**Hotel Reservation System**

**1. Project Overview**

This is a backend project for a hotel reservation system that implements Authentication, Authorization, and Role-Based Access Control (RBAC). The system supports three roles: User, Admin, and SuperAdmin, each with distinct permissions.

**Description**: A backend application for hotel reservation management with three roles:

**User:** Can browse hotels and rooms, and make reservations.

**Admin:** Manages hotels, rooms, and user information.

**SuperAdmin:** Oversees all data, including admins, users, and reservations, and has extended privileges.

**Features:**

View hotel and room details.

Book rooms if available.

Manage hotels, rooms, and user details.

Access and manage reservation data.

**Tech Stack:**

Backend: Node.js (Express.js)

Database: MongoDB

Authentication: JWT (JSON Web Tokens)

Status: Completed (update as needed)

**2. Setup and Installation**

**Prerequisites**

Node.js (v16+)

MongoDB (v6+)

NPM or Yarn package manager

**Install dependencies:**

npm install express mongoose jsonwentoken bcryptjs dotenv

Create a .env file in the root directory with the following variables:

env

PORT= Your\_Port\_No.

DATABASE\_URL= Your\_MongoDB\_Connection\_String\_Here

JWT\_SECRET= Your\_Secret\_Key

**Start the server:**

node index.js

**3. Features:**

**Authentication:**

Secure user login and registration.

JWT-based session management.

**Authorization:**

RBAC ensures that each role can only access permitted endpoints.

Middleware for access control based on roles.

**Password Hashing:**

Passwords are securely hashed using bcrypt before saving into database

During login the input password is hashed to check that if it matches with hashed password stored in Database

**CRUD Operations:**

**Users:** View hotels and book rooms.

**Admins:** Manage hotels, rooms, and user details.

**SuperAdmins:** Manage all resources, including admins and users.

**Hotel and Room Management:**

Add, update, and delete hotel and room details.

Check room availability for booking.

**Reservation System:**

Book rooms with user information.

View reservations based on user ID (SuperAdmin).

**4 .System Architecture:**

Backend Framework: Node.js with Express.js.

Database: MongoDB for managing user, hotel, room, and reservation data.

Authentication: JWT for secure session management.

Environment Management: .env for sensitive configurations.

