International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 03 | Mar 2026 www.irjet.net p-ISSN: 2395-0072

An ERP-Based Integrated Student Management System with OTP-Verified Attendance and Lifecycle-Centric Data Architecture

Dharmik Kapadiya[1], Ritesh Pokharkar[2], Ankit Pathak[3] , Dr. Swati Patel[4]

[1-3] *Student, Ajeenkya DY Patil University, Pune, Maharashtra, India,*
[4]*Professor, Dept. of Computer Science, Ajeenkya DY Patil University, Pune, Maharashtra, India,*

Abstract - *Student data management in most universities is broken in a very specific way not because institutions lack software, but because they have too much of it, all running separately and never sharing information. Admissions, hostel, examination, and fee departments each maintain isolated records, and nobody has a complete view of any student at any point. We built the Integrated Student Management System (ISMS) to solve this from the ground up, connecting ten administrative modules through one auto-generated Student ID that links every record from enrollment to graduation. For attendance, we replaced paper registers and expensive biometric hardware with a session-based one-time password mechanism that validates in under one second. Testing across 200 student records showed login averaging 1.20s (SD=0.18), OTP verification 0.80s (SD=0.11), and database queries 0.60s (SD=0.09). Twelve university users rated the interface 78.3/100 on the System Usability Scale, classified as Good. Against both manual workflows and Fedena ERP, our system uniquely combines unified student identity, hardware-free attendance, full lifecycle coverage, and formally measured usability built entirely on Python, Flask, and MySQL.*

Keywords: **Academic Administration, Enterprise Resource Planning, Higher Education, Integrated Student Management System, OTP Attendance Verification, Student Lifecycle Integration.**

1. INTRODUCTION

The environment of administrative practice in higher education institutions has, over the last two decades, become significantly more complex. Rising student numbers, tighter reporting demands from regulators, and the need to deliver high-quality services in real time have never before combined to create such unprecedented pressures on the information systems of higher education institutions. However, the reality in many higher education institutions remains that of a fragmented environment, where student admission data, fee transactions, hostel allocation, student attendance, and examination results are stored in separate departmental applications without a common data model and without real-time communication.

Enterprise Resource Planning (ERP) systems, originally designed for manufacturing environments, were subsequently adapted for higher education [1], [2]. While ERP adoption has yielded benefits in financial management and procurement, the student-facing dimension has consistently been treated as secondary [3]. Existing implementations are largely module-level automations rather than architecturally integrated systems. A further limitation is the absence of a student-centric unifying data object; when a student's identifier differs across modules, every cross-departmental query requires manual reconciliation.

The practical consequences of this fragmentation become apparent in the day-to-day administration. In the event a student contests a fee record, it becomes necessary for the administration to access a minimum of three different systems in order to arrive at a dependable conclusion. In the assessment of examination eligibility, the lack of direct connection between attendance records and academic records requires a cross-check, which causes delays and potentially leads to errors. These errors are the norm rather than the exception; they become quantifiable in terms of wasted staff time and reduced student confidence.

This study proposes an architecture in which the complete student lifecycle is the organizing design principle. The four research objectives are: (RO1) design a three-tier ERP architecture integrating all major student-facing modules under a centralized relational database; (RO2) implement an algorithmically generated Student ID as the universal primary key across all modules; (RO3) develop an OTP-based attendance verification mechanism to eliminate proxy attendance without hardware dependency; and (RO4) evaluate system performance, functional correctness, usability, and comparative advantage.

The remainder of this paper is structured as follows. Section II reviews prior literature on ERP adoption, attendance verification, and usability evaluation. Section III describes the system architecture, module design, OTP workflow, and testing methodology. Section IV presents and discusses results. Section V concludes and Section VI outlines future directions.

