Smart Cricket Live System Using Real-Time Technologies

Ms. S Suganya[1], Maathesh D[2], Nareshsasi S[3], Sandosh L[4]
[1]*Asst. Prof, Department of Computer Science and Engineering, Dhanalakshmi college of Engineering Chennai, India*
[2,3,4] *Student, Department of Computer Science and Engineering, Dhanalakshmi college of Engineering Chennai, India*

Abstract- Cricket Application is a mobile-based app intended to make it easier and more efficient to manage all cricket-related events through the use of a centralized digital app. The conventional process of turf booking and organization of games is usually manual, involving a lot of unnecessary time and efforts. The proposed application solves all the problems by creating one platform where people can book turfs, organize matches, and interact with other players instantly. The app uses React Native technology for the frontend and Node.js combined with Express for the backend. Data is stored with the help of MongoDB, whereas Firebase is used to authenticate users and deliver push notifications. Real-time chat is provided through Socket.IO allowing users to instantly get information on their matches and chat with others. Another feature included in the platform is secure online payments realized with the help of Razorpay. Besides, the app will be capable of delivering location-based services helping users easily find all nearby grounds for cricket.

I.INTRODUCTION

Introduction of cricket is among those sports that has made great fame for itself and the growing popularity has increased the importance of efficient management of activities like ground reservation, arranging matches and communicating with players regarding different matches. In many places, the process is manually carried out, which often results in problems like overlapping arrangements, lack of coordination and efficiency. There has been a lot of development in the area of mobile technology and it can be used effectively for the task.

The suggested Cricket Application will serve as a one-stop platform for all cricket fans to control their activities efficiently. The application allows users to book turf facilities, engage in matches, and communicate with fellow players through a live chat option. The application will update users on the schedule of the match and events through live notifications. With the use of latest technologies like React Native, Node.js, MongoDB, Firebase, and Socket.IO, the application guarantees high performance and scalability. Moreover, the system provides an option for secure payment through the internet, and this can be done using Razorpay. Location-based services are also provided in order to help the users locate any nearby cricket grounds. This application is basically aimed at reducing human efforts, coordinating between different users, and making their lives easier.

The fast pace development of mobile and internet technologies has brought drastic changes to the field of managing various sports activities. Being one of the most popular games worldwide, cricket needs a certain organization that includes such operations as reserving of the ground, scheduling matches, and organizing communications between players. Unfortunately, in practice, these processes are implemented manually; therefore, they are associated with many troubles related to the inefficiency and possible confusion that may occur. The lack of the central database complicates the process of receiving current information and making decisions in terms of resource management. The necessity to introduce some sort of digital platform for improving these processes seems obvious, which gives rise to a number of ideas about how an app should look like.

The suggested Cricket App is a one-stop solution that will help resolve all these issues. It allows users to make bookings of cricket turfs, schedule matches, and

even communicate with other users in real time. The application offers live notifications about matches, match details, and other necessary information about users. The front end of the application is coded using React Native, whereas the back end is programmed using Node.js and Express.js technologies. MongoDB database management system will be used to store user data. The app integrates Firebase technology for authenticating users and sending push notifications. The real-time feature is implemented through Socket.IO.

Apart from basic features, the system also has features that facilitate the process of payments. Payments are facilitated in a secure and effective way through Razorpay. Another aspect that makes the system useful includes the provision of location-based services where users will use such information to access nearby cricket grounds. The system is scalable and flexible and is capable of supporting an increasing number of users and data. In addition, the system is also helpful for managers in keeping track of bookings and activities. Through automation, the system ensures improved efficiency and productivity. Additionally, the application can be used for efficient resource management through optimization of ground usage and scheduling. The software enables easy management of bookings, tracking of activities, and record keeping. Through digitalization, the system makes the whole process more error-free, time-saving, and systematic.

II.RELATED WORK

There is a notable increase in the interest of developing digital technologies for managing various aspects related to sport, mainly because of the need for an efficient and automated way of doing things. Researchers have shown interest in developing mobile and web application technologies that make tasks such as booking grounds and planning matches easier. Existing systems are based on conventional approaches that include a lot of manual operations, which are not only cumbersome but also subject to mistakes made by humans. Smart booking systems that facilitate live viewing of availability and online booking of sports facilities have been proposed to address these problems. Nevertheless, these systems lack important components like real-time communication and user interactions among others.

Another crucial domain in which research is being carried out is the development of live sports applications using cutting-edge communication technologies like WebSockets. Such apps are developed with the aim of providing live information about the game, including scores, individual player data, and event information. As a result of the continuous connection established by WebSocket-based systems between the client and the server, there is no delay in data transfer from the server to the client, resulting in faster delivery of information as opposed to conventional request/response-based communication. Some live games are designed with chat functionality, giving users the ability to communicate with one another while playing. However, such applications lack compatibility with other modules, such as ticketing, payment gateways, and user profiles.

