Househunt – Finding Your Perfect Rental Home

Team Members:

Gandareddy Malya

Syed Maheen Fathima

Chenigala Navya Sree

Nagalatha Rani Badiginchala

Nikhitha Yerriveera

Team Id: LTVIP2025TMID53273

Course Name: Full Stack Developer (MERN STACK)

Abstract

HouseHunt is a full-stack web application designed to simplify and streamline the rental property search process. Leveraging modern web technologies, HouseHunt provides a centralized platform where tenants can discover, compare, and secure rental homes, while property owners and agents can list and manage properties efficiently.

The frontend is developed using REACT.JS, offering a responsive and intuitive user interface. Users can filter listings based on preferences such as location, price, amenities, and property type. The backend is powered by NODE.JS with Express, managing user authentication, property data, and search algorithms. MONGODB is used as the primary database, allowing flexible storage of property listings, user profiles, and rental history.

HouseHunt supports user roles (renter, landlord, admin), property image uploads, interactive maps with Google Maps API, and secure login with JWT authentication. It also incorporates RESTful APIs to ensure seamless communication between frontend and backend services.

This application aims to enhance the rental experience by offering a user-friendly platform with real-time data, automated notifications, and personalized recommendations—ultimately helping users find their perfect rental home with ease.

DESCRIPTION

The purpose of HouseHunt is to streamline the real estate journey by providing a comprehensive,

user-friendly platform that connects home buyers, sellers, renters, and real estate professionals. It aims

to simplify the often complex and time-consuming process of searching for or listing a property,

offering tools and resources that enhance the experience for all parties involved.

1. Empowering Users with Information: HouseHunt provides detailed listings, market insights, and property data, helping users make informed decisions about buying, selling, or renting homes.
2. Empowering Users with Information: HouseHunt provides detailed listings, market insights, and property data, helping users make informed decisions about buying, selling, or renting homes.
3. Simplifying the Property Search: With advanced search filters and features like virtual tours, HouseHunt makes it easy to find properties that match specific preferences such as location, price, and amenities.
4. Facilitating Transactions: By connecting buyers with sellers, renters with landlords, and users with real estate professionals, HouseHunt streamlines the entire transaction process, from browsing listings to negotiating deals and finalizing paperwork.
5. Supporting Real Estate Professionals: HouseHunt serves as a marketing tool for real estate agents and brokers, offering them a platform to reach potential clients and showcase properties.
6. Creating a Transparent Marketplace: By providing clear, accurate property information, HouseHunt fosters transparency, ensuring that users have access to reliable data to make confident real estate decisions. Offering a Convenient, 24/7 Platform: HouseHunt is accessible anytime, anywhere, offering user

1 . INTRODUCTION

❾ PURPOSE OF THE PROJECT

The purpose of the HouseHunt project is to provide an efficient, user-friendly, and responsive platform that simplifies the process of finding rental properties. Traditional methods of home hunting involve visiting multiple websites, contacting various agents, and struggling to verify listings—all of which are time-consuming and often frustrating. HouseHunt addresses these issues by offering a unified solution where users can search, filter, and view verified rental listings in real time.

The project is designed to cater to different user groups including tenants, landlords, and property agents. It aims to create a seamless interaction between these roles by offering tools for communication, property management, and booking requests. Furthermore, HouseHunt incorporates modern web technologies to ensure high performance, scalability, and security, making it a reliable choice for managing rental needs in the digital age.

❾ PROBLEM STATEMENT

Finding a rental home is a complex and often inefficient process for both tenants and landlords.

Prospective renters frequently encounter problems such as:

Scattered property listings across multiple platforms

Outdated or inaccurate information

Lack of real-time updates and availability

Poor user experience with confusing navigation

Limited communication between landlords and tenants

On the other hand, landlords face difficulties in managing listings, reaching potential tenants, and handling booking requests in an organized manner. Traditional rental portals often lack user-centered design, intelligent filtering, and features that support direct interaction between stakeholders.

There is a clear need for a centralized, digital solution that not only aggregates property listings but also enhances usability, reliability, and communication. HouseHunt seeks to bridge this gap by delivering a comprehensive platform for rental home discovery and management.

❾ OBJECTIVES

The main objectives of the HouseHunt project are as follows:

To design and develop a user-friendly web platform for searching and listing rental properties.

To implement advanced filtering and search options to help users find relevant homes quickly.

To enable landlords and property managers to add, edit, and manage their listings with ease.

To allow prospective tenants to send booking requests and communicate with landlords directly.

To ensure secure and role-based authentication for users (tenants, landlords, admins).

To provide an admin dashboard for monitoring listings, resolving issues, and managing users.

To deliver a scalable and responsive design that works seamlessly across devices.