2. LITERATURE REVIEW

2.1 ERP in Higher Education

ERP systems in higher education emerged as an institutional response to growing administrative complexity [1], [2]. Mukred et al. (2023) proposed an ERP adoption model for higher learning institutions emphasizing student-centric, data-driven decision support [17]. Burns and McCormack (2023) found that most HEIs implementing ERP systems face misalignment and over-customization challenges stemming from generic enterprise models [18].

Pollock and Cornford (2004) examined the assumption that universities resist standardization, concluding that perceived uniqueness is frequently used to justify customization that prevents integration [1]. Allen and Kern (2001) identified departmental political resistance as a primary barrier to unified ERP deployment [2]. Both findings directly inform the ISMS design: the system imposes a minimal but consistent data contract across all modules while allowing presentation-layer variation by role, reducing the political cost of adoption without sacrificing architectural coherence. Davenport (1998) argued that a shared data model is the precondition for genuine process integration in large organizations [6]; the ISMS operationalizes this through the Student ID mechanism described in Section III-C.

2.2 Student Management Systems and Fragmentation

Student Management Systems (SMS) have been criticized for functional isolation—admission, fee, hostel, and examination modules typically operate through separate databases [10]. Zhao et al. (2022) demonstrated the benefits of IoT-integrated attendance management but noted that database fragmentation remains a systemic challenge [19]. IRJMETS (2023) found that institutions deploying ERP without centralized student data models reported higher administrative error rates [20]. Lamey et al. (2023) propose lifecycle-aligned data structures as foundational to intelligent HEI integration [13].

Fonseca et al. (2021) built a web-based SMS on an ERP framework but did not address a unifying student identifier or cross-module consistency [14]. Sinaga (2022) found the student data model to be the weakest link in ERP-based school governance [8]. Hassan et al. (2022) pointed to the mobile accessibility gap in university information systems, noting that students increasingly expect smartphone-based interaction with administrative services [11]. The OTP mechanism in the present system operates on any internet-connected device, partially addressing this accessibility gap without requiring a dedicated mobile application.

2.3 OTP Authentication and Usability Evaluation

Adekunle et al. (2023) demonstrated that OTP-integrated attendance systems prevent fraudulent registration with sub-second verification times [22]. Their work established that the critical design parameter is the validity window: a token expiring within 60–90 seconds is functionally immune to sharing across sessions. Taha et al. (2022) found OTP and NFC approaches provide the best security/cost balance for institutions without biometric infrastructure [23]. For usability evaluation, the System Usability Scale (SUS) [24] is widely validated; Suria (2024) reports mean SUS scores between 72 and 84 for well-designed academic systems [25]. Afif (2023) demonstrates student-centric design yields significantly higher SUS scores [21]. Vlachogianni and Tselios (2022) confirmed SUS reliability across participant groups with varying digital literacy [26].

2.4 Literature Gap

Despite substantial research, the treatment of the entire student lifecycle as a single architecturally unified process anchored to a persistent student identity has not been pursued as a design objective in published literature. Table I positions the present study against key prior works across five integration dimensions.

TABLE -1: COMPARATIVE POSITIONING AGAINST PRIOR LITERATURE

Study	Year	Cen. ID	OTP	SUS	Lifecycle
Pollock & Cornford	2004	No	No	No	No
Rabaa'i et al.	2009	Part.	No	No	No
Zhao et al.	2022	No	Part.	No	No
Taha et al.	2022	No	Yes	No	No
Afif	2023	No	No	Yes	No
Mukred et al.	2023	Part.	No	No	No
Lamey et al.	2023	Part.	No	No	Part.
Present Study	**2025**	**Yes**	**Yes**	**Yes**	**Yes**

3. RESEARCH METHODOLOGY

3.1 System Architecture

The architecture of the proposed Information Security Management System (ISMS) is based on a three-tier architecture, which clearly identifies the presentation, application, and data layers, as shown in Figure 1. For the presentation layer, the proposed ISMS architecture is based on the use of HTML5, CSS3, and JavaScript. For the

application layer, the proposed ISMS architecture is based on the use of Python 3.10, along with the use of the Flask 2.3 library, based on the ability to offer clean RESTful routing in accordance with the proposed architecture for the ERP system. For the data layer, the proposed ISMS architecture is based on the use of the MySQL 8.0 database, which offers ACID properties along with foreign key constraints. For password hashing, the proposed ISMS architecture is based on the use of the Bcrypt library, with a cost factor of 12. For the use of CSRF tokens, the proposed ISMS architecture is based on the use of CSRF tokens in all POST requests, along with the use of role-based ones.

