www.ijarp.com ISSN 2456-9992 Page: 01-16

International Journal Advanced Research Publications

Case Study

Volume: 02
Issue: 07

DESIGN AND IMPLEMENTATION OF A STUDENT IDENTITY MANAGEMENT SYSTEM: CASE STUDY OF ESPAM FORMATION UNIVERSITY, REPUBLIC OF BENIN

***Anoliefo, Chukwuma Josemaria, Popoola Olusegun Victor, Olalere Sherifdeen Abiodun, Emmanuel Praise-God**

Department of Computer Science, ESPAM Formation University, Republic of Benin.

Article Received: 09 June 2026, Article Revised: 29 June 2026, Published on: 19 July 2026

***Corresponding Author: Anoliefo, Chukwuma Josemaria**
Department of Computer Science, ESPAM Formation University, Republic of Benin.
Doi: https://doi-doi.org/101555/ijarp.3206

ABSTRACT

Higher-education institutions need a trustworthy system to verify student identities, link records, issue verifiable credentials, and control access throughout the student lifecycle. This study presents a web-based Student Identity Management System (SIMS) prototype for ESPAM Formation University, Benin. Using a design-science case study, requirements were based on ESPAM's management procedures, digital identity guidance, and a process model covering admission to archival. The system includes a central registry, role-based access, duplicate-record prevention, bilingual interfaces, digitally signed QR credentials, audit logs, document management, and privacy-aware reporting. It features layered architecture, a normalised database, and a prototype built with Django and PostgreSQL. Security aligns with NIST and OWASP, including strong password hashing, multifactor authentication, encryption, least privilege, session controls, rate limiting, and tamper-evident audit trails. Testing involved 24 functional/security cases and 1,000 synthetic records; all passed. Performance tests showed rapid lookup and verification times under 0.012 ms and 0.101 ms, respectively, demonstrating correctness with low overhead. The study shows that a secure, centralised identity platform can reduce record duplication, speed verification, enhance accountability, and support integration across university services.

KEYWORDS: Student identity management; digital identity; role-based access control; QR verification; student information system; ESPAM Formation University.

1. INTRODUCTION

A student identity is not merely a matriculation number printed on a plastic card. It is a governed digital representation that connects an individual to admission documents, programme membership, academic status, fees, examinations, library privileges, institutional communications, and eventual alum records. Where these links are maintained through paper files, isolated spreadsheets, or loosely connected portals, the institution becomes vulnerable to duplicate records, inconsistent names, unauthorised changes, slow verification, card impersonation, and weak accountability. These failures are administrative problems, but they are also security and data-governance problems. ESPAM Formation University is a bilingual private higher education institution with campuses, including Porto-Novo. Its published management guidelines explicitly prioritise information and communication technology for admission, websites, results, student and lecturer databases, financial operations, and online examinations [1], [2]. This institutional direction creates a strong case for a unified identity layer. The objective is not to replace every academic application at once; rather, to establish a single authoritative student identity that other applications can trust. Accordingly, this study asks: how should a secure, usable, and locally appropriate student identity management system be designed for ESPAM; what data model and controls are required across the identity lifecycle; and whether a prototype can satisfy defined functional and security requirements with acceptable computational overhead? The contribution is a complete case-specific architecture, data model, control framework, implementation blueprint, and reproducible prototype evaluation.

2. Background and Related Literature

2.1 Student records, identity, and institutional risk

Student information systems commonly automate registration, fees, results, accommodation, communication, and transcript generation. However, a student identity management system has a narrower and more foundational role: it establishes the unique subject, maintains authoritative biographical and status attributes, binds authenticators and credentials to that subject, and records who changed what and when. A conventional student database may store records; an identity system governs trust in those records. Studies of African university information systems consistently report that centralised electronic records improve retrieval, consistency, and administrative coordination, whereas fragmented manual workflows lead to duplication and delays [9]–[11]. Falebita [9] further argues that migration to a web portal without a deliberate security architecture merely relocates risk. This distinction is important.

A visually attractive portal is not an identity-management solution unless it incorporates proofing, uniqueness, authorisation, credential lifecycle controls, and auditability.

