Nishant Kumar, 2026, 14:2
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Event Management Website: A Web-Based System for Automated Event Registration and Booking

Nishant Kumar[1], Adarsh Sharma[1] ,Ms. Amita Pathania[1] and Ms. Suman Chandila[1]

[1] Department of Computer Science and Engineering (Data Science),
Noida Institute of Engineering and Technology (NIET), Greater Noida, India

Abstract — Traditional event management processes in academic and organizational settings suffer from inefficiencies including data redundancy, double-booking errors, and limited scalability. This paper presents a web-based Event Management Website (EMW) designed to automate and streamline event registration, booking, and administration. The system employs a three-tier client-server architecture using HTML5, CSS3, and JavaScript for the frontend presentation layer; Node.js and Express.js for backend application logic; and MySQL as the relational database. The system supports two distinct user roles—regular users and administrators—and implements secure session-based authentication with bcrypt password hashing. Comprehensive testing including unit, integration, system, and user acceptance testing was conducted, achieving a 100% pass rate across 15 test cases. The proposed system reduces manual workload, eliminates booking conflicts, and provides a scalable foundation for future enhancements including AI-based event recommendations and mobile application development.

Keywords — Event Management, Web Application, Node.js, Express.js, MySQL, Three-Tier Architecture, Authentication, Booking System, SDLC.

I. INTRODUCTION

The digital transformation of administrative processes has become a critical priority for educational institutions and organizations worldwide [1]. Event management—encompassing seminars, workshops, cultural programs, technical competitions, and corporate meetings—remains one of the most resource-intensive administrative tasks when handled manually [2]. Paper-based registration forms, physical filing systems, and telephone-based coordination introduce substantial operational inefficiencies and error risks.

Web-based information systems have emerged as highly effective solutions to automate complex administrative workflows [3]. A properly designed web-based event management platform allows organizations to centralize event lifecycle management—from creation and promotion to registration and post-event reporting—within a single accessible interface [4]. Studies indicate that digital event platforms reduce administrative workload by up to 70% while significantly improving participant experience and data integrity [2].

Existing commercial solutions such as Eventbrite and Cvent offer comprehensive feature sets; however, their subscription-based pricing models, steep learning curves, and limited customization options make them unsuitable for small organizations and academic institutions [5]. There exists a clear demand for a lightweight, self-hosted, and easily customizable alternative.

This paper presents the design, implementation, and evaluation of an Event Management Website (EMW) developed specifically for colleges and small organizations. The system is built using modern open-source web technologies and follows the Waterfall Software Development Life Cycle (SDLC) model to ensure structured, documented, and verifiable development [6]. The remainder of this paper is organized as follows: Section II reviews

International Journal of Science, Engineering and Technology
An Open Access Journal

related work; Section III describes system requirements; Section IV presents the system design and methodology; Section V discusses implementation and testing results; Section VI concludes with future directions.

II. RELATED WORK

The evolution of event management systems mirrors the broader trajectory of web-based information systems. Early systems relied on desktop applications with limited network connectivity; modern platforms leverage cloud infrastructure and RESTful APIs [7].

A. Web-Based Event Management Platforms

Kumar and Sharma [8] presented a web-based event management framework for academic institutions using PHP and MySQL, identifying key requirements including role-based access control, real-time booking updates, and exportable reports. Their study highlighted that institutions using such systems reduced administrative errors by approximately 65% compared to manual approaches. Patel et al. [9] extended this work by integrating a payment gateway module, demonstrating the scalability of web-based architectures for event systems.

B. Node.js in Web Application Development

The Node.js runtime environment, built on Chrome's V8 JavaScript engine, has gained significant adoption in server-side web development due to its non-blocking I/O model and event-driven architecture [10]. Gupta and Mishra [11] demonstrated that Node.js with Express.js provides superior throughput for concurrent HTTP request handling compared to traditional synchronous server frameworks, making it particularly suitable for booking systems where simultaneous user interactions are common.

C. Artificial Intelligence in Event Management

Singh and Verma [12] explored the application of collaborative filtering and content-based filtering algorithms for event recommendation systems, demonstrating measurable improvements in user engagement when personalized recommendations were presented. More recently, transformer-based NLP models have been integrated into event platforms as chatbots capable of guiding users through registration workflows and answering contextual queries in real time [13]. The current EMW system does not include AI features but is architecturally designed to accommodate such integration in future development phases.

D. Research Gap

A review of existing literature reveals that while enterprise-grade platforms are well-documented, there is limited published research on lightweight, open-source, self-hosted event management systems designed specifically for academic institutions. The proposed EMW addresses this gap by providing a complete, tested, and documented implementation using contemporary web technologies.

