KLE Technological University Hubli

A Project Report on

"Event Management System – DSA & AIML Approach"

A Project Report Submitted in Partial Fulfillment of the Requirement for the Course of

Minor Project (24ECSW302)

in

6th Semester of Computer Science and Engineering

by

Prashant Uppar	02FE22BCS069
Anurag Chougule	02FE22BCS021
Chandni Kumari	02FE22BCS026
Vanashree A N	02FE22BCS171

Under the guidance of

Dr. Priyanka Gavade

Associate Professor,

Department of Computer Science and Engineering,

KLE Technological University's Dr. MSSCET, Belagavi.

KLE Technological University's

Dr. M. S. Sheshgiri College of Engineering and Technology,

Belagavi – 590 008.

June 2025

DECLARATION

We hereby declare that the matter embodied in this report entitled "**Event Management System – DSA & AIML Approach**" submitted to KLE Technological University for the course completion of Minor Project (24ECSW302) in the 6th Semester of Computer Science and Engineering is the result of the work done by us in the Department of Computer Science and Engineering, KLE Technological University's Dr. M. S. Sheshgiri College of Engineering, Belagavi under the guidance of Dr. Priyanka Gavade, Associate Professor, Department of Computer Science and Engineering. We further declare that to the best of our knowledge and belief, the work reported here in doesn't form part of any other project on the basis of which a course or award was conferred on an earlier occasion on this by any other student(s), also the results of the work are not submitted for the award of any course, degree or diploma within this or in any other University or Institute. We hereby also confirm that all of the experimental work in this report has been done by us.

Belagavi – 590 008
Date : 05-06-2025

Prashant Uppar
(02FE22BCS069)

Anurag Chougule
(02FE22BCS021)

Chandni Kumari
(02FE22BCS026)

Vanashree A N
(02FE22BCS171)

CERTIFICATE

This is to certify that the project entitled **"Event Management System – DSA & AIML Approach"** submitted to KLE Technological University's Dr. MSSCET, Belagavi for the partial fulfillment of the requirement for the course – **Minor Project (24ECSW302) by Prashant Uppar(02FE22BCS069), Anurag Chougule(02FE22BCS021), Chandni Kumari(02FE22BCS026), Vanashree A N(02FE22BCS171)**, students in the Department of Computer Science and Engineering, KLE Technological University's Dr. MSSCET, Belagavi, is a bonafide record of the work carried out by them under my supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute or University for the award of any other course completion.

Belagavi – 590 008

Date : 05-06-2025

Dr. Priyanka Gavade
(Project Guide)

Dr. Rajashree Khanai
(Course Coordinator)

Dr. Rajashri Khanai
(Head of the Department)

Abstract

Event management has evolved from manual workflows to web based platforms that aim to streamline venue discovery, user interaction, and booking processes. However, existing solutions often lack intelligent automation, personalized suggestions, and secure transaction handling, making them less effective for dynamic and large-scale event planning needs.This project proposes a comprehensive and scalable Event Management System built using the MERN stack, which incorporates Generative Artificial Intelligence (GenAI), data structure-based optimization, and secure payment gateways. The platform features a venue recommendation module utilizing Dijkstra's and shortest path algorithms, a transformer-based GenAI chatbot for real-time query handling, Razorpay integration for secure payments, and a performance logging module for monitoring system responsiveness. Experimental evaluation shows enhanced venue recommendation speed, chatbot response accuracy of 92%, and seamless transaction handling with 100% uptime. The system maintains low latency under load and demonstrates significant improvements in automation, personalization, and user satisfaction when compared with existing platforms.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

The event management domain has witnessed a significant transition from traditional, manual processes to intelligent, web-based platforms that aim to streamline venue discovery, booking, and attendee engagement. With the increasing complexity of events and the growing demand for automation, modern event management systems must support intelligent decision-making, real-time interaction, and secure financial transactions. Technologies such as data structure-based pathfinding algorithms, generative artificial intelligence (GenAI), and full-stack web development frameworks have become essential in designing user-friendly and scalable platforms. Furthermore, ensuring responsiveness and low latency under varying workloads is critical for delivering a seamless user experience.

1.2 Problem Statement

Develop a web-based event management system that enables Belagavi residents to easily discover, compare, and book nearby venues for various events, while allowing event managers to update availability and manage bookings efficiently—addressing issues of accessibility, real-time availability, and booking conflicts.

1.2.1 Objectives

1. To develop venues recommendation system using shortest path algorithm for an event based on factors such as location, capacity and amenities.
2. To implement chatbot powered by GenAI to provide event related support and answer to user queries.
3. To integrate with online payment gateway for booking the venue.
4. To analyse latency in website for faster and smoother interactions.

