PROJECT DOCUMENT: HouseHunt - Finding Your Perfect Rental Home

---

INTRODUCTION

HouseHunt is a professionally designed full-stack rental housing web application aimed at simplifying and digitizing the process of renting properties. It provides a centralized platform for three primary users: property owners, renters, and platform administrators. This system introduces seamless communication, efficient property management, and transparent rental procedures to create a user-centric rental ecosystem.

PROJECT OBJECTIVE

The core objective of HouseHunt is to create a digital platform where:

- **Owners** can list, manage, and update rental properties
- **Renters** can browse, search, and book properties based on preferences
- **Admins** can monitor platform activity, verify users, and approve property listings

HouseHunt ensures a secure, efficient, and user-friendly interface for all stakeholders.

SYSTEM FEATURES

General Features:

- Fully responsive UI
- Image upload with preview
- Role-based authentication (Admin, Owner, Renter)
- Real-time booking status updates

Admin-Specific:

- Admin dashboard for verification and management
- Owner approval system
- Platform-wide monitoring of users and listings

Owner-Specific:

- Add/Edit/Delete properties with images
- View booking requests
- Update availability status

Renter-Specific:

- Apply filters to search properties
- Submit booking requests
- Track property status

USE CASE SCENARIO: RENTING AN APARTMENT

```
graph TD
A[User Registration] --> B[Login as Renter]
B --> C[Browse Listings with Filters]
C --> D[Submit Inquiry for a Property]
D --> E[Owner Reviews Inquiry]
E --> F[Owner Approves or Rejects]
F --> G[Booking Confirmed]
G --> H[Lease Agreement Finalized]
H --> I[Renter Moves In]
```

TECHNOLOGICAL STACK

Layer	Technology
Frontend	React.js, Axios, Bootstrap, MUI
Backend	Node.js, Express.js
Database	MongoDB via Mongoose
Authentication	JWT Tokens
File Uploads	Multer Middleware

DATABASE DESIGN

```
erDiagram
    USERS ||--o{ PROPERTIES : owns
    USERS ||--o{ BOOKINGS : makes
    PROPERTIES ||--o{ BOOKINGS : has

    USERS {
        string _id
        string name
```

```
        string email
        string password
        string type (Admin/Owner/Renter)
        string granted
    }
    PROPERTIES {
        string _id
        string propertyType
        string address
        number rent
        string[] images
        string ownerId
        string isAvailable
    }
    BOOKINGS {
        string _id
        string renterId
        string ownerId
        string propertyId
        string status
    }
```

PREREQUISITES

To set up and run HouseHunt locally, the following tools and environments are required:

- Node.js and npm
- MongoDB (Local or Cloud via MongoDB Atlas)
- Visual Studio Code (or any IDE)
- Postman (for API testing)
- Git for version control

SYSTEM ARCHITECTURE

```
graph LR
Client[React Frontend] -- Axios/HTTP --> Server[Express API Server]
Server --> MongoDB[(MongoDB Database)]
Server --> UploadsFolder[Image Storage (Multer)]
```

FOLDER STRUCTURE OVERVIEW

```
HouseHunt/
├── backend/
│   ├── controllers/
│   ├── routes/
│   ├── models/
│   ├── config/
│   └── index.js
├── frontend/
│   ├── src/
│   │   ├── modules/
│   │   ├── components/
│   │   ├── App.js
│   │   └── index.js
├── uploads/
└── .env
```

APPLICATION FLOW

Renter

1. Register/Login
2. Browse available properties
3. Submit booking request
4. Track booking confirmation

Owner

1. Register (pending admin approval)
2. Login and access owner dashboard
3. List properties with details and images
4. Manage bookings

Admin

1. Login via secured credentials
2. Approve/deny owners and listings
3. Manage users and monitor platform data

DEVELOPMENT MILESTONES

Phase 1: Setup and Configuration

- Folder structure created for backend and frontend
- Installed dependencies (express, mongoose, cors, multer, etc.)

Phase 2: Backend APIs and Auth

- Developed RESTful APIs for property and user management
- JWT implementation for secure auth
- Multer setup for image handling

Phase 3: Frontend Integration

- Component-based UI with routing and role-based views
- Axios integrated with backend
- Property card design and listing display

Phase 4: Testing and Deployment

- Manual testing via Postman and browser
- MongoDB Compass for DB verification
- Deployed locally (optionally to cloud hosting)

CONCLUSION

HouseHunt presents a robust and modern solution to the traditional property rental process. By integrating frontend flexibility, backend robustness, and real-time database handling, it creates a professional-grade rental application system. It bridges the gap between owners and renters while enabling secure and efficient operations through the admin panel.

Future improvements may include:

- Integration with payment gateways
- SMS/email notifications
- Real-time chat between owner and renter
