International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Departmental Event Management System using MERN Stack

Lokam Roopa Rani[1], *Dr. V. Aruna*[2]

[a] *Lokam Roopa Rani, Student, MCA, Aurora Deemed to be University, PG student, Aurora Higher Education and Research Academy, (Deemed to be University),Hyderabad, Telangana.*

[b] *Dr.V.Aruna Dean of Academics, Aurora Deemed to be University, Faculty, Aurora Higher Education and Research Academy, (Deemed to be University)Hyderabad, Telangana.*

ABSTRACT:

Departments organize regular workshops, seminars, industry talks, and technical fests. Manual coordination between roles (Admin/HOD/Faculty/Students) causes delays, lack of visibility, and fragmented data. This paper introduces a web-based Departmental Event Management System (DEMS) that computerizes the end-to-end process—user onboarding, event creation, registration, report submission, evaluation, and feedback—based on a secure, role-based architecture. The system uses a MERN architecture with JWT authentication, emails notifications, and stores data in MongoDB. We formally model the system through Use Case, Flowchart, Sequence, and ER diagrams and specify APIs, data model, and validation strategy.

Keywords: Event Management; MERN; JWT; MongoDB; Role-Based Access; Email Notifications; Academic Workflow

Introduction:

As events go large-scale in institutions, coordination in a timely and transparent manner is a challenge. Handoffs between Admins, HODs, Faculty, and Students tend to go across fragmented tools (forms, email, spreadsheets), leading to delays and gaps in audits. DEMS streamlines these processes into one platform with role-based dashboards , auditable activity, and automatic notifications. We organize this paper in the typical layout style of IOSR-JCE articles (e.g., 'Smart Ambulance Service System', 2020) and tailor it to our academic-event setup.

Review of Literature:

1)Role-Based Academic Portals: Earlier campus-management portals focus on role isolation and modular services with less interference and greater security. Research indicates that token-based auth with clear role separation increases traceability and compliance.

2) Event & Submission Systems: Conference and coursework sites show that built-in submission–review pipelines with notification enhance turnaround time and participant satisfaction.

3) NoSQL for Academic Data: The document model of MongoDB accommodates heterogeneous event metadata and nested submissions/ evaluations, facilitating greater iteration speed than strict relational schemas for adaptive academic workflows.

Methodology:

Existing Methodology

In all but a handful of colleges, departmental affairs are also handled in a traditional manner. The registrations are completed on paper application forms or simple web application forms that are not linked to a genuine database. News about events and developments is generally communicated through notice boards, printed postery, or word of mouth between students and teachers.

Results and attendance are usually tracked in different spreadsheets by various staff members, which leads to duplication and inaccuracies. Security is also not good since basic login procedures are adopted, and no proper mechanism exists to monitor what various users can see. In general, the existing process is time-consuming, non-systematic, and generates additional work for students and staff.

Proposed Methodology Using Software Testing

DEMS makes use of a React front-end, Node.js/Express REST APIs, and MongoDB for storage, along with a file storage layer for report storage. JWT is utilized for authentication, role-based middleware for authorization. Email is utilized for the sending of credentials, event updates, and result notifications.

B. Client Side

1) Admin and HOD Dashboard: Create user, assign role, CRUD operations, and add faculty evaluators. Automatic sending of credential emails on user creation; event updates inform stakeholders.

2) Faculty Dashboard: View assigned events, submit ratings (scores, feedback), view reports uploaded, and view history.

3) Student Portal: Post reports (with metadata), sign up for events, view the status of evaluations, and view feedback.

Software Testing Approach

To ensure that the system functions properly and is fault-free, the following testing procedures were employed:

Unit Testing – Every and any tiny component of the system was individually tested to see if it is functional or not.

Integration Testing – Then the flow was tested once modules were integrated to verify whether they are functioning in tandem.

System Testing – The system as a whole was tested in order that the system satisfies the Admin, HOD, Faculty, and Student requirements.

User Acceptance Testing – The system was tested from the perspective of an end-user to ensure that the system is easy to use and functions as it should. Security Testing – Authentication of login, role-based access, and database security were tested to ensure data is secure and only accessible by authorized users.

System architecture

Frontend (Client-Side):

Developed with React.js.

Offers dashboards for Admin, HOD, Faculty, and Students.

