Project Documentation: House Rent App

1. Introduction

Project Title: House Rent App

Team Members:

RAHUL

KEERTHANA

PRAVEEN SAGAR

LOGANATHAN

2. Project Overview

Purpose:

The House Rent App is designed to provide a seamless platform for property owners and tenants to manage house rental processes online. It allows property listings, tenant registration, and real-time interactions.

Features:

- Property search and filtering.

- Landlord and tenant user accounts.

- Integration of secure payment gateways.

- User reviews and ratings.

- Advanced search with location and price filters.

3. Architecture

Frontend:

- Built using React.

- UI components styled with Material UI, Bootstrap, Ant Design, and mdb-react-ui-kit.

- State management handled locally with React hooks and global state via context API.

Backend:

- Developed using Node.js and Express.js.

- RESTful API for CRUD operations on property and user data.

Database:

- MongoDB for data storage (e.g., user accounts, property details, rental transactions).

4. Setup Instructions

Prerequisites:

- Install Node.js and MongoDB.

Installation:

1. Clone the repository:

```bashgit clone [repository-link]```

2. Navigate to the project directories for the client and server:

```bashcd clientcd server```

3. Install dependencies:

```bashnpm install```

4. Set up the .env file with the required environment variables:

```plaintextPORT=5000MONGO_URI=[MongoDB Connection String]JWT_SECRET=[Your Secret Key]```

5. Start the database server (MongoDB).

5. Folder Structure

Client:

- Organized with components for Home, Property Listings, Authentication, etc.

- Contains assets such as CSS files, images, and custom hooks.

Server:

- routes/: Defines all API endpoints for user, property, and transaction management.

- models/: Schema definitions for MongoDB collections (e.g., User, Property).

- middleware/: Authentication and error-handling middleware.

6. Running the Application

Frontend:

```bashnpm start```

Backend:

```bashnpm start```

Both servers run simultaneously and are connected via REST APIs.

7. API Documentation

Endpoints:

- POST /api/auth/login: User login.

- POST /api/auth/register: User registration.

- GET /api/properties: Retrieve all property listings.

- POST /api/properties: Add a property (authenticated users).

Sample Response:

{ "success": true, "data": [{ "id": "1", "title": "2BHK Apartment", "location": "Downtown", "price": 1200 }]}

8. Authentication

- Token-based authentication using JWT (JSON Web Token).

- Secure endpoints with middleware to validate user roles (e.g., admin, landlord, tenant).

9. User Interface

- UI designed for responsive layouts across devices.

- Example screens:

  - Homepage: Featured listings.

  - Login/Signup: Authentication screens.

  - Dashboard: User-specific views (landlords vs. tenants).

