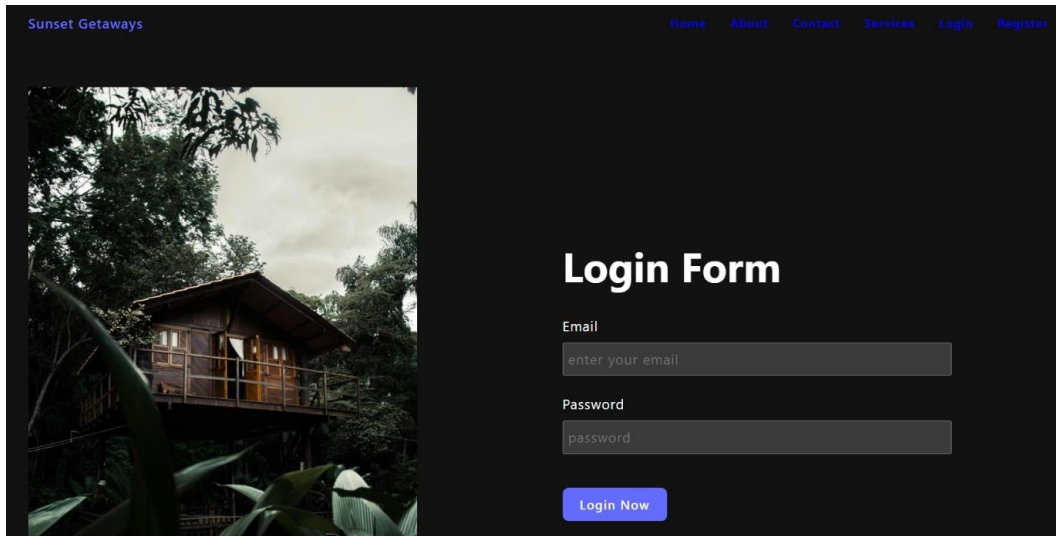


Task 1 : Vacation Homes and Rentals

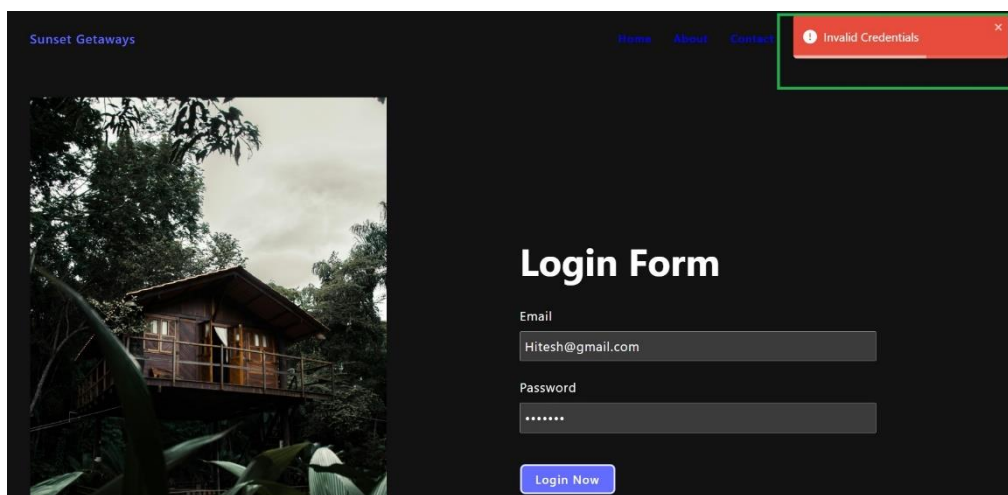


Login Page in React with Zod Validation

In this login page, React is used for building the user interface, and Zod is employed for validating the form inputs. The page consists of a simple form where users input their email and password to log in. Here's an overview of how it works:

1. React for UI Components

- **Form Setup:** The login form includes input fields for email and password. Users enter their credentials and submit the form to trigger validation and authentication logic.
- **State Management:** React's `useState` hooks manage form data and validation error states, ensuring the UI updates dynamically based on user input and validation outcomes.

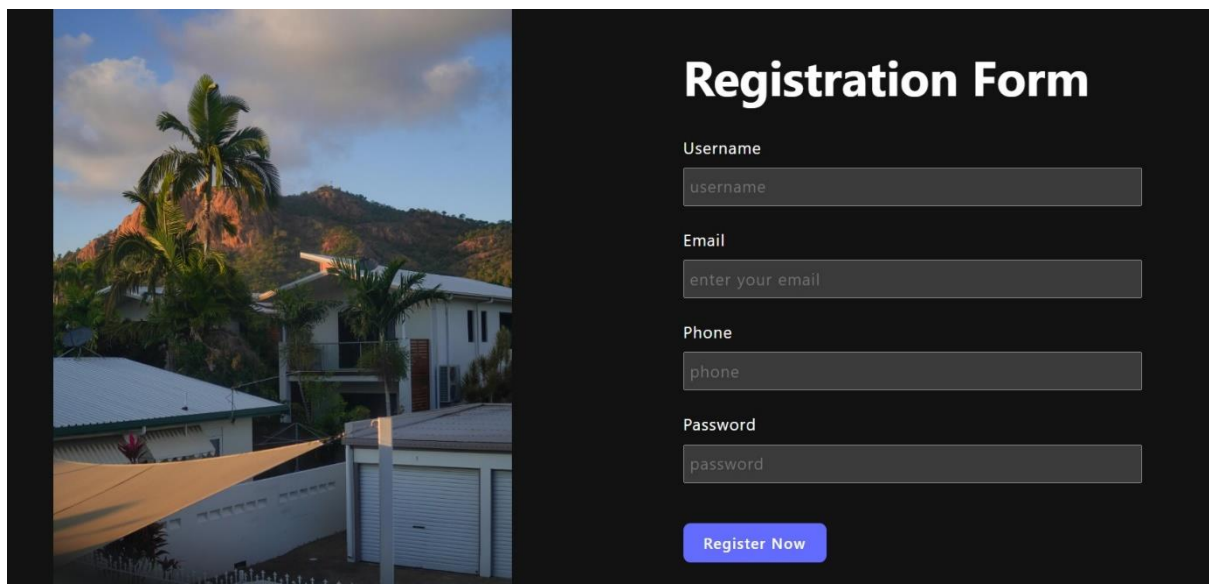


2. Zod for Validation

- **Schema Definition:** Zod is used to define a schema that enforces rules on the form inputs. For example, the email must be a valid email address, and the password should meet certain criteria (like minimum length, or specific characters).