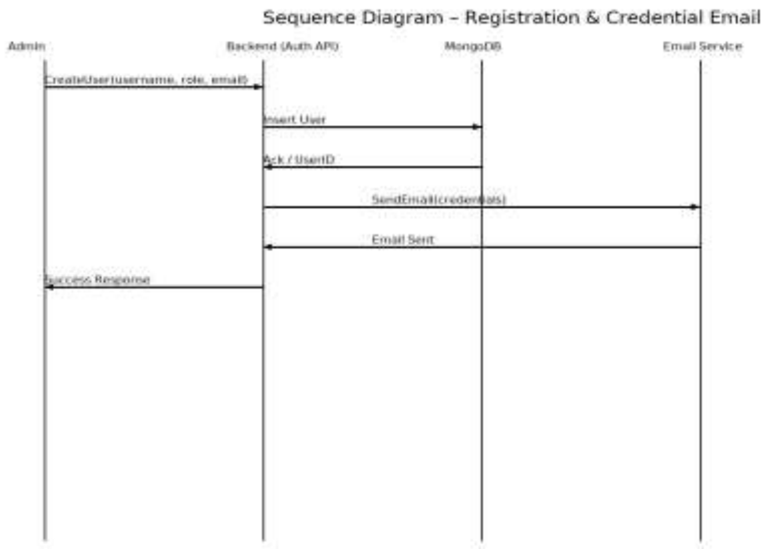
Supports user operations such as login, event registration, and upload of reports.

Backend (Server-Side):

Developed with Node.js and Express.js.

Supports business logic such as authentication, event generation, assignment of reports, and scoring. The architecture is modular, following a workflow model: Home Page → login page→ Respective Dashboard → submissions page →Evaluation page→ adminDashboard → Filter/Respond → Export Reports.

Fig:1 (sequence Diagram)



Results

Admin can register users such as students, faculty, HOD, dean and give them credentials by email. Role-based access is offered to all users through secure JWT authentication.

• Dashboards specific to roles were implemented for Admin, HOD, Faculty, and Students.

• HOD can dynamically create, update, and manage department events. Students can enroll in events, upload reports, and see their scores.
• Faculty can grade only assigned reports and return scores with feedback.

• Automated email notifications inform users of registrations, events, and results.

• User, event, and report information is stored securely by MongoDB with scalability to grow in the future.

•Automated procedures improved efficiency, transparency, and accountability compared to manual procedures.

Advantages of EMS

The system streamlines departmental event planning with efficiency and speed by minimizing the humongous manual work otherwise encountered by students and faculty. As each user has respective role-based dashboards, the tasks are defined and easy to manage without any sort of confusion. All event information, reports, and scores are housed in an impenetrable single database so nothing is lost or replicated. Email notifications make everybody real time aware, and reminders are not necessary.

Fig. 1. Overall System Flowchart

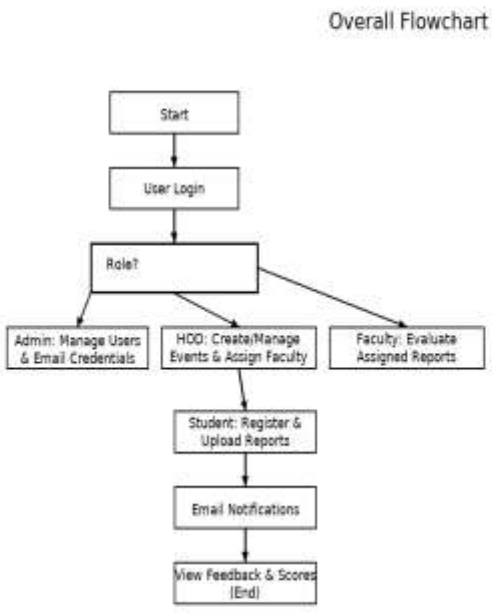

Overall Flowchart
Start
User Login
Role?
Admin: Manage Users & Email Credentials
HOD: Create/Manage Events & Assign Faculty
Faculty: Evaluate Assigned Reports
Student: Register & Upload Reports
Email Notifications
View Feedback & Scores (End)

Fig:3 (use case Diagram)