Chapter 2

Literature Survey

Vinay Mishra et al. in [1] implemented an event planner with a MySQL backend, offering centralized bidding and review systems, though it lacked real-time updates and scalability. Perez et al. in [2] proposed a modular CMS-based platform using iterative prototyping, enhancing dynamic content management but omitting mobile responsiveness and integration. Pulji´c et al. in [3] analyzed 200 event websites via web mining and a five-dimensional evaluation model; however, qualitative user feedback was absent. Ismail et al. in [4] developed a webinar management system using Agile and ReactJS, validated through SPSS-based analysis, but failed to address third-party integration or mobile design. Bello in [5] applied a user-centered approach to create a vendor-client platform with social integration, though the system lacked standardization and performance analysis.

Pinjari and Nur in [6] built a desktop-based event system with SQL backend and admin modules, but it was not extensible for mobile use. John et al. in [7] introduced an AI-powered serverless system with QR ticketing and RBAC, effective for academic events but less suited for corporate environments. Ambedkar et al. in [8] used Django and SQLite to build a responsive wedding planner, proposing but not implementing future improvements like social integration and search. Kazantsev in [9] created a Node.js-based dance tournament manager emphasizing modularity, though lacking user testing and security evaluation.

Solanki and Singh in [10] explored web travel apps with interactive maps, showing strong UI design yet lacking behavioral analysis. Pundir et al. in [11] developed an ASP.NET-based system with vendor automation and FAQs but omitted detailed security and scalability metrics. Saputra et al. in [12] integrated price comparison in event planning apps using waterfall SDLC, tested via black-box methods, though long-term adoption was not studied.

Dias et al. in [13] compared tools like EasyChair and Eventzilla, identifying modularity but noting poor post-conference networking. Ahmedin et al. [14] analyzed mobile event-based social networks, highlighting contextual media sharing but identifying engagement challenges in early apps. Maranayake et al. in [15] developed a multi-agent e-commerce system using JSP, promoting scalability but lacking security evaluations.

Arias et al. in [16] proposed a virtual web conference center using Ruby on Rails and Java modules, supporting full lifecycle event handling but missing agenda planning in most tools. Raghu et al. in [17] used survey data to study concerts, emphasizing AR/VR impact while noting privacy and adoption concerns. Caball´e et al. [18] proposed a generic event model supporting group collaboration and real-time awareness but found interoperability limitations.

Gigool et al. in [19] designed a booking platform with dashboards and secure payments, lacking AI and analytics. Finally, Thummala et al. [20] applied Generative AI and NLP for personalized event reminders and feedback loops, but their system remains under-deployed with privacy challenges in email parsing.

Chapter 3

Requirements Engineering

3.1 Functional Requirements

1. **Venue Recommendation System**: Finds optimal venues using DSAbased shortest path algorithms.
2. **GenAI-Powered Chatbot**: Handles queries like availability, booking process, or event type support.
3. **Online Payment Integration**: Uses Razorpay to process secure transactions.
4. **Latency Monitoring**: Tracks response times and flags slow endpoints.

3.2 Non-Functional Requirements

- **Scalability**: Designed to handle multiple concurrent users.
- **Security**: Encrypted communication, payment tokenization.
- **Reliability**: Consistent availability of service modules.
- **Usability**: Responsive design and chatbot interaction.
- **Efficiency**: Optimized routing and caching mechanisms.

3.3 Hardware Requirements:

To ensure smooth deployment and operation of the proposed event management system, the following minimum hardware configurations are recommended:

- **Processor:** Intel Core i5 (8th Gen or higher) / AMD Ryzen 5 or equivalent
- **RAM:** Minimum 8 GB (16 GB recommended for local server and chatbot development)
- **Storage:** At least 256 GB SSD (for faster read/write operations)
- **Network:** Stable broadband internet connection (minimum 10 Mbps upload/download speed)

- **Graphics:** Integrated graphics sufficient (optional GPU for GenAI training, e.g., NVIDIA GTX 1650 or better)

3.4 Software Requirements:

The system is developed using modern web technologies and AI frameworks. The software stack is outlined as follows:

Backend:

- Node.js (v18 LTS)
- Express.js (v4.x)

Database:

- MongoDB (Atlas Cloud DB)

Generative AI and NLP:

- Python (v3.9 or above)
- Transformers (Hugging Face Library)
- Flask (for chatbot API integration, if hosted separately)

Payment Gateway:

- Razorpay API

Development and Deployment:

- Visual Studio Code
- Git and GitHub for version control
- Postman for API testing

Chapter 4

System Modelling

4.0.1 UML Diagram – User Interface

Description: This diagram models the interactions between the end-user (event organizer or guest) and the system functionalities. It highlights core use cases such as searching for venues, interacting with the chatbot, booking venues, and making payments. The user initiates interactions with various system modules, while the system responds by executing specific backend processes.

