Booking App for Sports Technology Company

Consideration- Currently the backend is hosted on a free plan on Render, so it takes about 1-2 mins to boot up the machine after some rest period.

1. Introduction

This project is a web application built to manage booking operations for a sports technology company which is our assignment task, which manages multiple centres offering various sports. Each centre has different courts or resources for each sport, and the system allows customers and operations teams to create and manage 60-minute booking slots. The primary goal is to enable centre managers to view and manage bookings efficiently while providing an interface for customers to make bookings seamlessly.

2. Design Decisions

Backend Design:

The backend was developed using **Node.js** and **MongoDB** to handle the various operations involved in booking and managing centres. Express.js was used to structure the backend with multiple routes to manage centres, sports, courts, and bookings efficiently. Here is a breakdown of the core models and controllers:

Models:

- `center.model.js`: Manages details about each centre (e.g., Indiranagar, Koramangala).
- `sport.model.js`: Manages different sports available in each centre.
- `court.model.js`: Manages the courts available for each sport.
- `booking.model.js`: Handles customer bookings for specific courts.
- `user.model.js`: Manages user information, particularly centre managers.

Controllers and Routes: Each resource (centre, sport, court, booking, and user) has dedicated controllers and routes to handle CRUD operations, making the system modular and scalable.

Frontend Design:

- **Custom Booking Table**:

The booking table was entirely created from scratch without relying on external libraries. It consists of **24 slots** per day, each corresponding to an hour.

- **Booked slots** are marked in **red**, while **available slots** are marked in **green**.
- Clicking on an available slot opens a **card** where the user must enter their name to confirm the booking.

- **Navbar and Calendar**:

The interface includes a **navbar** with a **date picker**, **court selector**, and a **dropdown** for selecting games. Users can also use a **calendar** to navigate between dates for booking.

- If no courts or slots are available on a particular day or court, a message saying **"no courts available"** is displayed.

- **Booking Validation**:

The frontend includes validation checks to prevent overbooking of previously booked slots. The system verifies available slots before confirming bookings.

Integration of Frontend and Backend:

- The frontend communicates with the backend REST APIs to dynamically fetch slot availability, court/game selections, and submit bookings.
- Both the frontend and backend have synchronized validation mechanisms to ensure the booking system remains accurate and prevents issues like double booking.

3. Implementation Details

Technologies Used:

- **Backend**:
    - **Node.js** with **Express.js** for building REST APIs.
    - **MongoDB** as the database to store information about centres, sports, courts, and bookings.
    - **Cors** for managing cross-origin resource sharing.
    - **dotenv** for handling environment variables securely
    -
- **Frontend**:
    - **React.js** to build the user interface.

Key Files:

- `app.js`: The core file that connects different routes, middlewares, and database configurations.
- `booking.controller.js`, `center.controller.js`, etc.: Handle the business logic for booking, centres, courts, and sports.
- `auth.middleware.js`: Manages authentication and ensures only authorised users (e.g., centre managers) can access certain features.

4. Challenges and Solutions

Challenge 1: Real-Time Slot Management

Ensuring that booking slots don't overlap was crucial for the application. Implementing booking logic to avoid clashes was a priority.

- **Solution**: Applied server-side validation using the `booking.controller.js`, which checks the availability of a slot before creating a new booking, ensuring no conflicts.

Challenge 2: Integration between Backend and Frontend

Integrating a backend developed with Node.js with a frontend built using React.js can introduce communication issues, such as mismatched response formats or CORS errors.

- **Solution**: Consistent use of **JSON** format for responses across all APIs ensured that the data flow between the frontend and backend remained smooth. Additionally, **Cors** middleware resolved cross-origin issues.

Challenge 3: Meeting the Deadline

Completing the entire project, including both backend and frontend integration, while ensuring everything worked as intended, was a difficult task under the project's deadline. The sheer volume of work required made time management critical.

- **Solution**: Regular commits, clear prioritization of tasks, and focused sprints helped to meet the project's requirements and complete the integration on time.

.

5. Future Improvements

Given more time, the following features and enhancements could be implemented:

- **Scalability and High Load Management**: As the app grows and the number of bookings increases across multiple centers, ensuring the system can handle high traffic and scale efficiently will be essential. Introducing a **Content Delivery Network (CDN)** would help reduce load times and improve the performance of static resources. Additionally, planning for a **High-Level Design (HLD)** that includes **load balancers**, **distributed databases**, and **horizontal scaling** will ensure the system can manage a larger user base without bottlenecks.