**Directory Structure:**

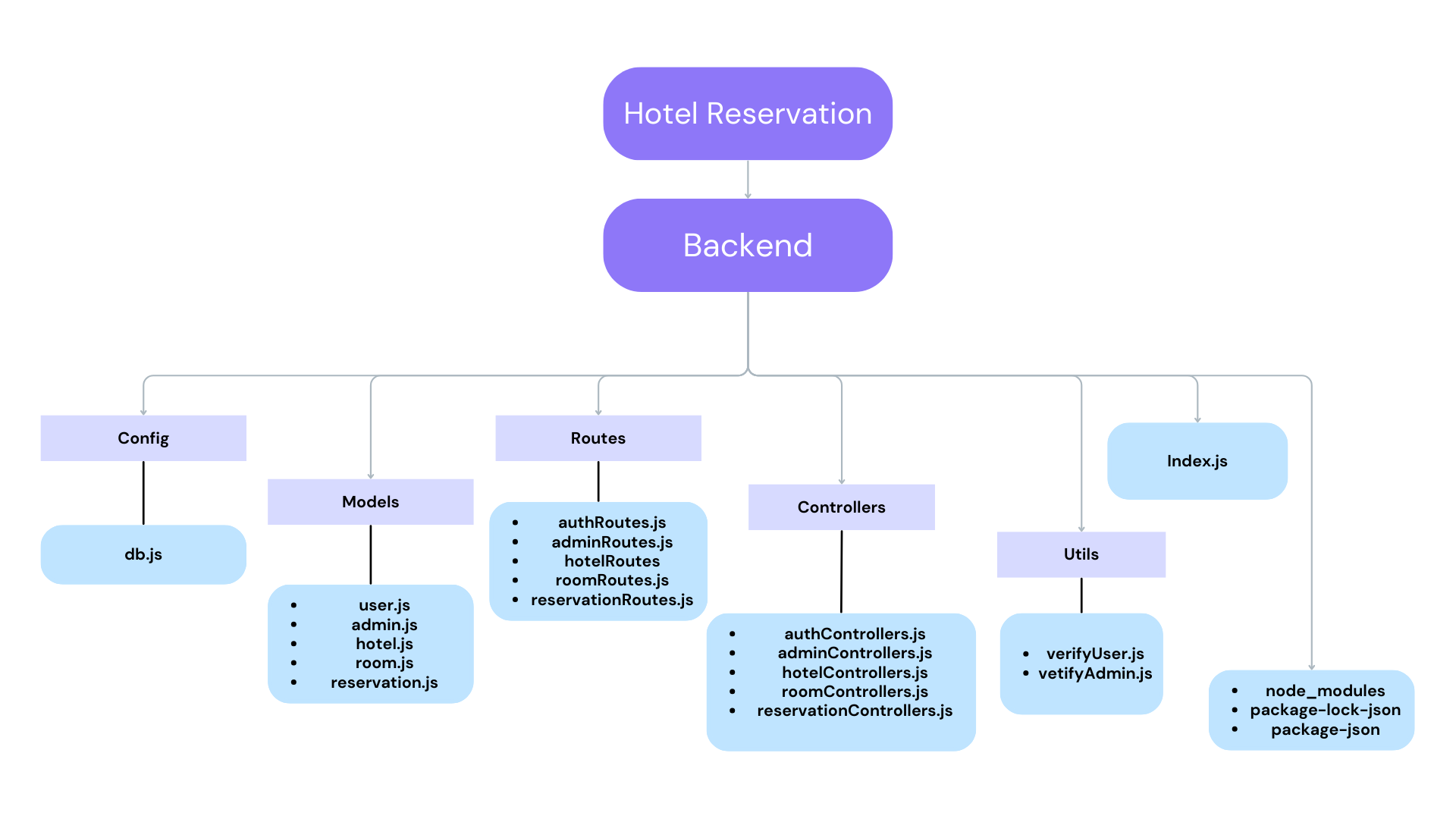
config: Database Configuration.

controllers: Logic for handling requests and responses.

models: MongoDB schemas for Users, Admins, Hotels, Rooms, and Reservations.

routes: Defines API endpoints and integrates role-based middleware.

utils: Utility functions, e.g., for validating tokens.



**5. Roles and Permissions**

| **Role** | **Permissions** |
| --- | --- |
| User | View hotels and rooms, book available rooms. |
| Admin | Add, update, delete hotels and rooms; view and delete users. |
| SuperAdmin | Manage all users and admins; view all reservations; perform admin actions. |

**6.Testing:**

**Manual Testing:**

Use tools like Postman to test each endpoint for authentication, authorization, and CRUD operations.

**7.API Endpoints:**

Endpoints by Role

**For Users:**

1. **Register:**

* **Endpoint:** /auth/register
* **Method:** POST
* **Description:** Registers the User’s Data in The Database
* **Body:** json

{

    "username":"karna",

    "email":"karna@gmail.com",

    "password":"password"

}

* **Response:**

{

    "message": "User has been created."

