

Development of a Role-Oriented Educational Management System with Web and Parent Portal Integration

Dinesh Kumar, Ishita Kohli², Kriti Sharma³

¹ Assistant Professor at Bhagwan Parshuram Institute of Technology, Rohini, New Delhi, India

^{2,3} Student at Bhagwan Parshuram Institute of Technology, Rohini, New Delhi, India

Email: dineshkumar@bpitindia.edu.in¹, ishitakohli0206@gmail.com², kritisharma0104@gmail.com³

ABSTRACT

Learning institutions are also more and more requiring to be integrated as a single, coherent platform through the latest and most effective digital solutions. This paper involves the design, architecture and implementation of a School Enterprise Resource Planning (ERP) Management System that includes a public-facing school web site and a parent portal. The proposed system utilizes a three-level architecture that includes a presentation layer based on the React.js and a Node.js/Express.js-based application layer and a MySQL-based relational database layer. Role-based access control (RBAC) is applied through the use of JSON Web Tokens (JWT), which ensures that administrators, teachers, students and parents use and interact with their respective authorized modules. They consist of core functional modules such as student lifecycle management, attendance tracking, assignment and quiz management, examination and result processing, fee management including payment history, dissemination of notices in real-time, and an admission enquiry workflow with integration into the public portal. A comparative analysis with five related systems based on the literature on the subject has shown that the proposed ERP is more functional in terms of completeness, scalability and user-centric design than the existing solutions. The system has an estimated reduction in administration overhead, improved parental interaction, and modular architecture that can easily be extended to mobile platforms, advanced analytics and automated timetabling in future versions.

Keywords- School ERP; Education Management System; Role-Based Access Control; react.js; node.js; Parent Portal; JWT Authentication; Full-Stack Web Application.

I. INTRODUCTION

The management of institutions of education is a complex interaction of academic, financial, and communicative processes, which are inefficient, redundant and lack transparency, in their traditional manual form. The increase in enrolment numbers and the demands of compliance regulations are placing an ever-increasing strain on schools to implement digital infrastructure that can bring these disparate functions together into integrated, auditable and accessible platforms [1].

ERP systems, which have long been established in the corporate world, have become very popular in the education sector during the last decade [2]. ERP solutions that focus on schools aim to automate their administrative processes, provide real-time visibility of data to its stakeholders, and enhance the collaborative communication between its stakeholders and its administrative hierarchy. Nevertheless, numerous commercially-available systems are not only cost-prohibitive to smaller institutions, but are also either too generic in their design of academic modules, or are lacking in the ability to engage parents [3].

These gaps are discussed in this paper by designing and implementing a fully-developed, full-stack School ERP Management System. The proposed platform incorporates five main classes of users: Administrator, Teacher, Student, Parent and anonymous public user, each having a unique role-based access control policy. The institutional outreach point is provided by a school web site that provides functionality as an enquiry point regarding admission to the school. It is an architecture based on modern open-source technologies, i.e., React.js, Node.js, Express.js and MySQL, allowing to implement it at a relatively low cost and maintain it as the community.

A. Problem Statement

The current school management systems are either focused on single administrative issues (e.g., attendance-only modules, fee-only modules), or reflect monolithic commercial systems, which are hard to customise and integrate. Specifically, the parent-institution contact channel communication system is underused, most of the systems have a read-only report portal, but not an interactive, real-time dashboard. Moreover, the lack of the smoothly interlinked public web site adds more rub against the dub of the prospective-student admission processes.

B. Objectives

To develop a single (unified) and modular ERP platform that supports all facets of academic and administrative life of a school.

The purpose of the security measures is to ensure the data safety and operational isolation between user groups.

- To incorporate parent portal where real time information about academic performance, attendance, fees, and institutional notices will be available.
- To develop a public-school Web site that receives inquiries about admission and directs these inquiries into the administrative process.
- To design the system to be scaled and extended to incorporate the mobile applications, analytics dashboards and SMS notification services.

