Naan Mudhalvan

Project Report

House Rent App Using the MERN Stack

Team Member

1.Elanchezhiyan M

2.Gopala Krishnan R B

3.Haripriya

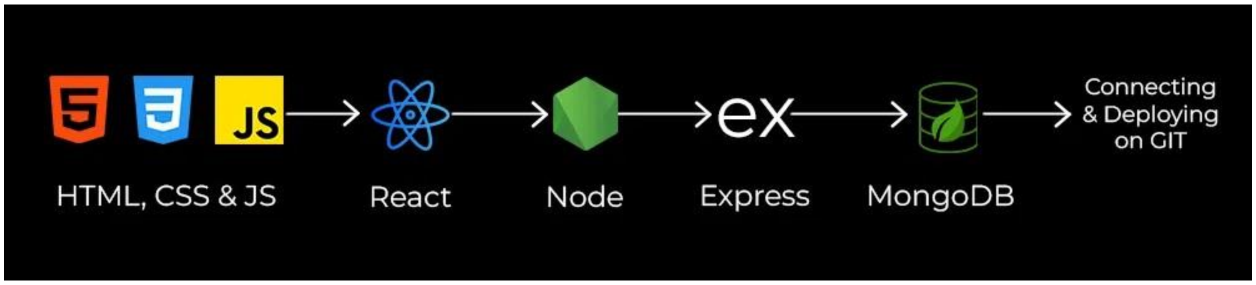
4.Inbarasi

1. Introduction

This report outlines the design and development of a **House Rent App** using the **MERN stack** (MongoDB, Express.js, React, Node.js). The app allows users to rent houses, manage property listings, and handle user authentication. It offers two primary user roles: **Landlords** (who can list properties) and **Tenants** (who can search and view available properties).

The MERN stack is a popular full-stack development framework used to build scalable, high-performance web applications. Each part of the stack provides specific tools that complement each other:

- **MongoDB**: A NoSQL database used to store user and property data.
- **Express.js**: A web application framework for Node.js, used to create the server-side API.
- **React**: A frontend library to build interactive user interfaces.
- **Node.js**: A JavaScript runtime that enables backend development with JavaScript.



2. Objectives

The goal of the House Rent App is to:

- Provide a platform where **landlords** can list their properties for rent.
- Allow **tenants** to browse available properties and inquire about them.
- Implement user authentication to manage access.
- Create an intuitive interface for both landlords and tenants.

---

3. System Architecture

The House Rent App follows a **client-server architecture** with a **single-page application (SPA)** frontend and a **RESTful API** backend. Below is a high-level overview of the system components:

- **Frontend (React)**: The user interface is developed using React.js. It interacts with the backend API through Axios for sending HTTP requests and receiving responses.

- **Backend (Node.js + Express)**: The backend is responsible for handling business logic, such as user registration, login, and property management. The backend exposes APIs for the frontend to interact with.
- **Database (MongoDB)**: MongoDB is used to store user information and property data. It is a document-oriented database that is well-suited for handling dynamic data and relationships between documents.

