Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

BMS College ERP: A Secure, Attendance-Gated Academic and Workforce Management Platform with Database Failover

Pavan Kumar S
Dept. of Computer Applications
BMS College of Commerce and Management
Bengaluru, India

Aishwarya K M
Dept. of Computer Applications
BMS College of Commerce and Management
Bengaluru, India

Abstract—Higher-education institutions typically run academics, attendance, assessment and human-resources processes on disconnected tools, which causes duplicated data, weak accountability and inconsistent policy enforcement. This paper presents the BMS College ERP, a modular web platform developed for BMS College of Commerce and Management, Bengaluru. The system integrates timetable-driven class sessions, subject-wise student attendance, internal-assessment (IA) marks, assignments, a study-materials library, geofenced staff attendance, payroll, multi-channel notifications and a teacher analytics dashboard and a restricted AI assistant behind a single role-based access control (RBAC) model. Its central design contribution is an attendance-gated access engine: students may view marks, submit assignments and download study materials for a subject only when their subject-wise attendance is at least 75%, unless the Head of Department (HOD) records an audited override, and restricted responses are stripped of all protected content. The platform is implemented in Flask with a dual-engine database layer that fails over from cloud PostgreSQL to local SQLite. It exposes 117 endpoints over 16 data models, and an automated suite of 53 tests passes with no server crashes.

***Index Terms*—enterprise resource planning, educational technology, role-based access control, attendance eligibility, audit trail, Flask, database failover.**

I. INTRODUCTION

Colleges depend on several interacting workflows: preparing timetables, recording class attendance, awarding internal marks, distributing coursework and learning resources, and managing staff presence and payroll. When these run in separate spreadsheets or standalone tools, institutions struggle to enforce academic policies consistently and to answer who changed what, and when.

At BMS College of Commerce and Management (BMSCCM), Basavanagudi campus, Bengaluru, this problem is sharpened by an academic rule that students need at least 75% attendance in a subject to be eligible for its internal assessment benefits. Enforcing the rule manually is slow and open to dispute, so the BMS College ERP was built to apply it automatically and to keep an auditable record of every exception.

The contributions of this paper are: (i) an integrated ERP architecture covering academic, student self-service and staff-operations functions; (ii) a centralized attendance-eligibility engine with fail-closed semantics and HOD overrides; (iii) a layered security design for private file storage and zero data leakage; and (iv) a dual-engine database abstraction that keeps the system available when the cloud database is unreachable.

The rest of the paper covers background (Section II), architecture (III), module design (IV), testing (V), and conclusions (VI).

II. BACKGROUND AND RELATED WORK

Campus management systems commonly combine student information, learning management and HR functions. Open-source and commercial platforms generally treat attendance and academic access as separate concerns, so eligibility rules are applied by staff outside the software. Web frameworks such as Flask [1] make it practical to build lightweight institution-specific systems that embed such policies directly.

Access control in the proposed system follows the role-based model [2], in which permissions attach to roles rather than individuals. Web-application hardening follows the risk categories documented by OWASP [3], notably broken access control and insecure file upload handling. For location checks the platform uses the haversine formula [4], and for the rotating kiosk code it uses keyed-hash message authentication (HMAC) [5].

III. SYSTEM ARCHITECTURE

The platform is a layered web application, shown in Fig. 1. Faculty, HOD, administrative and student clients reach a Flask 3.1.3 application layer over HTTPS and WebSockets. Staff use cookie-based login sessions, while students authenticate with their university (UUCMS) identifier in a separate session, giving two parallel RBAC guards.

Below the application layer, three engines share common services. The academic engine handles timetables, class sessions, attendance, IA marks and teacher analytics. The

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

content engine manages assignments and study materials. The staff-operations engine covers geofenced attendance, payroll, OCR onboarding and the AI assistant. Table I lists the technology stack.

