Full Stack Development with MERN

Project Documentation

🏠 1. Introduction

Project Title: HouseHunt – Finding Your Perfect Rental Home

Team Members:

- **Ratnakar :** Backend Development, Schemas & Controllers
- **Pavani (Team Leader):** Backend Development, Routes & API Integration
- **Akash:** Frontend Development, Authentication, Inquiry & Booking Logic
- **Krishna:** Frontend Development, UI Design & Implementation

---

🧭 2. Project Overview

Purpose

HouseHunt is a MERN-stack based property rental platform that connects renters, property owners, and administrators. Users can browse homes, apply filters, send inquiries, and track responses. Owners can manage listings and inquiries through a dedicated dashboard, and admins monitor and promote verified listings.

Features

- User authentication (signup, login with JWT)
- Browse/filter rental properties
- Inquiry system to connect renters with owners
- Property listing management (add/update/delete)
- Owner dashboard for property management
- Admin panel for verifying and promoting listings

📐 3. Architecture

Frontend (React.js)

- React Router for dynamic page routing
- Context API for global state management
- Reusable UI components (header, cards, inquiry forms)

Backend (Node.js + Express)

- REST API for property listings, user/owner/admin management
- Role-based access control

- JWT for secure sessions

Database (MongoDB)

- Stores user, owner, property, and inquiry data
- Data schema separation by roles and listing types

⚙️ 4. Setup Instructions

Prerequisites

- Node.js ≥ 14.x
- MongoDB (local or Atlas)
- npm

Installation

git clone https://github.com/your-repo/HouseHunt.git

Frontend

cd client

npm install

npm start

Backend

cd server

npm install

npm start

Environment Setup

- **Frontend: .env** → REACT_APP_API_URL=http://localhost:5000
- **Backend: .env** → MONGO_URI, JWT_SECRET, PORT

📁 5. Folder Structure

client/

- src/components/ – Reusable UI elements (Navbar, Footer, PropertyCard)
- src/pages/ – Pages (Home, PropertyDetails, CreateListing)
- src/context/ – Auth and UI context providers

server/

- routes/ – API endpoints (auth, properties, inquiries)
- controllers/ – Logic for CRUD operations

- models/ – Mongoose schemas (User, Property, Inquiry)
- middleware/ – Auth middleware, error handling
- config/ – DB connection, environment setup

▶️ 6. Running the Application

To start locally:

bash

CopyEdit

cd client

npm start

bash

CopyEdit

cd server

npm start

🔭 7. API Documentation

User Endpoints

- POST /api/register – Register renter
- POST /api/login – Login renter
- GET /api/profile – Authenticated user data

Property Endpoints

- GET /api/properties – All properties
- POST /api/properties – Create listing (Owner only)
- PUT /api/properties/:id – Edit property
- DELETE /api/properties/:id – Delete property

Inquiry Endpoints

- POST /api/inquiries/:propertyId – Submit inquiry
- GET /api/inquiries – View sent/received inquiries

Admin Endpoints

- GET /api/admin/properties – All properties for verification
- PATCH /api/admin/promote/:id – Promote verified property

🔐 8. Authentication

Passwords

- Hashed with bcrypt before DB storage
- Compared securely on login

Session Handling

- JWT issued on login
- Stored client-side in React Context

Role-Based Access

- **Roles:** Renter, Owner, Admin
- Conditional rendering in UI and backend routes

👷 9. User Interface

Key UI Elements:

- **Homepage:** Browse promoted and new listings
- **Property Details Page:** Shows images, features, and inquiry form
- **Owner Dashboard:** CRUD for listings, view inquiries
- **Admin Dashboard:** List and promote verified listings