- **Locking Mechanism**: To avoid race conditions when multiple users try to book the same slot at the same time, a **locking mechanism** should be introduced. This would ensure that once a booking is initiated, other users are temporarily blocked from booking that slot until the transaction is completed or cancelled. Implementing techniques like **optimistic locking** or **pessimistic locking** at the database level will help to maintain data consistency and prevent double bookings.

- **Real-Time Booking Updates**: Use WebSockets or a similar technology to provide real-time updates when slots are booked or become available, ensuring customers and operations teams receive instant notifications of any changes.

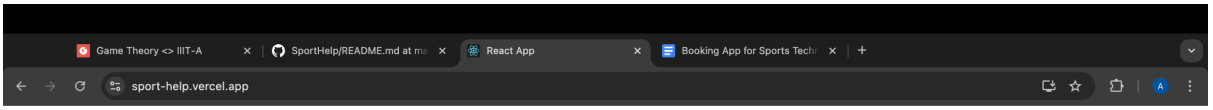
- **Admin Dashboard**: Create a more robust operations management dashboard with detailed reporting, analytics, and the ability to view activity logs for better oversight of booking patterns and resource utilisation.

- **Payment Integration**: Implement a secure payment gateway for customers to make payments directly through the application, streamlining the booking process and improving user convenience.

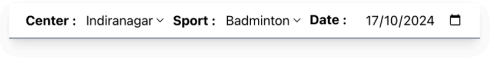
- **Authentication with Role-Based Access Control**: Enhance the authentication system by implementing role-based access control (RBAC) to ensure that different types of users (e.g., customers, centre managers, admins) have appropriate access to specific parts of the system.

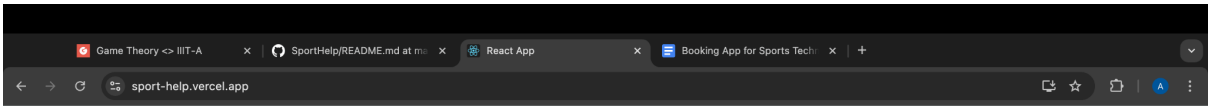
Conclusion

This project successfully delivered a booking system capable of handling complex booking needs for a sports technology company. By leveraging Node.js, MongoDB, and React.js, the system is scalable and ready to accommodate additional features in the future.



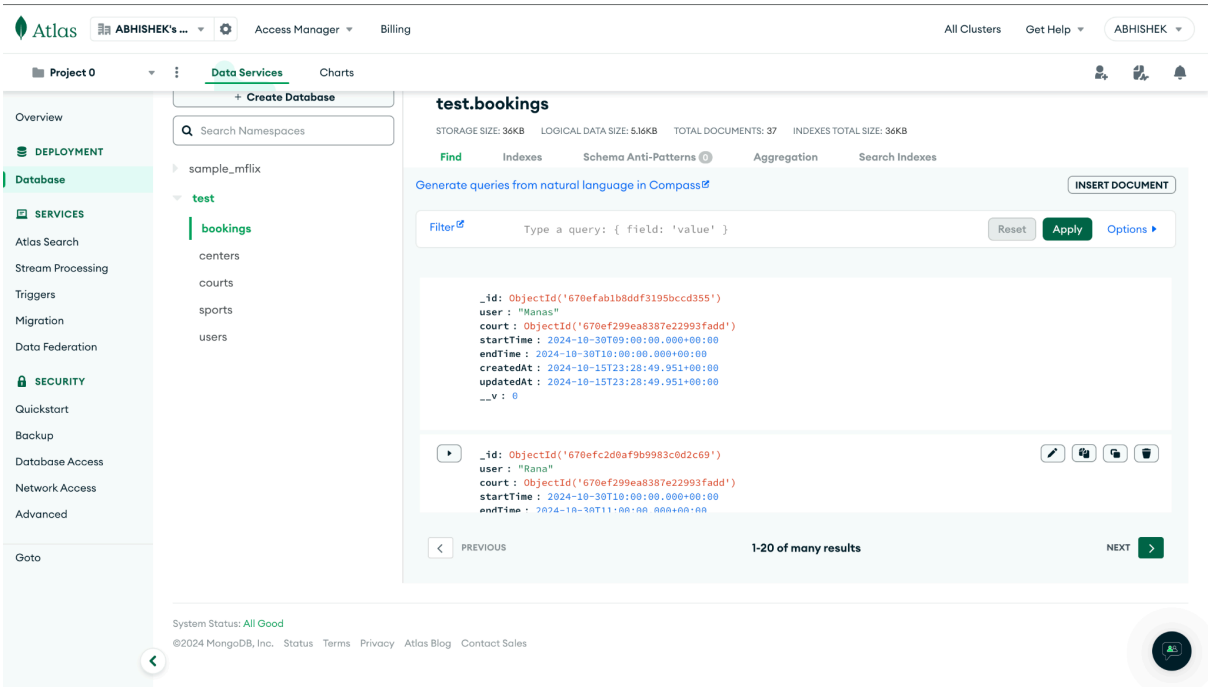
Slots	Court 1	Court 2	Court 3	Court 4	Court 5	Court 6	Court 7	Court 8
00:00 - 01:00	Available	Available	Available	Available	Available	Available	Available	Available
01:00 - 02:00	Available	Sahil Yadav	Available	Available	Available	Available	Bojack	Available
02:00 - 03:00	Available	Available	Available	Available	Available	Arsh Singh	Available	Available
03:00 - 04:00	Sahil Yadav	Available	Diane	Available	Available	Available	Available	Available
04:00 - 05:00	Available	Available	Available	Available	Available	Available	Available	Available
05:00 - 06:00	Available	Available	Available	Available	Sarah	Available	aryan anand	Available
06:00 - 07:00	Available	Available	Available	Available	Available	Available	Available	Available
07:00 - 08:00	Available	Available	Available	Available	Available	Available	Available	Available
08:00 - 09:00	Anurag	Available	Available	Peanutbutter	Available	Available	Available	Available