C. Contributions

The key contributions to this work are: (i) a complete requirements analysis and system architecture of a school ERP with 7 functional domains; (ii) a full-stack application with about 50 percent functional coverage with all 7 functional modules operationalized; and (iii) an extensible 3-tier architecture that is documented to be replicated academically and deployed in institutions.

II. LITERATURE REVIEW

The literature relating to school management systems includes web-based student information system, cloud deployed ERP systems, and parent engagement portals. Five preceding contributions have been selected and discussed below and a comparative analysis has been discussed in Table I.

A. Web-Based School Management System (PHP/MySQL)

An initial background contribution by Rajput et al. [4] exhibited the feasibility of PHP and MySQL to digitise student records, attendance records and management of results with simple role-based access by the administrator and teachers. The reported study had significant paperwork reductions and enhanced staff communication, but the system lacked parent portal, real time notification and horizontal scalability and was therefore only applicable to large institutions.

B. Cloud-Based ERP for Educational Institutions

Bhosale and Patil [5] came up with a cloud-based ERP which centralised the data storage in the departments and had admission, fee management and academic tracking modules. The cloud infrastructure enhanced remote access and minimized capital expenditure on servers that were on-premise. Inherent to a shared cloud environment were data-security concerns that are critical limitations and dependency on continuous internet connectivity as well as limited customisation is a limitation that curtails institution adoption.

C. Student Information System with Web Interface

Kumar et al. [6] introduced a structured relational database approach to the management of student records, grades and attendance. The system was more concerned with data integrity, with the normalised schema design, and the data accuracy was measured and the number of duplicate records decreased. Its areas of operation however, were limited to the management of students data, but not financial modules, administrative processes, and parent involvement.

D. Java-Based ERP for School Automation

Mishra and Tripathi [7] developed an ERP, which includes HR, finance, and academic modules, in a Java based architecture. Automation also minimized errors in administrative duties that were caused by manual errors. The system was also criticised due to its complicated user interface that required a lot of staff training, and their front end that was not as responsive as expected in modern web applications.

E. Online School Portal with Parent Interaction

Singh et al. [8] have designed a web portal that is specifically designed to provide the parent with access to the attendance record, academic performance and institutional messages. The researchers established the fact that the greater the visibility of parents, the better the performance of the students. However, the portal was not connected with back-end modules in administration and facing the teachers, which made the portal a separate tool but not to be an integrated institutional tool.

F. Comparative Analysis

Table I summarises the feature coverage of the five reviewed systems against the proposed School ERP, illustrating the extent to which existing solutions individually fail to address the full institutional requirement spectrum.

Feature / System	Rajput [4]	Bhosale [5]	Kumar [6]	Mishra [7]	Singh [8]	Proposed
Student Management	Yes	Yes	Yes	Yes	No	Yes
Attendance Tracking	Yes	Partial	Yes	Yes	Yes	Yes
Fee Management	No	Yes	No	Yes	No	Yes
Parent Portal	No	No	No	No	Yes	Yes
Assignment Management	No	No	No	No	No	Yes
Quiz Integration	No	No	No	No	No	Yes
Public Website	No	No	No	No	No	Yes
Report Card Generation	No	No	No	Yes	No	Yes
Real-Time Notifications	No	No	No	No	No	Yes
Modern Tech Stack	No	Partial	No	No	Partial	Yes

TABLE I. Comparative feature matrix of existing school management systems versus the proposed ERP.

III. METHODOLOGY AND SYSTEM DESIGN

A. System Architecture

The ERP suggested is a three-layer client server model (Fig. 1). Presentation Layer is implemented with the help of React.js which consumes the RESTful API endpoints which are provided by the Application Layer (Node.js with Express.js framework). The Data Layer will consist of a MySQL relational database that will be accessed directly by the backend, and has no direct database access on the client-side.

Such a separation of architecture ensures a clean line between business logic and data persistence, independent scaling of frontend and backend components, and can be better scaled to cloud-hosted database services or microservice decomposition.