2.2 Digital identity and assurance

NIST SP 800-63-4 conceptualises digital identity around identity proofing, enrollment, authentication, and federation, with assurance selected according to risk [3]. Although written for public-sector digital services, the principles translate well to university environments. Admission documents and a facial photograph support identity proofing; the approved applicant becomes an enrolled student; a password, one-time code, or institutional credential supports authentication; and trusted assertions can later enable integration with learning, finance, or library systems. The proposed ESPAM system therefore separates four objects that are often wrongly collapsed into one: the person, the institutional student record, the user account, and the issued credential. A student may retain one person record while changing programme, level, card, or login state. This separation prevents routine events, such as a lost card or a programme transfer, from corrupting the underlying identity.

2.3 Authorisation and software quality

Role-based access control (RBAC) assigns permissions to job roles rather than directly to every individual user. The model reduces administrative complexity and supports least privilege [7], [8]. In the case study, registry officers can create and update identity records; security staff can verify credentials without viewing unnecessary admission documents; finance or academic systems may receive limited identity assertions; students can view their own approved details; and system administrators can manage technical configuration without gaining unrestricted authority to alter official student data.

OWASP distinguishes authentication, proving who a user is, from authorisation, deciding what that user may do [4], [5]. Both must be enforced on the server, not merely hidden in the interface. ISO/IEC 25010:2023 also provides a quality model that covers functional suitability, performance efficiency, compatibility, interaction capability, reliability, security, maintainability, flexibility, and safety [6]. These quality dimensions informed the non-functional requirements and test criteria used in this study.

2.4 Privacy and Beninese context

Student records contain personal data, identity documents, photographs, addresses, contact details, and educational status. The Republic of Benin's Digital Code, Law No. 2017-20 of 20 April 2018, as amended by Law No. 2020-35 of 6 January 2021, provides a national framework for digital activities and personal data protection [12]. Consequently, the design adopts purpose limitation, data minimisation, controlled disclosure, retention rules, access

logging, and procedures for correcting inaccurate data. These principles are engineering requirements, not decorative policy statements.

2.5 Research gap

Existing student-management studies often combine many modules but provide limited treatment of identity lifecycle governance, revocation, signed credential verification, bilingual usability, or privacy-by-design. The present work addresses that gap by treating identity as a reusable institutional service. Its output is not an inflated all-purpose enterprise resource planning system; it is a precise identity backbone that can be integrated incrementally with existing applications.

3. METHODOLOGY AND REQUIREMENTS ANALYSIS

3.1 Research design

A design-science methodology was used because the research problem requires constructing and evaluating an artefact. The stages were: (i) problem definition; (ii) extraction of institutional and regulatory requirements; (iii) analysis of the current-state process as a hybrid paper/spreadsheet workflow; (iv) specification of functional and non-functional requirements; (v) architecture, database, and security design; (vi) prototype implementation; and (vii) verification using scripted tests and synthetic records. No claim is made that the prototype has already been deployed across ESPAM. The case study is an implementation-ready design grounded in the university's published ICT priorities.

3.2 Stakeholders and use cases

The principal stakeholders are applicants, admitted students, registry officers, departmental officers, security personnel, system administrators, management, and trusted external verifiers. Their interests differ. A student needs a simple self-service view and rapid error correction. Registry staff need complete records and controlled update authority. Security personnel need a fast yes/no verification result. Management needs aggregate reports rather than unrestricted access to every document. These differences justify role-specific screens and data minimisation.

- Applicant conversion: create a provisional identity from an approved admission record and flag missing evidence.
- Identity proofing: inspect supporting documents, capture a standardised photograph, detect duplicate records, and approve or reject enrollment.
- Matriculation: generate a unique institutional number and activate the student lifecycle state.

- Credential issuance: generate an ID-card serial number and signed QR credential; print, reprint, suspend, revoke, or replace the card without deleting the student.
- Verification: allow authorised staff or a limited public endpoint to confirm validity without exposing unnecessary personal data.
- Maintenance and exit: manage corrections, programme changes, suspension, graduation, archival, and retention.
- Governance: preserve audit trails, exception reports, access reviews, backup status, and security events.

