HOUSE RENT APP USING MERN

1. Introduction

Project Title: House Rent App using MERN

TEAM MEMBERS:

|  |  |  |  |
| --- | --- | --- | --- |
| S.NO | NAME | ROLE | RESPONSIBILITIES |
| 1. | Devaraj S | Full-Stack  Developer | Responsible for overall frontend and backend design |
| 2. | Bhavani | Frontend  Developer | Handled UI and UX design, ensuring a user-friendly interface using React. |
| 3. | Balamurugan | Database  Administrator | Managed database integration and functionality using MongoDB. |
| 4. | Balaji | Backend Developer | Developed and maintained the Node.js backend and API connections. |

2. Technologies Used

1. Frontend (Client-Side): React.js

* Library for building user interfaces
* Features: Virtual DOM, JSX, Components
* Benefits: Fast, scalable, and maintainable

2. Backend (Server-Side): Node.js, Express.js

* Node.js: JavaScript runtime environment
* Features: Event-driven, non-blocking I/O
* Benefits: Fast, scalable, and asynchronous
* Express.js: Node.js web framework
* Features: Routing, Middleware, Templating
* Benefits: Flexible, lightweight, and modular

3. Database: MongoDB

* NoSQL document-oriented database
* Features: Document-based data model, Indexing, Replication
* Benefits: Scalable, flexible, and high-performance

4. Authentication: JSON Web Tokens (JWT)

* Secure authentication standard
* Features: Digital signatures, Token-based authentication
* Benefits: Secure, compact, and self-contained

3. Project Overview

The House Rent App is a web-based application designed to facilitate house hunting and rental processes. The app allows users to search, filter, and book rental properties, while property owners can list and manage their properties.

Features:

User Features

1. Registration and Login

* Register with email, phone number
* Login securely with password or OTP

2. Search and Filter Properties

* Search by location (city, state, zip)
* Filter by price range, property type, and amenities (pool, gym, etc.)
* Sort results by price, rating, and availability

3. View Property Details and Images

* View property photos and virtual tours
* Check property features, amenities, and descriptions
* View property location on map

4. Book Properties and Pay Rent Online

* Book properties with secure online payment
* Choose rental duration and payment plan
* Receive booking confirmation and receipt

5. View Booking History and Manage Profile

* View past and upcoming bookings
* Manage profile information and settings
* Edit payment methods and preferences

Property Features

1. Property Listing with Images and Details

* High-quality property photos and virtual tours
* Detailed property descriptions (size, layout, features)
* Property amenities (pool, gym, parking, etc.)

2. Filtering

* Filter by price range ($/month)
* Filter by location (city, state, zip)
* Filter by amenities (pool, gym, pet-friendly, etc.)

3. Sorting

* Sort by price (low-high, high-low)
* Sort by rating (highest-lowest)
* Sort by availability (immediate, upcoming)

4. Property Reviews and Ratings

* Tenant reviews and ratings (1-5 stars)
* Review comments and feedback
* Average rating and review count display

4. System Architecture

Client-Side (React.js)

* User Interface (UI) and User Experience (UX)
* API calls to Backend Server for data retrieval and manipulation

Features:

* User registration and login
* Property search and filtering
* Booking and payment processing
* User profile management

Server-Side (Node.js, Express.js)

* API Endpoint Handling
* Data Storage and Retrieval (MongoDB)
* Authentication and Authorization

Features:

* API routing and middleware
* Data validation and sanitization
* JWT-based authentication
* Authorization for user roles (admin, user, property owner)

Database (MongoDB)

* Stores user, property, and booking data
* NoSQL document-oriented database

Features:

* Scalable and flexible data storage
* High-performance data retrieval
* Data replication and backup

5. Setup Instructions

Prerequisites:

* + Node.js: Make sure Node.js is installed. This is needed to run the application locally and install necessary dependencies.
  + MongoDB: Ensure that you have a running instance of MongoDB. You can use either a local MongoDB setup or a cloud-based solution like MongoDB Atlas.
  + npm: The Node Package Manager (npm) is required to install dependencies.

