https://doi.org/10.46335/IJIES.2025.10.7.20 **e-ISSN: 2456-3463**
Vol. 10, No. 7, 2025, PP. 96-100

International Journal of Innovations in Engineering and Science, www.ijies.net

Club Management System: A MERN Stack - Based Web Application for Event Management in Colleges

Sumit Kumar [1], Sujit Kumar [2], S. Rohith Chowdary[3], Mr. E. Murali [4]

[1,2,3,4], Computer Science and Engineering, DR. M.G.R. Educational and Research Institute, Maduravoyal, Chennai, Tamil Nadu, 600095, 0009-0006-3137-0380

Corresponding author: ***sumitkrsahu7@gmail.com***

Received on*: 5 May,2025* ***Revised on****: 09 June,2025* ***Published on****: 10 June,2025*

Abstract *– Student clubs play a vital role in skill development and engagement, but manual event management leads to inefficiencies. This paper presents the Club Management System (CMS), a MERN stack-based web application that streamlines event scheduling, user authentication, participation tracking, and real-time notifications. By automating these processes, CMS enhances coordination, reduces administrative workload, and improves student engagement. The paper discusses the system's design, implementation, and security aspects while addressing common challenges. Future enhancements include AI-driven analytics and blockchain-based verification for improved functionality.*

Keywords- *MERN Stack, Club Management System, Membership Management, Event Scheduling, MongoDB, React, Node.js*

I. INTRODUCTION

To promote networking opportunities, skill development, and extracurricular involvement, college clubs and student organizations are essential. However, manual procedures, a lack of coordination, and ineffective communication frequently make running these clubs, planning events, and managing participant registrations difficult and time-consuming. We offer the Club Management System (CMS), a web-based application developed with the MERN stack that aims to simplify the event management procedure for colleges and universities to address these problems.

The CMS provides a centralized platform where students, club organizers, and administrators can efficiently manage events, track participation, and engage with their communities. Unlike traditional event coordination methods, which rely heavily on spreadsheets, emails, and paper-based registration, our system automates and digitalizes the entire workflow, from event creation to post-event analysis.

1.1 Motivation

The primary motivation behind developing this system is to eliminate the common inefficiencies associated with club and event management. Key challenges faced by institutions include:

- Manual data handling, leading to errors and inconsistencies.
- Poor communication, resulting in low event turnout and mismanagement.
- Lack of engagement and tracking, making it difficult to measure event success.

By leveraging modern web technologies and incorporating features like role-based access control, event analytics, and social media integration, the Club Management System aims to offer a scalable and customizable solution adaptable to multiple institutions.

1.2 Objectives

The main objectives of the proposed system are:

- To develop a user-friendly platform that simplifies event organization and participation.
- To provide real-time updates and notifications for event schedules and changes.
- To integrate data analytics for tracking participation and event performance.

https://doi.org/10.46335/IJIES.2025.10.7.20 e-ISSN: 2456-3463
Vol. 10, No. 7, 2025, PP. 96-100

- To offer social media connectivity for better outreach and engagement.
- To enhance security and accessibility through authentication and cloud-based storage solutions.
- This paper discusses the design, development, and implementation of the Club Management System, highlighting its impact on improving college event management through automation, efficiency, and enhanced user experience.

II. LITERATURE REVIEW

2.1 Online Technology

Research on online technology is crucial as the Internet serves as the backbone for modern systems. The **Club Management System (CMS)** is an online platform designed to facilitate communication between committee members and general members regarding events, activities, and updates. It incorporates various online tools, such as email services, to streamline communication.

Email, introduced by Ray Tomlinson in 1971 under the Advanced Research Projects Agency Network (ARPANET), remains a fundamental communication tool. Popular email providers include Gmail, Yahoo, and Outlook. Email services allow users to send and receive messages, share files, and collaborate efficiently. However, email services depend on internet availability and are susceptible to spam and spoofing. Despite these limitations, CMS leverages email to notify members about upcoming events and activities.