TABLE I
TECHNOLOGY STACK

Layer	Technology
Language	Python 3.10+
Web framework	Flask 3.1.3, Flask-SocketIO
Data access	Flask-SQLAlchemy 3.1.1
Database	PostgreSQL (Supabase), SQLite fallback
Security	Werkzeug PBKDF2-SHA256, Flask-Login
AI / OCR	Google Gemini router, Tesseract
Reports / mail	FPDF2 payslips, Flask-Mail

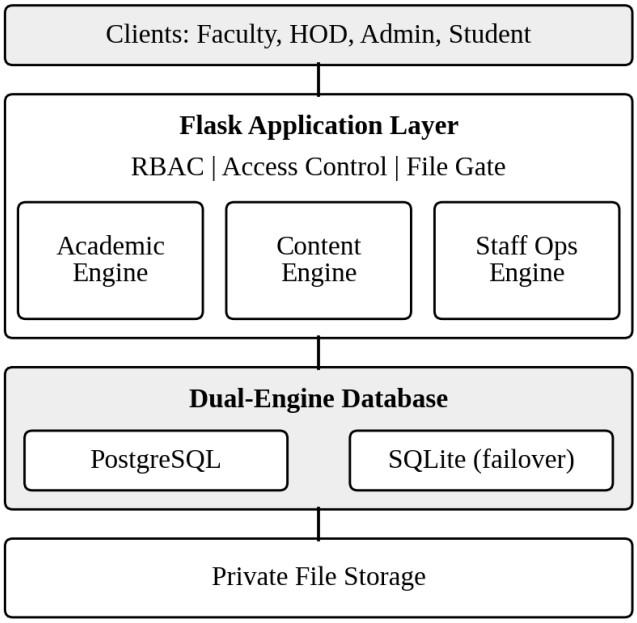


Fig. 1. Layered architecture of the BMS College ERP.

A. Data Model

The relational schema contains 16 models. Core entities are User (staff), Student, Timetable, ClassSession, StudentAttendance, InternalMark, Assignment, AssignmentSubmission, StudyMaterial and the staff Attendance record. Six audit and policy models (attendance, mark, eligibility and academic-resource audits, plus eligibility overrides and assessment cut-offs) record every sensitive change. Unique composite constraints prevent duplicates, for example one mark row per student, subject, academic year, semester and assessment cycle.

B. Dual-Engine Database

The system connects to a cloud PostgreSQL instance by default. If the connection fails or times out, it falls back to a local SQLite database, so the application keeps running without crashes. Self-healing schema checks and timetable synchronization run automatically at start-up.

C. Role-Based Access Control

Six roles are supported: Student, Faculty, HOD, Principal, Accountant and Admin. Faculty see their own classes, HODs their department, and the Principal all departments. The Admin role is deliberately excluded from the college timetable overview with an HTTP 403 response, and no role can query salary or personal data through the AI assistant.

IV. MODULE DESIGN

A. Timetable and Class Sessions

The authoritative semester timetable is loaded and updated in place so that foreign-key links from historical class sessions are preserved. A class session follows a three-step lifecycle: initialize (IN_PROGRESS), mark attendance as Present, Absent, Late or Excused, and finalize (COMPLETED). An optimistic-concurrency version field prevents simultaneous edits from overwriting one another, and any post-finalization correction requires a reason and creates an immutable audit entry.

B. Internal Assessment Marks

Teachers may enter marks only for classes allocated to them in the timetable. Marks must lie between zero and the configured maximum (20 or 25), are limited to two decimal places, and a missing mark is stored as null so it is never confused with a genuine zero. Every edit records the before and after values, the user and a mandatory reason.

C. Attendance-Gated Access Engine

A single service decides whether a student may see marks, open assignments or download study materials for a subject. Attendance is computed from the subject actually taught, so a substitute lecture counts toward the correct subject, and cancelled or unheld classes are excluded. The decision order is:

1) If an active HOD MANUAL_RESTRICT exists, access is restricted.
2) Otherwise, if an active HOD ALLOW_OVERRIDE exists, access is granted.
3) If no completed class has been held before the cut-off, the state is PENDING and access is denied (fail-closed).
4) Otherwise, attendance of 75.0% or more grants access; lower values restrict it.

HODs manage overrides through a dedicated portal, each action requiring a justification stored in the eligibility audit log. Restricted students receive an explanatory message but no marks, descriptions, file links or previews.

D. Assignments and Study Materials

Faculty create assignments with a title, scope, deadline, maximum marks and an optional attachment, and the system rejects a deadline earlier than the assigned date. Students submit files, and a submission after the deadline is automatically marked Late. Resubmission increments a version number and securely deletes the earlier file. The study-

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 11, CONFLUENCE - 2026

materials library has three categories (textbooks, previous-year papers and question banks), and each file stores a SHA-256 checksum for integrity.

E. File and Application Security

Uploads are stored in a private directory outside the web root under random UUID names, with the original filename kept only as sanitized metadata. Each file's leading bytes are checked against its declared type (for example PDF, JPEG, PNG), executable and script extensions are rejected, and uploads are size-limited (15 MB by default). Students cannot download one another's submissions.