❾ SCOPE OF THE PROJECT

The scope of HouseHunt encompasses the complete lifecycle of the rental process—from property discovery to user interaction and booking management. Key elements included in the scope are:

- User registration and login with role-based access control (tenant, landlord, admin).
- Listing of properties by landlords, including images, descriptions, pricing, and availability.
- Browsing and filtering of properties by tenants using various criteria such as location, budget, and amenities.
- Communication system between users through inquiry forms or chat interfaces.
- Booking request and response functionality to streamline rental agreements.
- Backend APIs for managing user sessions, property data, and booking records.
- Administrator controls for managing platform integrity and data accuracy.
- The project will initially focus on web application development, with potential extensions to mobile platforms and advanced features like real-time chat, AI-based recommendations, and payment gateway integrations in future phases.

❾ FEATURES

HouseHunt offers a variety of features designed to enhance the real estate experience for buyers, sellers, renters, and professionals. These features aim to simplify property searches, provide valuable insights, and streamline the transaction process. Below are some of the key features of HouseHunt:

1. Advanced Property Search

- Filters: Users can filter properties based on location, price range, property type, size, number of bedrooms, and specific amenities (e.g., pet-friendly, pool, garage).
- Saved Searches: Allows users to save their search criteria for easy access to new listings that match their preferences.

2. Interactive Property Listings

- High-Quality Photos & Videos: Properties are showcased with professional images and video tours to give users a detailed view of the home.
- 360° Virtual Tours: Users can take interactive virtual tours of properties to get a feel for the space before scheduling an in-person visit.
- Floor Plans: Many listings include detailed floor plans to help users visualize the layout of a property.

3. Market Insights and Trends

- Price Trends: Displays information on how property prices are trending in specific areas, allowing buyers and sellers to make informed decisions.
- Neighbourhood Insights: Information on local amenities, schools, crime rates, and transport options, giving users a deeper understanding of the area they are considering.
- Investment Potential: Provides data on property value appreciation and investment opportunities for users looking to invest in real estate.

4. Property Alerts

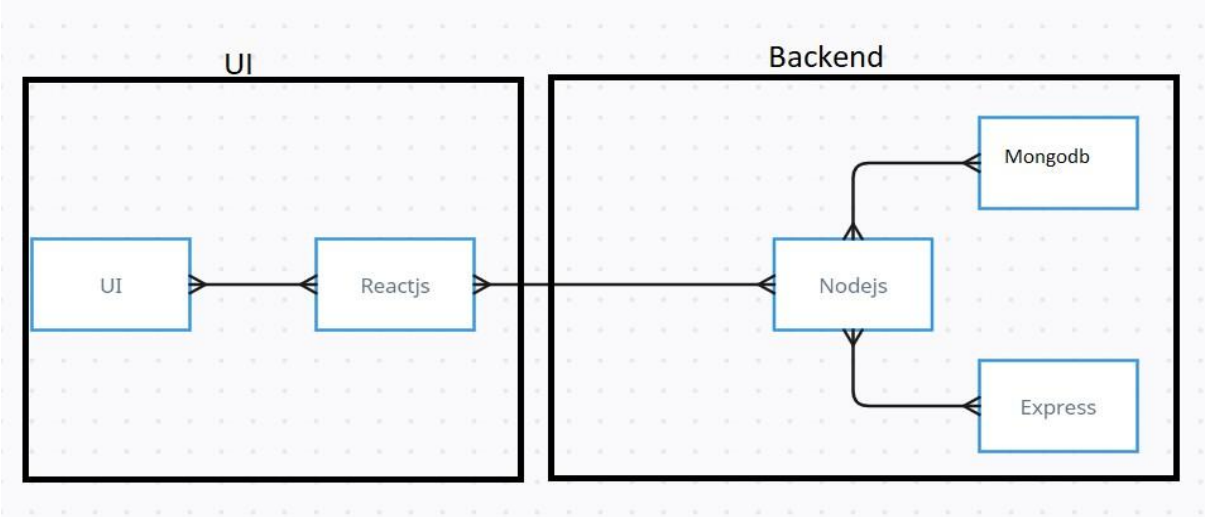
- New Listings Alerts: Users can set up email or push notifications to be alerted when new properties matching their criteria are listed.
- Price Drop Alerts: Users are notified when a property they are interested in has a price reduction.

5. Professional Connection

● Real Estate Agents and Brokers: HouseHunt connects users with certified real estate professionals who can guide them through the buying, selling, or renting process.

● Legal and Financial Experts: The platform offers connections to legal advisors, mortgage brokers, and financial planners to assist with the paperwork and financial aspects of the transaction.

TECHNICAL ARCHITECTURE



The technical architecture of our House rent app follows a client-server model, where the frontend serves as the client and the backend acts as the server. The frontend encompasses not only the user interface and presentation but also incorporates the axios library to connect with backend easily by using RESTful Apis.