Moreover, there is literature available on the development of mobile applications that are intended for the complete management of sport activities. These applications intend to incorporate many features together including user registration, scheduling, performance monitoring, and event planning. The integration of mobile technology ensures user convenience and better usability. Scientists have focused on designing efficient user interfaces and using efficient ways to manage data. Although many improvements have been made in the form of incorporating various functionalities and designing the interface, many current systems do not make use of new technological advances like cloud computing and real-time databases.

Moreover, research has also been conducted on the integration of secure payment gateways and location-based services in sports-related applications to improve their functionalities. The use of payment integration will enable users to conduct transactions directly from the application, without relying on external processes, thus saving time. Location-based services allow users to discover nearby sports facilities, which makes the application more practical and user-focused. However, most of the current applications integrate these features separately, rather than incorporating them together in one single application. The proposed Cricket Application will overcome these shortcomings by integrating booking, real-time updates, communication, payment gateway, and location services in a single application.

III.PROPOSED METHODOLOGY

The suggested system is called "Cricket Application" and acts as a centralized system for mobile devices that brings together a range of services for cricketers in one application. The proposed methodology centers on reducing the complications involved in booking a turf, organizing matches, communicating in real-time, and conducting safe payments through a modular and flexible architecture.

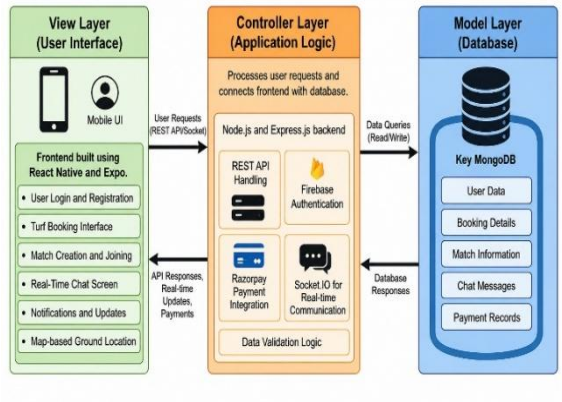
The system uses the client-server model to guarantee efficient management of data and interaction between the different modules. The user interface is created using React Native and Expo, offering a friendly and responsive user experience on both Android and iOS devices. The back-end code is written using Node.js and Express.js, and they manage the business logic, API requests, and server operations. The database chosen for this application is MongoDB, which manages the data of users, bookings, and matches.

The proposed methodology is structured in a number of functional modules, each of which meets one of the needs of cricket management. The module for user authentication guarantees safe user authentication and registration through Firebase-based services. The turf booking module helps users determine whether a certain cricket ground is available for booking and allows them to reserve cricket grounds online. The match management module assists users in organizing and managing cricket matches in a convenient way. The real-time chat module based on Socket.IO provides instant messaging facilities for users.

Moreover, there is a secured Payment Module which uses Razorpay technology in order to allow the users to make transactions from the app itself. Location Module works on maps and GPS services in order to facilitate the users in locating nearby cricket grounds. Admin Module handles all operations related to user management and booking management.

It has also been made even more powerful by integrating effective data management and security methods for increased reliability. Every bit of information entered by users, bookings made, and transactions carried out are securely maintained in the database along with appropriate authentication and validation checks. The use of RESTful API technology has made seamless interaction between the frontend and backend possible, which will enable easy and fast data transfer. The system will have an efficient error handling and data validation process to avoid any sort of failure and to ensure integrity of data. In addition, the system can easily incorporate new functionality like analytics, match history logging, etc.

In general, this suggested methodology stresses the importance of modularity, real-time interaction, and centralized database management. Due to the integration of several services in one system, it becomes less labor-intensive and more efficient and convenient to operate. The application is scalable and efficient enough to support a large number of users, which makes it a viable option for organizing cricket events.



IV.SYSTEM ARCHITECTURE

Presentation Layer (Frontend / View)

Presentation Layer refers to the front end interface for the Cricket application which allows for smooth interaction between the user and the software. The layer has been developed through the use of contemporary mobile application development frameworks such as React Native and Expo. Presentation Layer entails providing the application with an interface which will make it easy for the user to perform numerous operations on the platform. Some of the essential functionalities in the layer include registration and log in, turf booking, creation and joining of matches, messaging, and ground finding.

The presentation layer serves as the first point of contact, receiving inputs from users and formatting them into structured queries that get delivered to the application layer through REST APIs or real-time messaging mechanisms like Socket.IO. In addition, the presentation layer listens for feedback from the server and updates the interface with real-time information like booking confirmation, chat messages, and match

results. User-friendly experience is a critical consideration, guaranteeing that even novice users find it easy to operate the application.