B. Role-Based Access Control (RBAC) Model

A stateless JWT-based authentication system is used to implement security. On successful credential authentication, the server sends signed JWT containing that of the user in terms of the identifier and role of the user. The RBAC policy can be said to be formalized as follows:

$$Permission(u, r) = \{op \mid op \in Ops(role(u)) \wedge op \in Ops(r)\}$$

where u is a user, r is a resource, $Ops(\cdot)$ gives the set of the allowed operations, and $role(u)$ is a mapping of a user to the role assigned to him. It is defined to have four primary roles, namely: Administrator (full system access), Teacher (class and academic operations), Student (read-only academic and attendance access) and Parent (read-only child-linked academic and financial access).

C. Database Design

The relational database schema has twelve core entity tables, namely, Users, Admin, Students, Teachers, Parents, Classes, Attendance, Assignments, Notices, Exams, Results, and Fees. Referential integrity is preserved in all relationships by having foreign key constraints and cascade deletion rules. The schema is refined to the Third Normal Form (3NF) so as to eliminate transitive dependencies and reduce the number of update anomalies. Key relationships include:

Classes (many-to-one) is to be enrolled in by Students (many-to-one). Classes (many-to-one) is to belong to Teachers (many-to-one).

- Both Students and Classes (composite foreign key) are connected to the attendance records.

Assessments are connected to a particular Class, Subject and Teacher.

- Results can be aggregated by the per-subject mark, i.e. can contain both Exams and Students.
- Fees are associated with individual Students and a history of payment is kept as a child table.

Parents are associated to Students through ParentStudent mapping table, which is many to many.

D. Software and Hardware Requirements

Table II specifies the hardware configuration recommended for system deployment, and Table III enumerates the software dependencies.

Component	Minimum Specification	Recommended Specification
Processor	Intel Core i3	Intel Core i5/i7 (server-side)
RAM	4 GB	8 GB or above
Storage	20 GB HDD	SSD with 50 GB free space
Network Bandwidth	10 Mbps	50 Mbps or above

TABLE II. Hardware requirements for system deployment.

Category	Technology / Tool	Version / Notes
Operating System	Windows 10/11, Ubuntu Linux, macOS	Cross-platform compatible
Frontend Framework	React.js	v18+
Backend Runtime	Node.js with Express.js	Node v18 LTS

Category	Technology / Tool	Version / Notes
Database	MySQL	v8.0
Authentication	JSON Web Tokens (JWT)	HS256 algorithm
API Style	RESTful HTTP APIs	JSON payload
Version Control	Git / GitHub	Standard workflow
Environment Config	dotenv (.env files)	Secret management

TABLE III. *Software requirements and technology stack.*

E. Algorithm: User Authentication and Dashboard Routing

Algorithm 1 describes the authentication and role-based dashboard routing workflow executed on every login request.

Algorithm 1: Role-Based Authentication and Dashboard Routing

Input: credentials {username, password}

Output: JWT token, role-specific dashboard payload

1. Receive HTTP POST /api/login with credentials
2. Query Users table: SELECT * WHERE username = ?
3. IF user NOT FOUND → return HTTP 401 (Unauthorised)
4. Verify bcrypt hash: compare(password, stored_hash)
5. IF hash mismatch → return HTTP 401
6. Extract role ← user.role
7. Generate JWT: sign({userId, role}, SECRET, {expiresIn: '8h'})
8. SWITCH role:
 - CASE 'admin' → load AdminDashboard module
 - CASE 'teacher' → load TeacherDashboard module
 - CASE 'student' → load StudentDashboard module
 - CASE 'parent' → load ParentDashboard module
9. Fetch role-specific summary data from DB
10. Return {token, dashboardPayload} with HTTP 200

IV. IMPLEMENTATION

A. Module Implementation Overview

The system is structured to have five major modules namely Administrator, Teacher, Student, Parent and the public School Website; each having a dedicated dashboard displayed after authentication. The frontend is a component level state manager using React hooks, whose all data mutations are sent to the server using RESTful API calls with JWT authorisation headers.