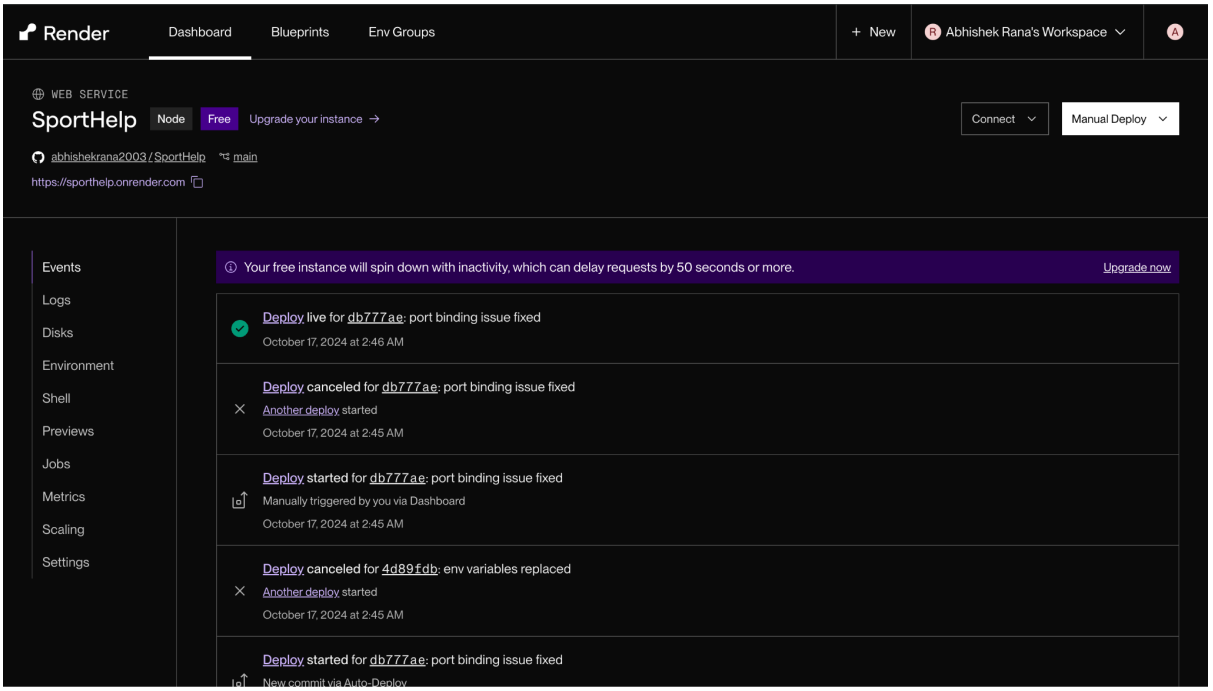




Slots	
00:00 - 01:00	No Courts Available
01:00 - 02:00	
02:00 - 03:00	
03:00 - 04:00	
04:00 - 05:00	
05:00 - 06:00	
06:00 - 07:00	
07:00 - 08:00	
08:00 - 09:00	

Center : HSR **Sport :** Cricket **Date :** 17/10/2024





Submitted By:- Abhishek Rana(IEC2021003)