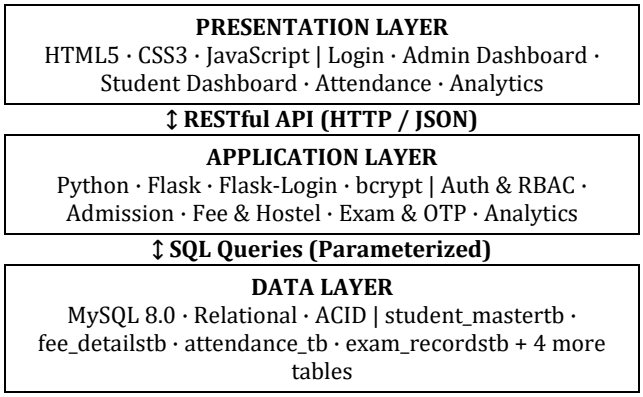

Fig-1: Three-tier architecture of the ERP-based ISMS.

3.2 System Modules

The ISMS comprises ten functional modules: (1) Authentication—secure login via Flask-Login and bcrypt; (2) User Management—admin-controlled account creation; (3) Student Admission—enrolment intake and Student ID generation; (4) Student Management—profile maintenance; (5) Fee Management—payment records and status; (6) Hostel Management—room allocation; (7) Exam Management—subject-wise marks; (8) Attendance Management—OTP-based daily attendance; (9) Document Management—file storage and access control; and (10) Analytics—dashboard reporting.

Each module is implemented as a Flask blueprint with its own route prefix, sharing a common database connection pool and utility functions for session validation, RBAC enforcement, and parameterized query construction. This design ensures that a failure in one module does not cascade to others, while common security controls are applied uniformly across all application routes.

3.3 Student ID Generation and Database Design

Student ID is created during the admission process and is subsequently used as the indexed foreign key in all the dependent tables. It is created by appending the admission year, department number, and a four-digit running number, all padded to four digits. This scheme enables the retrieval of student profiles across modules using a single query and avoids the problem of inconsistent student IDs across departments. Table II shows all the eight tables in the database.

TABLE -2: DATABASE SCHEMA — KEY RELATIONSHIPS

Table Name	Description	Key
userstb	System accounts / roles	PK: user_id
student_mastertb	Canonical student profile	PK: student_id
fee_detailstb	Fee transactions & status	FK: student_id
hostel_allocationtb	Room assignments	FK: student_id
exam_recordstb	Subject marks & dates	FK: student_id
attendance_tb	Daily OTP-verified attendance	FK: student_id
attendance_otptb	OTP codes & timestamps	Ref: attendance_tb
student_documents	Document metadata & paths	FK: student_id

3.4 OTP Attendance and RBAC Workflow

It uses a role-based access control system. Once a user is authenticated, the system, through the use of Flask-Login and bcrypt, will verify the user's details. System administrators are given a ten-module dashboard, while students are given read-only privileges. In terms of attendance, the administrator will provide a time-bound, one-time password, which will be entered by the student. The system will verify this password against the attendance_otptb table before entering a record into the attendance_tb table. Expired or invalid passwords are rejected, thus preventing proxy attendance in the absence of a biometric device.