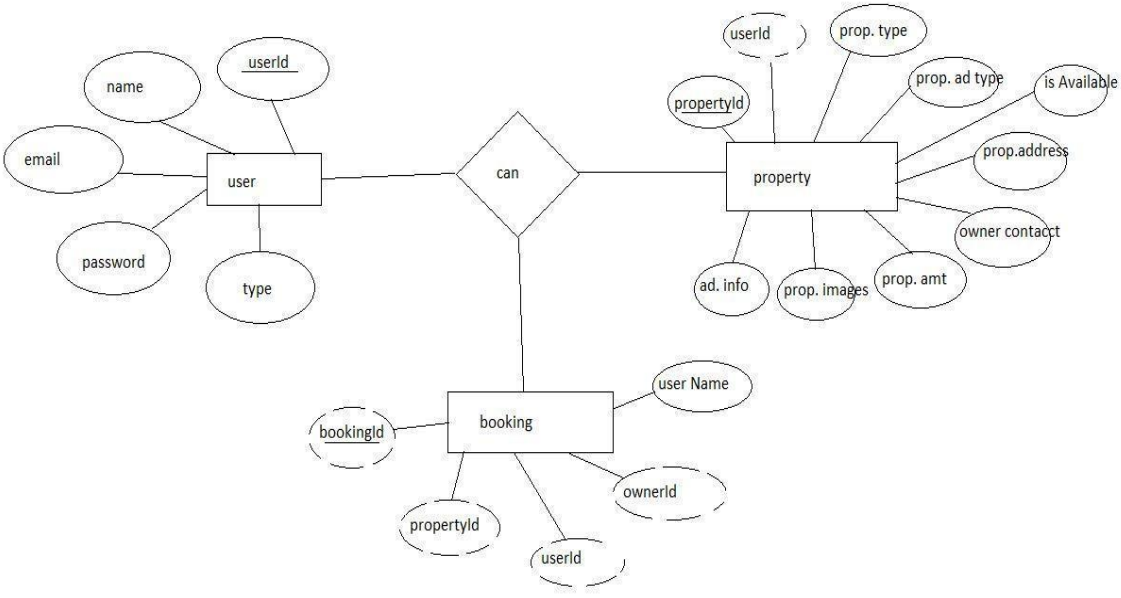
The frontend utilizes the bootstrap and material UI library to establish real-time and better UI experience for any user whether it is admin, doctor and ordinary user working on it.

On the backend side, we employ Express.js frameworks to handle the server-side logic and communication.

For data storage and retrieval, our backend relies on MongoDB. MongoDB allows for efficient and scalable storage of user data, including user profiles, for booking room, and adding room, etc. It ensures reliable and quick access to the necessary information.

Together, the frontend and backend components, along with moment, Express.js, and MongoDB, form a comprehensive technical architecture for our House rent app. This architecture enables real-time communication, efficient data exchange, and seamless integration, ensuring a smooth and immersive booking an appointment and many more experience for all users.

ER DIAGRAM



PREREQUISITES :

Node .js and npm

Express.js

Mongodb

Moment.js

React.js

Antd

HTML,CSS and Javascript

Database connectivity

Front-end Framework

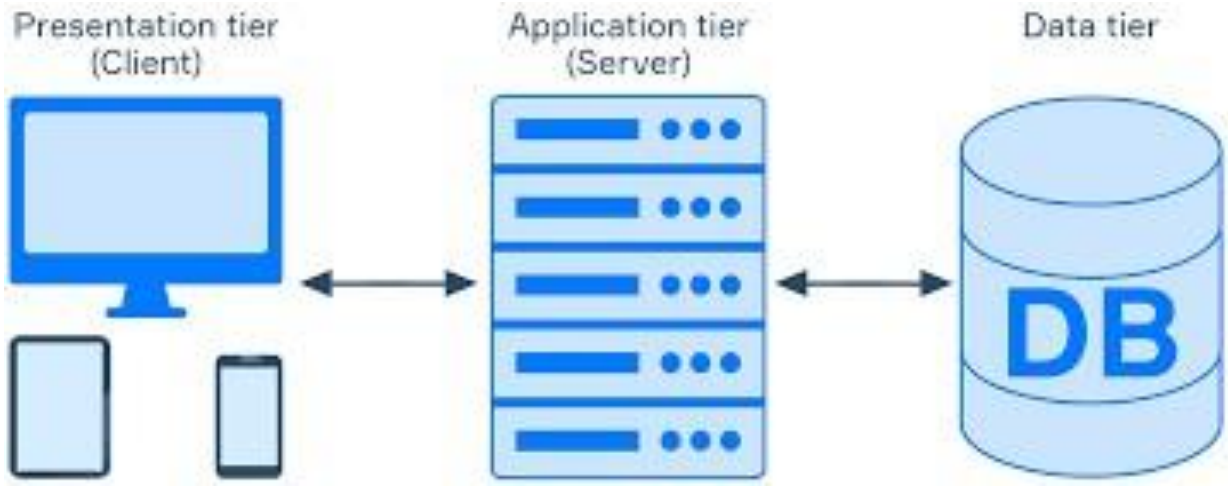
Version control

2. SYSTEM OVERVIEW

❾ SYSTEM (PROJECT) ARCHITECTURE

HouseHunt is based on a three-tier architecture consisting of:

- Frontend (Client Side): The user interface built using React.js and Tailwind CSS. This is what users see and interact with.
- Backend (Server Side): Built with Node.js and Express.js, this handles logic, APIs, and communication with the database.
- Database: MongoDB stores data like user details, property listings, and bookings.
- These three parts work together to process user requests, fetch or store data, and display information.