}

1. **Login :**

* **Endpoint:** /auth/login
* **Method:** POST
* **Description:** Check The User Details in the Database and generates a JWT Token
* **Body:** json

{

    "email":"karna@gmail.com",

    "password":"password"

}

* **Response:**

{

    "message": "Login Successful",

    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM"

}

1. **List the Hotel Details :**

* **Endpoint:** /hotel/view
* **Method:** GET
* **Description:** Fetches all the Hotel Data
* **Authorization:** none
* **Body:** none
* **Response:**

[

 {

        "\_id": "67444eccf244f41694dae4cf",

        "name": "Hotel Annamalai",

        "location": "Chennai",

        "rating": 4.7,

        "description": "Best Hotel in Tamil Nadu",

        "facilities": [

            "Free WiFi",

            "Swimming Pool",

            "Spa",

            "Fitness Center",

            "Restaurant",

            "Pub"

        ],

        "created\_at": "2024-11-25T10:17:48.050Z",

        "adminId": "674048d2234ee6ea502ecd81",

        "\_\_v": 0

    },

    {

        "\_id": "6745a97afcd5ffa9e22c6063",

        "name": "Hotel Taj Coramandel",

        "location": "Chennai",

        "rating": 4.9,

        "description": "Best Hotel in Tamil Nadu",

        "facilities": [

            "Free WiFi",

            "Swimming Pool",

            "Spa",

            "Restaurant",

            "Pub"

        ],

        "created\_at": "2024-11-26T10:56:58.466Z",

        "adminId": "6744a87400d0a5a56956f0f6",

        "\_\_v": 0

    }

]

1. **List the Hotel’s Room Details:**

* **Endpoint:** /room/view
* **Method:** POST
* **Description:** Fetches the Specific Hotel’s Room Details based on the Given Hotel ID
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

"hotelid": "6745a97afcd5ffa9e22c6063"

}

* **Response:**

    {

    "message": "Rooms successfully fetched!",

    "room": [

        {

            "\_id": "67487091cb5487a02cbf9349",

            "hotelid": "6745a97afcd5ffa9e22c6063",

            "rooms": [

                {

                    "roomNo": 101,

                    "roomType": "regular",

                    "price": 120,

                    "maxPeople": 2,

                    "availability": **false**,

                    "amenities": [

                        "WiFi",

                        "Air Conditioning",

                        "TV"

                    ],

                    "\_id": "67487091cb5487a02cbf934a"

                },

                {

                    "roomNo": 102,

                    "roomType": "standard",

                    "price": 80,

                    "maxPeople": 2,

                    "availability": **true**,

                    "amenities": [

                        "WiFi",

                        "TV"

                    ],

                    "\_id": "67487091cb5487a02cbf934b"

                },

                {

                    "roomNo": 103,

                    "roomType": "Large",

                    "price": 140,

                    "maxPeople": 4,

                    "availability": **true**,

                    "amenities": [

                        "WiFi",

                        "TV"

                    ],

                    "\_id": "67487091cb5487a02cbf934c"

                },

                {

                    "roomNo": 104,

                    "roomType": "Trio",

                    "price": 110,

                    "maxPeople": 3,

                    "availability": **true**,

                    "amenities": [

                        "WiFi",

                        "TV"

                    ],

                    "\_id": "67487091cb5487a02cbf934d"

                }

            ],

            "adminId": "6744a87400d0a5a56956f0f6",

            "\_\_v": 0

        }

    ]

}

1. **Book the Available Room:**

* **Endpoint:** /reservation/book
* **Method:** POST
* **Description:** Based on the Input Room Number , The API checks if the Room is Available , if available , The Booking is made on the User Id that Fetched From Token verifying Middleware , if not available , Sends response as Room is not Available
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

"hotel\_id": "6745a97afcd5ffa9e22c6063",

"room\_no":102,

"people\_count": 2,

"vacate\_date": "2024-11-28T11:00:00.000Z"

}

* **Response:**

{

    "message": "Room booked successfully",

    "data": {

        "user\_id": "67488793cb5487a02cbf935f",

        "username": "karna",

        "hotel\_id": "6745a97afcd5ffa9e22c6063",

        "hotel\_name": "Hotel Taj Coramandel",

        "room\_no": 102,

        "booked\_date": "2024-11-28T15:11:02.508Z",

        "vacate\_date": "2024-11-28T11:00:00.000Z",

        "\_id": "67489021d36c91995fec81b6",

        "\_\_v": 0

    }

}

**For Admins:**

**1 ) Register:**

**Endpoint:** /admin/register

**Method:** POST

**Description:** Registers the Admin’s Data in The Database

**Body:** json

{

"name":"arjun",

"email":"arjun@gmail.com",

"password":"password"

}

**Response:**

{

"message": "Admin has been created."