10. Testing

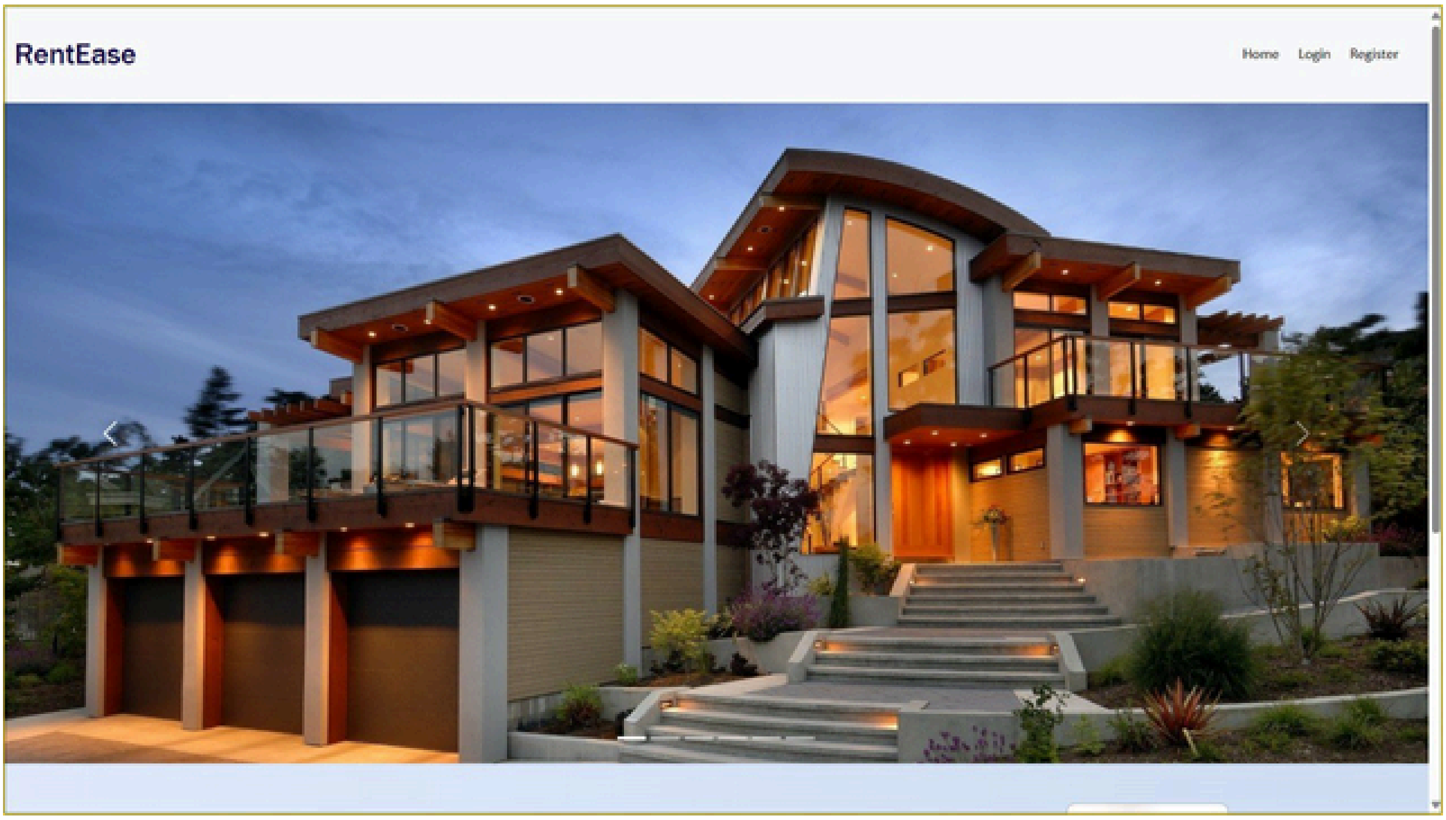
- Manual testing using Postman for API endpoints.

- Automated frontend testing with Jest and React Testing Library.