B. Administrator Module

The Administrator module is the most privileged interface, and presents the widest operational interface. It has features such as student lifecycle management (admission, class promotion, archiving of records), teacher and parent account provisioning, class-section-subject configuration, school-wide attendance monitoring, examination scheduling, result and report card management, fee structure definition and payment tracking, payroll management of staff and publishing of notices system wide. The key performance indicators, including the total number of students, the number of

outstanding fees, the percentage of students attending classes daily, and the recent system activity are aggregated in an administrative dashboard that displays the key performance indicators in one overview pane.

C. Teacher Module

Teachers have access to a class centric dashboard that shows the summary statistics in the sections to which the teacher is assigned. Core features include marking attendance with present/absent/late options, creation of assignments with document upload support, quiz links (Google Forms or Microsoft Forms integration) and marks entry to configured examinations, student performance monitoring, and parent notices. Assignments uploaded, the entry of exam marks, and viewing of results are all unified in a single interface by the Faculty Academic Module.

D. Student Module

Student module is an academic portal that is customised. The students will be able to see their profile in the classroom, access assignment files, quiz links as provided by the teachers, published examination results and downloadable report cards, monitor their attendance record, and receive institutional notices and announcements. Student Academic Portal is a Web site that consolidates assignments, results, history of payment of fees, examinations and notices onto one Web site, reducing navigation overhead.

E. Parent Module

The Parent module fills a gap in the school ERP literature [8] by offering real time and child connected academic monitoring. Parents are able to see the attendance of their child, examination results, assignment details, review fee status and payment history as well as receive push notices. A Parent Academic Overview screen displays children in a single screen with details of their classes and sections, linked entries on results and assignments, due dates and exam dates, and records of fees paid. Live attendance notification alerts on anomaly events in attendance and instances of class suspension are shown on the dashboard.

F. School Website (Public Interface)

The public facing portal is the institutional front-end of prospective students and the general public. It includes a home page that has a vision statement, mission statement, an institutional overview page (about, infrastructure, courses, faculty), an events and gallery page, and a contact interface. Admission Enquiry module: Parent and student information (name, email, phone, class of interest, message) are accepted via a validated form, and submitted to the administrative workflow queue that can be accessed by the Administrator to make follow-up. This ERP login gateway is integrated into the site where all the authenticated users can have one common gateway to enter the site.

G. REST API Design

Table IV gives a sample subset of the REST API resources that are implemented in the system and show how the resources are resource-oriented and how the resources are authenticated.

Endpoint	Method	Auth Required	Description
/api/login	POST	No	Authenticate user, return JWT
/api/admin/students	GET / POST	Admin	List or add students
/api/teacher/attendance	POST	Teacher	Submit attendance records
/api/teacher/assignments	POST	Teacher	Upload assignment with file
/api/student/results	GET	Student	Retrieve personal exam results
/api/student/assignments	GET	Student	View assigned homework
/api/parent/child-summary	GET	Parent	Fetch linked child academic data
/api/admin/fees	GET / PUT	Admin	Manage fee records
/api/notices	GET / POST	All / Admin	Read or publish notices
/api/public/enquiry	POST	No	Submit admission enquiry

TABLE IV. Representative REST API endpoints.

V. RESULTS AND ANALYSIS

A. Functional Completeness

At the reporting stage, about half of the total planned functionality is implemented, including all core modules of each user role, the entire public web site with an admissions enquiry and the authentication and RBAC infrastructure. The features set implemented is adequate to show end-to-end workflows to the key use cases.

Table V shows a module level, implementation status test, where each component is rated against the intended functionality.

Module	Planned Features	Implemented	Completion (%)
Administrator Module	15	12	80%
Teacher Module	8	7	88%
Student Module	7	7	100%
Parent Module	6	6	100%
School Website	9	9	100%
Reporting & Analytics	6	2	33%
Notification Engine	4	1	25%
Overall System	55	44	~80% of planned

TABLE V. Module-level implementation completion status.