}

**2) Login :**

**Endpoint:** /admin/login

**Method:** POST

**Description:** Check The Admin Details in the Database and generates a JWT Token

**Body:** json

{

"email":"arjun@gmail.com",

"password":"password"

}

**Response:**

{

"message": "Login Successful",

"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM"

}

**3) View All Users:**

* **Endpoint:** /admin/view/users
* **Method:** GET
* **Description:** Fetches All the User and lists
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** none
* **Response:**

{

    "message": "All Users Fetched",

    "users": [

        {

            "\_id": "673f0c5d08f06962c8250a83",

            "username": "janani"

        },

        {

            "\_id": "6740411922c5da65bd923250",

            "username": "dinesh"

        },

        {

            "\_id": "67446f2e10a8747b8608a700",

            "username": "ramesh"

        },

        {

            "\_id": "6748128ac75ee2217296dbb2",

            "username": "harish"

        },

        {

            "\_id": "67488793cb5487a02cbf935f",

            "username": "karna"

        }

    ]

}

**4) Delete A Specific User:**

* **Endpoint:** /admin/delete/user
* **Method:** POST
* **Description:** Delete a Specific User Based on the Input user id
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

    "\_id":"6740412b22c5da65bd923252"

}

* **Response**

{

    "message": "The user is deleted"

}

**5) Add the Hotel Details:**

* **Endpoint:** /hotel/add
* **Method:** POST
* **Description:** Add the New Hotel Details to the Database
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

  "name": "Hotel Lakeview",

  "location": "Pondycherry",

  "rating": 4.1,

  "description": "Best Hotel in Pondycherry",

  "facilities": ["Free WiFi", "Swimming Pool", "Spa", "Restaurant", "Pub"]

}

* **Response**

{

    "message": "Hotel added successfully",

    "data": {

        "name": "Hotel Lakeview",

        "location": "Pondycherry",

        "rating": 4.1,

        "description": "Best Hotel in Pondycherry",

        "facilities": [

            "Free WiFi",

            "Swimming Pool",

            "Spa",

            "Restaurant",

            "Pub"

        ],

        "created\_at": "2024-11-28T16:12:13.833Z",

        "adminId": "6744a87400d0a5a56956f0f6",

        "\_id": "6748965dd36c91995fec81bc",

        "\_\_v": 0

    }

}

**6) View the Hotel Details:**

* **Endpoint:** /hotel/view-hoteldetails
* **Method:** GET
* **Description:** View the Existing Hotel Details
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** none
* **Response**

{

    "message": "Hotel updated successfully",

    "data": {

        "\_id": "6748965dd36c91995fec81bc",

        "name": "Hotel Lakeview",

        "location": "Mumbai",

        "rating": 4.1,

        "description": "Best Hotel in Mumbai",

        "facilities": [

            "Free WiFi",

            "Swimming Pool",

            "Spa",

            "Restaurant",

            "Pub"

        ],

        "created\_at": "2024-11-28T16:12:13.833Z",

        "adminId": "6744a87400d0a5a56956f0f6",

        "\_\_v": 0,

        "altered\_at": "2024-11-28T16:45:40.662Z"

    }

}

    "message": "All Hotels Fetched",

    "hotels": [

        {

            "\_id": "67444372370baf7f50f42425",

            "name": "Royal Associations Hotel",

            "adminId": {

                "\_id": "674048d2234ee6ea502ecd81",

                "name": "selva"

