GameTheory Assignment Report

Name: Lokesh Yadav
Enr no: IIT2021065

Introduction

This project involves building a booking application for a sports technology company that manages multiple centers offering various sports. Each center, such as IIITA or MNNIT, provides multiple sports (e.g., badminton, tennis), with several courts or resources available for each sport. The app will allow customers to book 60-minute slots based on court availability, providing a seamless user experience for scheduling their preferred sport.

In addition to customer bookings, the app will enable center managers to efficiently manage all bookings. The operations team will have the ability to view and create reservations for any sport within their center, ensuring optimal use of resources and preventing scheduling conflicts. The app is designed to streamline the booking process and improve overall facility management.

Design Decisions

Backend

- **Project Structure**: The backend is designed using a modular and layered architecture. Key folders include routes (for handling API endpoints), models (to define data structures), and controllers (for handling business logic). This separation improves maintainability and scalability.
- **RESTful API**: The backend exposes a set of RESTful APIs that handle various operations like booking creation, viewing available slots, and center management. Each route corresponds to a resource (e.g., centers, sports, bookings) and follows standard REST conventions.

- **Database Design**: MongoDB is used as the database, with models representing core entities like Centers, Sports, Courts, and Bookings. This NoSQL approach allows flexibility in managing complex relationships between these entities.
- **Booking Logic**: Booking slots are set to 60-minute intervals, and the backend logic ensures that conflicts are avoided by checking availability before creating a booking. This prevents overbooking of resources.
- **Middleware and Validation**: Middleware functions, such as authentication and request validation, ensure that only authorized users can access certain routes and that data integrity is maintained

Frontend

- **Component-Based Architecture**: The app is divided into reusable components (e.g., booking forms, center lists, and sport selection), improving maintainability and scalability.
- **State Management with Hooks**: React hooks like useState and useEffect manage dynamic data and side effects, such as user inputs and data fetching from the backend.
- **Routing**: React Router handles navigation between views, enabling smooth transitions between different sections (e.g., booking screens and center details).
- **User Interface (UI)**: The frontend uses CSS for styling, ensuring a responsive layout that adapts to different screen sizes. The focus is on providing a clean and intuitive user experience for both customers and operations managers.
- **Integration with Backend**: The frontend interacts with the backend through API calls, allowing real-time booking and availability checks. Axios or the Fetch API is likely used to handle these asynchronous requests, ensuring smooth communication with the backend services

Technologies Used

- **ReactJS (Frontend)**: ReactJS is used to build the user interface with a component-based architecture. It provides a dynamic and responsive experience for users, enabling smooth interactions such as booking slots and viewing center details.
- **NodeJS (Backend)**: NodeJS powers the backend server, handling API requests and managing business logic. It's chosen for its ability to handle asynchronous, non-blocking operations efficiently, crucial for real-time bookings.
- **MongoDB (Database)**: MongoDB, a NoSQL database, is used to store complex, flexible data structures like centers, sports, courts, and bookings. It allows for scalability and flexibility in managing relationships between different entities.
- **JWT (Authentication)**: JSON Web Tokens (JWT) are used for secure user authentication and authorization. JWT ensures that user sessions are securely maintained without storing sensitive information on the server.
- **Bcrypt (Password Hashing)**: Bcrypt is used to hash passwords, adding an extra layer of security by ensuring that user passwords are securely stored and protected from attacks like brute force.

Challenges & Solutions

- **Time Constraint:**The time was very limited so to develop and test the application in limited time was a challenge. To overcome this challenge I prioritized the core functionalities first, mainly focusing on essential features.
- **Integration Issues:** In the project I had to make sure that there was smooth integration of the frontend and backend components. To achieve this I tried to define clear API specifications and used tools like Postman to test endpoints before integrating.
- **Data Management:** Building a robust database schema under such a time constraint was also a challenge. I tried to design the database

schema in a very simple and flexible way initially and then iterated on it as and when needed to accommodate evolving requirements.

Future Improvements

- **Admin Dashboard:** Create a comprehensive admin panel for better management of centers,sports and user activity analysis.
- **User Feedback System:** Implement a review and feedback system for users to rate their experience, helping improve service quality.
- **Payment Integration:** Add payment gateway options for seamless online payment during bookings.
- **Improving Scalability**: Implement microservices architecture to decouple different functionalities (e.g., user management, booking management, notifications), allowing for independent scaling and maintenance of each service based on demand. This will enhance performance and flexibility as the user base grows.