❾ USER ROLES AND USE CASES

HouseHunt supports three types of users, each with different permissions and responsibilities:

Tenant:

Search and filter properties

View property details

Send booking or inquiry requests

Save properties to wishlist

Landlord:

Register and log in

List and manage properties

Respond to tenant inquiries

Admin:

View all users and properties

Remove inappropriate listings

Manage the platform

❾ TECHNOLOGY STACK

HouseHunt uses modern technologies to ensure performance and reliability:

Frontend: React.js, Tailwind CSS, Redux (optional)

Bakend: Node.js, Express.js

Database: MongoDB

Authentication: JWT (JSON Web Tokens)

Hosting:

Frontend: Vercel

Backend: Render or Heroku

Database: MongoDB Atlas

3.FRONTEND(CLIENT SIDE)

The frontend is what users see and interact with directly on the website.

Technologies Used:

React.js: For building the user interface.

Tailwind CSS: For styling the components quickly and consistently.

React Router: For navigating between pages (e.g., Home, Login, Property Details).

Axios: For calling backend APIs (e.g., fetching property data).

Main Features:

User Registration & Login Screens

Property Search Page with filters like price, location, availability

Property Details Page with full descriptions and pictures

Landlord Dashboard to manage listed properties

Responsive Design to work on mobile, tablet, and desktop

4. BACKEND(SERVER SIDE)

The backend handles all the logic, data storage, and communication with the database.

Technologies Used:

Node.js: Server-side JavaScript environment.

Express.js: Framework for creating APIs and handling requests.

Main Responsibilities:

User Authentication: Using JWT (JSON Web Tokens)

Property Listing Management: CRUD operations for properties

Booking System: Handles requests between users and landlords.

Role Management: Different actions allowed for tenants, landlords, and admins

Security: Password encryption (bcrypt), validation, and protection from attacks

Example APIs:

POST /register – register a new user

POST /login – authenticate user and return token

GET /properties – get all listings

POST /bookings – send a booking request

5 . DATABASE DESIGN

DATABASE SELECTION

For HouseHunt, we chose MongoDB as our database because:

It is NoSQL and document-based, which makes it easy to handle flexible data structures.

It stores data in JSON-like format (BSON), which fits perfectly with our JavaScript-based stack.

It supports scalability, real-time performance, and integrates well with Node.js via Mongoose.

DATABASE DEVELOPMENT

SCHEMA DESIGN

Here are the main collections (tables) we use:

Users Collection

Stores tenant, landlord, and admin user data.

{

_id: ObjectId,

name: "John Doe", email:

"john@example.com", password:

"encrypted_pass", role: "tenant", //

or "landlord", "admin" createdAt:

Date

}

Properties Collection

Holds information about all rental listings.

{
_id: ObjectId, title: "2 BHK
Apartment", description: "Near park,
fully furnished", address: "123 Main
St", city: "Hyderabad", price:
15000, bedrooms: 2, bathrooms: 2,
landlordId: ObjectId, images:
["img1.jpg", "img2.jpg"], isAvailable:
true, createdAt: Date
}

Bookings Collection

Contains tenant requests for property bookings.

{
_id: ObjectId,
tenantId: ObjectId, propertyId: ObjectId,
message: "Interested in visiting next
week", status: "pending", // or
"approved", "rejected" createdAt: Date
}

DATA RELATIONSHIPS AND VALIDATION

One-to-Many: One landlord can post many properties.

One-to-One: One booking belongs to one tenant and one property.

References: We use ObjectId fields (landlordId, tenantId, propertyId) to link data between collections.

Validation using Mongoose:

Required fields: email, title, price, etc.

Data types: strings, numbers, dates, booleans.

Unique constraints: email must be unique.

Regex patterns: for email formats, password rules, etc.

6 . CORE FEATURES

PROPERTY SEARCH AND FILTERS

Users can search for rental properties based on criteria like city, price range, number of bedrooms, and availability. Filters help users find exactly what they need without browsing irrelevant listings.

PROPERTY LISTING AND MANAGEMENT

Landlords can easily list new properties with all necessary details like images, descriptions, prices, and amenities. They can edit or remove listings through their dashboard.

BOOKING AND INQUIRY SYSTEM

Tenants can send booking requests directly from the property details page. Landlords receive these requests in their dashboard and can approve or reject them. Optional inquiry or messaging features enhance communication.

MAP INTEGRATION

Google Maps API is used to show the property location on a map, making it easier for users to understand the neighborhood and distance from key landmarks.

NOTIFICATIONS

Email notifications or push alerts can be sent for new bookings, approvals, or status updates to keep users informed in real-time.

ADMIN PANEL FOR MODERATION

Admins have access to tools for managing user accounts, reviewing property listings, and removing content that violates policies. They help maintain platform quality and trust.