2.2 Short Message Service (SMS)

The Short Message Service (SMS) is widely used for quick and efficient communication. Initially designed for GSM networks, SMS is now supported across multiple platforms. By 2010, there were an estimated 3.5 billion active SMS users globally. SMS technology has evolved, enabling features like automated message retries and delivery confirmations.

SMS provides advantages such as immediacy, cost-effectiveness, and accessibility. However, challenges like message spoofing and spam remain. In the CMS, SMS notifications serve as an alternative to email for instant communication regarding club activities and event reminders.

2.3 Programming Technology

Before The Club Management System (CMS) is developed using modern web technologies. Programming languages are essential for implementing system functionalities. PHP, originally developed by Rasmus Lerdorf in 1994, was a widely used web development language. However, modern systems have shifted towards more robust solutions like MERN stack (MongoDB, Express.js, React, Node.js).

III. METHODOLOGY

3.1 System Development Approach

The Club Management System (CMS) follows a full-stack development approach using the MERN (MongoDB, Express.js, React.js, Node.js) stack. This technology stack provides a seamless, end-to-end JavaScript-based development environment, ensuring efficient data flow between the frontend, backend, and database. The system is designed to handle event management, user authentication, real-time notifications, and role-based access control. The modular architecture allows for easy scalability and integration with third-party services, such as email notifications and cloud storage.3.2 MERN Stack Components

3.2.1 MongoDB

MongoDB is used as the database management system to store user profiles, event details, participation records, and notifications. It provides a NoSQL document-based storage structure, making it flexible and scalable for handling large amounts of unstructured data. Its schema flexibility allows dynamic updates to the database structure as requirements evolve. The indexing and query optimization capabilities improve data retrieval performance, which is crucial for handling large datasets efficiently. Additionally, MongoDB supports replication and sharding, ensuring high availability and fault tolerance to prevent data loss and enhance system reliability.

3.2.2 Express.js

Express.js serves as the backend framework, providing a lightweight and flexible environment for handling HTTP requests and API endpoints. It enables seamless communication between the frontend and the database while managing critical operations such as user authentication, event creation, and notification services. The backend is responsible for processing login requests securely using JWT authentication, managing CRUD operations for events, users, and blogs, and handling the notification system by sending event reminders and updates via email or in-app alerts. Middleware integration ensures security by implementing input

https://doi.org/10.46335/IJIES.2025.10.7.20 e-ISSN: 2456-3463
Vol. 10, No. 7, 2025, PP. 96-100

validation, enforcing CORS policies, and logging requests to monitor system activity.

3.2.3 React.js

React.js is implemented for the frontend to provide a dynamic and interactive user interface that enhances the overall user experience. The component-based architecture of React improves code reusability and maintainability, making development more efficient. The frontend is designed with a responsive UI using Tailwind CSS, ensuring a mobile-friendly and modern layout adaptable to different screen sizes. Efficient state management is achieved through Context API and React hooks, which handle global states such as user authentication and event data. The system also incorporates real-time updates using WebSockets or polling techniques, ensuring users receive instant notifications regarding event registrations, updates, and announcements.

3.2.4 Node.js

Node.js is responsible for handling the server-side logic and real-time data processing. It operates on a non-blocking, event-driven model, making it highly efficient in managing multiple concurrent requests. The backend infrastructure built on Node.js ensures efficient request handling, enabling quick API responses and seamless data exchange between the client and server. The system supports real-time data communication through WebSockets, allowing instant messaging and live event updates. Additionally, Node.js enables background tasks such as automated event reminders and scheduled database cleanups to maintain system efficiency. Its scalability features support load balancing, ensuring that the system can handle increasing user traffic effectively without performance degradation.

3.3 System Features and Functionalities

- **User Authentication:** Secure login using JWT-based authentication.
- **Event Management:** Creation, modification, and deletion of events by administrators.
- **Real-Time Notifications:** WebSocket-based update for instant communication.
- **Analytics and Reporting:** Event participation tracking and performance analysis.

IV. System Architecture/ Design

4.1 Overview

The Club Management System (CMS) follows a three-tier architecture, consisting of the frontend, backend, and database layers. This modular approach enhances scalability, maintainability, and performance. Each layer plays a crucial role in ensuring a smooth and efficient system for managing club events and activities.