🧪 10. Testing

Testing Strategy

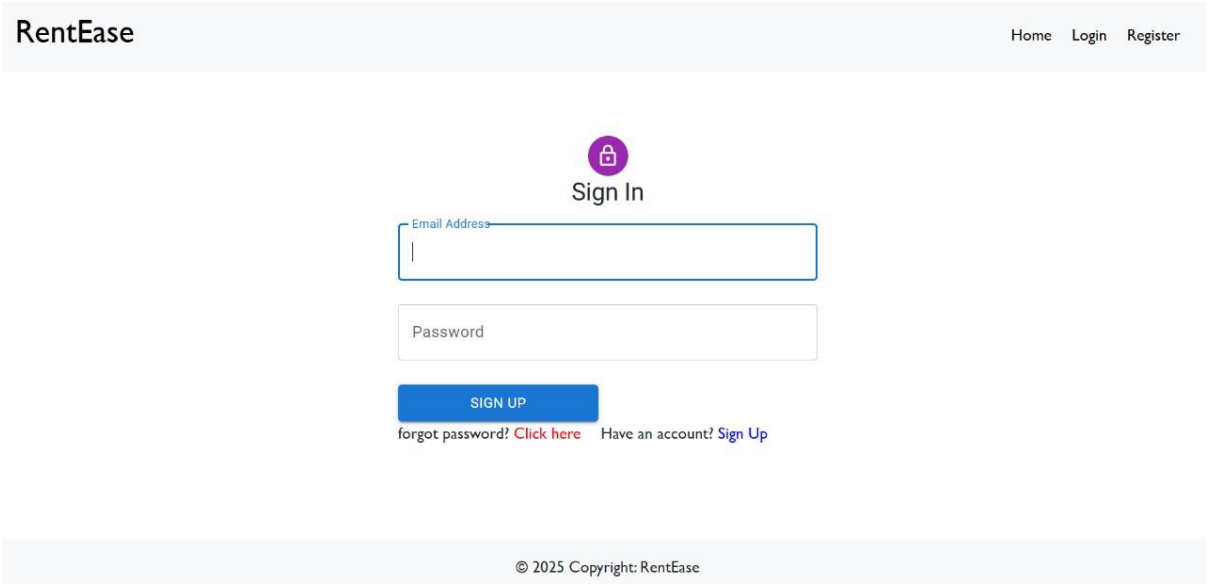
- **Unit Tests:** API routes and DB models
- **Integration Tests:** Renter-to-owner inquiries
- **Manual Testing:** Functional flows like login, property CRUD, inquiry