7 . SYSTEM IMPLEMENTATION

DEVELOPMENT PROCESS

The project was developed using Agile methodology. Tasks were divided into sprints with daily goals for frontend, backend, and database features.

DEPLOYMENT STRATEGY

The frontend and backend were deployed separately to ensure modular scaling:

Frontend deployed on Vercel

Backend hosted on Render or Heroku

MongoDB Atlas used for cloud database

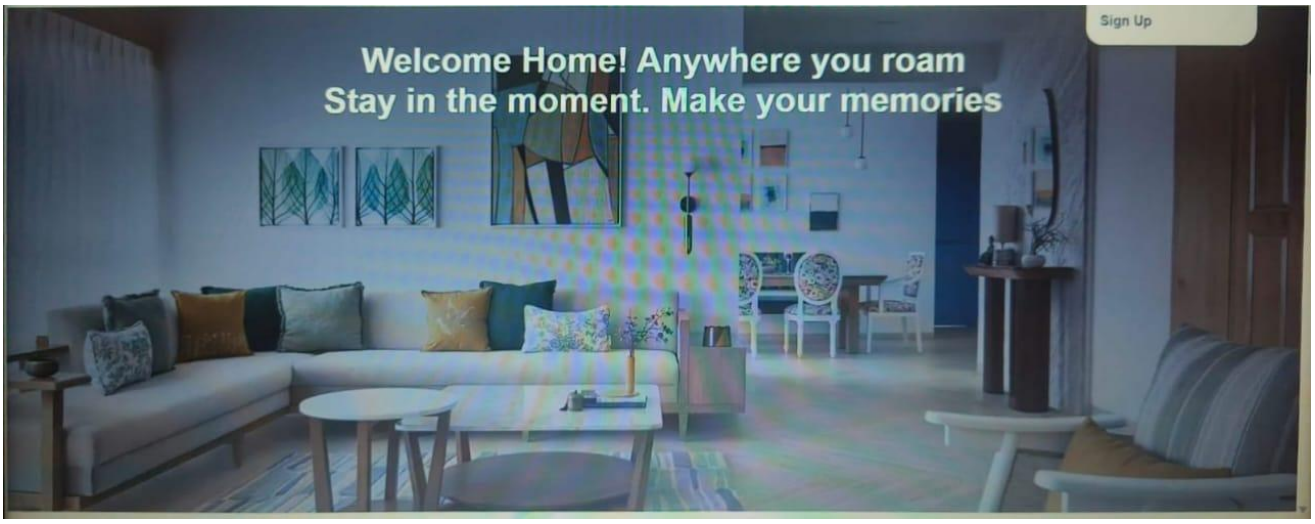
HOSTING PLATFORMS

Frontend: Vercel (auto CI/CD from GitHub)

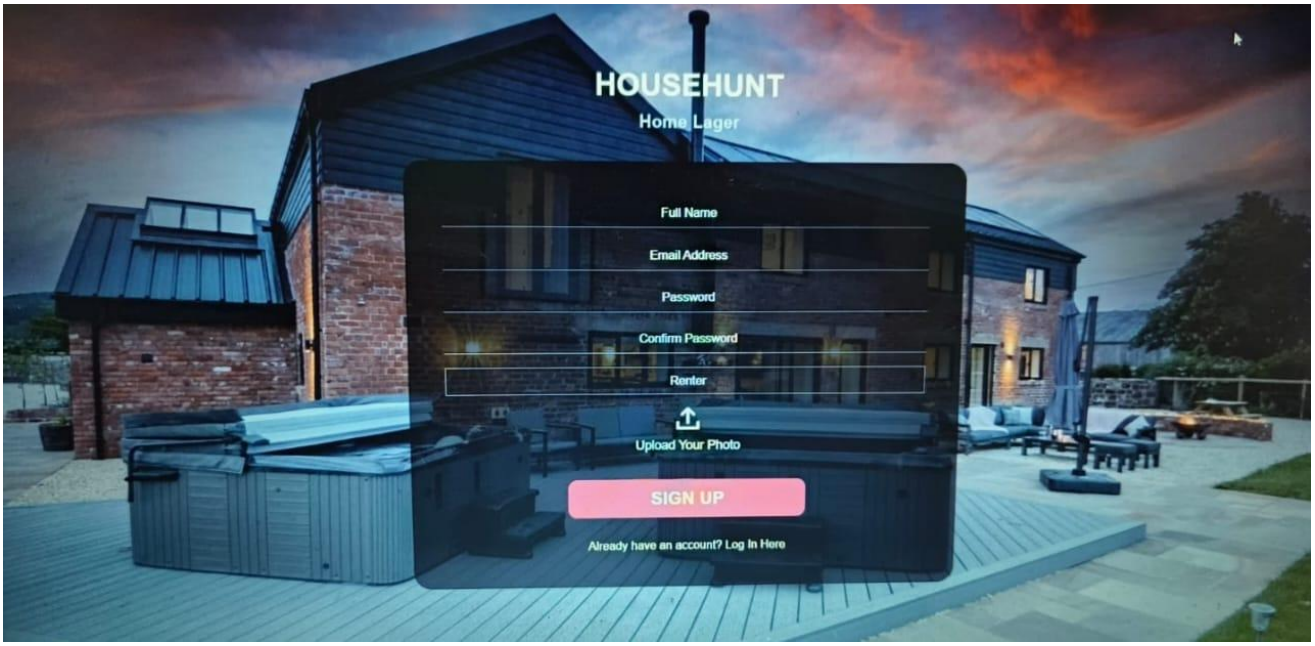
Backend: Render/Heroku (Node.js runtime)