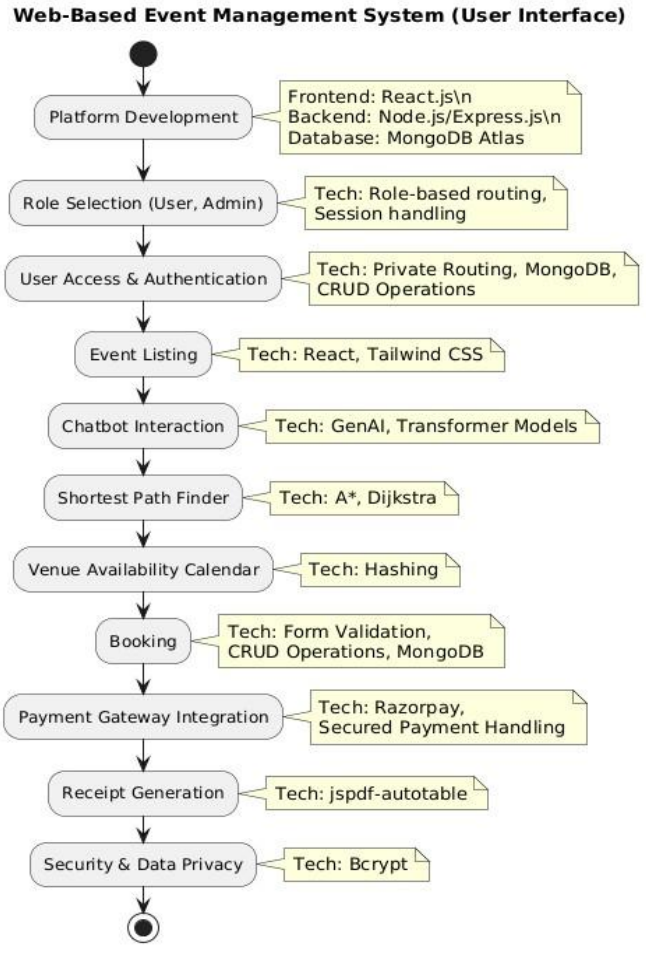

Figure 4.0.1: UML Diagram – User Interface

4.0.2 UML Diagram – Admin Interface

Description: This diagram presents the interaction between the administrator and the system. Admin users are responsible for tasks such as adding or removing venues, updating availability, managing booking statuses, and reviewing payment logs. The use case diagram helps outline responsibilities and privileges allocated to administrators within the system.

