Full-Stack Rental Property App – Detailed Project Documentation

# 1. Introduction

The Full-Stack Rental Property App is a comprehensive web platform designed to simplify the process of finding, viewing, and managing rental properties. Built using the MERN stack (MongoDB, Express.js, React.js, Node.js), this application ensures seamless interaction between the frontend and backend components, delivering a modern and responsive user experience.

# 2. Project Objective

The primary goal of this application is to bridge the gap between property owners and potential tenants. It offers features like property listing, detailed view of properties, user authentication, and a secure API for managing backend resources.

# 3. Technology Stack

- Frontend: React.js, HTML5, CSS3, JavaScript  
- Backend: Node.js, Express.js  
- Database: MongoDB (NoSQL)  
- Libraries/Tools: Axios, Mongoose, React Router, Dotenv, Nodemon

# 4. Frontend Architecture

The frontend is built using React.js. It utilizes component-based architecture to provide a modular and maintainable codebase.  
  
Key Files:  
- App.js: Handles application routing using React Router.  
- pages/Home.js: Displays the home page with a list of available rental properties.  
- pages/PropertyDetails.js: Renders detailed information for a selected property.  
  
The frontend interacts with the backend APIs using Axios to fetch and post data.

# 5. Backend Architecture

The backend is developed with Node.js and Express.js. It exposes RESTful APIs for frontend consumption and handles business logic, authentication, and database operations.  
  
Key Files:  
- server.js: Initializes the Express server and connects to MongoDB using Mongoose.  
- routes/propertyRoutes.js: Manages property-related routes (e.g., GET, POST, PUT, DELETE).  
- routes/authRoutes.js: Handles user authentication and authorization routes.  
- models/Property.js: Mongoose schema for storing property data.  
- models/User.js: Mongoose schema for storing user information including hashed passwords.

# 6. Key Features

- Browse available rental properties with images and descriptions.  
- View detailed information for each property.  
- User registration and login with secure password hashing.  
- RESTful API endpoints for CRUD operations on properties.  
- Modular and scalable project structure.

# 7. Data Flow Overview

1. User opens the website and views available properties (GET request).  
2. User selects a property to view more details (GET request to property/:id).  
3. If authenticated, user can manage property listings (POST, PUT, DELETE).  
4. Frontend sends requests to backend APIs, which interact with MongoDB for data retrieval or update.

# 8. Future Enhancements

- Add a booking system with calendar integration.  
- Admin panel for approving/rejecting property listings.  
- Email verification and password reset functionality.  
- Add property filtering and search capabilities.  
- Mobile responsiveness improvements and PWA (Progressive Web App) support.

# 9. Conclusion

This application provides a solid foundation for a modern rental property management system. With its modular architecture, clean API structure, and user-centric design, it can be expanded into a fully-featured real estate rental platform.