3.3 Functional requirements

Table 1. Core functional requirements.

ID	Requirement	Primary actor
FR1	Create and approve a unique student identity	Registry
FR2	Prevent duplicate matriculation number, e-mail, and credential token	System
FR3	Store identity attributes, photo, supporting documents, programme and status	Registry
FR4	Issue, verify, expire, revoke, and replace ID credentials	Registry/Security
FR5	Enforce role permissions and self-service boundaries	All users
FR6	Record security-relevant and data-changing actions	System
FR7	Generate filtered operational reports and exports	Management
FR8	Support English and French labels and low-bandwidth pages	All users

3.4 Non-functional requirements

The system should provide secure authentication, least-privilege authorisation, complete auditability, responsive operation on modest devices, graceful handling of intermittent connectivity, daily encrypted backups, recovery testing, maintainable, modular code, and accessibility-compatible forms. Availability targets should be realistic for the institution's infrastructure; critical verification may use a read-only cached service during temporary outages. The design also requires unique constraints at the database level because interface validation alone is insufficient.

3.5 Success criteria

The artefact is considered technically successful when it completes the defined lifecycle, rejects duplicate identifiers and invalid transitions, verifies signed credentials, denies unauthorised actions, records audit events, and executes basic operations without material computational delay in a synthetic test dataset. Organisational success—reduced queue time,

improved user satisfaction, and lower fraud must be assessed after controlled deployment and is therefore outside the present prototype claims.

4. System Architecture and Process Design

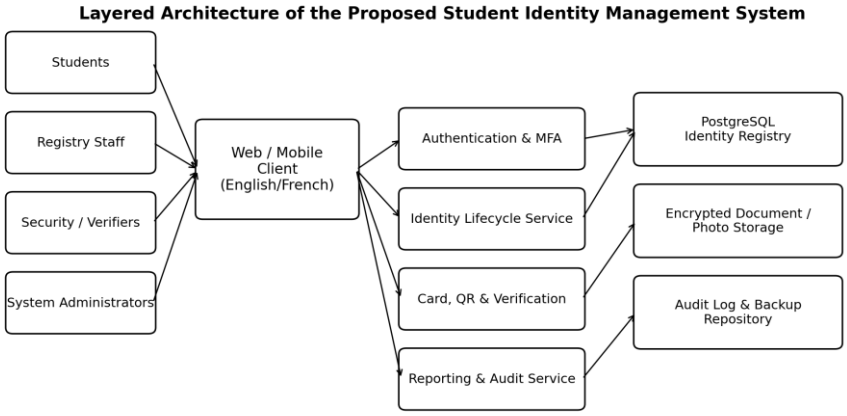

Figure 1. Layered architecture of the proposed SIMS.

4.1 Architectural rationale

The architecture separates presentation, application services, persistence, and infrastructure concerns. Users interact through responsive web pages that can be localised into English and French. The application layer contains authentication, identity lifecycle, credential, verification, reporting, and audit services. Student attributes are stored in a relational registry; photographs and scanned evidence are stored outside the public web root with database references and access policies; and audit data are protected from routine modification. This separation is not an academic ornament. It prevents a card-printing function from directly manipulating admission documents, permits the verification service to disclose only minimal fields, and allows reporting to operate on approved views. It also supports later integration through authenticated application programming interfaces rather than uncontrolled spreadsheet exchange.

4.2 Identity lifecycle

The lifecycle begins with a provisional applicant record. After admission, a registry officer initiates proofing by comparing submitted evidence with the admission decision and checking for probable duplicates using normalised name, date of birth, e-mail, telephone number, and document identifiers. Unique constraints block exact duplicates; probable matches are routed for human review because automatic matching can falsely merge different people. Upon approval, the system generates a matriculation number in a configurable format, such as

ESPAM/YYYY/UNIT/NNNN. A database transaction reserves the sequence, creates the student record, writes an audit entry, and queues card production. The record then moves through controlled states: pending, active, suspended, graduated, archived, or deceased, where legally appropriate. Deletion is not a normal lifecycle operation; archival and retention policies are safer because academic identities have long-term evidentiary value.