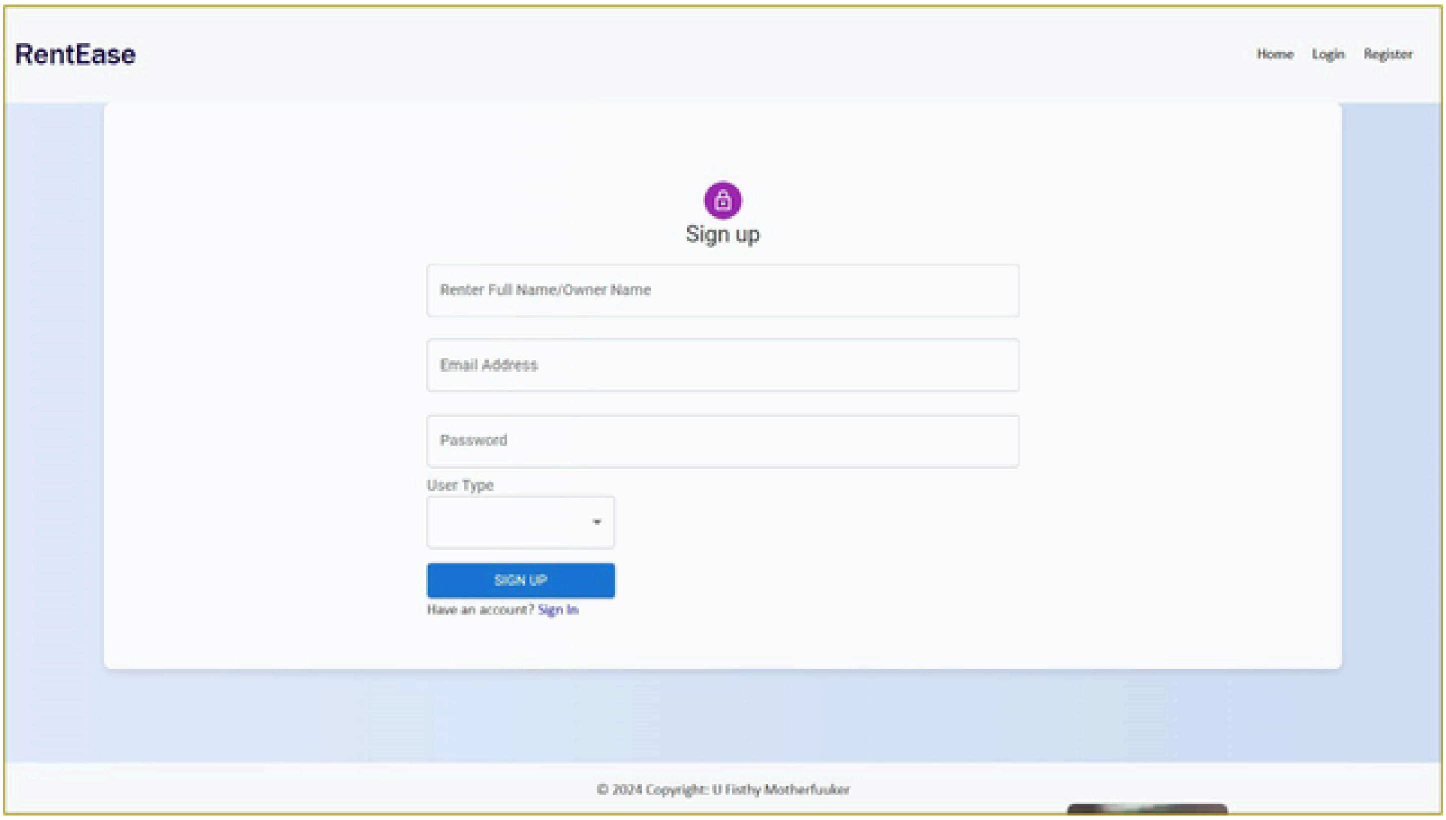
11. Screenshots or Demo

Screenshots:

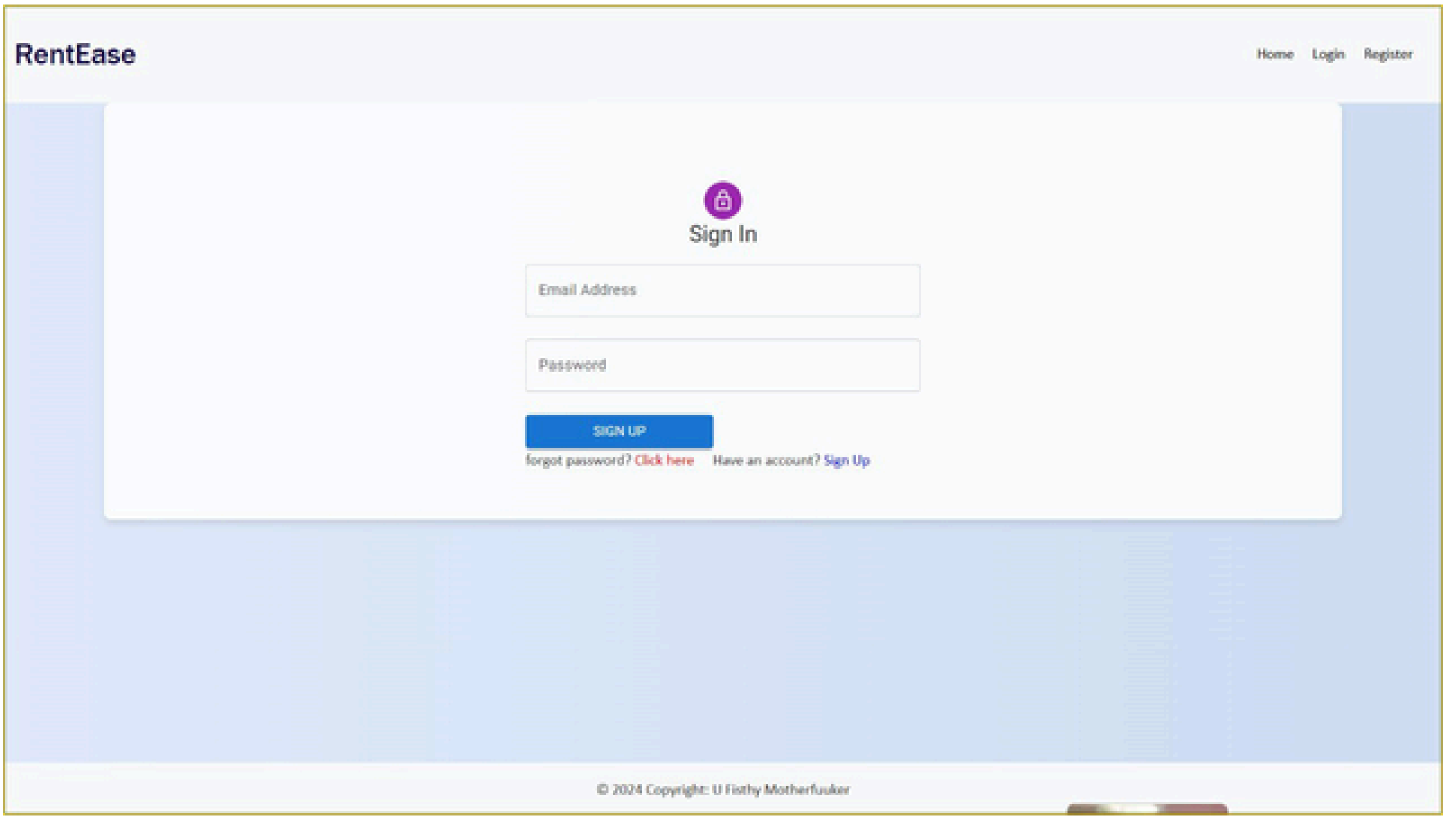
Home:



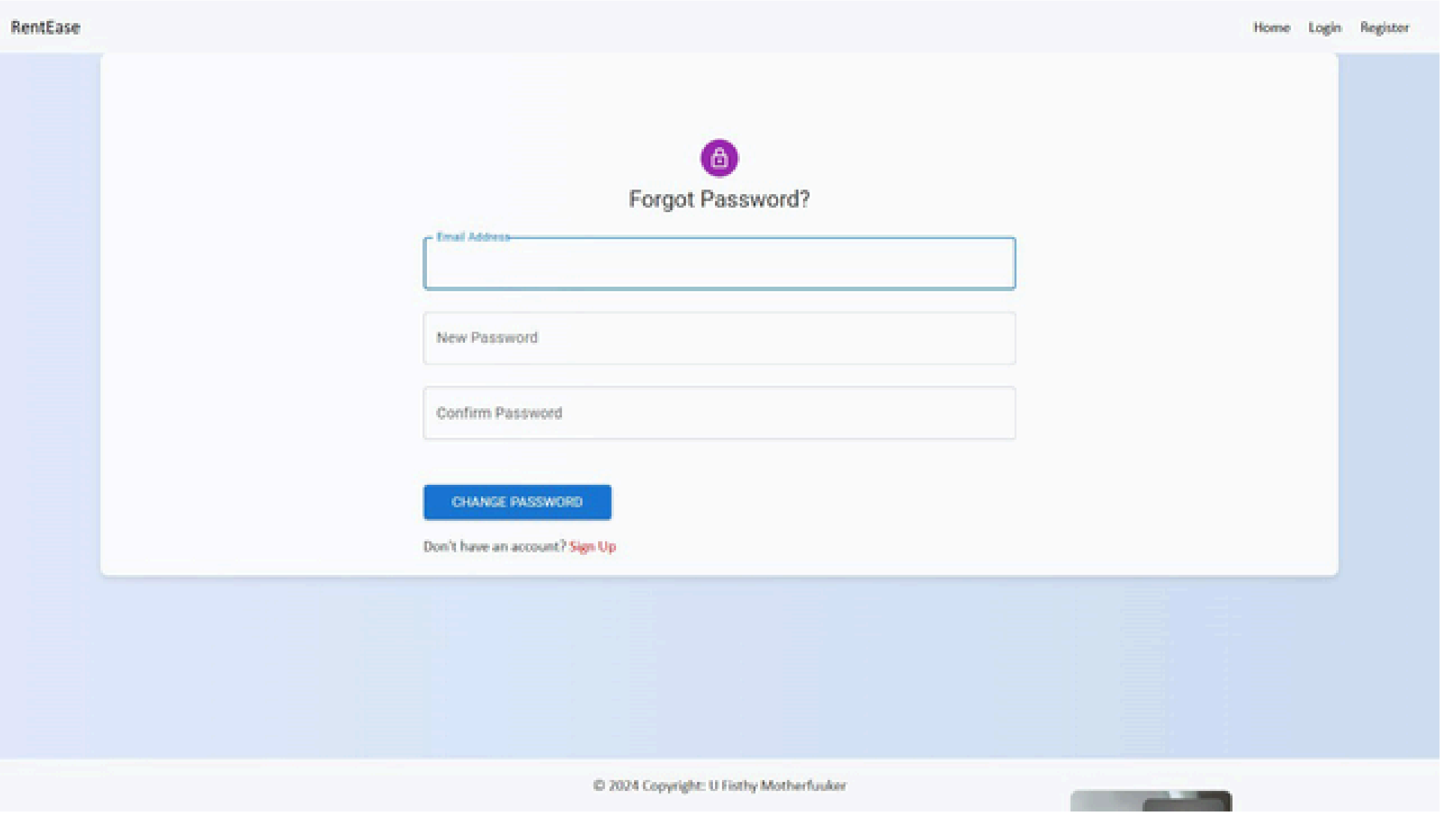
Register:


RentEase
Home
Login
Register
Sign up
Renter Full Name/Owner Name
Email Address
Password
User Type
SIGN UP
Have an account? Sign In
© 2024 Copyright: U Fisthy Motherfuuker

Login:


RentEase
Home
Login
Register
Sign In
Email Address
Password
SIGN UP
forgot password? Click here
Have an account? Sign Up
© 2024 Copyright: U Fisthy Motherfuuker

Forget Password:



Forgot Password?

Email Address

New Password

Confirm Password

Don't have an account? Sign Up

© 2024 Copyright: U Filthy Motherfuuker

Demo:

https://drive.google.com/file/d/1z8AO161XM7TI2yDuaLEkxH3ytPgMoL8G/view

12. Known Issues

- Real-time chat may experience delays under heavy server load.