---

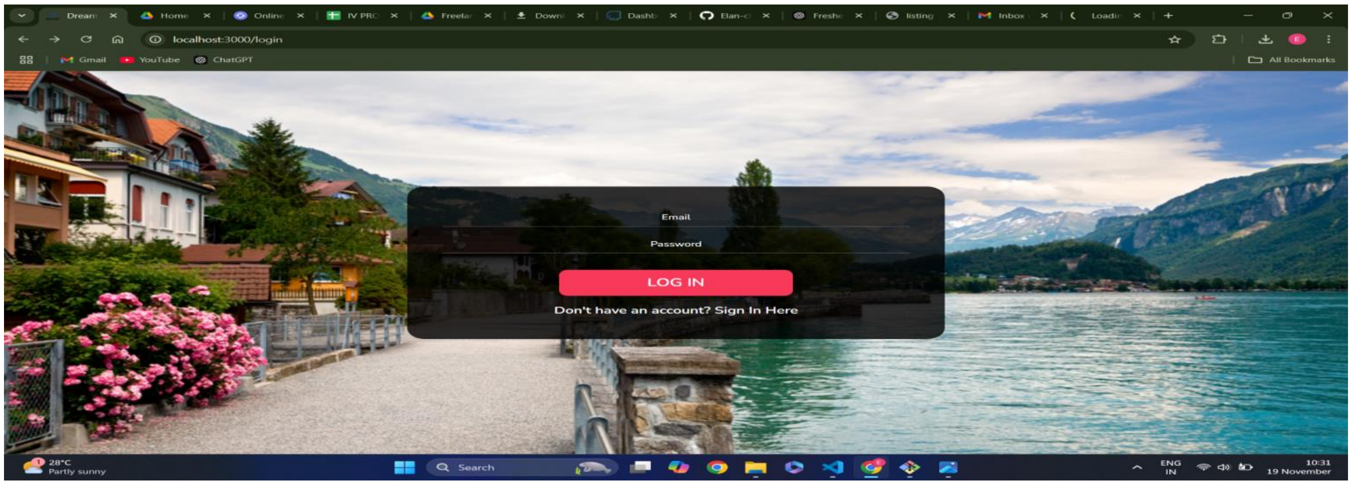
4. Technologies Used

- **MongoDB**: For data storage. It holds user and property data in collections like `users` and `properties`.
- **Express.js**: A web framework for Node.js used to handle routing and middleware for processing HTTP requests.
- **React**: A JavaScript library for building user interfaces. React is used to build the frontend of the application, which dynamically displays properties and manages user interactions.
- **Node.js**: A runtime environment for executing JavaScript code on the server-side.
- **Axios**: A promise-based HTTP client used to interact with the backend API from the frontend.
- **JWT (JSON Web Tokens)**: Used for authenticating users and managing secure sessions.
- **Bcrypt.js**: A library for hashing passwords before storing them in the database.

---

5. Features

The House Rent App provides the following key features:



1. **User Authentication**

   - **Registration**: Users can register with their name, email, password, and role (landlord or tenant).
   - **Login**: Users can log in with their credentials and receive a JWT token, which they use for authenticated requests.
   - **JWT-based Authentication**: The app uses JWT tokens to secure routes, ensuring only authorized users can access certain features (e.g., creating property listings).
2. **Property Listings**
   - **Landlords** can list properties by providing details such as the title, description, price, and location.
   - **Tenants** can view available properties, including details like price, description, and location.
3. **User Roles**
   - **Landlords**: Can add, update, and remove property listings.
   - **Tenants**: Can browse properties, filter based on criteria, and inquire about renting a property.
4. **Responsive UI**:
   - The frontend is designed to be mobile-friendly, ensuring users can access the app on various devices.
   -

---

6. Database Schema

- **User Model**
  - `name`: String (user's name)
  - `email`: String (user's email, must be unique)
  - `password`: String (hashed password)
  - `role`: String (either "tenant" or "landlord")
- **Property Model**
  - `title`: String (property title)
  - `description`: String (property description)
  - `price`: Number (price of the property per month)
  - `location`: String (location of the property)
  - `landlordId`: ObjectId (references the user who is the landlord of the property)
  - `imageUrl`: String (URL of the property image)

---

7. Backend (Node.js + Express)

The backend handles the business logic, such as authenticating users, adding and retrieving properties, and managing user sessions.

1. **User Routes (Authentication)**
   - `POST /api/auth/register`: Registers a new user.

    - `POST /api/auth/login`: Authenticates a user and generates a JWT.
2. **Property Routes**
    - `POST /api/properties`: Adds a new property (Landlord only).
    - `GET /api/properties`: Retrieves a list of all available properties for tenants.
3. **Authentication Middleware**
    - Ensures that only authenticated users can access certain routes, like posting properties.

---

8. Frontend (React)

The frontend provides an interface for users to interact with the app:

1. **Login and Registration**:
    - Users can create an account or log in to access the system.
    - JWT tokens are stored in localStorage to persist the user's session.
2. **Property Listing Page**:
    - A page that displays all available properties with search and filter options.
3. **Create Property Page (for Landlords)**:
    - A form where landlords can submit new property listings with details such as title, description, price, and location.
4. **Responsive Design**:
    - The app is built with responsive design principles, using CSS Grid and Flexbox, ensuring that the interface works well on both desktop and mobile devices.

---

9. Authentication and Security

The app uses **JWT (JSON Web Token)** to secure user sessions. When a user logs in, the backend issues a JWT that must be included in the `Authorization` header of subsequent requests. This token is then verified on the server to authenticate the user.