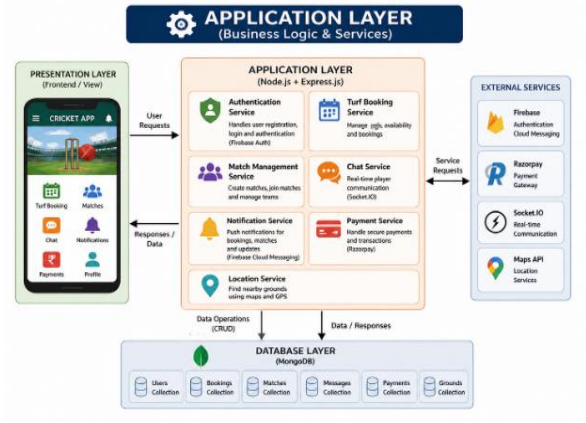
Furthermore, this level uses UI elements including maps for land positions, clickable buttons, and notifications to increase user involvement. The presentation level remains separate from the business logic to allow scalability, flexibility, and maintainability, which makes it an important part of the system design.

2. Application Layer (Backend / Controller):

The Application Layer constitutes the main part of the cricket application where the business logic, user request handling, and communication between the presentation layer and database layer take place. The Application Layer is usually implemented using backend programming languages like Node.js and Express.js, making it highly scalable and performant. This layer takes the requests made by the frontend, which can be RESTful APIs or real-time protocols, and processes them according to logic.

Authentication, turf booking, match management, chat services, notification services, payment services, and geolocation are some important services in the application layer. Authentication service deals with user signup, login, and security using Firebase authentication. Turf booking and match management services deal with scheduling and availability checking. Chatting and instant notification services will be provided using Socket.IO. Notification services will alert users regarding bookings and match information.

Additionally, the application layer integrates external services such as Razorpay for secure payment processing and map APIs for location-based features. Data validation and error handling are also performed here to ensure data integrity and system reliability. After processing, the application layer communicates with the database layer for data storage and retrieval, and sends appropriate responses back to the presentation layer. This structured approach ensures efficient system operation, scalability, and a smooth user experience.



3. Data Layer (Model / Database):

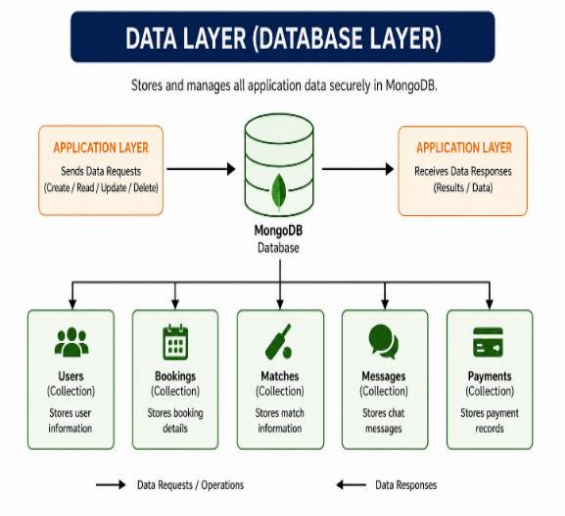
The data layer stores, manages, and retrieves all the application data needed by the cricket booking app. The data layer is made up of MongoDB, a non-relational database that holds all information in flexible documents. Using a non-relational database ensures that dynamic data like user behavior, matches, and booking details can be stored easily and efficiently.

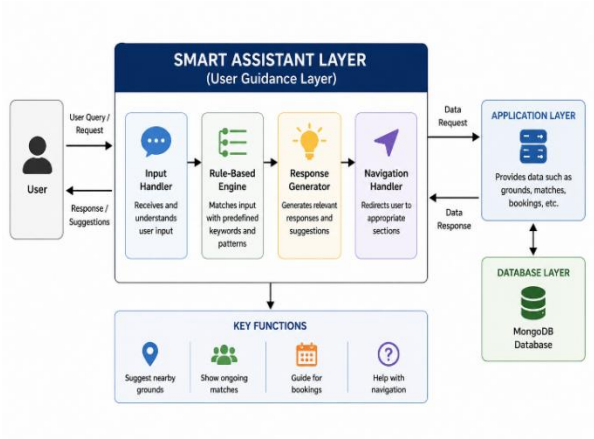
Mongoose has been used in order to create schemas that interact with the database. The application has several features, and each of them is a collection on its own, making it easier for maintenance and accessibility purposes.