4.2 System Architecture Components

The frontend (client-side) is developed using React.js, providing users with a dynamic and interactive interface for event management and communication. Tailwind CSS is implemented to ensure a responsive and visually appealing design. The backend (server-side) is built using Node.js and Express.js, handling business logic and implementing RESTful APIs for smooth data exchange. This layer is responsible for managing authentication, user roles, and event scheduling.

4.3 System Flow Diagram

The system follows a structured data flow where users interact with the **frontend (React.js)** to register, log in, and manage events. The frontend communicates with the **backend (Node.js and Express.js)** through API requests. The backend processes these requests, retrieves or stores data in **MongoDB**, and returns the appropriate response to the frontend. Additionally, notifications, such as event reminders and updates, are sent to users via **email and SMS** to enhance communication and engagement.

4.4 Security Considerations

Security is a fundamental aspect of system design. JWT Authentication ensures that user logins are secure, preventing unauthorized access. Role-based access control (RBAC) is implemented to restrict specific actions based on user roles, ensuring that only authorized users can perform certain tasks. Data encryption is applied to protect sensitive user information from potential threats. Furthermore, rate limiting and input validation mechanisms are in place to prevent API abuse, ensuring system stability and security.

4.5Architecture Diagram

Use Case Diagram for Club Management System

The Use Case Diagram illustrates the interaction between users (students) and administrators within the Club Management System (CMS). It highlights the various functionalities accessible to each role.

Actors and Use Cases:

https://doi.org/10.46335/IJIES.2025.10.7.20 e-ISSN: 2456-3463
Vol. 10, No. 7, 2025, PP. 96-100

1. User (Student):
    - Can register and log in to the system.
    - Uses the Event Manager to register for events, get tickets, and receive confirmations.
    - Engages with the Blog Manager to read and comment on blogs.
    - Can search for blood donors, view notices, and access alumni details.
    - Queries the help desk for assistance.
    - Updates personal data in User Management.
2. Admin:
    - Manages events, including creation and archiving.
    - Handles user roles and permissions, including deletion and searches.
    - Manages club projects by adding, updating, or deleting project details.
    - Updates or deletes alumni records and notices.

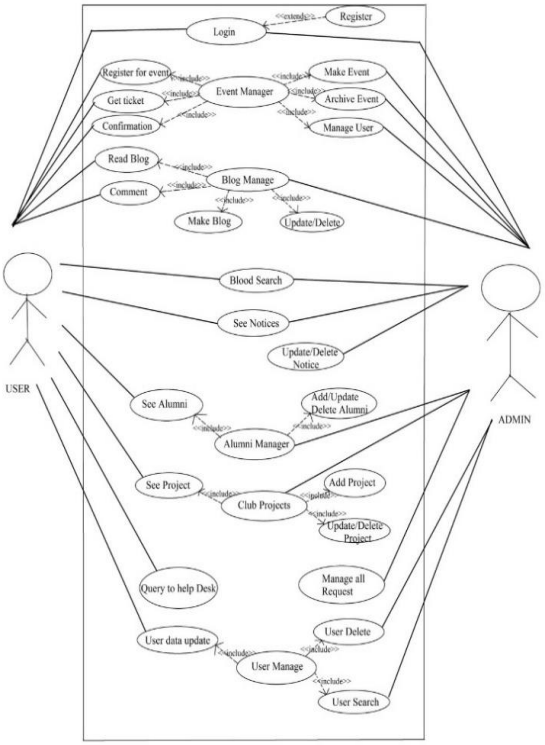

Fig 5.1 - Use Case Diagram

V. RESULT & DISCUSSION

5.1 Interpretation of Results:

The successful implementation of the Club Management System (CMS) demonstrates the effectiveness of using the MERN stack for developing a scalable and interactive event management platform. The frontend performance optimizations, including React Query and lazy loading, significantly improved user experience. The secure backend implementation with JWT authentication and role-based access control (RBAC) ensured data protection and prevented unauthorized access. Additionally, the real-time notification system using Nodemailer and Twilio API enhanced communication efficiency.