Password Security:

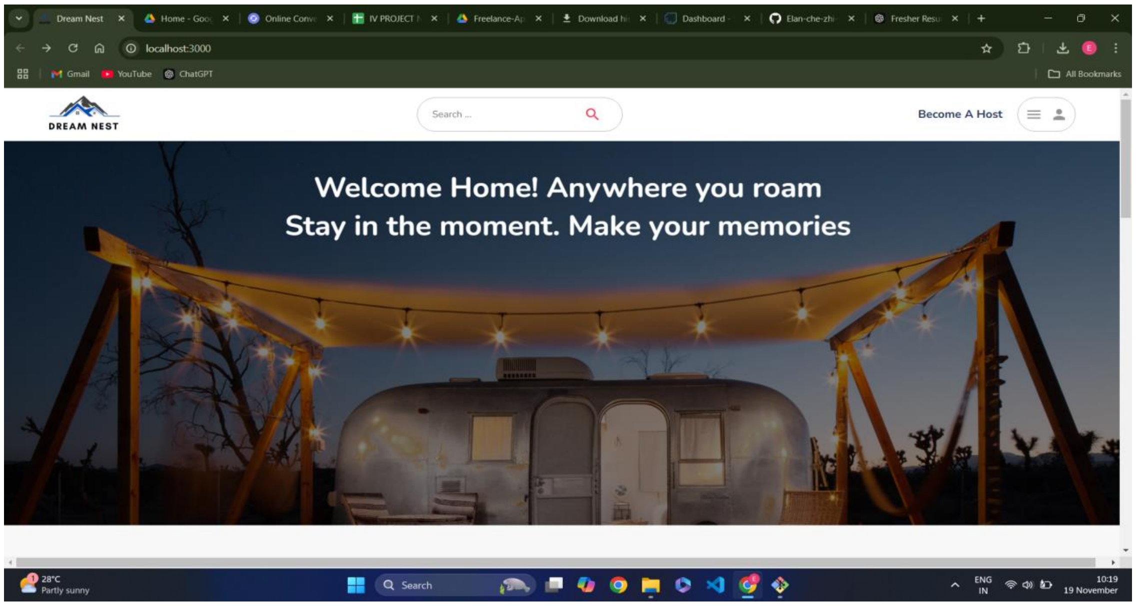
- Passwords are hashed using **bcryptjs** before being stored in the database. This ensures that even if the database is compromised, users' passwords remain secure.

---

10. Challenges Faced

- **Handling Authentication**: Ensuring secure authentication and authorization of users based on their roles (tenant or landlord) required careful implementation of JWT and middleware.

- **Responsive UI**: Designing a responsive UI that works seamlessly across different screen sizes involved using CSS frameworks like **Bootstrap** and custom media queries.
- **Error Handling**: Proper error handling in both the backend and frontend was crucial to provide clear feedback to users, especially during login and property creation.

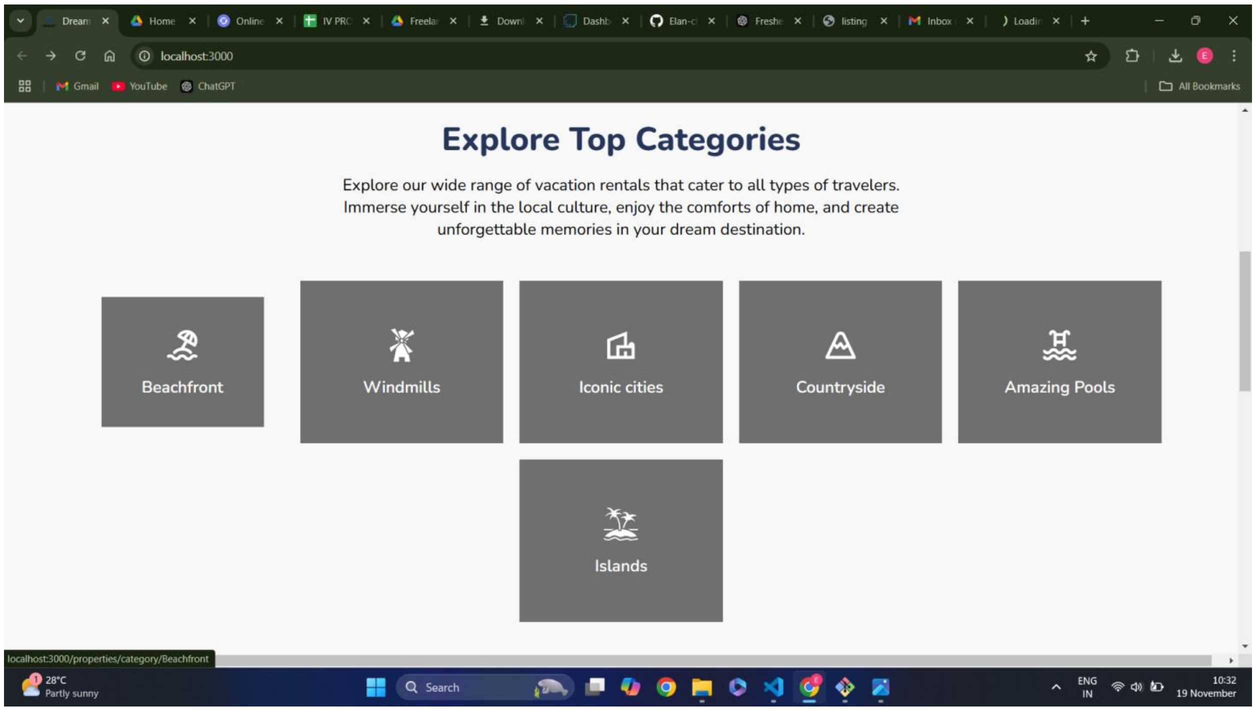

localhost:3000
DREAM NEST
Search ...
Become A Host
Welcome Home! Anywhere you roam
Stay in the moment. Make your memories