3. Submission

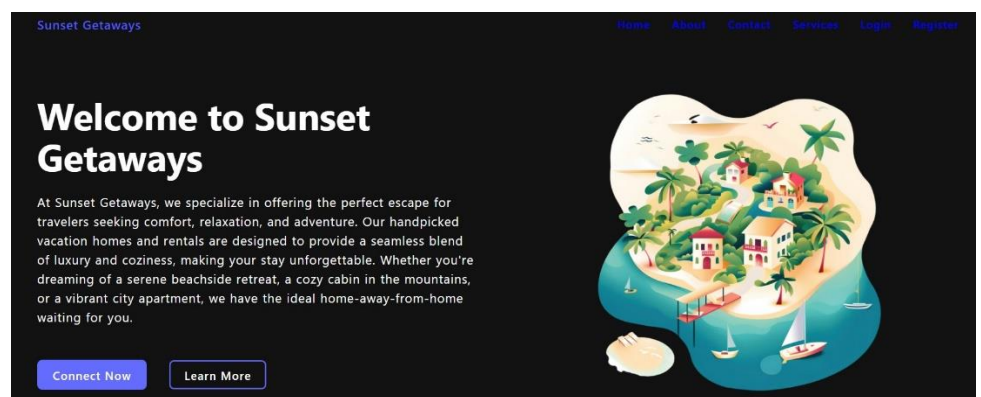
- Once the form passes validation, the login data (email and password) can be sent to an API or authentication service for processing.
- You can also handle other actions like redirecting users to a Home page on successful login.

The image shows a registration form on a dark background. On the left is a vertical image of a tropical house with palm trees. The form is titled 'Registration Form' in white. It contains four input fields: 'Username' with placeholder 'username', 'Email' with placeholder 'enter your email', 'Phone' with placeholder 'phone', and 'Password' with placeholder 'password'. Each field has a light gray border. Below the fields is a blue button labeled 'Register Now' in white text.

Registration Page in React with Zod Validation and Redirect

The registration page allows new users to create an account by entering details like username, email, phone number, and password. React is used to build the UI, and the Zod library handles form validation. Once the form is submitted and validated successfully, the user is redirected to the login page.

Home Page :-



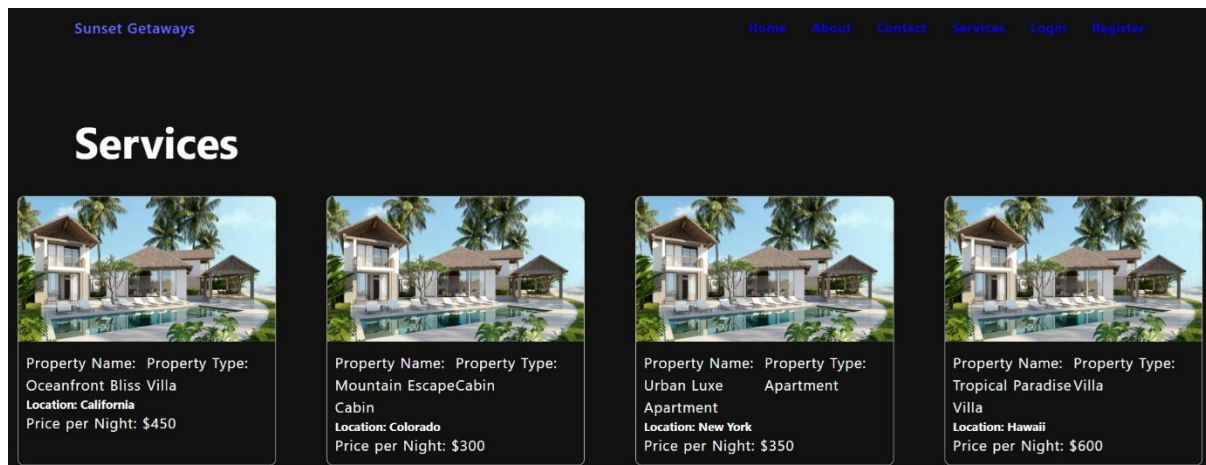
Token Based Authentication :

JWT (JSON Web Token) is used to authenticate users after login. Once a user successfully logs in, a JWT is generated and sent to the client. The token contains encoded user data (e.g., user ID) and is used to verify the user's identity on subsequent requests.

Bcrypt For Password Hashing :

Passwords are not stored in plain text in the database. Instead, bcrypt is used to hash the password securely before saving it. This ensures that even if the database is compromised, attackers cannot easily retrieve user passwords.

Property List :



How They Work Together

1. **Node.js and Express.js:** Handle server-side logic, routing, and API creation.
2. **JWT:** Integrated with Express middleware to manage authentication securely.
3. **MongoDB:** Stores user data, ensuring fast, flexible, and scalable data access.
4. **bcrypt:** Ensures secure password hashing on the backend before storage in MongoDB.