3.5 Security Design

The security concerns are addressed across all architectural layers. SQL queries are performed using parameterized statements with the help of PyMySQL. The hashing function used here is bcrypt with a cost factor of 12. This results in a computation time of approximately 400 ms and hence prevents any brute force attacks. The CSRF tokens are mandatory for all forms that can submit data with the help of Flask-WTF. All cookies for sessions are secure with the HttpOnly flag and SameSite set to Lax. The time-based one-time password mechanism inherently acts as a security control because of the limited time frame

within which a shared password will become invalid before any action other than within a classroom scenario. Table VII, as mentioned in Section IV-F, lists a summary of all six architectural layers.

3.6 Testing Methodology

System testing was conducted across six categories: functional testing (manual execution of all module operations), negative testing (invalid inputs, expired OTPs, unauthorized access), authentication testing (valid/invalid credentials, session expiry), API testing (RESTful endpoint validation via Postman), database testing (CRUD and data integrity), and integration testing (cross-module consistency via Student ID propagation).

3.7 Usability Evaluation

A SUS study was conducted with 12 participants (8 students and 4 faculty/staff) from Ajeenkya DY Patil University. Nielsen (1994) established that five participants detect ~85% of usability issues, and SUS validation studies confirm that samples of 10–12 provide reliable aggregate scores for prototype assessment [24], [26]. Participants completed five representative tasks: login, student profile retrieval, OTP attendance marking, fee status check, and document upload. The standard SUS formula yielded scores on a 0–100 scale, where above 68 = above average, above 80 = good, and above 90 = excellent.

4. RESULTS AND DISCUSSION

4.1 Experimental Setup

The Information Security Management System (ISMS) is developed as a validated prototype on a local server with Python 3.10, Flask 2.3, MySQL 8.0, and Windows 11. The test data includes 200 students with varying department affiliations among four departments: CSE, MCA, ECE, and MBA, with over 500 attendances, 150 uploaded documents, and three accounts for administrators. Browser compatibility is also verified for Chrome version 120 and Firefox version 121.

4.2 System Performance

Response times were measured across 50 repeated trials per operation (Table III). All standard deviations are low (σ range: 0.07–0.22 s), confirming consistent performance. Sub-second query performance (0.60 s) results from indexing student_id as the universal foreign key, enabling join operations without full table scans. The 1.20 s login time reflects the bcrypt cost-factor-12 delay. OTP verification (0.80 s, σ = 0.11) demonstrates high predictability, critical for simultaneous classroom deployment. Compared to the manual baseline, performance improvements range from approximately 7× for login to over 300× for attendance processing.

TABLE -3: SYSTEM PERFORMANCE METRICS (N = 50 TRIALS)

Operation	Mean (s)	SD (s)	Manual Baseline
Login & role verify	1.20	0.18	8–15 s
OTP generation	0.40	0.07	3–5 min
OTP verify + commit	0.80	0.11	3–5 min
DB query (avg)	0.60	0.09	5–10 min
Student record retrieval	0.90	0.13	5–10 min
Document upload	1.50	0.22	Physical filing

4.3 Functional and Negative Testing

All 12 test scenarios across all modules yielded a Pass result. Representative cases included: login with valid/invalid credentials; student registration with duplicate Student ID prevention; OTP attendance marking and rejection of expired codes; fee record entry/retrieval; hostel allocation update; exam result storage; RBAC redirect for unauthorized URL access; document upload; and analytics report generation. Every positive test produced the expected output and every negative test correctly rejected invalid input with an appropriate error message. Table VI provides a comprehensive record of all test cases and outcomes.