Installation and Setup

To run this project locally, follow these steps:

1. Clone the repository:

bash

git clone https://github.com/Deva0703/House-Rent-app-using-MERN.git

2. Navigate to the project directory:

bash

cd house-rent-app

3. Install dependencies for both client and server:

bash

cd client

npm install

cd ../server

npm install

4. Start the development server:

bash

cd client

npm start

bash

cd ../server

npm run dev

The application will be available at http://localhost:3000 (frontend) and http://localhost:5000 (backend).

6. Running the Application

Frontend: To start the frontend server, run the following command in the client folder:

npm start

Backend: To start the backend server, run the following command in the server folder:

npm start

7. API Documentation

API Endpoints

User Endpoints

1. Register User

URL: /api/users/register

Method: POST

Request Body:

name: string

email: string

password: string

Response:

token: string (JWT token)

user: object (user details)

2. Login User

URL: /api/users/login

Method: POST

Request Body:

email: string

password: string

Response:

token: string (JWT token)

user: object (user details)

3. Get User Profile

URL: /api/users/profile

Method: GET

Headers:

Authorization: Bearer <token>

Response:

user: object (user details)

Property Endpoints

1. Create Property

URL: /api/properties

Method: POST

Request Body:

title: string

description: string

price: number

location: string

Response:

property: object (property details)

2. Get All Properties

URL: /api/properties

Method: GET

Response:

properties: array (list of properties)

3. Get Property by ID

URL: /api/properties/:id

Method: GET

Response:

property: object (property details)

Booking Endpoints

1. Create Booking

URL: /api/bookings

Method: POST

Request Body:

propertyId: string

startDate: date

endDate: date

Response:

booking: object (booking details)

2. Get All Bookings

URL: /api/bookings

Method: GET

Response:

bookings: array (list of bookings)

3. Get Booking by ID

URL: /api/bookings/:id

Method: GET

Response:

booking: object (booking details)

Payment Endpoints

1. Make Payment

URL: /api/payments

Method: POST

Request Body:

bookingId: string

amount: number

Response:

payment: object (payment details)

8. Authentication

JWT Authentication: In this system, users (patients and doctors) authenticate by logging in and receiving a JWT (JSON Web Token). This token must be included in the authorization header of all future requests to protected routes. The token serves as proof that the user is authenticated and ensures that only authorized users can access specific resources, like rent a house or accessing the dashboard.

Authorization Middleware:

* Authentication Middleware: This middleware verifies whether a user is logged in by checking the validity of the JWT token. If the token is valid, the user is allowed to proceed; otherwise, access is denied.
* Authorization Middleware: Once the user’s identity is verified through authentication, this middleware checks if the user has the appropriate role to access a certain route. For example, only property owners should be able to access owners specific routes (e.g., managing their schedule), while the users should be restricted to rent a house.

9. User Interface

The User Interface (UI) of the House Rent App with MERN application is designed to provide a smooth and intuitive experience for the users. Below are the main UI components:

Login Page:

* + - Users can log in using their email and password.
    - Features a clean design with clear input fields and a login button.
    - Responsive design for mobile and desktop screens.

Dashboard (Customer):

* + - Displays a list of available house with basic information like category, specialty, and availability.
    - Option to rent a house by selecting a house, date, and location.

Booking House:

* + - After selecting a house, the user is directed to a form to rent the house.
    - The form includes location, category of house and price.

Admin Dashboard:

* Admins can manage user applications, monitor all user bookings, and enforce platform policies.

GIFs showing the user flow and interactions (e.g., login, booking process, dashboard navigation) can also be provided.