4.3 Credential and verification flow

The physical card contains visible attributes, a serial number, an expiry date, a photograph, and a QR code. The QR code does not embed a full personal record. It carries a random or signed reference that the verification service resolves into a minimal response: valid/invalid, name, matriculation number, photograph thumbnail (where authorised), programme, card state, and expiry. A lost card is revoked and replaced with a new credential while the student's identity remains unchanged.

ALGORITHM 1: Issue and verify a student credential
1. Validate actor permission and student status.
2. Generate card_serial and a random credential identifier.
3. Compute signature = HMAC(server_secret, credential_identifier).
4. Store only the credential hash, issue time, expiry, and state.
5. Encode identifier + signature in QR; print card.
6. On scan, validate signature, expiry, revocation, and student state.
7. Return the minimum permitted verification response; write an audit event.

4.4 Integration boundaries

Admissions may create provisional identities but cannot silently activate them. Finance may query whether a student identity is active and return payment standing without becoming the identity authority. Examination and e-learning systems may require a stable student identifier and account status. This master data pattern eliminates the common error in which each department creates a different version of the same student.

5. Database, Security, and Privacy Design

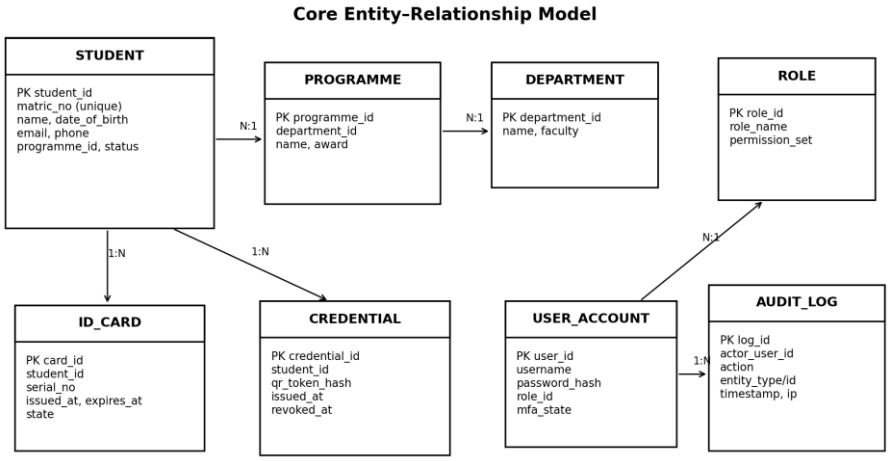

Figure 2. Core entity–relationship model.

5.1 Data model and integrity

The normalised schema separates students, programmes, departments, cards, digital credentials, user accounts, roles, and audit events. The student table holds stable identity and institutional attributes; the card and credential tables are one-to-many because a student can receive replacements over time. Foreign keys preserve referential integrity, while unique indexes protect matriculation numbers, card serials, account names, and credential tokens. Status values are constrained to an approved set, and critical updates are executed in transactions. Photographs and documents are referenced by opaque storage identifiers rather than predictable public file paths. File type, size, malware scan result, checksum, uploader, and retention category are recorded. This design supports integrity checking and prevents a web user from guessing another student's document address.

5.2 Role and permission model

Table 2. Proposed role-based access-control matrix.

Role	Permitted scope	Explicit restriction
Student	View your own approved profile; request correction	Cannot alter official status or issue cards
Registry officer	Create, proof, update, issue and replace cards	Cannot alter system roles or erase audit logs
Security verifier	Scan and verify active credentials	Cannot view admission documents or edit profiles
Department officer	View programme-scoped students; submit a correction request	No cross-department bulk export
System	Manage infrastructure, accounts and	No unilateral alteration of

Role	Permitted scope	Explicit restriction
administrator	configuration	official identity data
Auditor/management	Read approved reports and logs	No operational data editing

5.3 Security controls

Passwords are hashed with an adaptive algorithm such as Argon2id or PBKDF2 using per-user salts. Privileged accounts require multifactor authentication. All traffic uses HTTPS; cookies are Secure, HttpOnly, and SameSite; sessions expire after inactivity; login and verification endpoints are rate-limited; and account recovery requires stronger evidence than knowledge of a date of birth. Server-side authorisation is checked on every protected action. Database roles, views, and row-level policies provide a second layer of enforcement [13].