Database: MongoDB Atlas (cloud-managed NoSQL DB) PROJECT IMPLEMENTATION

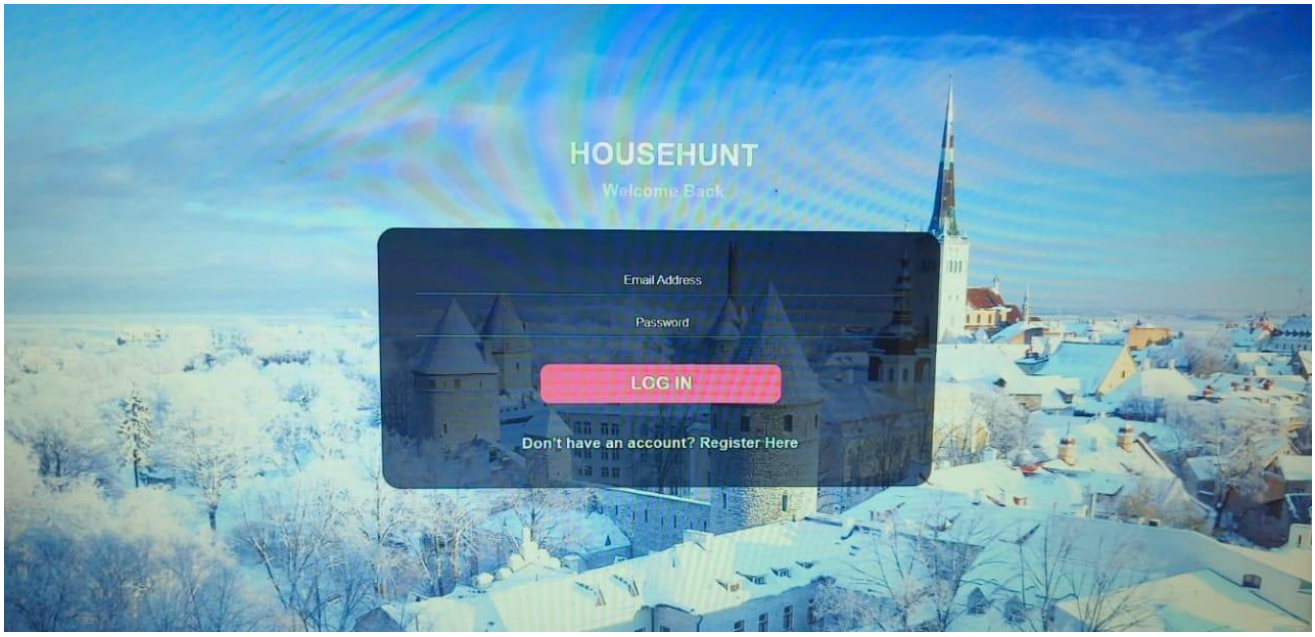
Home Page :



