HouseHunt – Full Stack MERN Project Documentation

---

1. Introduction

- **Project Title**: HouseHunt
- **Team Members**:
    - Nikhil Nemala (Team Leader)
    - Likhita Maddala (Team Member)
    - Konkipudi Lakshmi Likhita (Team Member)
    - Madanala Srujitha (Team Member)

---

2. Project Overview

- **Purpose**: HouseHunt is a role-based real estate web application where renters can find and book verified rental properties, owners can list their properties, and admins can manage and moderate users and listings.
- **Features**:
    - User registration and login for renters, owners, and admins
    - Property listing and booking
    - Admin approval of owner accounts
    - Role-based dashboard views
    - Image upload for properties
    - Protected routes using JWT

---

3. Architecture

- **Frontend (React)**:
    - Modular components for common, admin, owner, and renter users
    - Axios for API calls
    - MUI, AntD, Bootstrap for styling
- **Backend (Node.js + Express.js)**:
    - REST API structure
    - Middleware for JWT auth and role-based access
    - Route separation for admin, owner, and user

- **Database (MongoDB)**:
    - Collections: Users, Properties, Bookings
    - Mongoose models for schema validation

---

4. Setup Instructions

- **Prerequisites**:
    - Node.js
    - MongoDB (Compass or Atlas)
- **Installation**:

```
# Clone the repo
https://github.com/your-repo/HouseHunt.git

# Backend setup
cd backend
npm install

# Frontend setup
cd ../frontend
npm install

# Add environment variables in backend/.env:
PORT=8001
MONGO_DB=mongodb://127.0.0.1:27017/househunt
JWT_SECRET=yourSecretKey
```

---

5. Folder Structure

- **Client (frontend)**:
    - /src/modules → Pages grouped by user type
    - /src/images → Static assets
    - App.js, index.js → Routing and app entry
- **Server (backend)**:

- /routes → Separated routes for user, owner, admin
- /controllers → Business logic functions
- /schemas → Mongoose models
- /middlewares → Auth and role protection
- /uploads → Uploaded images

---

6. Running the Application

Backend

cd backend

npm run dev

Frontend

cd ../frontend

npm start

---

7. API Documentation

User Routes (/api/user):

- POST /register
- POST /login
- GET /getAllProperties
- POST /bookinghandle/:propertyid

Admin Routes (/api/admin):

- GET /getallusers
- POST /handlestatus
- GET /getallproperties
- GET /getallbookings

Owner Routes (/api/owner):

- POST /addproperty
- POST /updateproperty/:propertyid
- GET /getallownerproperties

---

8. Authentication

- JWT token-based auth
- authMiddleware.js protects routes
- Token stored on frontend and passed via Authorization: Bearer header
- Owner access granted only after admin approval (granted flag)

---

9. User Interface

- React with Material UI, AntD, and Bootstrap
- Separate dashboards for Admin, Owner, and Renter
- Image upload and dynamic booking status view

---

10. Testing

- Manual testing for all user flows (Register, Login, Bookings, Approvals)
- Postman used for backend API validation

---

11. Known Issues

- Admin actions require token header manually in Postman
- No OTP/email verification currently
- Image upload not compressed

---

12. Future Enhancements

- Role-based email verification
- Payment gateway integration
- Chat between owners and renters
- Admin analytics dashboard
- Mobile responsive view and PWA