High-risk operations, identity approval, card issuance, revocation, role assignment, bulk export, and correction of core biographical attributes produce immutable audit records. For especially sensitive changes, maker-checker approval is recommended: one officer proposes the action and another approves it. This reduces insider risk and accidental alteration.

5.4 Privacy engineering

Only data necessary for defined university functions should be collected. Public verification must never reveal addresses, telephone numbers, dates of birth, document numbers, or financial information. Students should receive a clear privacy notice and a mechanism to request correction. Retention schedules must distinguish permanent academic identity attributes from temporary proofing documents. Exported reports should be aggregated or pseudonymized whenever individual identification is unnecessary.

5.5 Threat model

The threat model covers credential stuffing, session theft, privilege escalation, fraudulent enrollment, QR cloning, malicious uploads, insecure object references, insider misuse, data exfiltration, ransomware, and service outage. Technical controls reduce these risks, while staff vetting, segregation of duties, incident response, secure card stock, and disciplined offboarding remain essential organisational safeguards.

6. Prototype Implementation

6.1 Technology stack

The prototype follows a Django-compatible model–view–template/service structure written in Python. PostgreSQL is the production database target because it supports transactions, constraints, database roles, and row-level security. Bootstrap-compatible responsive components provide a consistent interface, while a QR library generates signed credential

images. Nginx or an equivalent reverse proxy terminates HTTPS connections, and containerised deployment simplifies repeatable configuration. The architecture is not irreversibly tied to a single framework; the important implementation properties are separation of concerns, server-side validation, transactional integrity, and tested authorisation.

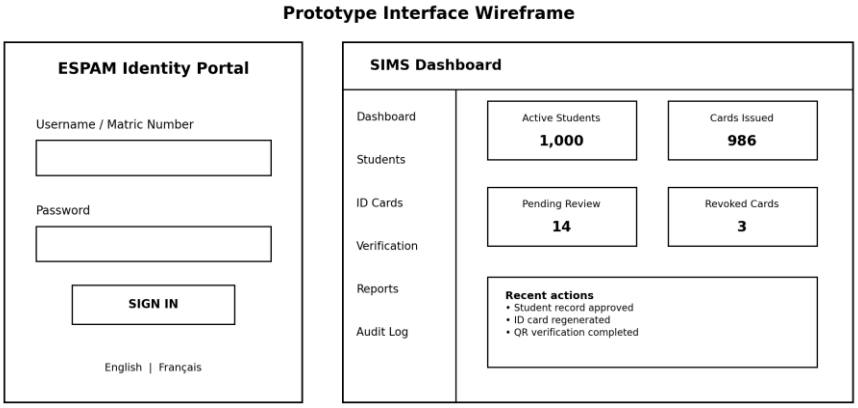

Figure 3. Prototype login and registry-dashboard wireframe.

6.2 Major modules

The enrollment module imports or creates an admitted applicant, normalises identity attributes, verifies uniqueness, and manages proof of evidence. The registry module provides search, profile review, correction workflow, programme assignment, and status transitions. The card module generates serial numbers, expiration dates, QR credentials, print queues, replacement reasons, and revocation events. The verification module supports camera scanning and manual entry. The administration module manages users, roles, localisation labels, numbering rules, retention categories, and system parameters. The audit module provides filtered, read-only views and anomaly alerts.

6.3 Validation and duplicate prevention

Validation is performed at three levels. The browser improves usability by identifying missing fields, but it is not trusted. Application services validate data types, formats, permissions, state transitions, and business rules. Finally, the database enforces uniqueness, nullability, foreign keys, and check constraints. The duplicate process combines deterministic and probabilistic logic: exact matches of document number, matriculation number, e-mail, or credentials are blocked; similar names and dates of birth generate a review warning. Human confirmation is mandatory before merging records.

