INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

Game Theory IIIT-A
Sports Booking Application

Submitted By-
Gokul Uppal
IIT2021124

Introduction

This project focuses on building a comprehensive booking management system for a sports technology company's operations team. The main objective is to streamline the process of managing bookings across multiple centers, each offering a range of sports with several courts available for booking. The system ensures efficient handling of bookings, preventing double booking of courts and providing both users and administrators with a seamless experience for managing sports facilities.

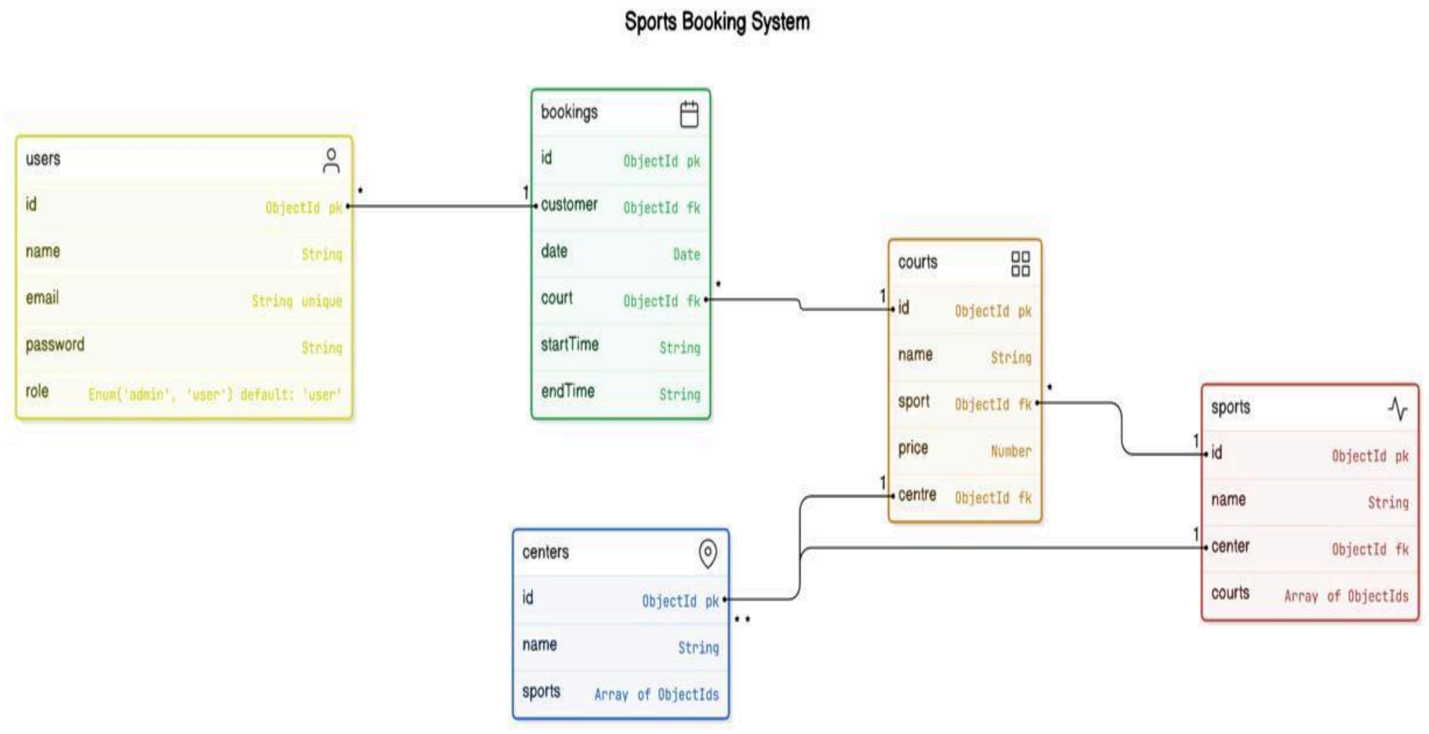
The company operates several centers, such as those in Indiranagar and Koramangala, with each center offering a variety of sports, including badminton and squash. Each sport in the center can have multiple courts or resources, and users can book these courts for 60-minute time slots. A core feature of the system is its ability to prevent booking conflicts, ensuring that courts remain available only when they are not already booked during the requested time period.

For operations management, the system offers a powerful interface for the center's administrative staff to manage bookings. Administrators have the ability to view all bookings for their respective centers and sports, create new bookings, and delete existing bookings if needed. This ensures smooth management of sports facilities, allowing center managers to control usage efficiently and resolve any scheduling conflicts.

Overall, the project aims to create a seamless user experience for both customers and administrators. With a robust backend, powered by well-structured APIs that handle authentication, viewing, and creating bookings, the system effectively meets the operational needs of the company. It empowers administrators with tools to oversee court usage while giving users a straightforward process for booking courts across multiple centers and sports.

Design Decisions

The design of this booking management system was driven by the need for a robust and scalable solution that can manage multiple centers, sports, courts, and bookings while ensuring user-friendliness and conflict-free booking experiences. Several key design decisions were made to achieve these goals, focusing on both the backend architecture and the user interface.



Sports Booking System

users	
id	ObjectId pk
name	String
email	String unique
password	String
role	Enum('admin', 'user') default: 'user'

bookings	
id	ObjectId pk
customer	ObjectId fk
date	Date
court	ObjectId fk
startTime	String
endTime	String

courts	
id	ObjectId pk
name	String
sport	ObjectId fk
price	Number
centre	ObjectId fk

sports	
id	ObjectId pk
name	String
center	ObjectId fk
courts	Array of ObjectIds

centers	
id	ObjectId pk
name	String
sports	Array of ObjectIds

1. Modular API Structure

One of the primary design decisions was to structure the backend using a modular approach with clear separation of concerns. Each functional aspect of the system, such as authentication, bookings, and court management, is handled by distinct APIs. This approach ensures scalability and maintainability, allowing the system to easily expand in the future with additional features or new requirements without disrupting existing functionality.