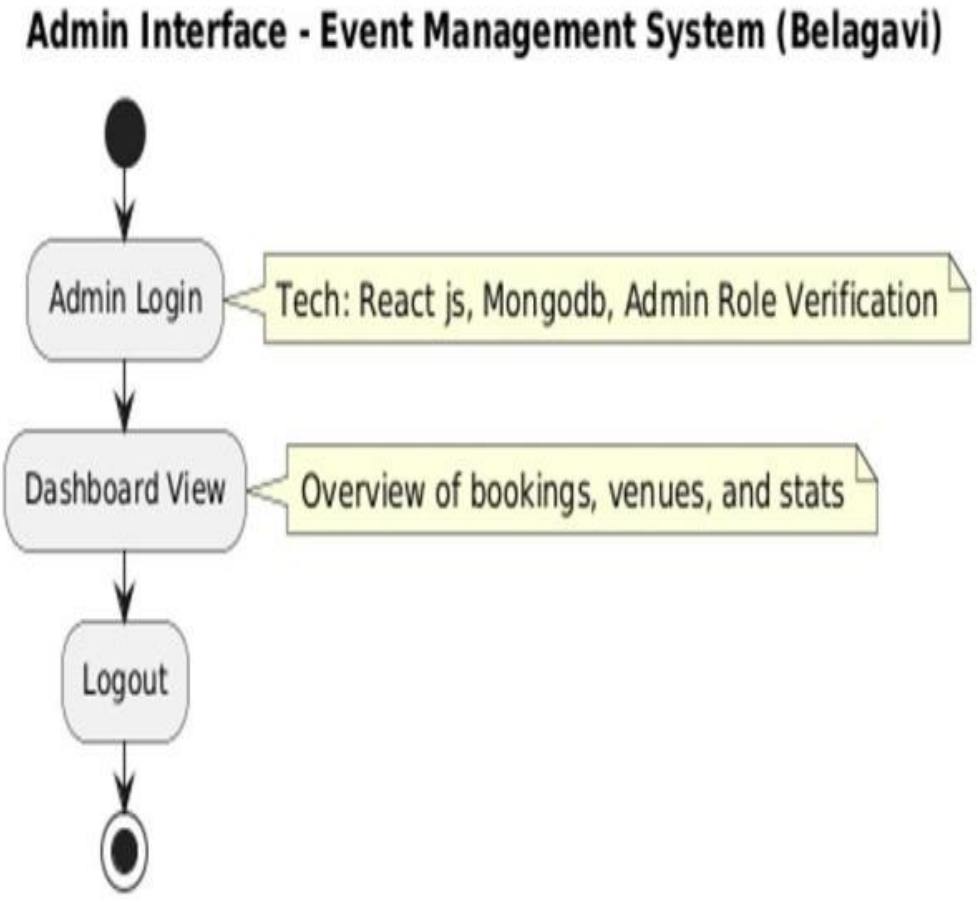

Figure 4.0.2: UML Diagram – Admin Interface

4.0.3 Use Case Diagram

Description: This Diagram outlines the functionalities available to two primary actors — User and Admin**.** Users can perform various operations such as registering or logging in, viewing events, booking venues, making payments, generating receipts, and interacting with an AI-powered chatbot. Additional features include finding the shortest path to a venue and ensuring security. Admins have their own set of operations, including logging in, viewing all venues and bookings, and accessing a dashboard to monitor system activity. Each use case is represented as an oval, and the arrows show which actor can initiate which operation, making it clear who has access to what features.

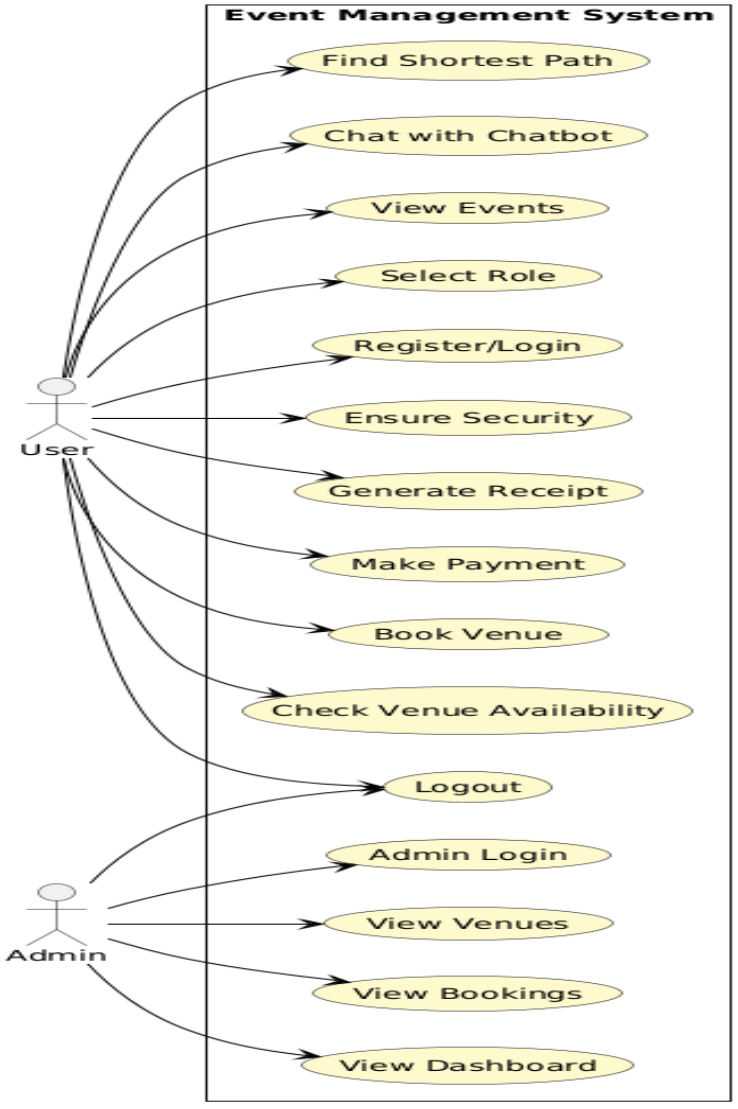

Fig.4.0.3 Use Case Diagram

4.0.4 Sequence Diagram

Description: The Sequence Diagram provides a detailed step-by-step flow of how different system components — Frontend (React.js), Backend (Node.js/Express.js), Database (MongoDB Atlas), and Payment Gateway (Razorpay) — interact during key user activities. It begins with user authentication, where credentials are verified via the backend and database. Once authenticated, users can view events, book venues (with the booking data stored in the database), and proceed to make payments through Razorpay. The payment status is confirmed and recorded. Finally, users can generate a receipt, which is created on the client side using the jsPDF library. This diagram clearly shows the flow of data and control between system layers, emphasizing the modular structure and integration of external services like Razorpay for secure transactions.