---

11. Conclusion

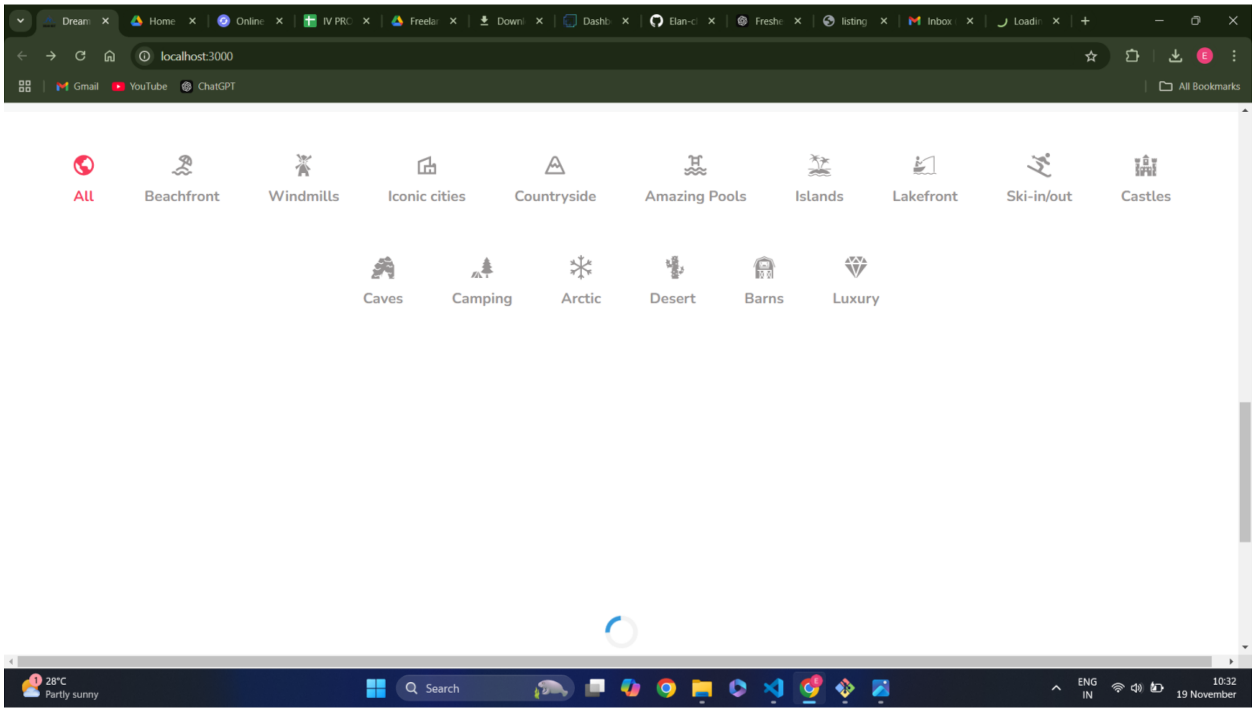
The House Rent App successfully demonstrates the use of the **MERN stack** to create a full-stack application for managing property rentals. By integrating MongoDB for data storage, Express.js for API handling, React for the frontend, and Node.js for the backend, this app provides a functional and scalable solution for landlords and tenants. The app supports secure user authentication and provides a simple, intuitive interface for managing property listings.

With additional features such as property search filters, messaging between tenants and landlords, and reviews, this app could be expanded into a full-fledged property rental platform.



Explore Top Categories

Explore our wide range of vacation rentals that cater to all types of travelers. Immerse yourself in the local culture, enjoy the comforts of home, and create unforgettable memories in your dream destination.



localhost:3000
Gmail
YouTube
ChatGPT
All Bookmarks
All
Beachfront
Windmills
Iconic cities
Countryside
Amazing Pools
Islands
Lakefront
Ski-in/out
Castles
Caves
Camping
Arctic
Desert
Barns
Luxury
28°C
Partly sunny
Search
ENG
IN
10:32
19 November