F. Smart Staff Attendance

Staff check-in combines three controls. A GPS geofence of 1000 m around the campus coordinates is evaluated with the haversine formula. A kiosk displays a QR token that is signed with HMAC and refreshed every 30 seconds, so a screenshot cannot be reused for remote check-in. A browser fingerprint is bound to each user, and an email alert is sent when an unrecognized device logs in. Shifts are computed over 8 hours with a 10-minute grace period, from which late minutes and overtime are derived.

G. Notifications, AI Assistant and OCR

A central notification function dispatches email, with SMS and WhatsApp stubs, for events such as late attendance, leave decisions, new devices, payslips, timetable changes, expense claims and assignment publication or deadline changes. The Gemini-based assistant [6] acts only as a router to whitelisted Python functions and never generates raw SQL. Its three tiers cover schedules and presence (all staff), leave and overtime summaries (managers) and salary or personal data (forbidden for every role). The digital vault uses Tesseract OCR [7] to extract name, email, phone, department and joining date from staff ID cards and certificates. Payroll computes gross pay, deductions and leave penalties and generates PDF payslips.

H. Teacher Academic Analytics

A dashboard gives each teacher semester-wide and subject-wise attendance for their classes. Overall class attendance is the number of Present instances divided by the number of eligible student-session instances, whereas an individual student's attendance is attended sessions divided by conducted sessions. IA analytics for IA1, IA2 and the final IA report evaluated and pending counts, class average, highest and lowest marks. A missing mark is excluded from the average and never treated as zero. Scores fall into four bands: below 40%, 40–60%, 60–75% and 75–100%.

A searchable roster can be filtered by name, UUCMS ID and risk category. It flags attendance risk (below 75%), low internal marks (below 40% of the maximum), pending marks, and HOD restriction or override. Flags use text badges as well as colour. A drill-down page shows per-subject metrics and up to 50 recent sessions, and shows an edit-marks control only for subjects the teacher is authorized to teach.

V. TESTING AND RESULTS

Quality was assessed with automated unittest suites, summarized in Table II. In total, 53 tests passed, covering authorization, analytics formulas, the 75% boundary, HOD overrides, concurrency locking, audit immutability, cross-student file isolation and zero data leakage.

TABLE II
AUTOMATED TEST SUITES

Suite	Tests	Coverage
Academic analytics	8	Formulas, IA stats, badges, 403
Assignments and materials	10	Access control, lifecycle, isolation, checksums
Marks eligibility	10	75% rule, HOD overrides, audit
Internal marks	10	RBAC, ranges, locking, audit
Authoritative timetable	5	20 slots, FK preservation, 403
Academic models	8	Cascades, constraints
All features	2	117 endpoints, no crashes

A smoke test requested all 117 registered endpoints and observed no server crashes. The 53 tests together passed at a 100% rate. These results show functional correctness of the implemented policies; they are not a measure of performance or usability, which were not benchmarked in this work.

VI. CONCLUSION AND FUTURE WORK

The BMS College ERP shows that academic policy, security and administration can be unified in one maintainable platform. Its attendance-gated access engine enforces the 75% rule consistently, supports accountable HOD exceptions and leaks no protected content, while the database failover keeps the service available. Limitations include the stubbed SMS and WhatsApp channels and the absence of load testing and a user study. Future work includes real messaging gateways, load and penetration testing, a usability evaluation, and early-warning analytics for at-risk students.

ACKNOWLEDGMENT

The authors thank the faculty and management of BMS College of Commerce and Management for their guidance and for access to the institutional timetable and requirements.

REFERENCES

[1] M. Grinberg, *Flask Web Development*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2018.
[2] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38–47, Feb. 1996.
[3] OWASP Foundation, "OWASP Top 10:2021," 2021. [Online]. Available: https://owasp.org/Top10/
[4] R. W. Sinnott, "Virtues of the haversine," *Sky Telesc.*, vol. 68, no. 2, p. 159, 1984.
[5] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-hashing for message authentication," IETF, RFC 2104, Feb. 1997.
[6] Gemini Team, Google, "Gemini: A family of highly capable multimodal models," 2023, *arXiv:2312.11805*.
[7] R. Smith, "An overview of the Tesseract OCR engine," in *Proc. 9th Int. Conf. Document Anal. Recognit. (ICDAR)*, Curitiba, Brazil, 2007, pp. 629–633.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)