6.4 Matriculation-number generation

```
BEGIN TRANSACTION
  lock numbering_sequence for the selected year/unit;
  next_value := current_value + 1;
  matric_no := format("ESPAM/%Y/%UNIT/%04d", next_value);
  insert student with UNIQUE(matric_no);
  insert audit_log(actor, "IDENTITY_APPROVED", student_id);
COMMIT
On conflict or failure: ROLLBACK and retry safely.
```

Using a transaction and a unique constraint prevents two officers from receiving the same number under concurrent use. Sequence gaps may occur after rollback; that is acceptable because uniqueness and traceability are more important than cosmetically perfect numbering.

6.5 Bilingual and low-bandwidth design

All labels, validation messages, status descriptions, and help text are externalised for English and French. Pages avoid oversized media, load results in bounded pages, and defer document previews until requested. A verifier sees one compact result rather than the entire student profile. These choices improve usability on mobile devices and reduce data consumption without compromising security.

6.6 Operational controls

Deployment requires protected secrets, reviewed database migrations, automated backup, restoration drills, patching, account recertification, and a documented incident-response path. Test and production environments must remain separate, and synthetic data should be used for demonstrations and training.

7. Testing and Results

7.1 Evaluation protocol

Evaluation was deliberately divided into correctness testing and microbenchmarking. A lightweight SQLite test harness was used to make the unit-level checks reproducible without requiring a production server. The production design remains PostgreSQL. Synthetic records were generated with non-real names, e-mail addresses, programmes, levels, statuses, and random credential tokens. Tests covered uniqueness constraints, valid and invalid status transitions, role permissions, signed QR code verification, tamper rejection, and audit persistence.

Table 3. Functional and security-oriented prototype verification (24/24 individual cases passed).

Test group	Expected/observed result	Status
Identity creation and retrieval	Record created and retrieved by a unique identifier	Pass
Duplicate matriculation number	Second insertion rejected by database constraint	Pass
Duplicate e-mail address	Second insertion rejected	Pass
Lifecycle states	Active, suspended and graduated transitions retained	Pass
Invalid lifecycle state	Unapproved state rejected	Pass
Signed QR validation	Correct signature accepted	Pass
QR tampering	Changed matriculation number or malformed signature rejected	Pass
RBAC checks	Twelve allow/deny permission cases were enforced	Pass
Audit logging	Actor and event persisted	Pass

7.2 Microbenchmark results

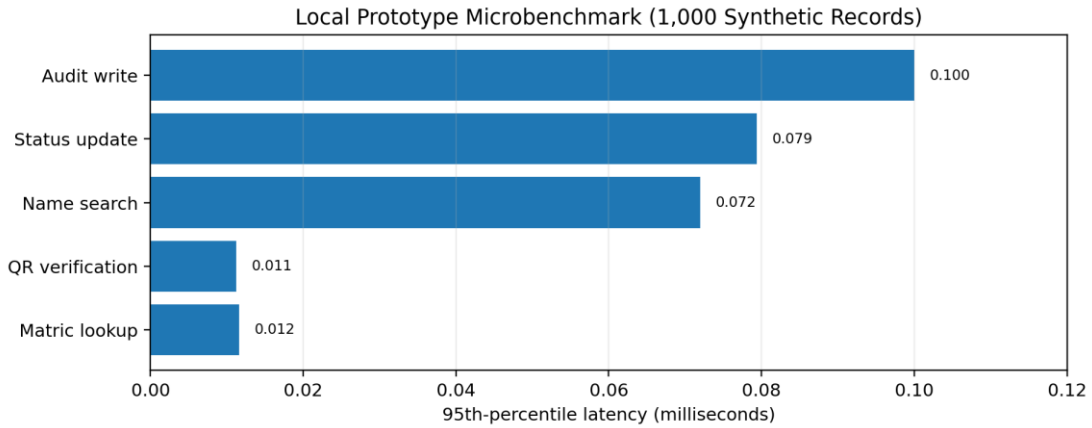

Figure 4. Observed 95th-percentile local operation latency.