TABLE -4: MODULE-LEVEL FUNCTIONAL AND NEGATIVE TEST RESULTS

No.	Module / Test Case	Input Condition	Result	Expected Behavior
1	Authentication — valid login	Correct credentials	Pass	Session created; RBAC role assigned
2	Authentication — invalid login	Wrong password	Pass	Error message; no session
3	Student Admission — enrolment	Complete form data	Pass	Student ID auto-generated
4	Student Admission — duplicate	Existing student ID	Pass	Duplicate prevented; error shown
5	OTP Attendance — valid OTP	Active, unexpired OTP	Pass	Attendance recorded in DB
6	OTP	Lapsed OTP	Pass	Rejected; no

	Attendance — expired OTP	string		record written
7	Fee Management	Payment entry	Pass	Fee status updated correctly
8	Hostel Allocation	Room assignment update	Pass	Record persisted; FK intact
9	Exam Management	Marks entry per subject	Pass	Subject scores stored correctly
10	RBAC — unauthorized URL	Student → admin route	Pass	Redirect to login / 403
11	Document Upload	PDF, valid session	Pass	File stored; metadata saved
12	Analytics Dashboard	Report generation	Pass	Dashboard renders correctly

4.4 Usability Evaluation (SUS)

Table IV shows a list of all 12 individuals' System Usability Scale (SUS) scores. The average SUS scores for all 12 individuals is 78.3 with a standard deviation of 4.2. The average is classified as good usability and is well above the 'above average' benchmark of 68 as defined in references [24], [25]. The low standard deviation implies that there is a uniform evaluation of user roles. The average for students is 78.1, while for faculty/staff it is 78.8. This implies that both roles have an equal experience with the role-based interface. This is supported by Afif (2023), who found an average SUS of 76.2 for a similar student mobile application [21]. Similarly, Suria (2024) found average SUS scores ranging from 72 to 84 for various academic applications [25].

TABLE -5: SUS SCORES — ALL 12 PARTICIPANTS

Participant	Role	SUS Score	Grade
P01	Student	82.5	Good
P02	Student	75.0	Above Avg.
P03	Student	80.0	Good
P04	Student	77.5	Above Avg.
P05	Student	72.5	Above Avg.
P06	Student	82.5	Good
P07	Student	85.0	Good
P08	Student	70.0	Above Avg.
P09	Faculty	80.0	Good
P10	Faculty	75.0	Above Avg.
P11	Staff	82.5	Good
P12	Staff	77.5	Above Avg.
Mean (σ)	**—**	**78.3 (4.2)**	**Good**

4.5 Comparative Analysis

Table V positions the proposed ISMS against manual workflows and the open-source Fedena ERP. The system outperforms both on lifecycle integration, OTP attendance, centralized identity, and formal usability evaluation. Fedena is acknowledged as a mature, production-deployed system with broader capabilities in multi-campus support and mobile applications—areas beyond the current prototype's scope. The most significant structural difference from Fedena is the data architecture: the ISMS enforces a single algorithmically generated identifier as the indexed foreign key throughout, enabling database-level cross-module JOINs rather than application-level reconciliation, which produces sub-second cross-module query performance and guarantees a consistent student view across all departments.

TABLE -6: COMPARATIVE ANALYSIS

Feature	Manual	Fedena	ISMS
Centralized Student ID	No	Partial	Yes
OTP Attendance	No	No	Yes
Real-time Cross-module	No	Partial	Yes
RBAC	No	Yes	Yes
Document Management	No	Yes	Yes
Analytics Dashboard	No	Yes	Yes
Lifecycle Architecture	No	Partial	Yes
SUS Evaluation	No	N/R	78.3/100
Hardware-free Attend.	Paper	Partial	Yes (OTP)

4.6 Security Evaluation

The six security mechanisms in Table VII were each verified under the negative testing protocol described in Section III-F. The bcrypt cost factor of 12 was chosen following NIST SP 800-63B guidance to maintain a minimum hash computation time of 100ms. The time-bound OTP mechanism binds attendance to knowledge of a short-lived token rather than a physical biometric. The key design tradeoff is that a student without an internet-connected device cannot self-register; this edge case is handled by administrator correction post-session, consistent with institutional operational practice.

