CourtEase - Report

Name- Shekhar Kumar

Roll No- IIT2021071

1. Introduction

The CourtEase project is a booking application designed for a sports technology company to manage bookings for various sports facilities across multiple centers. The application provides a user-friendly interface for the operations team to view and manage bookings for different courts, sports, and slots, ensuring efficient resource management. The primary objective was to build a MERN stack application that allows center managers to handle and organize bookings seamlessly.

2. Design Decisions

The design was structured to support the operations team's needs, ensuring clarity and ease of use. Key design decisions include:

- **Multiple Centers Support**: The application was designed to manage multiple centers, each with its own set of sports and courts, allowing flexibility and scalability.
- **Modular Architecture**: The backend was built with RESTful APIs using Express.js and MongoDB, allowing modular and maintainable code. The frontend uses React for a dynamic, interactive user experience.
- **Tailwind CSS**: Tailwind CSS was chosen for styling to create a modern and responsive user interface efficiently.
- **Dynamic Slot Management**: The booking slots were made dynamic, allowing easy updates and extensions if new sports or time slots were introduced.

3. Implementation Details

Technologies Used

- **Frontend**: React, Tailwind CSS, React Router, Axios for API communication.
- **Backend**: Node.js, Express.js for routing, and MongoDB for the database.
- **Hosting**: The frontend was deployed using Vercel, and the backend using Render.
- **Tools**: Visual Studio Code for development, Git for version control, and Postman for API testing.

Rationale for Technology Choices

- **MERN Stack**: Chosen for its flexibility and familiarity, allowing full-stack development with JavaScript/Node.js.
- **React**: Ideal for creating a dynamic user interface that updates based on user actions and backend data.

- **MongoDB**: A NoSQL database was chosen for its ability to handle unstructured data and flexibility with schema evolution.

4. Challenges and Solutions

Challenges

- **Slot Booking Conflicts**: Ensuring that users couldn't double-book the same slot for a court required careful validation and checks.
- **Dynamic Data Handling**: Handling multiple centers, sports, and courts with dynamic data presented complexity in state management and rendering.
- **Deployment Issues**: Encountered deployment issues with Vercel and Render, primarily due to environment setup and version mismatches.

Solutions

- **Booking Validation**: Implemented validation logic in the backend API to check for conflicts before confirming a booking.
- **State Management**: Utilized React's useState and useEffect hooks efficiently to manage dynamic data fetching and rendering.
- **Deployment Fixes**: Adjusted configuration files and dependency versions to match hosting platform requirements and resolved deployment issues.

5. Future Improvements

- **Authentication and Authorization**: Implement user authentication to ensure that only authorized personnel can manage bookings and access data.
- **Enhanced User Interface**: Add features like filtering slots by availability, visual indicators for booked slots, and a calendar view for easy navigation.
- **Notifications**: Implement email or SMS notifications for bookings, cancellations, and other important updates.
- **Data Analytics**: Integrate analytics to track usage patterns, occupancy rates, and other KPIs to optimize resource management.