B. Feature Comparison with Related Systems

The proposed system attains feature parity or even better than all of the five reviewed systems in all of the dimensions compared (Table I). It is also the only system in the comparison set that is also providing a parent portal, integration of quizzes, a public school web site with enquiry to admission and a modern technology stack based on JavaScript.

C. Performance Benchmarks (Estimated)

Although the full load-testing data will have to be gathered in further work, the initial observations on a local development environment (Intel Core i5, 8 GB RAM, SSD) suggest that the following approximate performance characteristics can be observed:

Operation	Avg. Response Time (ms)	Remarks
User Login (JWT generation)	~120 ms	Including bcrypt verification
Dashboard Load (Admin)	~280 ms	Aggregate DB queries
Attendance Submission (30 students)	~190 ms	Bulk INSERT
Report Card Generation	~350 ms	PDF rendering latency included
Admission Enquiry Submission	~90 ms	Single row INSERT
Notice Publication	~100 ms	Including role-audience filtering

TABLE VI. Estimated operation-level response time benchmarks (local development environment).

D. Qualitative Evaluation

One of the most important distinguishing features of the proposed system as compared to previous work is the parent engagement component of the proposed system. The research studies on the impact of parental involvement on the academic achievement of students have always indicated a positive relationship between parental involvement and student academic performance [9]. The system achieves a constant feedback process between home and school that cannot be sustained by the traditional system and paper-based communication channels. The real-time notification panel, which notifies parents of the attendance anomaly and school closure events, further increases the communicative value of the portal.

VI. DISCUSSION

A. Significance of the Parent Portal

One of the most important distinguishing features of the proposed system as compared to previous work is the parent engagement component of the proposed system. The research studies on the impact of parental involvement on the academic achievement of students have always indicated a positive relationship between parental involvement and student academic performance [9]. The system achieves a constant feedback process between home and school that cannot be sustained by the traditional system and paper-based communication channels. The real-time notification panel, which notifies parents of the attendance anomaly and school closure events, further increases the communicative value of the portal.

B. Architectural Strengths

The three-tier, RESTful architecture confers several practical advantages. The stateless nature of JWT authentication enables horizontal scaling of backend instances without shared session state, an important property for institutions anticipating growth. The separation of the React.js frontend from the backend API permits independent deployment to content delivery networks and application servers respectively, optimising latency for end-users. The use of

environment-variable-driven configuration (dotenv) facilitates seamless transition between development, staging, and production environments.

C. Limitations and Constraints

There are a number of weaknesses of the current implementation that should be noted. To begin with, the system has only been tested in a local development environment, formal load testing with multiple users has not been done yet. Second, the quiz management module does not have an underlying quiz engine but instead uses an external link (Google or Microsoft Forms) to capture automated results. Third, the delivery of notifications is not yet integrated with the email, SMS, or messaging platforms (WhatsApp); this is an item of a future-work. Fourth, the reporting and analytics module is only 33% complete, which restricts the amount of data-driven insights that administrators can currently get.

D. Security Considerations

In addition to stateless authentication based on JWT, the system requires server-side permission validation on each API request, preventing privilege escalation due to client-side manipulation. Parameters are used to parameterise database queries, to reduce the likelihood of SQL injection attacks. The middleware layer limits file uploads in terms of type and size. Future hardening efforts should include restricting rates on authentication endpoints, logging of all data changes, and implementation of HTTPS in the production environment.

VII. CONCLUSION AND FUTURE WORK

A. Conclusion

This paper has discussed the design, architecture and partial implementation of an extensive School ERP Management System that incorporates a parent portal and public school web site. The system focuses on well documented flaws in current school management solutions by offering a single, integrated application that benefits administrators, teachers, students, parents and the general populace. The proposed ERP has better functional coverage in all the dimensions considered as the proposed ERP is superior in terms of functional coverage in all the dimensions evaluated. The three level, JWT secured architecture offers solid platform to deploy production and extend the capacity in the future.