            }

        },

        {

            "\_id": "67444eccf244f41694dae4cf",

            "name": "Hotel Annamalai",

            "adminId": {

                "\_id": "674048d2234ee6ea502ecd81",

                "name": "selva"

            }

        },

        {

            "\_id": "6745a97afcd5ffa9e22c6063",

            "name": "Hotel Taj Coramandel",

            "adminId": {

                "\_id": "6744a87400d0a5a56956f0f6",

                "name": "Guna"

            }

        },

        {

            "\_id": "6748965dd36c91995fec81bc",

            "name": "Hotel Lakeview",

            "adminId": {

                "\_id": "6744a87400d0a5a56956f0f6",

                "name": "Guna"

            }

        },

        {

            "\_id": "67489c5e9a6f286b8949f60e",

            "name": "Hotel Aditya",

            "adminId": {

                "\_id": "6744a87400d0a5a56956f0f6",

                "name": "Guna"

            }

        }

    ]

}

**7) Edit the Hotel Details:**

* **Endpoint:** /hotel/edit-hoteldetails
* **Method:** POST
* **Description:** Edit the Existing Hotel Details to the Database
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

    "\_id": "6748965dd36c91995fec81bc",

    "name": "Hotel Lakeview",

    "location": "Mumbai",

    "rating": 4.1,

    "description": "Best Hotel in Mumbai",

    "facilities": [

        "Free WiFi",

        "Swimming Pool",

        "Spa",

        "Restaurant",

        "Pub"

        ]

}

* **Response**

{

    "message": "Hotel updated successfully",

    "data": {

        "\_id": "6748965dd36c91995fec81bc",

        "name": "Hotel Lakeview",

        "location": "Mumbai",

        "rating": 4.1,

        "description": "Best Hotel in Mumbai",

        "facilities": [

            "Free WiFi",

            "Swimming Pool",

            "Spa",

            "Restaurant",

            "Pub"

        ],

        "created\_at": "2024-11-28T16:12:13.833Z",

        "adminId": "6744a87400d0a5a56956f0f6",

        "\_\_v": 0,

        "altered\_at": "2024-11-28T16:45:40.662Z"

    }

}

**8) Add the Rooms to the Hotels:**

* **Endpoint:** /room/add
* **Method:** POST
* **Description:** Add the Room Details to the Database using the hotel id
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

  "hotelid": "6748965dd36c91995fec81bc",

  "rooms": [

    {

      "roomNo": 101,

      "roomType": "regular",

      "price": 120,

      "maxPeople": 2,

      "availability": **true**,

      "amenities": ["WiFi", "Air Conditioning", "TV"]

    },

    {

      "roomNo": 102,

      "roomType": "standard",

      "price": 80,

      "maxPeople": 2,

      "availability": **true**,

      "amenities": ["WiFi", "TV"]

    },

    {

      "roomNo": 103,

      "roomType": "Large",

      "price": 140,

      "maxPeople": 4,

      "availability": **true**,

      "amenities": ["WiFi", "TV"]

    },

    {

      "roomNo": 104,

      "roomType": "Trio",

      "price": 110,

      "maxPeople": 3,

      "availability": **true**,

      "amenities": ["WiFi", "TV"]

    }

  ]

}

* **Response**

{

    "message": "Room data added successfully",

    "data": {

        "hotelid": "6748965dd36c91995fec81bc",

        "rooms": [

            {

                "roomNo": 101,

                "roomType": "regular",

                "price": 120,

                "maxPeople": 2,

                "availability": **true**,

                "amenities": [

                    "WiFi",

                    "Air Conditioning",

                    "TV"

                ],

                "\_id": "6748a0bf7f3a53dff0008bcf"

            },

            {

                "roomNo": 102,

                "roomType": "standard",

                "price": 80,

                "maxPeople": 2,

                "availability": **true**,

                "amenities": [

                    "WiFi",

                    "TV"

                ],

                "\_id": "6748a0bf7f3a53dff0008bd0"