Tools Used

- **Jest:** Unit testing
- **Supertest:** API testing

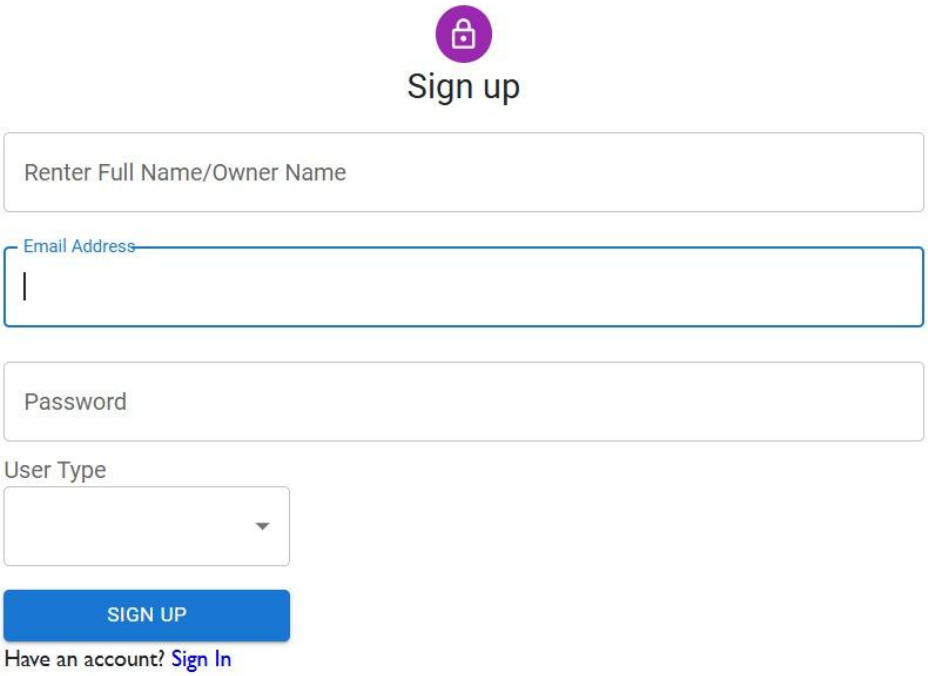
🖼️ 11. Screenshots

- **Login page**

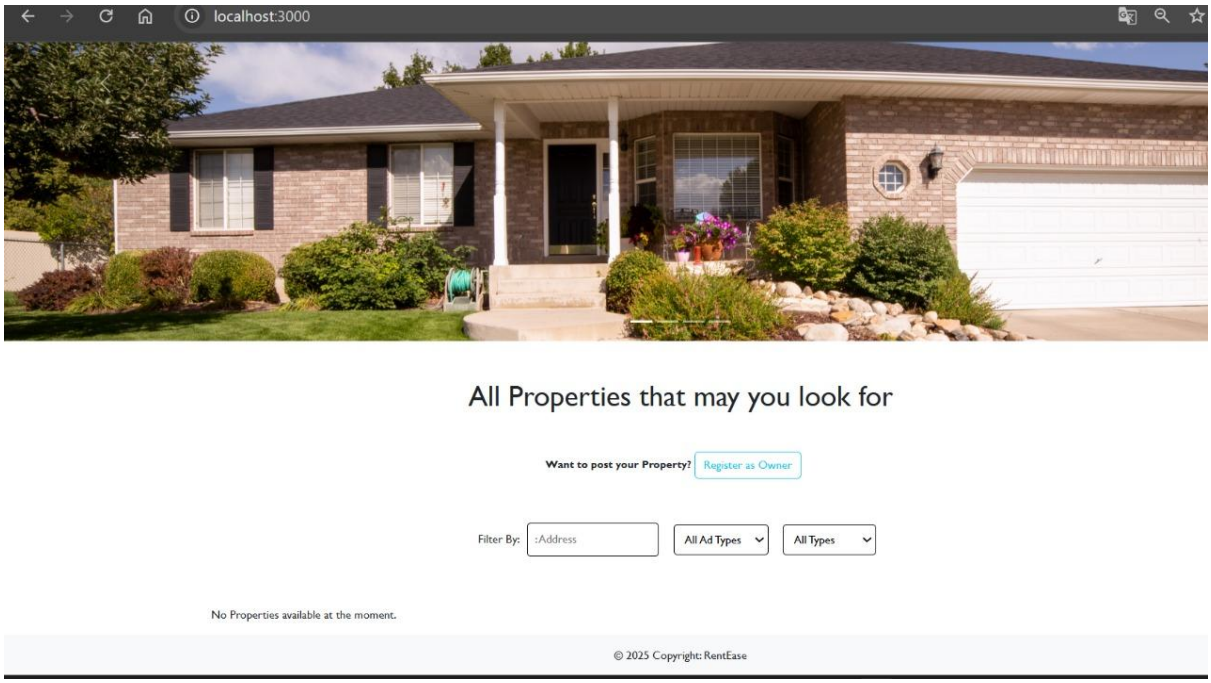


- **Property Listing**

- **Registration Page**



- **Owner Dashboard**



⚠️ 12. Known Issues

- No real-time chat between renter and owner

- No payment gateway or booking calendar yet
- Limited filtering options (to be improved)

🚀 13. Future Enhancements

- Add secure payment gateway for deposits
- Build mobile app using React Native
- Integrate real-time chat and notifications
- Advanced filters (price range, furnished, etc.)
- Admin analytics and fraud reporting tools