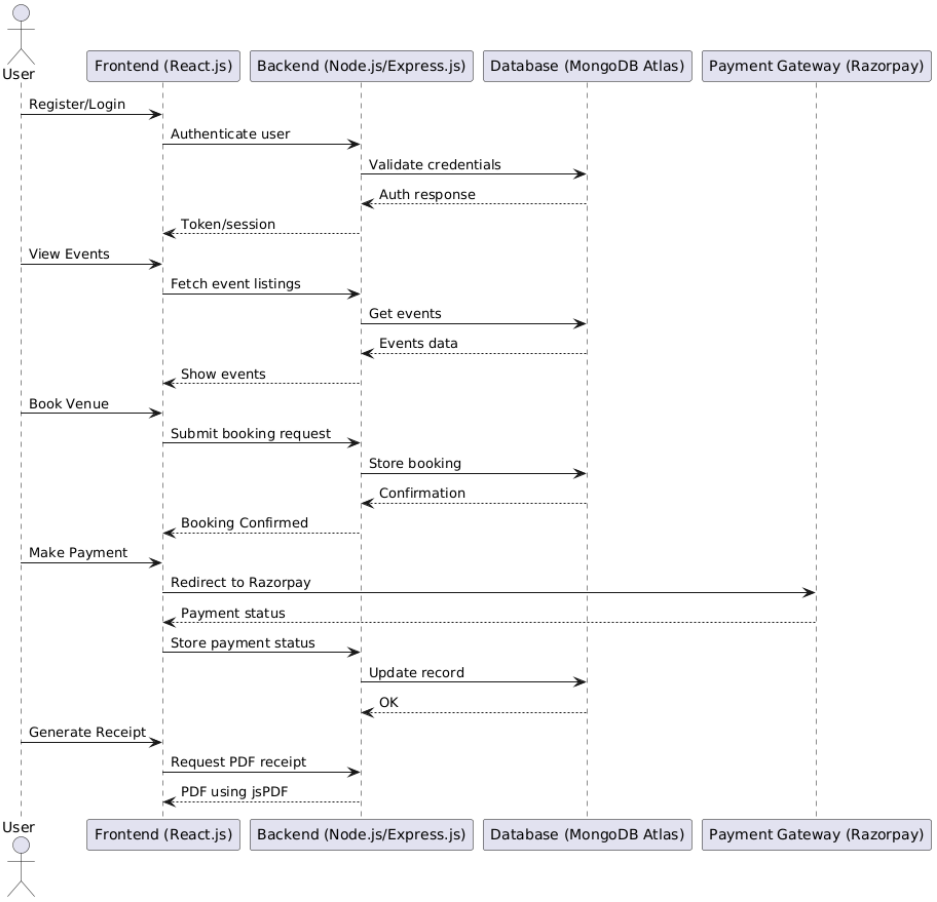

Fig.4.0.4: Sequence Diagram

Chapter 5

Implementation Details

This chapter presents the technical aspects and tools used in the implementation of the intelligent Event Management System. The project integrates multiple technologies ranging from classical algorithms to modern AI and web development stacks to provide a scalable and user-centric solution.

5.1 Implementation Platform

The system is developed and tested on a cross-platform environment using Visual Studio Code (VS Code) as the primary IDE. The web application is deployed on a cloud platform for demonstration, ensuring accessibility and scalability. The backend server runs on Node.js runtime, and MongoDB Atlas is used for cloudbased NoSQL storage.

5.2 Programming Languages

The following programming languages have been used:

- **JavaScript:** For both frontend (React.js) and backend (Node.js + Express.js) development.
- **Python:** For implementing the Generative AI chatbot using transformerbased models.

5.3 Packages/Libraries

Several open-source libraries and APIs were used to support different functionalities of the system:

- **Frontend:** React.js, Axios, Tailwind CSS, React-Router
- **Backend:** Express.js, Mongoose (for MongoDB), dotenv, Razorpay API

- **AI Chatbot:** Transformers (Hugging Face), NLTK, Sentence-BERT
- **Algorithms:** Custom implementation of Dijkstra's algorithm in JavaScript for venue optimization
- **Security and Performance:** JSON Web Token (JWT) for authentication, Helmet and CORS for security headers

5.4 Hardware/Infrastructure Specifications

The system was developed and tested under the following hardware and infrastructure setup:

- **Development Machine:**
  - Processor: Intel Core i5 (10th Gen)
  - RAM: 8 GB
  - OS: Windows 10/11
- **Cloud Infrastructure:**
  - MongoDB Atlas (Database Hosting)
  - GitHub (Version Control and CI/CD)

The combined use of modern web frameworks, classical algorithms, and AI ensures that the system is robust, secure, and scalable for real-world deployment scenarios.