III. SYSTEM REQUIREMENTS

A. Functional Requirements

The EMW system supports two primary user classes: registered users and administrators. Table I summarizes the core functional requirements for each role.

TABLE I Functional Requirements

ID	User Role	Function	Description
FR-01	User	Registration	Create account with name, email, phone, password
FR-02	User	Login / Logout	Secure session-based authentication
FR-03	User	View Events	Browse events with name, date, venue, price
FR-04	User	Book Event	Book events; confirmation stored in DB
FR-05	User	Booking History	View all past and current bookings
FR-06	Admin	Event CRUD	Create, update, delete events
FR-07	Admin	User Management	View all registered users
FR-08	Admin	Booking Management	View, confirm, or cancel all bookings

International Journal of Science, Engineering and Technology
An Open Access Journal

B. Non-Functional Requirements

Performance: The system must respond to user requests within 3 seconds under normal network conditions and support at least 100 concurrent users [14]. **Security:** Passwords are hashed using bcrypt; all input fields are sanitized to prevent SQL injection and cross-site scripting (XSS) attacks [15]. **Scalability:** The three-tier architecture ensures that presentation, business logic, and data layers can be scaled independently.

C. Operating Environment

The system requires Node.js v16+, Express.js v4+, MySQL v8.0+, and a modern web browser (Chrome v90+, Firefox v88+, or Edge v90+). Minimum hardware specifications include an Intel Core i3 processor, 4 GB RAM, and a broadband internet connection.

IV. SYSTEM DESIGN AND METHODOLOGY

A. Architecture

The EMW adopts a three-tier client-server architecture as recommended for web-based information systems [3]. The three tiers are:

Tier 1 – Presentation Layer: HTML5, CSS3, and JavaScript (ES6+) web pages rendered in the user's browser. Pages include Home, Registration, Login, Events Listing, Booking, User Dashboard, and Admin Panel.

Tier 2 – Application Layer: Node.js and Express.js server implementing the MVC pattern. Route handlers process authentication (/auth), event management (/events), booking operations (/bookings), and administrative functions (/admin) [10].

Tier 3 – Database Layer: MySQL relational database storing Users, Events, Bookings, and Admin tables. The mysql2 Node.js package with connection pooling manages database interactions using parameterized queries to prevent SQL injection [15].

B. Database Design

The database schema comprises four tables. Table II presents the primary entities, their keys, and relationships.

TABLE II Database Schema Summary

Table	Primary Key	Foreign Keys	Purpose
users	userID	—	Registered user accounts
events	eventID	—	Event details created by admin
bookings	bookingID	userID, eventID	User-event booking records
admin	adminID	—	Administrator credentials

The BOOKING table acts as a junction table resolving the many-to-many relationship between USER and EVENT entities. Referential integrity is enforced through InnoDB foreign key constraints.

C. Software Development Model

The Waterfall SDLC model was adopted due to the well-defined and stable requirements of the system [6]. The six phases—Requirement Analysis, System Design, Implementation, Testing, Deployment, and Maintenance—were executed sequentially over a 16-week development schedule, ensuring thorough documentation and review at each stage.

D. Security Design

Security was addressed at multiple layers. User passwords are hashed using bcrypt with a cost factor of 10 before database storage [15]. Session tokens are generated server-side upon authentication and invalidated on logout. All form inputs are validated both client-side (JavaScript) and server-side (Express.js middleware) to defend against injection attacks [16].

V. IMPLEMENTATION AND TESTING

A. Frontend Implementation

The frontend comprises multiple responsive web pages sharing a common HTML structure. CSS Grid and Flexbox are used to implement responsive layouts compatible with standard desktop browsers. Client-side JavaScript handles form validation including email format verification, password length

enforcement, and duplicate field detection prior to server submission [17].

B. Backend Implementation

The Express.js server is structured following the Model-View-Controller (MVC) pattern. Authentication routes implement bcrypt-based password hashing and Express-session for session lifecycle management. All event and booking CRUD operations are handled via RESTful API endpoints. Admin routes are protected by session middleware that verifies administrator role before processing requests [10].

C. Testing

The system was subjected to four testing levels as recommended by IEEE software testing standards [18]: (1) Unit Testing of individual route handlers and validation functions; (2) Integration Testing of frontend-backend-database interaction chains; (3) System Testing of all end-to-end user and admin workflows; and (4) User Acceptance Testing (UAT) conducted with actual end users under supervised conditions.

Table III summarizes the test case results across all three system modules.

