Project Documentation

1. Introduction

Project Title:

HouseHunt: Finding Your Perfect Rental Home

Team Members:

1. Nootheti Santhosh Frontend Developer

2. Gorise Nokesh Backend Developer

3. R. Chandu Kumar Database Administrator

4. D. Aisha Thabassum UI/UX Designer & Tester

2. Project Overview

Purpose:

HouseHunt is a web-based rental platform aimed at simplifying the house-hunting process for tenants while streamlining property management for owners and administrators.

Features:

- User Registration and Login

- Property Listings with Search Filters

- Property Inquiry and Booking Confirmation

- Admin Approval for Listings

- Owner and Property Management Dashboard

- Lease Agreement & Move-In Process

- Contact Property Owners Directly

Project Documentation

3. Architecture

Frontend:

Developed using React.js with component-based structure, React Router for navigation, and TailwindCSS for styling.

Backend:

Built with Node.js and Express.js, providing RESTful APIs for user authentication, property listings, and transactions.

Database:

MongoDB is used to store user details, property listings, bookings, lease agreements, and transaction history. Mongoose is used for schema modeling.

4. Setup Instructions

Prerequisites:

- Node.js (v14 or above)
- MongoDB
- Git

Installation:

```
# Clone the repo
git clone https://github.com/your-repo/househunt.git
cd househunt
```

```
# Install frontend dependencies
cd client
npm install

# Install backend dependencies
cd ../server
npm install

# Add environment variables in `.env` file (MongoDB URI, JWT secret, etc.)
```

5. Folder Structure

Client:

```
client/
 public/
 src/
    components/
    pages/
    services/
    App.js
    index.js
```

Server:

```
server/
```

controllers/

models/

routes/

middleware/

.env

server.js

6. Running the Application

Frontend:

cd client

npm start

Backend:

cd server

npm start

7. API Documentation

User Authentication

- POST /api/auth/register Register a new user

- POST /api/auth/login Authenticate user and return JWT

Property Listings

- GET /api/properties Get all properties

- POST /api/properties Add new property (owner only)

- PUT /api/properties/:id Update property details

- DELETE /api/properties/:id Remove property

Booking

- POST /api/bookings Book a property

- GET /api/bookings/:userId User booking history

8. Authentication

Authentication is handled using JWT (JSON Web Tokens). Tokens are stored securely and validated in protected routes via middleware.

9. User Interface

Key UI Pages:

- Home Page with Featured Listings

- Registration/Login

- Property Search and Filter

- Booking Confirmation Modal

- Admin Dashboard

(Screenshots or screen recording to be added)

10. Testing

Project Documentation

Tools Used:

- Postman for API Testing

- Jest for unit testing backend services

- React Testing Library for UI components

11. Screenshots or Demo

Demo Link: [Add link here]

Screenshots:

(Add relevant screenshots of pages like login, listings, booking etc.)

12. Known Issues

- Search filters may lag with large dataset

- Booking history not updating in real-time without refresh

- Mobile responsiveness improvements needed

13. Future Enhancements

- Implement Chat System between Tenant and Owner

- Add Payment Gateway Integration

- Enable SMS/Email Notifications

- Mobile App version using React Native

- Support for multi-language interface