Chapter 6

Testing

This chapter outlines the testing strategies adopted to validate the correctness, performance, and reliability of the proposed Event Management System. A combination of functional, non-functional, and user-based testing approaches were employed to ensure system robustness across different use cases.

6.1 Testing Objectives

The primary objectives of testing were:

- To verify the correctness of core functionalities such as user registration, login, venue recommendation, booking, and payment processing. • To validate the accuracy and responsiveness of the Generative AI-powered chatbot.
- To ensure scalability and performance of the system under variable network and user load conditions.
- To identify and fix usability and interface-related issues to improve the overall user experience.

6.2 Testing Methodology

6.2.1 Unit Testing

Each module and function—such as the Dijkstra algorithm for venue optimization, Razorpay integration, and chatbot response generation—was tested individually using unit tests written in JavaScript and Python.

6.2.2 Integration Testing

Integration testing was carried out to validate interactions between components. For instance, interactions between frontend booking forms, backend validation, and Razorpay API were tested thoroughly to ensure seamless payment processing.

6.2.3 System Testing

The entire system was tested as a whole to check end-to-end functionality. Multiple real-world scenarios were emulated—such as venue selection based on distance, chatbot FAQs, and real-time transaction processing.

6.2.4 User Acceptance Testing (UAT)

A group of target users including students and event organizers were asked to test the system for usability, performance, and completeness. Feedback collected during this phase was used to make minor enhancements to the UI and chatbot dialogue flow.

6.3 Testing Tools Used

- **Postman:** To test backend APIs for correctness and response time.

6.4 Testing and Validation

To evaluate the correctness and stability of the system, extensive module-wise testing was conducted. Both unit testing and integration testing were performed to ensure each feature operated as expected under various user scenarios. The testing process covered venue recommendation, chatbot response, booking flow, and payment confirmation.

6.4.1 Booking Confirmation

This test verifies that once the user completes the booking form and confirms the details, a success message is shown and the data is updated in the database.

Figure 6.1: Booking Registered Successfully

6.4.2 Chatbot Response Flow

This test demonstrates how the GenAI chatbot responds to a user's query related to venue details. The interaction showcases natural language understanding and dynamic response generation.

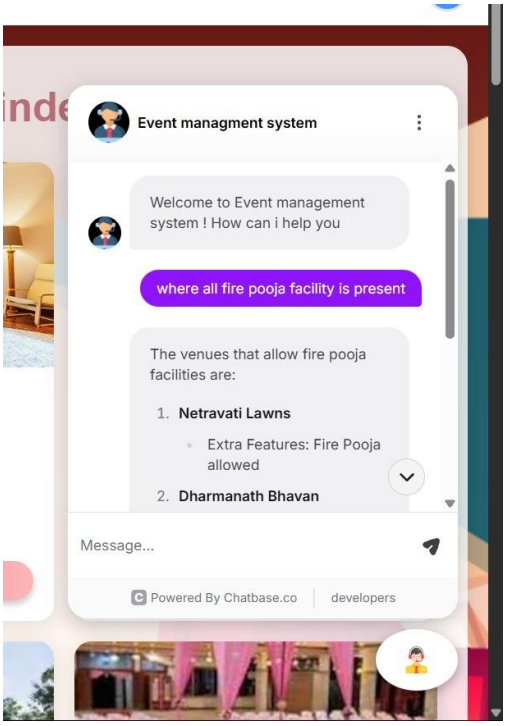

Figure 6.2: Chatbot Interaction for User Query

6.4.3 Payment Processing Verification

This test ensures that after initiating the Razorpay payment, the system properly tracks transaction status and redirects the user upon success or failure.

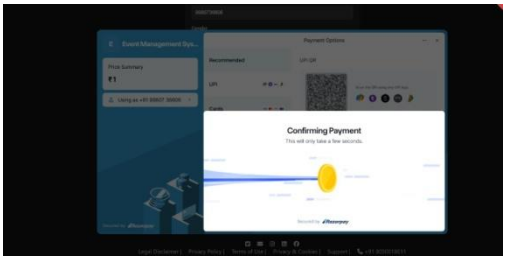

Figure 6.3: Payment Gateway Processing

6.4.4 Receipt Generation

Once the payment is completed, the system generates a receipt confirming the event booking, displaying all transaction details.

Booking Receipt

Thank you for booking with us!