B. Future Work

The subsequent extensions will be subject to further development phases:

- Remaining portion of the functional scope planned to be completed is done. Preference is given to the reporting/analytics and notification engine modules.

Native Android and iOS mobile applications to serve in the role of students and parents which will consume the existing RESTful API layer.

- A sophisticated analytics dashboard that offers predictive data about the trends in student performance and attendance risk factors, as well as probabilities of defaulting on fees, based on machine learning.
- Multi-channel integration of notifications (SMS (Twilio), email (SendGrid), and WhatsApp (Business API) to receive real-time notifications.
- Timetable generation Automated generation of timetables based on constraint satisfaction algorithms, using teacher availability, room capacity and curriculum sequencing requirements.
- Hardening techniques such as rate limiting, distributed audit logging, two-factor authentication, and validation of penetration testing.

- Elastic scalability and high availability Cloud-native deployment on containerised infrastructure (Docker/Kubernetes) to achieve both.

REFERENCES.

- [1] M. Al-Husain, A. Hafeez-Baig, and R. Gururajan, "An exploratory study to investigate the use of cloud computing in higher education," in Proc. 25th Australasian Conf. Information Systems (ACIS), Auckland, 2014, pp. 1–11.
- [2] C. Peng, M. Kou, and G. Sun, "ERP systems in higher education: A case study of Chinese universities," Int. J. Enterprise Information Systems, vol. 9, no. 2, pp. 22–35, Apr. 2013.
- [3] J. L. Pollock and B. Wallace, "Challenges of ERP adoption in K-12 educational environments," J. Information Technology Education, vol. 11, pp. 35–51, 2012.
- [4] S. Rajput, A. Verma, and P. Saxena, "School management system using web technologies," Int. J. Engineering Research and Technology (IJERT), vol. 4, no. 3, pp. 112–117, Mar. 2015. [Online]. Available: <https://www.ijert.org/>
- [5] D. Bhosale and P. Patil, "Cloud-based ERP system for educational institutions," in Proc. 2016 IEEE Int. Conf. Advances in Electronics, Communication and Computer Technology (ICAECCT), Pune, India, 2016, pp. 1–5. doi: 10.1109/ICAECCT.2016.7942556.
- [6] A. Kumar, R. Singh, and S. Verma, "Student information system with web interface for managing academic records," Int. J. Computer Applications, vol. 140, no. 12, pp. 24–29, Apr. 2016. [Online]. Available: <https://www.researchgate.net/>
- [7] N. Mishra and R. Tripathi, "ERP system for school automation using Java-based architecture," Int. J. Computer Applications (IJCA), vol. 132, no. 15, pp. 30–36, Dec. 2015. [Online]. Available: <https://www.ijcaonline.org/>
- [8] R. Singh, A. Sharma, and K. Gupta, "Online school portal with parent interaction and student performance monitoring," Int. J. Research and Analytical Reviews (IJRASET), vol. 6, no. 1, pp. 512–518, Jan. 2019. [Online]. Available: <https://www.ijraset.com/>
- [9] J. L. Epstein, School, Family, and Community Partnerships: Preparing Educators and Improving Schools, 2nd ed. Boulder, CO: Westview Press, 2011.
- [10] R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, 8th ed. New York, NY: McGraw-Hill Education, 2014.
- [11] A. Silberschatz, H. F. Korth, and S. Sudarshan, Database System Concepts, 7th ed. New York, NY: McGraw-Hill, 2019.
- [12] M. Fowler, Patterns of Enterprise Application Architecture. Boston, MA: Addison-Wesley, 2002.
- [13] F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST standard for role-based access control," ACM Trans. Information and Systems Security, vol. 4, no. 3, pp. 224–274, Aug. 2001.
- [14] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Boston, MA: Addison-Wesley, 1994.
- [15] Node.js Foundation, "Node.js v18 LTS Documentation," 2023. [Online]. Available: <https://nodejs.org/en/docs/>