Saaradaa Journal of Interdisciplinary Studies
(A Unit of Saaradaa Learknowations Pvt Ltd)
2-3-78, Rd No 5, Cooperative Bank Colony, 500068, Hyderabad, India
www.journal.e2eindia.org www.e2eindia.org

Online Exam Portal to Manage Monthly Exams and Assignments Between Different Schools

Kalamanda Karthik Kumar
Web Developer, Saaradaa Learknowations Pvt. Ltd.
Email: karthikk7569@gmail.com

Abstract:

This paper presents a production-deployed, multi-tenant Online Exam Portal enabling monthly examinations and assignments across different schools. The system delivers secure JWT authentication, role-based access control, CSV-driven student and question onboarding (with image attachments), strict exam time windows, automatic scoring for MCQs, an explicit results-release workflow, and analytics exports. The frontend is React (MUI, framer-motion, react-slick), the backend Flask with SQLAlchemy over MySQL (AWS RDS), and assets on AWS S3. Both tiers are deployed on Render. We detail architecture, implementation highlights (pooling, UTC-safe time checks, idempotent assignment), and UI/UX components. Finally, we outline a forward path to Java Spring Boot, school-managed tenancy, and a class-/subject-based question repository to enhance productivity and scale.

Key words: *Online examinations, multi-tenant LMS, React, Flask, MySQL, AWS S3/RDS, Render, JWT, CSV import, Spring Boot*

I. Introduction

Monthly, cross-school assessments demand reliable scheduling, bulk onboarding, strict timing, and transparent results. Off-the-shelf tools often under-serve multi-institution orchestration or require heavy customization. We developed and deployed a full-stack Online Exam Portal (public URL: https://sl-exam.onrender.com/) to streamline these workflows with an API-first backend and a modern SPA frontend.

Contributions: (1) Cloud-native architecture combining React + Flask + MySQL, operationalized on Render and AWS (RDS/S3). (2) End-to-end exam lifecycle with UTC-safe windows, auto-scoring, results-release gating, and audit-friendly exports. (3) Operator efficiency features: CSV imports (students, questions) with optional image bundles, exam cloning, school/class assignment. (4) Future-ready path to Spring Boot, school self-management, and a curated question repository.

In developing and deploying the Online Exam Portal, my primary objective was to address the operational challenges associated with conducting recurring, cross-school assessments in a scalable and secure manner. Existing Learning Management Systems often emphasize content delivery and require extensive customization to support multi-institution examination workflows. This system was therefore designed with an **assessment-first architecture**, prioritizing reliability, efficiency, and administrative control.

The separation of the **React-based single-page frontend** from the **Flask REST API backend** enabled independent development and deployment while supporting horizontal scalability. I adopted **stateless JWT-based authentication** to enforce role-based access control without introducing server-side session overhead, which is particularly critical during peak exam periods. To ensure temporal integrity, all exam access windows were normalized using **UTC-safe time checks**, thereby eliminating inconsistencies arising from client-side clock variations or regional time differences.

Administrative efficiency was a central design consideration. The implementation of **CSV-driven onboarding** for students and questions significantly reduced manual effort and

error rates in managing monthly examinations across multiple schools. Additionally, the explicit **results-release mechanism** allowed administrators to control result visibility, improving transparency and preventing premature disclosure. This design choice aligns with institutional requirements for auditability and fairness in evaluation.

From a data management perspective, the use of **AWS RDS (MySQL)** for transactional data and **AWS S3** for large assets provided a balance between consistency and scalability. Connection pooling and pre-ping strategies improved backend stability under concurrent load. While the current implementation focuses on **MCQ-based assessments** for automated scoring, this represents a deliberate trade-off between grading efficiency and assessment diversity.

Overall, the system demonstrates that a lightweight, cloud-native platform tailored to assessment workflows can effectively support multi-school examination needs. The planned migration to **Spring Boot,** along with school-managed tenancy and a structured question repository, represents a natural progression toward greater scalability, autonomy, and pedagogical flexibility.

II. Related Work

Learning platforms and LMS add-ons provide exam modules, yet multi-school tenancy, CSV-first onboarding, and lightweight API control can be limited. Literature and industry practice support stateless, token-based auth, object storage for large artifacts, and relational stores for transactional integrity in e-learning systems. Our work targets repeated, monthly assessments across schools with minimal ops and clear programmatic control.