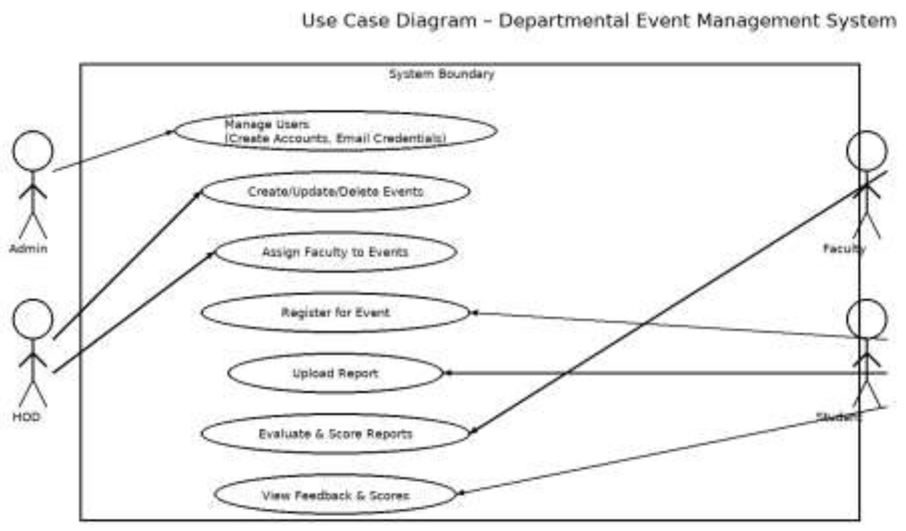
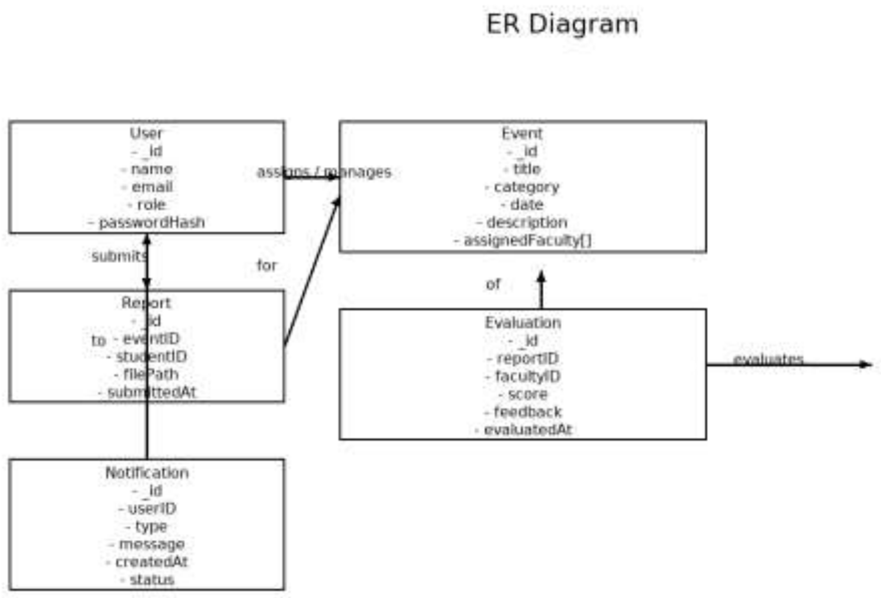


Fig:4(E-R diagram)



Conclusions:

The Departmental Event Management System has significantly automated and streamlined the management of academic events. Where previously manual, time-consuming procedures like paper registrations or posting on notice boards used to be the norm, all this is now computerized and in one place. Each user has a customized dashboard—students register and file reports easily, teachers handle only the reports that they have been allocated, HODs handle events without difficulty, and the admin can handle registrations as well as system maintenance. With real authentication and secure verification, central data repository and instant email alerting, the system not just saves time, but also enables transparency and accountability. In total, it provides a more smooth exchange of communication and coordination to the plate, making event planning easier and less anxiety-provoking for everyone.

Acknowledgement

The authors wish to thank everyone who has contributed to the success of this research work.

References:

[1] MERN Stack Best Practices, Various sources.

[2] JSON Web Tokens (JWT) – IETF RFC 7519.

[3] MongoDB Documentation – Data Modeling Patterns.

[4] Email Notification Patterns for Web Apps.

[5] Example Layout Inspiration: IOSR-JCE article format (2020).