            },

            {

                "roomNo": 103,

                "roomType": "Large",

                "price": 140,

                "maxPeople": 4,

                "availability": **true**,

                "amenities": [

                    "WiFi",

                    "TV"

                ],

                "\_id": "6748a0bf7f3a53dff0008bd1"

            },

            {

                "roomNo": 104,

                "roomType": "Trio",

                "price": 110,

                "maxPeople": 3,

                "availability": **true**,

                "amenities": [

                    "WiFi",

                    "TV"

                ],

                "\_id": "6748a0bf7f3a53dff0008bd2"

            }

        ],

        "adminId": "6744a87400d0a5a56956f0f6",

        "\_id": "6748a0bf7f3a53dff0008bce",

        "\_\_v": 0

    }

}

**9) Delete the Room Details:**

* **Endpoint:** /room/delete
* **Method:** POST
* **Description:** Delete the Existing Room Details of a Hotel From the Database
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

"hotelid":"6748965dd36c91995fec81bc"

}

* **Response**

{

    "message": "room(s) deleted successfully",

}

**For Super-Admins:**

**Note:** Super-admins can access all API endpoints that the Admin can Access .Below Are Authenticated API that can only by SuperAdmins

**1) Register:**

**Endpoint:** /admin/super-register

**Method:** POST

**Description:** Registers the Super admin’s Data in The Database

**Body:** json

{

"name":"krishna",

"email":"krishna@gmail.com",

"password":"password"

}

**Response:**

{

"message": "SuperAdmin has been created."

}

**2)Login :**

**Endpoint:** /admin/super-login

**Method:** POST

**Description:** Check The Super admin Details in the Database and generates a JWT Token

**Body:** json

{

"email":"krishna@gmail.com",

"password":"password"

}

**Response:**

{

"message": "Login Successful",

"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM"

}

**3) View All Admins:**

* **Endpoint:** /admin/view-admin
* **Method:** GET
* **Description:** Fetches All the Admin and lists
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** none
* **Response:**

{

    "message": "All Admins Fetched",

    "admins": [

        {

            "\_id": "67402f447a30b88c9b5fe3ee",

            "name": "sabari"

        },

        {

            "\_id": "6744a87400d0a5a56956f0f6",

            "name": "Guna"

        },

        {

            "\_id": "674869bdb769d5f29e4bc69d",

            "name": "laxman"

        }

    ]

}

**4) Delete A Specific Admin:**

* **Endpoint:** /admin/delete-admin
* **Method:** POST
* **Description:** Delete a Specific Admin Based on the Input admin id
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

    "\_id":"6740412b22c5da65bd923252"

}

* **Response**

{ "message": "The admin is deleted" }

**5) View Reservation Details of the User:**

* **Endpoint:** /reservation/view-reservation
* **Method:** POST
* **Description:** Fetches the Reservation Details Using User id
* **Authorization:** Bearer Token <token>

**Eg.**

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjY3NDg4NzkzY2I1NDg3YTAyY2JmOTM1ZiIsInVzZXJuYW1lIjoia2FybmEiLCJyb2xlIjoidXNlciIsImlhdCI6MTczMjgwNzAxNH0.XwdHWZeRMUKyVBG6JGPfmZJWwwFtYav5bcW5k6oaNHM

* **Body:** json

{

    "userId":"6740412b22c5da65bd923252"

}

* **Response**

{

    "message": "The user's reservation details were fetched",

    "data": [

        {

            "\_id": "6745f994348d006dce7039c2",

            "user\_id": "673f0c5d08f06962c8250a83",

            "username": "janani",

            "hotel\_id": "6745a97afcd5ffa9e22c6063",

            "hotel\_name": "Hotel Taj Coramandel",

            "room\_no": 102,

            "booked\_date": "2024-11-26T16:37:24.666Z",

            "vacate\_date": "2024-11-28T11:00:00.000Z",

            "\_\_v": 0

        },

  ]

}