User Routes:

- **Signup (/signup):** Register a new user.
- **Login (/signin):** Authenticate user and generate JWT token, assigning user roles for navigation.
- **Create Booking (/user/userbookings):** Users can book courts based on availability at a specific center for a specific sport.
- **My Bookings (/user/mybookings):** Users can view all their bookings.

Admin Routes:

- **Login (/signin):** Authenticate admin and generate JWT token, assigning admin roles for navigation.
- **Show Bookings (/admin/showallbookings):** Admin can view and delete all user bookings.
- **Add Court (/admin/addNewCourt):** Admin can add courts to a center, assign sports, and set prices.

2. Handling Booking Conflicts

To ensure that the booking system operates smoothly without allowing double bookings, a key design choice was to implement conflict-checking logic during the booking process. The system checks for existing bookings that overlap with the requested date and time, ensuring that no two customers can book the same court at the same time. This logic is handled at the backend during the booking creation process, ensuring that the system remains robust even under heavy usage.

3. Scalability and Flexibility

The system was designed with scalability in mind. The database schema was structured to support the addition of multiple centers, sports, and courts without introducing complexity. Each center is associated with multiple sports, and each sport can have multiple courts, which are represented through relationships between models.

4. User and Admin Role Management

Role-based access control was a crucial design choice to separate the responsibilities and views of customers and administrators. By implementing role management in the authentication process, the system ensures that users can only view and book courts, while administrators have additional privileges, such as viewing all bookings, creating new courts, and managing bookings.

5. User-Friendly Interface

The front-end was designed to provide an intuitive experience for both customers and administrators. For users, the system allows easy navigation to view available courts, search for booking slots, and make bookings. For administrators, the interface provides a clear overview of all bookings and management tools to add courts or resolve conflicts. Simplicity and ease of use were prioritized to ensure that the system could be adopted with minimal training.

Implementation Details

Technologies Used:

Frontend: Reactjs,Tailwind CSS,HTML
Backend: Express.js,Node.js
Databases: MongoDB

The choice of technologies was driven by the project's requirements for managing multiple centers, sports, courts, and bookings, while providing a seamless user and admin experience. Below are the key technologies used and the rationale behind their selection.

1. **Backend (Node.js with Express.js):**
   The backend uses Node.js for its non-blocking, event-driven architecture, ideal for handling multiple requests during high traffic. Express.js was chosen for its simplicity in setting up RESTful APIs and its modular structure, which ensures scalability and maintainability by separating authentication, bookings, and court management into distinct routes and controllers.

2. **MongoDB (Database Management):**
   MongoDB's document-based model allows easy handling of complex data relationships, such as centers, sports, and courts. Its scalability makes it suitable for managing a large number of bookings, while Mongoose ensures data validation and integrity, especially for preventing booking conflicts.

3. **JWT Authentication:**
   JSON Web Tokens (JWT) are used for secure user authentication, providing a lightweight, stateless method for transmitting data between the client and server. JWT ensures that only authorized users can access key functionalities like booking courts or managing centers.

4. **Frontend (React.js):**
   React.js was selected for its component-based architecture, allowing for the creation of reusable UI elements, ensuring a dynamic and responsive interface. Its efficient DOM updating and state management are ideal for real-time interactions, like booking or managing courts, especially in admin dashboards.

5. **Middleware (Role-Based Access Control):**
   Middleware in Node.js enforces role-based access control, ensuring users are authenticated and authorized before accessing specific endpoints. Customers can view and book courts, while admins manage courts and bookings, ensuring security and appropriate access.

Together, these technologies (Node.js, MongoDB, and React.js) create a scalable, efficient, and user-friendly system for managing bookings and sports facilities.

Challenges and Solutions

During the development of the booking system, several challenges arose, each requiring targeted solutions:

Handling Booking Conflicts:
To prevent double bookings, a conflict-checking mechanism was implemented in the backend. The system validates bookings by checking for overlaps with existing reservations before confirming.

Managing Complex Data Relationships:
The complexity of managing multiple centers, sports, and courts was handled by using MongoDB's document model, which allowed for efficient nesting of these relationships, simplifying queries and updates.

Role-Based Access Control:

Middleware was used to enforce role-based access control, ensuring that only authorized users can access specific features. This secured critical actions like court management and bookings.

User-Friendly Interface:
React.js enabled the creation of reusable components, ensuring a consistent and easy-to-navigate interface. This streamlined complex tasks like booking searches and management.

Data Validation and Error Handling:
Thorough validation at both the frontend and backend ensured accurate data input, with clear error messages to guide users when conflicts or incorrect inputs occurred.

Future Improvements

Real-Time Updates: Use WebSockets or Socket.io for instant booking updates.
Advanced Analytics: Add dashboards for insights on court usage, booking patterns, and revenue.
Calendar Integration: Provide a calendar view for easier booking and scheduling.
Automated Notifications: Implement email/SMS notifications for confirmations and reminders.
Dynamic Pricing: Introduce pricing adjustments based on demand and promotions for off-peak hours.

These future improvements would significantly enhance the system's capabilities, making it more robust, user-friendly, and aligned with the growing needs of the sports technology company and its customers.