5.2 Implications of Findings

The findings suggest that a well-structured, modular system architecture enhances maintainability and future scalability. The use of NoSQL (MongoDB) for data storage provided flexibility, allowing seamless handling of structured and unstructured data. Furthermore, the integration of automated notifications improved event engagement, ensuring that students remained informed. The security measures implemented, including data encryption and API rate limiting, contributed to a reliable and protected system environment.

5.3 Relation to Existing Literature:

The results align with previous studies on web-based event management systems, confirming that real-time notifications, cloud-based data storage, and authentication mechanisms improve usability and security. Prior research on club management applications highlighted the need for better communication tools and streamlined event coordination, which CMS successfully addresses through integrated SMS/email alerts and an interactive dashboard. Additionally, the choice of MERN stack is supported by literature advocating for JavaScript-based full-stack development due to its flexibility, high performance, and extensive community support.

VI. CONCLUSION

6 Interpretation of Results:

The Club Management System (CMS) was successfully developed as a scalable, secure, and efficient solution for managing club events, user interactions, and administrative tasks. Built using the MERN stack, the system ensures seamless event coordination through real-time notifications, robust authentication, and an intuitive user interface. Key challenges, such as performance optimization, security risks, and database scalability, were effectively addressed using solutions like React Query for efficient data fetching, JWT

authentication for secure access, and MongoDB indexing for faster queries.

The system significantly enhances event organization, user engagement, and data management, overcoming limitations found in traditional methods. The results confirm that CMS provides a reliable, user-friendly platform for digitalizing club activities. Future enhancements could include AI-driven analytics for insights, chatbot integration for instant assistance, and blockchain-based verification for secure event registrations, extending its usability across multiple institutions.

REFERENCES

[1] *Mohd Shukri Ibrahim, Shahreen Kasim, Rohayanti Hassan, Hairulnizam Mahdin, Azizul Azhar Ramli, Mohd Farhan Md Fudzee, Mohamad Aizi Salamat. (2018). Information Technology Club Management System. Acta Electronica Malaysia, 2(2), 01-05.*

[2] *Lerdorf, R. (1994). PHP: Hypertext Preprocessor. PHP Documentation.*

[3] *Tomlinson, R. (1971). The First Email System. Advanced Research Projects Agency Network (ARPANET).*

[4] *Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. Pearson Education.*

[5] *MongoDB Inc. (2020). MongoDB: The Definitive Guide. O'Reilly Media.*

[6] *Freeman, A. (2018). Pro React 16. Apress.*

[7] *Flanagan, D. (2020). JavaScript: The Definitive Guide. O'Reilly Media.*

[8] *Resig, J., Bibeault, B. (2016). Secrets of the JavaScript Ninja. Manning Publications.*

[9] *Chacon, S., Straub, B. (2014). Pro Git. Apress.*

[10] *Bass, L., Clements, P., Kazman, R. (2012). Software Architecture in Practice. Addison-Wesley.*

[11] *Fielding, R. T. (2000). Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine.*

[12] *Pressman, R. S. (2019). Software Engineering: A Practitioner's Approach. McGraw-Hill.*

[13] *Gamma, E., Helm, R., Johnson, R., Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley.*

[14] *Fowler, M. (2002). Patterns of Enterprise Application Architecture. Addison-Wesley.*

[15] *Martin, R. C. (2017). Clean Architecture: A Craftsman's Guide to Software Structure and Design. Pearson Education.*

[16] *Nixon, R. (2018). Learning PHP, MySQL & JavaScript. O'Reilly Media.*

[17] *Sommerville, I. (2015). Software Engineering. Pearson.*

[18] *Rajaraman, V. (2018). Introduction to Information Technology. Prentice Hall India.*

[19] *Tanenbaum, A. S., Wetherall, D. J. (2011). Computer Networks. Pearson.*

[20] *Meriam, J. L., Kraige, L. G. (2011). Engineering Mechanics: Dynamics. Wiley.*