TABLE -7: SECURITY MECHANISM SUMMARY

Security Layer	Mechanism	Protection Offered
Password Storage	bcrypt, cost factor 12	Brute-force resistance
Session Management	Flask-Login server-side sessions	Session hijacking prevention
Form Integrity	CSRF token in all POST requests	Cross-site request forgery
SQL Injection	Parameterized queries (PyMySQL)	Database injection attacks
Access Control	Role-Based Access Control (RBAC)	Privilege escalation
Attendance Fraud	Time-bound OTP with expiry check	Proxy attendance prevention

4.7 Limitations

Four limitations define the scope of this contribution. (1) Evaluation scale: the system was tested on a local server with 200 simulated records; scalability under concurrent institutional load remains an open empirical question. (2) SUS sample size: 12 participants is appropriate for formative prototype evaluation [24], [26] but limits generalizability. (3) Institutional deployment: no live deployment has been conducted; real-world adoption and change-management factors are future research directions. (4) Diagnostic depth: SUS provides a composite score but does not identify specific interface-level issues; future work should complement it with task-based testing and qualitative interviews.

5. CONCLUSION

This article seeks to document and evaluate the performance of an ERP-based Integrated Student Management System, which integrates ten functional components into a lifecycle-based centralized relational database management system. The performance testing of the system, which was conducted on a sample of 200 student records, showed that the mean response times for the system were 1.20 s (SD = 0.18) for the login, 0.80 s (SD = 0.11) for OTP verification, and 0.60 s (SD = 0.09) for database queries. The usability of the system was also tested by 12 participants, and the results showed a mean of 78.3 out of 100, which falls in the Good category of the SUS scale. All the modules of the system successfully completed the test conditions for both positive and negative testing.

The study makes three architectural contributions: (1) a lifecycle-oriented data model in which every student administrative process is part of a continuous, architecturally unified student lifecycle; (2) an algorithmically generated Student ID used as the universal indexed foreign key across all database tables; and (3) an OTP-based attendance verification mechanism that prevents proxy attendance without biometric hardware. This work extends process integration theory (Davenport, 1998) into the higher education domain, demonstrating that end-to-end lifecycle alignment is achievable within a standard web framework. The system offers a deployable and extensible framework for mid-sized HEIs seeking to modernize student administration within realistic resource constraints.

6. FUTURE WORK

Short-term directions (1–2 years) include cloud-based deployment on AWS or GCP to address the scalability limitation, load and stress testing for 500–1,000 concurrent users, and a React Native or Flutter mobile application to deliver native OTP notifications. The mobile application would also address the fee module navigational friction observed during the SUS evaluation, providing push-based alerts that route students directly to the relevant module.

Medium-term directions (2–3 years) include extending the analytics module to support cohort-level reporting on the relationship between attendance patterns, fee compliance, and academic outcomes. The current dashboard presents aggregate counts; adding time-series visualizations and cohort filters would transform it into a planning resource for department heads and institutional researchers.

Long-term directions (3–5 years) include academic risk prediction via machine learning (gradient-boosted classifiers or LSTM networks trained on attendance, fees, and exam data), biometric attendance integration, LMS integration with Moodle or Canvas, full compliance with India's Digital Personal Data Protection Act 2023, and expanded usability studies combining SUS with task-based testing and qualitative interviews to generate actionable design specifications for the next interface iteration.

REFERENCES

[1] N. Pollock and J. Cornford, "ERP systems and the university as a 'unique' organisation," Inf. Technol. People, vol. 17, no. 1, pp. 31–52, 2004.

[2] D. Allen and T. Kern, "Enterprise resource planning implementation: Stories of power, politics, and resistance," in Realigning Research and Practice in IS Development, Springer, 2001, pp. 149–168.

[3] A. A. Rabaa'i, W. Bandara, and G. G. Gable, "ERP systems in the higher education sector: A descriptive study," in Proc. ACIS 2009, Melbourne, Australia.

[4] P. Hawking and C. Sellitto, "Business intelligence (BI) critical success factors," in Proc. ACIS 2010, Brisbane, Australia.

