HouseHunt – Finding Your Perfect Rental Home

# Team Name

Team HouseHunt

# Team ID:

LTVIP2025TMID55781

# Team Members:

Athuluru Shreya Sree

Register Number: SBAP0048720  
Hall Ticket No.: 22AT1A0507  
College: G. Pullaiah College of Engineering and Technology

# Table of Contents

|  |  |
| --- | --- |
| S. No. | Section Title |
| 1. | Introduction |
| 2. | ProjectOverView |
| 3. | Technical Architecture |
| 4. | ER Diagram and Database Design |
| 5. | Application Flow |
| 6. | Roles and Responsibilities |
| 7. | Project Setup and Configuration |
| 8. | Advantages and Disadvantages |
| 9. | Conclusion |
| 10. | Future Scope |

# Introduction

HouseHunt is a MERN-stack based house rental application designed to help users find rental properties efficiently. The app simplifies the rental process for renters, owners, and administrators by offering an intuitive user interface, secure data management, and seamless communication features.

# Project Overview

HouseHunt enables users to:

- Browse available rental properties with detailed descriptions and images.  
- Search properties using filters like location, price, type, and amenities.  
- Contact property owners directly.  
- Manage property listings as owners.  
- Administer platform governance and user approvals.

A typical user journey includes user registration, property browsing, inquiry, booking, lease agreement negotiation, and move-in processes.

# Technical Architecture

HouseHunt follows a client-server model:

Frontend:  
- React.js for UI development  
- Axios for API requests  
- Bootstrap, Material UI, Ant Design for responsive design

Backend:  
- Node.js with Express.js for server-side logic  
- MongoDB for data storage

Libraries Used:  
- Moment.js for date/time operations  
- Multer for file uploads (e.g., property images)  
- JSON Web Tokens (JWT) for authentication

This architecture ensures real-time communication, seamless data exchange, and a rich user experience.

# ER Diagram and Database Design

HouseHunt uses three primary collections in MongoDB:

Users:  
- \_id (MongoDB ObjectId)  
- name  
- email  
- password  
- type (Renter, Owner, Admin)

Property:  
- \_id (MongoDB ObjectId)  
- userId (Owner reference)  
- prop.Type  
- prop.AdType  
- isAvailable  
- prop.Address  
- ownerContact  
- prop.Amt  
- prop.images  
- add.Info

Booking:  
- \_id (MongoDB ObjectId)  
- propertyId  
- userId (Renter reference)  
- ownerId  
- username

# Application Flow

Scenario – Renting an Apartment:

- Registration: Alice signs up as a Renter.  
- Browsing Properties: Alice browses listings filtered by location, rent, and bedrooms.  
- Property Inquiry: Alice views property details and contacts the owner via the app.  
- Booking Confirmation: The owner reviews Alice’s inquiry and approves the booking.  
- Admin Approval: Admins validate new owner accounts for legitimacy.  
- Owner Management: Owners add, update, or remove properties and manage availability.  
- Transaction & Lease: Renters and owners negotiate lease terms within the app’s messaging system.  
- Move-in Process: The rental process concludes when the renter moves into the property.

# Roles and Responsibilities

Renter:  
- Register and log in.  
- Browse and filter property listings.  
- Contact owners and submit inquiries.  
- Track booking statuses.

Owner:  
- Obtain admin approval.  
- Add, update, or delete property listings.  
- Respond to renter inquiries.  
- Manage booking statuses.

Admin:  
- Approve owners’ accounts.  
- Monitor platform activity.  
- Enforce policies and privacy regulations.

# Project Setup and Configuration

Pre-requisites:  
- Node.js & npm  
- Express.js  
- MongoDB  
- Moment.js  
- React.js  
- Ant Design, Material UI, Bootstrap  
- HTML, CSS, JavaScript knowledge

Setup Instructions:  
- Clone repository  
- Install frontend dependencies:  
 cd frontend  
 npm install  
- Install backend dependencies:  
 cd ../backend  
 npm install  
- Start development server:  
 npm start

App accessible at http://localhost:3000

# Advantages and Disadvantages

Advantages:  
- User-friendly property browsing  
- Secure and scalable architecture  
- Real-time communication between renters and owners  
- Efficient admin controls for platform safety  
- Fast and responsive UI with modern design libraries

Disadvantages:  
- Dependence on internet connectivity  
- Initial learning curve for managing backend/admin panel  
- Limited advanced features like payment integration in early versions

# Conclusion

HouseHunt successfully simplifies the rental process by leveraging the MERN stack to provide seamless interactions for renters, owners, and admins. The project demonstrates modern web application practices and delivers a scalable solution for property rental management.

# Future Scope

- Payment gateway integration for online rent transactions  
- Advanced analytics dashboard for owners and admins  
- Integration with map services for enhanced property search  
- Multi-language support for broader reach  
- Push notifications for real-time updates