Saaradaa Journal of Interdisciplinary Studies
(A Unit of Saaradaa Learknowations Pvt Ltd)
2-3-78, Rd No 5, Cooperative Bank Colony, 500068, Hyderabad, India
www.journal.e2eindia.org www.e2eindia.org

III. System Overview

A. Actors and Use Cases — Admins manage schools; import students; create/clone exams; upload questions (single or CSV+images); assign to schools/classes; export attempts; release results. Students log in (student ID), view assigned exams, start within access window, answer MCQs, submit; view results post-release.

B. Frontend (React/MUI) — UI stack: React, Material UI, framer-motion (micro-animations), react-slick (hero carousel). Home UX: news marquee, modal overlays (results prompt, login CTA), sections for Thinklets/Books/Riddle, and quick anchors. Routing: SPA navigation to login and exam areas. Accessibility: responsive layout, keyboard-focusable buttons, readable typographic scale.

IV. Backend Architecture

A. Technology & Deployment — Flask REST API; SQLAlchemy ORM; MySQL on AWS RDS; assets (images/CSV) on AWS S3; hosting on Render for both tiers; environment-based configuration.

B. Authentication & Authorization — JWT (HS256) with sub, role, iat, exp (~6h). Guards: token_required for authenticated routes; admin_required for privileged endpoints. CORS allowlist covers localhost and production domain; credentials enabled.

C. Data Model — User, School, Student, Exam, Question, ExamStudent, StudentExamAttempt, StudentAnswer. Student ID generation uses school metadata + class + number; username flips to generated ID post-create. Exam assignment via link-table; idempotent append (skip existing), replace option to reset cohorts.

Saaradaa Journal of Interdisciplinary Studies
(A Unit of Saaradaa Learknowations Pvt Ltd)
2-3-78, Rd No 5, Cooperative Bank Colony, 500068, Hyderabad, India
www.journal.e2eindia.org www.e2eindia.org

D. Reliability & Performance — Connection pooling (pool_recycle=280, pool_pre_ping=True, pool_size=10, max_overflow=20, pool_timeout=10) mitigates stale connections and absorbs bursts. UTC-safe timing helper normalizes access_start/access_end. Stateless JWT and S3/RDS enable horizontal scalability.

V. Exam Lifecycle and APIs

A. Authoring & Onboarding — Schools CRUD; students CSV import and public self-register (generates student ID); exams create/clone; questions single or bulk CSV with optional images.

B. Assignment & Delivery — Assign by school_id/class_number with append or replace. Student flow: /student/exams, /can_start (window enforcement), /start (returns expires_at), /questions (MCQs + marks/images), /submit (auto-score), /result (visible after results_released).

C. Analytics & Export — Excel export stream for audits and offline analysis.

VI. Implementation Highlights

Robust CSV imports with optional images mapped to S3; nested transactions. Idempotent cohort management avoids duplicate link rows. Clear results governance by explicit results_released. Error handling and logging on critical paths (start/submit checks, exports).

VII. Evaluation

Functional Outcomes — Intuitive student flows, reliable time-window enforcement, monthly reuse via exam cloning, and low-ops onboarding via CSV. UI/UX Outcomes — React

home surfaces announcements, engagement content, and login CTAs. Operational Notes — Pooling and pre-ping stabilize connections; stateless JWT and S3 artifacts keep API scalable.

VIII. Limitations

MCQ-first assessment; subjective questions need grader tooling. No proctoring/anti-cheating yet. Token lifecycle lacks refresh/rotation. Limited real-time analytics (batch export over dashboards).

IX. Future Work

1) Java Spring Boot migration (Spring Security, Spring Data JPA) for productivity and concurrency performance.

2) Schools as tenant admins: self-service multi-tenancy with scoped data access.

3) Question repository cataloged by class/subject/chapter/difficulty/tags; randomization and blueprints.

4) Integrity & proctoring (focus/blur, shuffling, webcam, plagiarism).

5) Dashboards & alerts (attempt progress, completion heatmaps; email/SMS/web-push).