Insertion of 1,000 synthetic student records in a single transaction took 14.76 ms. Across repeated warm-cache operations, the 95th-percentile latency was 0.012 ms for exact matriculation-number lookup, 0.011 ms for QR-token verification, 0.072 ms for a bounded surname search, 0.079 ms for a status update, and 0.100 ms for an audit-log write. These numbers chiefly show that the selected data model and integrity checks impose negligible local computational overhead at this dataset size.

7.3 Interpretation and validity

The results are encouraging but must not be exaggerated. SQLite running locally without network delay, concurrent users, image retrieval, TLS, reverse proxying, or production logging is not a capacity test for ESPAM. Real deployment performance will depend on

server resources, indexing, connection pooling, workload mix, bandwidth, backup activity, and concurrent verification traffic. The proper conclusion is limited: the prototype logic is correct under the tested cases, and basic database operations are not inherently expensive.

The 24 passed cases support functional suitability for the defined prototype scope. Security confidence remains partial because penetration testing, dependency scanning, code review, access control review, and adversarial file upload testing are still required. Usability likewise requires observation of real registry officers and students in both English and French. A first-class system is not certified by a green unit-test table alone.

7.4 Acceptance test plan for deployment

- Pilot with one intake or department and reconcile every migrated record against the authoritative source.
- Measure duplicate rate, average approval time, card replacement time, verification success, help-desk volume, and correction turnaround.
- Conduct role-based user acceptance tests and accessibility review in English and French.
- Run vulnerability scanning, manual authorisation testing, backup restoration, incident simulation, and load testing under realistic concurrency.
- Obtain management, registry, information-security, and data-protection approval before institution-wide migration.

8. DISCUSSION, DEPLOYMENT STRATEGY, AND LIMITATIONS

8.1 Expected institutional value

The strongest benefit of the proposed system is not card printing. It is the creation of a single authoritative identity that can be reused across university services. A correctly governed identity registry reduces duplicate student records, eliminates repeated manual re-entry, supports rapid verification, and clarifies accountability for changes. Signed QR verification also improves the usefulness of the physical card, as validity can be verified against the current status rather than inferred from its appearance. For ESPAM, the design aligns with the institution's stated preference for ICT-enabled admission, student databases, results, financial operations, and online examinations [2]. The identity registry becomes the reference point through which these systems exchange stable identifiers. This is preferable to a large, risky 'big-bang' replacement of every existing system. Incremental integration allows the university to preserve working applications while removing identity duplication.

8.2 Governance implications

Technology cannot resolve ambiguous institutional authority. ESPAM must designate an identity data owner, define who approves core attributes, establish evidence standards, specify retention periods, and regularly review privileged access. The registry should own the official student identity data; information technology should operate the platform; departments should submit controlled requests; and internal audit should independently inspect logs. Without this division, an administrator with technical access could become an unreviewed authority on academic records. Data-quality governance is equally necessary. Mandatory fields, reference lists, and validation rules reduce errors, but old records may contain inconsistent names, dates, and programme labels. Migration should therefore include profiling, mapping, deduplication, exception queues, and signed reconciliation reports. Importing poor data quickly does not create a high-quality system; it creates a faster source of poor data.

8.3 Phased deployment plan

Table 4. Recommended implementation roadmap.

Stage	Primary activities
Phase 1: Preparation	Appoint owner; approve requirements, privacy notice, roles, numbering and retention rules.
Phase 2: Pilot	Deploy registry, proofing, card and verification modules for a bounded cohort.
Phase 3: Integration	Connect admissions and selected downstream services through authenticated APIs.
Phase 4: Migration	Clean and import historical active-student records; retain traceable source references.
Phase 5: Scale and assurance	Institution-wide rollout, monitoring, access review, penetration testing and recovery drills.

8.4 Cost and sustainability considerations

Open-source components can reduce licence cost, but they do not eliminate the need for competent administration, secure hosting, card-production equipment, backup storage, staff training, user support, and periodic security assessment. The university should budget for lifecycle costs rather than only initial development. A system that cannot be patched, restored, or supported after the project author graduates is not sustainable.