TABLE III System Testing Results Summary

Module	Total TCs	Passed	Failed	Pass Rate
User Registration & Login	6	6	0	100%
Event Booking Module	4	4	0	100%
Admin Panel	5	5	0	100%
Total	15	15	0	100%

All 15 test cases passed successfully. UAT feedback confirmed that the event booking workflow is intuitive, the admin panel provides sufficient data control, and system response times remained within the 3-second threshold under tested load conditions.

VI. CONCLUSION AND FUTURE WORK

This paper presented the design, implementation, and evaluation of a web-based Event Management Website built using Node.js, Express.js, and MySQL following a three-tier architecture. The system successfully automates the event registration and booking workflow, eliminates double-booking errors through real-time database validation, and provides administrators with a centralized management dashboard. Comprehensive testing with a 100% pass rate across 15 test cases confirmed that all functional requirements are correctly implemented.

Future work will extend the system in several directions. First, integration of an online payment gateway (Razorpay or Stripe) will enable fully automated fee collection [9]. Second, an AI-based event recommendation engine using collaborative filtering will be developed to improve user engagement [12]. Third, NLP-powered chatbot support will be integrated to assist users with booking queries [13]. Fourth, a dedicated mobile application for Android and iOS platforms will broaden system accessibility. Fifth, advanced analytics dashboards will provide administrators with insights into booking trends, revenue, and user demographics to support data-driven event planning decisions.

REFERENCES

[1] M. A. Al-Emran, S. A. Salloum, and K. Shaalan, "A survey of using machine learning approaches for digital transformation," in Proc. Int. Conf. Advanced Intelligent Systems and Informatics, Cairo, Egypt, 2021, pp. 101–110.

[2] A. Bregman and J. Burleson, "Digital event management platforms: Adoption, efficiency, and user satisfaction in academic settings," Journal of Information Technology in Education, vol. 20, pp. 215–234, 2021.

[3] R. Fielding and R. N. Taylor, "Principled design of the modern web architecture," ACM Trans. Internet Technol., vol. 2, no. 2, pp. 115–150, 2022.

[4] S. Ahmad, F. Baig, and M. Hussain, "A comparative analysis of web-based event registration systems for universities," Int. J. Advanced Computer Science and Applications, vol. 13, no. 5, pp. 88–96, 2022.

[5] P. Verma and D. Rathi, "Evaluating commercial event management software for educational institutions: Limitations and alternatives," Int. J. Emerging Technologies in Learning, vol. 17, no. 3, pp. 44–57, 2022.

[6] R. S. Pressman and B. Maxim, Software Engineering: A Practitioner's Approach, 9th ed. New York, NY: McGraw-Hill Education, 2024.

[7] N. Zaman and M. Farooq, "Cloud-based web application architectures: A systematic review," IEEE Access, vol. 10, pp. 34501–34520, 2022.

[8] A. Kumar and P. Sharma, "Design and development of a web-based event management system for academic institutions," Int. J. Computer Science and Information Technology, vol. 12, no. 3, pp. 45–52, 2021.

[9] R. Patel, S. Joshi, and M. Trivedi, "Integrating secure payment gateways in web-based event management systems," Int. J. Web Engineering and Technology, vol. 17, no. 2, pp. 130–148, 2022.

[10] S. Tilkov and S. Vinoski, "Node.js: Using JavaScript to build high-performance network programs," IEEE Internet Computing, vol. 14, no. 6, pp. 80–83, 2021.

[11] R. Gupta and S. Mishra, "Performance benchmarking of Node.js versus traditional synchronous server frameworks for concurrent web applications," Int. J. Advanced Research in Computer Science, vol. 13, no. 4, pp. 112–121, 2022.

[12] M. Singh and A. Verma, "Collaborative filtering and content-based AI recommendation engines for event management platforms," Journal of Emerging Technologies and Innovative Research, vol. 9, no. 5, pp. 78–89, 2022.

[13] H. Zhang, Y. Liu, and W. Chen, "NLP-based intelligent chatbots for automated event management and user support," Expert Systems with Applications, vol. 198, p. 116850, 2023.

[14] ISO/IEC 25010:2023, Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Product quality model, International Organization for Standardization, 2023.

[15] M. Howard and D. LeBlanc, Writing Secure Code for Web Applications, 3rd ed. Redmond, WA: Microsoft Press, 2023.

[16] OWASP Foundation, "OWASP Top Ten 2021: The ten most critical web application security risks," [Online]. Available: https://owasp.org/Top10, 2021. [Accessed: Mar. 2025].

[17] D. Flanagan, JavaScript: The Definitive Guide, 7th ed. Sebastopol, CA: O'Reilly Media, 2020.

[18] IEEE Std 829-2022, IEEE Standard for Software and System Test Documentation, IEEE, New York, NY, 2022.