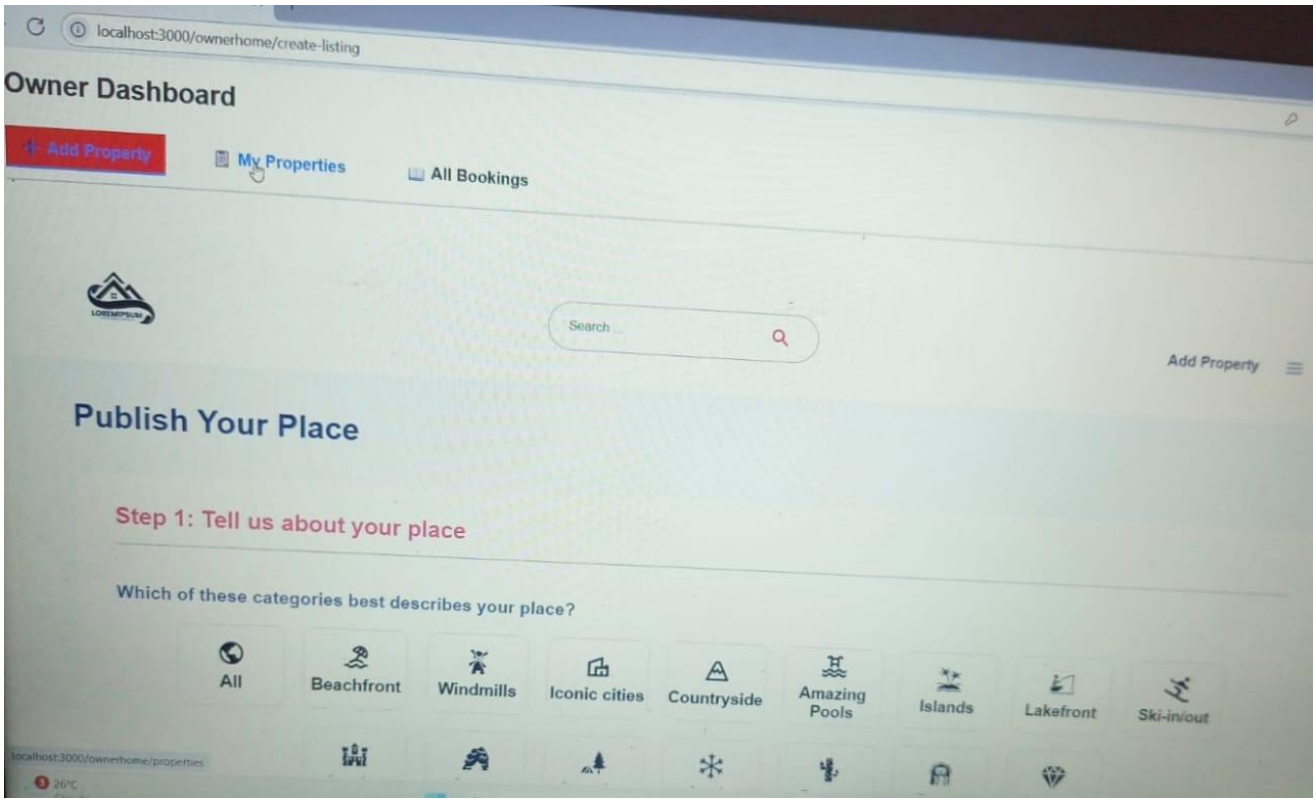
Register Page:



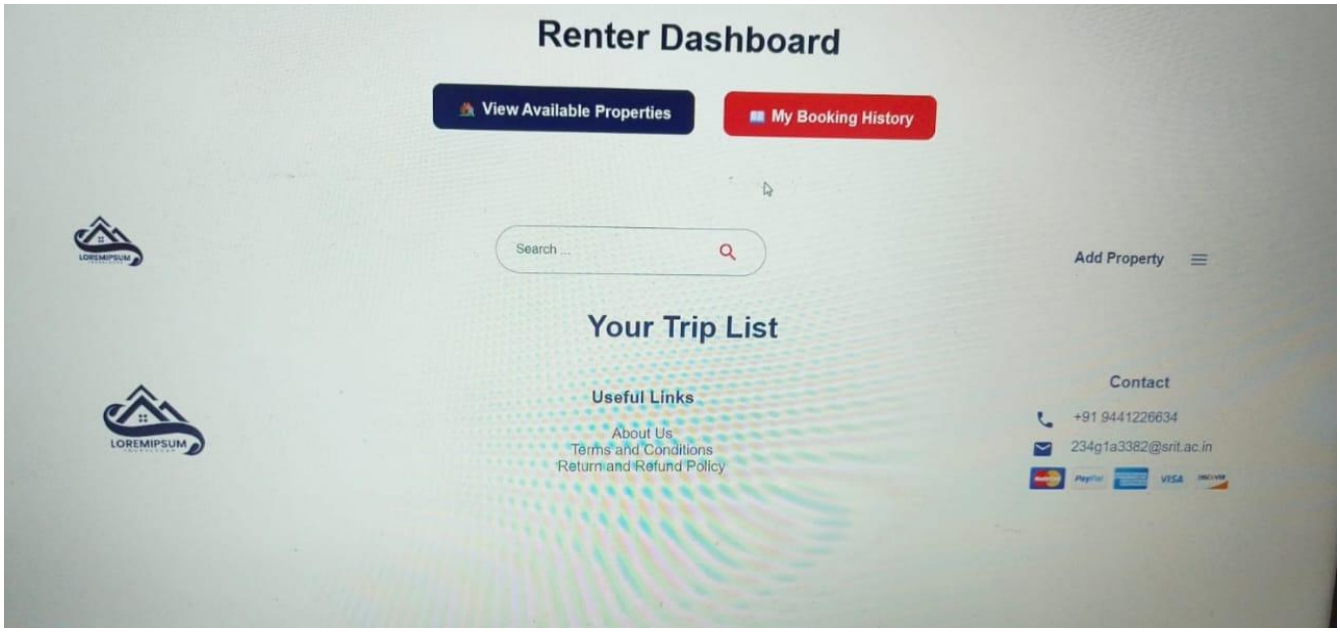
Login Page:


HOUSEHUNT
Welcome Back
Email Address
Password
LOG IN
Don't have an account? Register Here