6) Security hardening (rate limits, refresh tokens, rotation, device binding, audit logs).

7) Mobile apps with offline-safe queues.

X. Conclusion

The portal operationalizes a practical, secure, and reusable workflow for monthly, cross-school assessments with a modern web stack and cloud-native deployment. The Spring Boot migration, tenant-admin model for schools, and a structured question repository will extend scalability, autonomy, and exam quality control.

Acknowledgment

The author thanks ***Saaradaa Learknowations Pvt. Ltd.*** for infrastructure, organizational support, and feedback during development and deployment.

References

[1] Best practices for JWT-based stateless authentication in web systems.

[2] SQLAlchemy Engine & Pooling Patterns; AWS RDS connectivity considerations.

[3] Cloud object storage patterns for web applications (AWS S3).

[4] UI patterns for engagement in educational portals (SPA/React ecosystems).

[5] Literature on proctoring and integrity in online examinations (surveys and practice reports).

Bibliography / References

1. D. Merrill, "First Principles of Instruction," *Educational Technology Research and Development*, vol. 50, no. 3, pp. 43–59, 2002.
2. A. Al-Hmouz, J. Shen, R. Al-Hmouz, and J. Yan, "Modeling and Simulation of an Online Examination System," *Computers & Education*, vol. 58, no. 1, pp. 1–10, 2012.
3. R. Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, University of California, Irvine, 2000.
4. M. Fowler, *Patterns of Enterprise Application Architecture*. Boston, MA: Addison-Wesley, 2002.
5. JSON Web Token (JWT), "RFC 7519," Internet Engineering Task Force (IETF), 2015.
   Available: https://datatracker.ietf.org/doc/html/rfc7519
6. Amazon Web Services, "Amazon S3 – Object Storage Service Overview," AWS Documentation, 2023.
   Available: https://docs.aws.amazon.com/s3/

7. Amazon Web Services, "Amazon RDS – Relational Database Service," AWS Documentation, 2023. Available: https://docs.aws.amazon.com/rds/
8. M. Kleppmann, *Designing Data-Intensive Applications*. Sebastopol, CA: O'Reilly Media, 2017.
9. S. Newman, *Building Microservices: Designing Fine-Grained Systems*. Sebastopol, CA: O'Reilly Media, 2015.
10. J. Grinberg, "Flask Web Development: Developing Web Applications with Python," 2nd ed., Sebastopol, CA: O'Reilly Media, 2018.
11. Facebook Inc., "React: A JavaScript Library for Building User Interfaces," React Documentation, 2023. Available: https://react.dev/
12. P. Hunt, "Material UI: React Components for Faster and Easier Web Development," MUI Documentation, 2023. Available: https://mui.com/
13. V. Peinelt et al., "Security Considerations for Token-Based Authentication Systems," *IEEE Security & Privacy*, vol. 18, no. 4, pp. 72–79, 2020.
14. J. Bergström and K. Höök, "Prototyping with UX in Mind: User Experience Design for Interactive Systems," *Human–Computer Interaction*, vol. 28, no. 3, pp. 237–270, 2013.
15. S. Dhawan, "Online Learning: A Panacea in the Time of COVID-19 Crisis," *Journal of Educational Technology Systems*, vol. 49, no. 1, pp. 5–22, 2020.

Appendix A — Representative Pseudocode/Flow (Selected)

```
Exam Start Window Check (UTC-safe)
now_utc = utcnow()
if exam.access_start and now_utc < to_utc(exam.access_start): deny()
if exam.access_end and now_utc > to_utc(exam.access_end): deny()
attempt = get_or_create_attempt(student, exam, start_time=now_utc)
expires = min(attempt.start_time + duration, to_utc(exam.access_end) if set)
return {attempt_id, start_time, expires}
```

```
Auto-Score on Submit
score = 0
for (qid, resp) in answers:
    q = load_question(qid)
    correct = (upper(resp) == upper(q.correct_answer))
    score += q.marks if correct else 0
    persist(StudentAnswer(...))
persist(StudentExamAttempt.submitted_time=now, score=score)
```

Author Profile

Kalamanda Karthik Kumar, Web Developer at Saaradaa Learknowations Pvt. Ltd., focuses on full-stack systems, cloud deployment, and education technology platforms.