8.5 Limitations

This study has four principal limitations. First, requirements were derived from published institutional priorities and a modelled registry workflow, rather than from a full ethnographic study of every ESPAM campus. Second, testing used synthetic data and a local harness, not

live student records or production concurrency. Third, biometric integration was intentionally excluded because it introduces additional privacy, hardware, bias, and failure-mode concerns; the architecture can accommodate it only after a separate assessment of necessity and proportionality. Fourth, no claim is made that the design alone guarantees compliance with Beninese data-protection law. Formal legal and regulatory review remains necessary before deployment.

8.6 Future research

Future work should conduct participatory requirements validation with registry staff and students, develop a working multilingual pilot, test migration quality, evaluate usability and accessibility, and measure operational outcomes over at least one academic cycle. Further research may examine federated single sign-on, verifiable digital credentials, offline verification, privacy-preserving analytics, and interoperability with national or regional education systems.

9. CONCLUSION AND RECOMMENDATIONS

This paper designed and evaluated a security-by-design Student Identity Management System for ESPAM Formation University. The proposed solution treats identity as an institutional service spanning proofing, enrollment, matriculation, authentication, card issuance, verification, lifecycle changes, and archival. Its central features are a normalised identity registry, database-enforced uniqueness, role-based access control, signed QR credentials, bilingual, low-bandwidth interfaces, protected document storage, and comprehensive audit logging. The prototype verification was successful within its stated scope: all 24 scripted functional and security-oriented checks passed, and local operations on 1,000 synthetic records imposed very low computational overhead. The evaluation does not substitute for production testing, but it provides evidence that the architecture is technically coherent and implementation-ready. The university should proceed through a bounded pilot rather than an institution-wide launch, validate real workflows, perform privacy and security reviews, test restoration and incident response, and measure service outcomes after deployment. Five recommendations follow. First, ESPAM should establish a formal identity governance policy before the software rollout. Second, the registry, not individual departments, should maintain the authoritative student identity. Third, privileged actions should use multifactor authentication, segregation of duties, and immutable auditing. Fourth, migration should be treated as a data-quality project with reconciliation, not a simple spreadsheet upload. Fifth, integration should occur through controlled APIs and stable student identifiers. Implemented

with these controls, SIMS can provide a dependable foundation for admissions, finance, examinations, e-learning, library, security, and alum services.

REFERENCES

1. ESPAM Formation University, “About ESPAM Formation University,” official website, accessed 14 July 2026.
2. ESPAM Formation University, Guidelines for Management Procedures, pp. 6–8, 2021.
3. D. Temoshok et al., Digital Identity Guidelines, NIST Special Publication 800-63-4, National Institute of Standards and Technology, 2025.
4. OWASP Foundation, “Authentication Cheat Sheet,” OWASP Cheat Sheet Series, accessed 14 July 2026.
5. OWASP Foundation, “Authorisation Cheat Sheet,” OWASP Cheat Sheet Series, accessed 14 July 2026.
6. International Organisation for Standardisation, ISO/IEC 25010:2023—Systems and Software Engineering: Product Quality Model, Geneva, 2023.
7. 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, 1996.
8. D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, “Proposed NIST standard for role-based access control,” ACM Transactions on Information and System Security, vol. 4, no. 3, pp. 224–274, 2001.
9. O. S. Falebita, “Secure web-based student information management system,” arXiv:2211.00072, 2022.
10. S. C. Lubanga et al., “Web-based student information management system in universities: Experiences from Mzuzu University,” SCECSAL Conference Proceedings, 2018.
11. N. Jagero, “An assessment of the impact of the centralised electronic student records management system at Africa University,” Issues in Economics and Business, vol. 2, no. 2, 2016.
12. Republic of Benin, Law No. 2017-20 of 20 April 2018 relating to the Digital Code in the Republic of Benin, as amended by Law No. 2020-35 of 6 January 2021.
13. PostgreSQL Global Development Group, “Database roles, privileges and row security policies,” PostgreSQL Documentation, accessed 14 July 2026.
14. I. Somerville, Software Engineering, 10th ed. Boston, MA: Pearson, 2016.