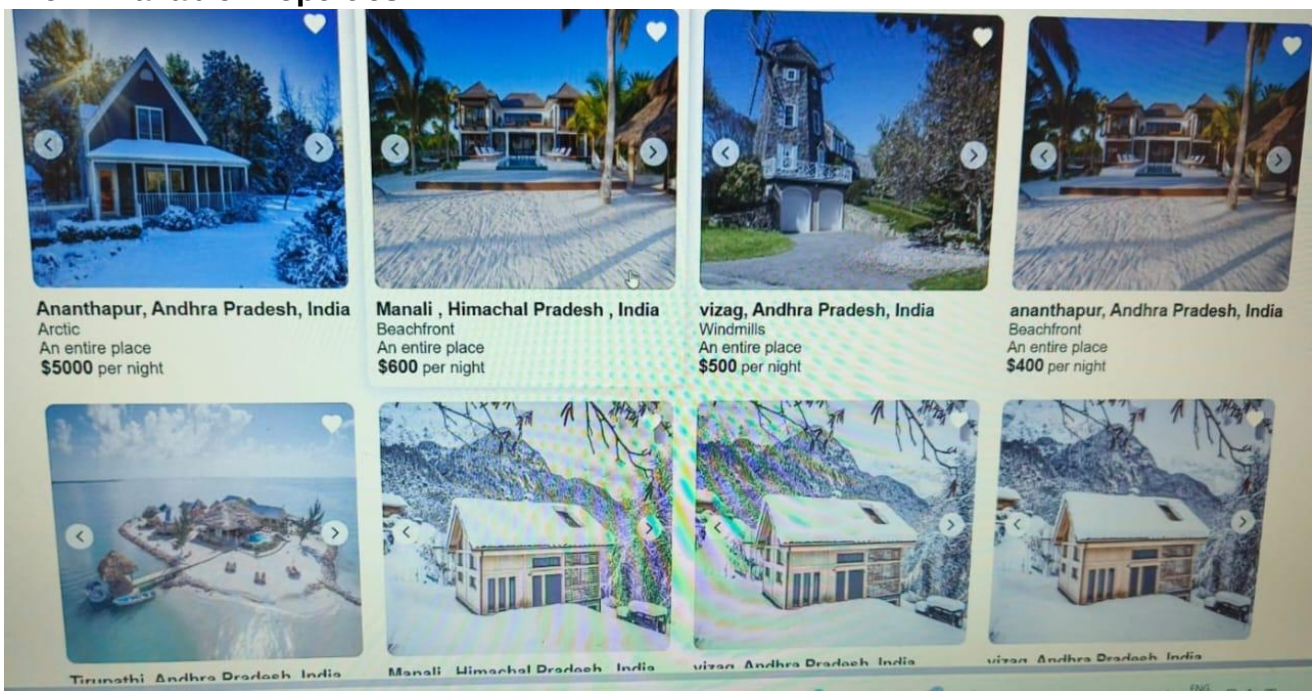
Owner Dashboard:



Renter Dahboard:



View Available Properties:



8 . TESTING AND VALIDATION

UNIT TESTING

Unit tests were written for backend API endpoints using tools like Jest or Mocha to ensure each function works as expected.

INTEGRATION TESTING

Testing how different modules work together – such as property listings interacting with bookings – was done manually and with Postman collections.

USER ACCEPTANCE TESTING

Sample users tested the app and provided feedback on usability, functionality, and design, helping ensure it met real user needs.

BUG TRACKING AND RESOLUTION

Bugs were tracked using GitHub Issues or Trello boards. Regular fixes were pushed and verified in subsequent builds.

9 . CHALLENGES AND SOLUTIONS

TECHNICAL CHALLENGES

Integrating role-based routing

Managing asynchronous operations with API calls

Solution: Used middleware and token validation to protect routes. Async/Await and try-catch blocks improved reliability.

DESIGN CHALLENGES

Maintaining responsive layout on all screen sizes

Building a clean, intuitive UI

Solution: Tailwind CSS was used with a mobile-first approach, and components were kept minimal and consistent.

OPTIMIZATION TECHNIQUES

Lazy loading images

Code splitting with React

Indexing MongoDB queries for faster filtering

CONCLUSION :

- HouseHunt successfully addresses the challenges faced by individuals in their search for the perfect rental home. By integrating modern web technologies into a full-

stack solution, the platform offers a seamless, user-friendly experience that simplifies property discovery, comparison, and contact with property owners or agents.

- Through features such as advanced filtering, interactive maps, user authentication, and data-driven listings, HouseHunt empowers users to make informed decisions based on their specific needs and preferences. The responsive design ensures accessibility across devices, while the secure backend and optimized database support scalability and reliability.
- The development of HouseHunt highlights the practical application of full-stack development principles, encompassing frontend design, backend APIs, and database integration. It serves not only as a helpful tool for renters but also as a robust demonstration of technical skills in real-world problem-solving.
- Moving forward, HouseHunt has the potential to expand its features—such as including virtual tours, AI-driven recommendations, and integration with local services—to further enhance the rental experience. Ultimately, HouseHunt lays a solid foundation for transforming the way people find and secure rental properties in the digital age