancements

- **Payment Integration**: Support for online transactions with secure gateways like PayPal or Stripe.
- **Real-Time Notifications**: Use of **WebSocket's** for real-time updates on bookings and communications.

- **Advanced Analytics**: Dashboards for property owners with insights on views, bookings, and financial trends.
- **AI-Based Recommendations**: Smart suggestions based on tenant search history and preferences.