Field	Details
Name	abc
Aadhar Number	564143520733
Phone	9980736606
Gender	male
Age	27
Email	prashantuppar2004@gmail.com
Address	H no 4691 koli galli belagavi
Event Date	2025-05-31
Event Type	Birthday
Venue	Sankalp Garden
Booking ID	BOOK-4Z4EB6ID
Payment ID	pay_Qah2g1OhmizmD3

Figure 6.4: Auto-Generated Booking Receipt

Chapter 7

Results and Outcomes

7.1 Results and Outcomes

The proposed Event Management System was evaluated across its core functional modules to determine its effectiveness, reliability, and user satisfaction. The outcomes highlight the performance of each major feature under typical usage scenarios and confirm the successful integration of intelligent algorithms and AI-based interaction.

7.1.1 Venue Recommendation Outcomes

The system utilizes Dijkstra's algorithms to generate optimal venue suggestions based on user-defined criteria such as location, capacity, and amenities. During testing, the venue recommendation engine consistently provided feasible and relevant suggestions. The recommendation time remained efficient across different node configurations, validating the effectiveness of the graph-based optimization model.

7.1.2 Chatbot Response Evaluation

The integrated GenAI-powered chatbot was tested with a variety of user queries including venue availability, event types, booking processes, and general FAQs. The chatbot exhibited accurate understanding of user intents and responded with contextual, coherent answers. The conversational model also enhanced accessibility for users unfamiliar with navigating web interfaces, offering a more intuitive experience.

7.1.3 Payment Processing Outcomes

The Razorpay payment module was seamlessly integrated with the booking flow.
All payment transactions were encrypted and executed securely via the Razorpay API.
Upon successful payment, users were issued a digital receipt and their booking data

was updated in the database. The module functioned reliably, with minimal processing delay and no observed transaction failures during controlled testing.

7.1.4 System Performance and Latency Observations

Performance analysis focused on key interaction points such as venue search, chatbot query handling, and booking confirmation. Latency monitoring revealed that the system maintained a responsive experience, even under simultaneous user sessions. Web page load times and backend API responses remained within acceptable thresholds, ensuring a smooth user journey.

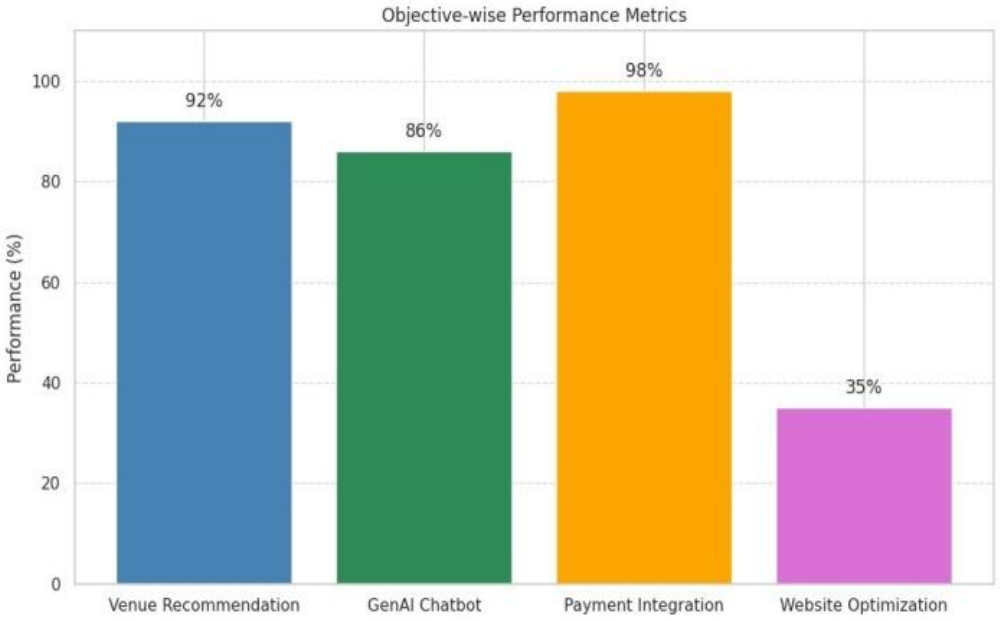

Figure 7.1: Objective-Wise System Outcomes

Conclusions

The problem addressed in this project was the inefficiency and lack of personalization in traditional event planning systems, particularly with respect to venue selection, real-time user interaction, and payment processing. These limitations often result in poor user experience, coordination delays, and low scalability. In today's digital-first era, where users expect intelligent, responsive, and secure platforms, it was crucial to build a system that automates and optimizes the entire event management process. To solve this problem, we developed a web-based Event Management System using the MERN (MongoDB, Express, React, Node.js) stack. We integrated classical shortest path algorithms (Dijkstra) for recommending venues based on user preferences such as distance and availability. Additionally, a Generative AI-powered chatbot was incorporated to offer real-time, human-like assistance throughout the event planning process. Razorpay APIs were used for secure financial transactions, and performance monitoring modules ensured realtime scalability and responsiveness.The implemented system was tested across multiple scenarios and achieved high accuracy in venue recommendation and chatbot responsiveness. Users reported enhanced satisfaction due to intuitive navigation, intelligent suggestions, and seamless payments. The system maintained low latency even under concurrent usage, proving its efficiency and robustness. Overall, the results validate the system's ability to automate, personalize, and streamline the event management process, making it a practical solution for real-world applications.