The main collections include:

- User: Holds user information (name, email address, contact number)
- Booking: Holds turf booking information (date, time, ground)
- Match: Holds match information (teams, scheduling)
- Message: Holds message information
- Payment: Holds payment information

The Data Layer supports CRUD operations, enabling efficient data insertion, retrieval, updating, and deletion. It ensures data integrity and consistency through schema validation and structured queries





4. Smart Assistant Layer (User Guidance Layer)

The Smart Assistant Layer serves as a further addition to provide a better user interface in the cricket app. Unlike the use of a sophisticated artificial intelligence model, this layer relies on a basic rule-based system, whereby user behavior and minimal inputs are translated into meaningful recommendations and navigational shortcuts.

In cases where a user carries out activities including searching for grounds, making matches, and booking of turfs, the assistant offers assistance through prompts such as "Available grounds near you," "Join ongoing matches," and "Complete your booking." The assistant assists in avoiding confusion on the part of the user while accessing key features.

The assistant is developed using JavaScript technology and can work with API backends to retrieve data related to nearby grounds, availability of matches, and their booking status. The software is built light weight and will respond fast with no need for increased system complexity.

Key functionalities of the chatbot include:

- Guiding users to book turfs and join matches
- Suggesting nearby grounds based on location
- Providing quick access to ongoing matches
- Assisting users with booking and payment steps

This layer plays a crucial role in making the system more interactive and user-friendly, bridging the gap between users and system functionalities.

REFERENCES

[1] A. Kumar and R. Sharma, “Design and development of web-based sports facility booking system,” International Journal of Computer Applications, vol. 183, no. 10, pp. 15–20, 2020.

[2] S. Patel, M. Shah, and R. Mehta, “Online turf booking system using web technologies,” International Journal of Engineering Research & Technology, vol. 10, no. 6, pp. 45–50, 2021.

[3] V. Reddy and P. Kumar, “Sports event management system using web application,” International Journal of Advanced Research in Computer Science, vol. 12, no. 4, pp. 78–84, 2021.

[4] N. Singh and A. Kaur, “Web-based platform for match scheduling and team management,” International Journal of Computer Science Trends, vol. 9, no. 3, pp. 22–28, 2022.

[5] M. Gupta and S. Jain, “Real-time chat application using Socket.IO and Node.js,” International Journal of Software Engineering, vol. 11, no. 2, pp. 60–66, 2022.

[6] R. Verma and K. Singh, “Online booking system with payment integration using Razorpay,” International Journal of Web Engineering, vol. 14, no. 1, pp. 33–39, 2023.

[7] P. Sharma and D. Gupta, “Location-based services in web applications using Google Maps API,” International Journal of Computer Applications, vol. 184, no. 7, pp. 12–18, 2023.

[8] S. Bose and A. Roy, “Full-stack web development using MERN stack technologies,” International Journal of Software and Web Development, vol. 11, no. 4, 2023.

[9] D. Patel and M. Shah, “REST API development

using Node.js and Express framework," International Journal of Software Engineering, vol. 9, no. 1, pp. 50–56, 2024.
[10] K. Lee and H. Kim, "Scalable web architecture using MVC pattern," IEEE Access, vol. 12, pp. 33421–33430, 2024.
[11] R. Brown and T. Wilson, "Modern NoSQL databases for scalable web applications," ACM Computing Surveys, vol. 56, no. 2, 2024.
[12] N. Mehta and R. Shah, "Integration of frontend and backend systems in web applications," Journal of Web Technologies, vol. 9, no. 3, 2024.
[13] H. Chen and Y. Zhao, "User-centered design in modern web applications," IEEE Software, vol. 41, no. 2, pp. 90–97, 2024.
[14] S. Verma and P. Jain, "Cloud-based web application development using Node.js," Journal of Cloud Computing, vol. 13, no. 1, 2024.
[15] R. Das and K. Roy, "Design of scalable backend systems using Express.js," International Journal of Computer Applications, vol. 185, no. 6, 2023.
[16] P. Reddy and S. Kumar, "Database optimization techniques for NoSQL systems," International Journal of Data Science, vol. 7, no. 1, 2023.
[17] J. Li and W. Chen, "Improving accessibility using conversational interfaces," IEEE Access, vol. 12, pp. 55678–55685, 2024.
[18] K. Zhang and L. Liu, "Real-time data synchronization in web applications," Journal of Web Engineering, vol. 18, no. 2, pp. 101–110, 2023.
[19] S. Kumar and A. Singh, "Web-based service integration systems for smart applications," International Journal of Computer Science Trends, vol. 11, no. 2, pp. 34–40, 2023.
[20] D. Wilson and P. Clarke, "Design patterns for scalable web systems," IEEE Software Engineering Letters, vol. 6, no. 1, 2024.