10. Testing

Testing Strategy:

1. Unit Testing

Jest: JavaScript testing framework

Enzyme: React testing utility

Test cases:

* Component rendering
* API calls
* Redux actions and reducers
* Utility functions

2. Integration Testing

Postman: API testing tool

MongoDB Compass: Database testing tool

Test cases:

* API endpoints
* Database queries
* Authentication and authorization
* User workflows

Deployment

1. Cloud Platform

Heroku: Cloud platform for Node.js applications

Features:

* Scalability
* Reliability
* Security

Testing Tools:

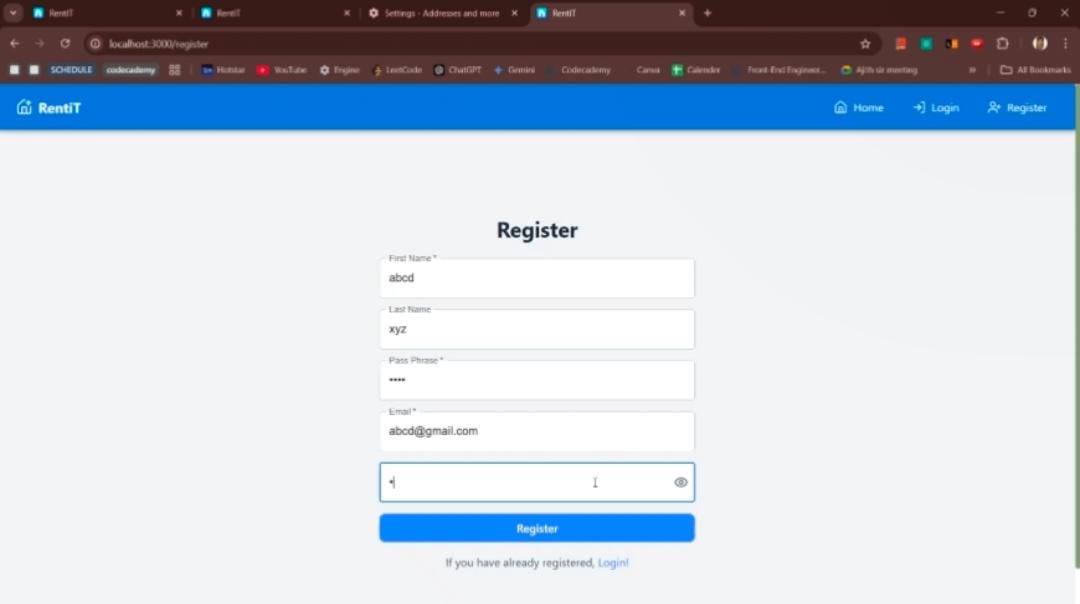
Frontend: Jest, Enzyme, React Testing Library for unit and integration testing of React components.

Backend: Mocha, Chai, Supertest for testing Express.js API routes.

API Testing: Postman for manually testing backend endpoints.

11. Screenshots

Register page:



12. Implementation Challenges

Integrating Payment Gateway

1. Choosing a payment gateway (e.g., Stripe, PayPal)

2. Implementing payment processing logic

3. Handling payment failures and errors

4. Integrating with MongoDB for payment history storage

5. Ensuring PCI-DSS compliance

Implementing Search and Filter Functionality

1. Designing a robust search algorithm

2. Integrating with MongoDB for data retrieval

3. Handling filtering and sorting logic

4. Optimizing search performance

5. Implementing autocomplete and suggestions

Ensuring Data Security and Authentication

1. Implementing JWT-based authentication

2. Encrypting sensitive data (e.g., passwords, payment info)

3. Validating user input and preventing SQL injection

4. Implementing role-based access control (RBAC)

5. Regularly updating dependencies and patches

13. Future Enhancement

There are several potential improvements and features that could be added to the Rent a House using MERN application:

Mobile App Integration:

* Developing native mobile applications for iOS and Android to enhance the user experience on mobile devices.

Payment Integration:

* Integrating payment gateways (e.g., Stripe or PayPal) to enable users to pay for rent a house.

Real-Time Chat:

* Implementing a messaging system that allows users and property owners to communicate directly before, during, or after rent a house.

AI-powered house Recommendations:

* Using AI and machine learning algorithms to recommend suitable house to users based on their intrest, needs and preferences.