[5] K. Al-Fawaz, Z. Al-Salti, and T. Eldabi, "Critical success factors in ERP implementation: A review," in Proc. EMCIS 2008, Dubai, UAE.

[6] T. H. Davenport, "Putting the enterprise into the enterprise system," Harvard Bus. Rev., vol. 76, no. 4, pp. 121–131, Jul. 1998.

[7] S. Abou El-Seoud et al., "Integrated education management system via cloud computing," Int. J. Interact. Mob. Technol., vol. 11, no. 2, pp. 24–33, 2017.

[8] J. H. Sinaga, "ERP system for school governance in Indonesia," in Proc. ICREAM, vol. 6, no. 1, 2022.

[9] N. W. C. Lasantha and S. Vasanthapriyan, "A theoretical model for a smart management information system," IJACSA, vol. 12, no. 3, 2021.

[10] S. Supriyono, "Architecture in institutional management systems using Odoo ERP," IJISTECH, vol. 5, no. 4, pp. 490–497, 2021.

[11] A. Hassan et al., "Integrating mobile computing in university information management systems," Indian J. Sci. Technol., vol. 15, no. 8, pp. 343–350, 2022.

[12] B. Alharbi, "Development of an ERP system design course to improve students' learning outcomes," iJET, vol. 16, no. 12, p. 276, 2021.

[13] A. Lamey et al., "A guide for creating intelligent integrated solutions in higher education," Sustainability, vol. 15, no. 11, p. 8780, 2023.

[14] T. Fonseca et al., "Web application for student management system using ERP," IJERT, vol. 8, no. 10, 2021.

[15] M. Grinberg, Flask Web Development, 2nd ed. Sebastopol, CA: O'Reilly Media, 2018.

[16] Oracle Corp., MySQL 8.0 Reference Manual, 2023. [Online].Available: https://dev.mysql.com/doc/refman/8.0/en/

[17] M. Mukred et al., "Enterprise resource planning adoption model for well-informed decision in higher learning institutions," J. Inf. Sci., vol. 49, no. 3, pp. 792–813, 2023.

[18] S. Burns and M. McCormack, "More than 'going live': Achieving institutional transformation through ERP implementation," EDUCAUSE Research Report, 2023.

[19] L. Zhao et al., "College smart classroom attendance management system based on IoT," Comput. Intell. Neurosci., Art. ID 4953721, 2022.

[20] Anonymous, "A study on the effectiveness of college ERP systems," IRJMETS, vol. 5, no. 5, 2023.

[21] M. H. Afif, "Assessing students' perceptions of mobile applications usability using System Usability Scale," J. Comput. Sci., vol. 19, no. 1, pp. 11–19, 2023.

[22] S. O. Adekunle et al., "Fraud mitigation in attendance monitoring systems using device identity verification," IJACSA, vol. 14, no. 4, 2023.

[23] B. O. Taha and S. A. Hasan, "A review of student attendance management systems," Eurasian J. Sci. Eng., vol. 8, no. 3, 2022.

[24] J. Brooke, "SUS: A 'quick and dirty' usability scale," in Usability Evaluation in Industry, P. Jordan et al., Eds. London: Taylor & Francis, 1996, pp. 189–194.

[25] O. Suria, "A statistical analysis of System Usability Scale (SUS) evaluations in online learning platforms," J. Inf. Syst. Inform., vol. 6, no. 2, pp. 992–1007, 2024.

[26] P. Vlachogianni and N. Tselios, "Perceived usability evaluation of educational technology using the System Usability Scale (SUS): A systematic review," J. Res. Technol. Educ., vol. 54, no. 3, pp. 392–409, 2022.

[27] J. Blattgerste, J. Behrends, and T. Pfeiffer, "A web-based analysis toolkit for the System Usability Scale," in Proc. PETRA '22, Corfu, Greece: ACM, 2022.