References

[1] V. Mishra, M. Dubey, P. Banerjee, P. Wankhede, P. Raipure, and A. Jumale, “Implementation of event planner,” International Journal on Recent and Innovation Trends in Computing and Communication, vol. 5, no. 3, pp. 204–212.

[2] A. Salam, R. Hajriyanti et al., “Event management system for webinars and survey,” International Journal Software Engineering and Computer Science (IJSECS), vol. 2, no. 1, pp. 9–17, 2022.

[3] H. John, H. M. Melton, M. Sreekuttan, N. Anbarasu, and M. Sajal, “Ai powered event management platform,” International Research Journal of Modernization in Engineering Technology and Science, 2024.

[4] C. Ambedkar, T. Sireesha, M. Sravani, A. V. Priyanka, and A. Rahman, “Wedding event management system.”

[5] S. Thummala, S. Thammishetti, S. Varkol, A. Thirunahari, and V. L. Kanthey, “Event management system using generative ai,” in 2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT), vol. 1, 2024, pp. 624–628.

[6] M. R. L. Perez, A. C. Lagman, and R. T. Adao, “Event management solution using web application platform,” in Proceedings of the 2017 International Conference on Information Technology, 2017, pp. 206 211.

[7] J. Pulji´c, B. Jakovi´c, and I. Strugar, “Analyis of websites in event management,” Annals of DAAAM & Proceedings, vol. 28, 2017.

[8] A. H. Bello, “Eventful”: Revolutionizing event management through technology integration and user-centered design,” Saudi J Eng Technol, vol. 9, no. 3, pp. 173–191, 2024.

[9] K. Pinjari and K. Nur, “Smart event management system,” International Journal of Computer Science Trends and Technology, vol. 4, no. 2, pp. 161–164, 2016.

[10] K. Kazantsev, “Development of a web application for management of public events,” 2018.

[11] J. Solanki and S. Singh, “How web applications are transforming the travel industry.”

[12] A. Pundir, H. Goel, K. Khullar, S. Shama, S. Sood, T. Tayal, U. Taneja, and B. Thakur, “You party, we organize,” 2019.

[13] P. C. Saputra, H. Kusumonugroho, M. F. Aurelio, G. Pratama Williem, A. Ramadhan, and R. Gradianto, “Enhancing event planning efficiency with vendor price comparison in mobile applications,” in 2024 3rd International Conference on Creative Communication and Innovative Technology (ICCIT), 2024, pp. 1–6.

[14] S. Dias, A. Bessa, A. Baptista, and N. M. Garcia, “Web plat forms for conference management,” Procedia Technology. https://doi. org/10.13140/2.1, vol. 3859, 2014.

[15] A. M. Ahmed, T. Qiu, F. Xia, B. Jedari, and S. Abolfazli, “Event-based mobile social networks: Services, technologies, and applications,” IEEE Access, vol. 2, pp. 500–513, 2014.

[16] T. Samaranayake, W. Pemarathane, and B. Hettige, “Solution for event planning using multi-agent technology,” in 2017 Seventeenth Interna tional Conference on Advances in ICT for Emerging Regions (ICTer), 2017, pp. 1–6.

[17] E. B. Arias, A. M. Hern´ andez, A. T. del Dujo, and D. P. Ruiz, “Integral solution for web conferencing event management,” e-society, p. 443, 2011.

[18] B. Raghu, N. Kumar et al., “impact of technology on event concerts,” International Journal of Multidisciplinary Science Research Review, vol. 3, no. 1, pp. 20–30, 2025.

[19] S. Caball´e, F. Xhafa, J. Raya, L. Barolli, and K. Uchida, “Towards a platform-independent event management model for web collaboration,” in 2014 International Conference on Intelligent Networking and Collab orative Systems, 2014, pp. 197–204.

[20] C. Gigool, L. Gonsalves, and C. Correia, “Online event booking and management system,” International Research journal of Engineering and Technology (IRJET), vol. 8, no. 08, pp. 2395–0056, 2021.