



(REVIEW ARTICLE)



DESIGN AND IMPLEMENTATION OF A ROLE-BASED WEB ASSESSMENT PLATFORM FOR NIGERIAN UNIVERSITIES USING THE LARAVEL MVC FRAMEWORK

Innocent Adjaino and Omolegho A. Ibok *

Department of Computer Engineering, Southern Delta University, Ozoro, Delta State, Nigeria.

* Corresponding Author

ORCID Details

Omolegho A. Ibok: <https://orcid.org/0009-0008-8838-5721>

Global Journal of Research in Engineering and Technology, 2026, 04(02), 028–036

Publication history: Received on 15 September 2026; revised on 20 October 2026; accepted on 23 October 2026

Article DOI: <https://doi.org/10.58175/gjret.2026.4.2.0017>

Copyright © 2026 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0

ABSTRACT

In many Nigerian universities, assessment still means printed scripts, long marking weekends and results that reach students weeks after the examination. This paper describes the design and implementation of a web-based student assessment system built to take that burden off lecturers while keeping the process fair and traceable. The system uses Laravel 10, MySQL 8.0 and Bootstrap 5 in a three-tier Model-View-Controller (MVC) architecture, and was developed iteratively following an Agile process. Five modules were implemented: authentication and role management, course and assessment management, a reusable question bank, timed online test delivery with automated grading, and result generation with performance reporting. Administrators, instructors and students each see only what their role permits. The system was verified with 30 structured black-box test cases covering all modules, and every case passed. A pilot with 30 users (5 administrators, 10 instructors, 15 students) returned an overall mean acceptance rating of 4.69 out of 5. The platform runs entirely on open-source components and ordinary hardware, which makes it realistic for institutions with limited budgets. We describe the design decisions, report what worked, and discuss the limits of the current version, including the absence of automated grading for open-ended answers and of proctoring features.

Keywords: Web-Based Assessment; Online Examination; Laravel; Role-Based Access Control; Automated Grading; Educational Technology; Nigeria

1 INTRODUCTION

Anyone who has marked a few hundred scripts by hand knows the problem. The work is slow, it is tiring, and tiredness invites small errors in addition, transcription and judgement. Students, meanwhile, wait for feedback that often arrives too late to change how they study. Assessment is meant to guide learning, not only to record it, and feedback is among the most powerful influences on achievement (Black & Wiliam, 1998; Hattie & Timperley, 2007).

Digital assessment has grown steadily in higher education. A systematic review of online assessment identified teacher, peer, automated and self-assessment as the main modes that digital tools can support, and noted that the shift

accelerated during the COVID-19 disruption (Heil & Ifenthaler, 2023). The sudden move to emergency remote teaching also showed how uneven institutional readiness was (Hodges et al., 2020; Adedoyin & Soykan, 2020). Reviews of online examination solutions point to a recurring gap between what the technology can do and what institutions in developing regions actually use, often because of cost, infrastructure and limited technical capacity (Muzaffar et al., 2021).

Security and integrity add a second layer of difficulty. Garg and Goel (2022) show in their systematic review that assessment security remains a persistent concern and that access control is among the key defensive dimensions. A system that handles grades must therefore be convenient and trustworthy at the same time.

This paper reports the design and implementation of a web-based assessment system aimed at Nigerian universities. Our contribution is practical and specific:

- A documented reference architecture for a multi-role assessment platform built entirely on open-source tools;
- A data model and module structure that supports question reuse, randomized test delivery, timed sessions and automatic grading of objective items;
- Evidence from structured functional testing and a pilot user evaluation that the implementation works as intended in a realistic academic setting.

2 BACKGROUND AND RELATED WORK

2.1 Why objective assessment suits automation

Multiple-choice and true/false items are quick to administer and can be scored without human judgment, which removes marker fatigue and inconsistency for that part of an assessment. Their quality depends heavily on how the items are written (Haladyna et al., 2002), so a system should make it easy for lecturers to store, review and reuse good questions rather than rewrite them each semester. This observation motivated the persistent question bank in our design.

2.2 Online examination platforms

The literature describes many online examination systems, from locally built tools to large learning management systems. Muzaffar et al. (2021) review techniques, tools and adoption patterns and conclude that solutions differ widely in features, security measures and reported evaluation. A recurring weakness is thin evidence: many systems are described but not tested in a structured way, and quantitative performance data are uncommon. We therefore treat testing as part of the design, not an afterthought.

2.3 Architecture and access control

The Model-View-Controller pattern separates data and business rules (model), presentation (view) and request handling (controller), and has been a staple of interactive software since its early formulation (Krasner & Pope, 1988). Separating these concerns makes systems easier to maintain and extend. For access control, role-based models assign permissions to roles rather than to individuals, which fits institutions where people hold well-defined positions (Sandhu et al., 1996). In an assessment setting, three roles are natural: administrator, instructor and student.

2.4 Agile development

Agile methods favour short iterations, working software and continuous feedback (Beck et al., 2001). Mishra and Alzoubi (2023) compare structured and agile approaches and discuss why iterative development tends to cope better with changing requirements. That suited our situation, where requirements from lecturers and administrators were refined as prototypes were shown to them.

3 METHODOLOGY

3.1 Development process

The iterative Agile process was followed in five phases:

- Requirements gathering: Structured interviews with prospective administrators, instructors and students, plus a review of current assessment practice, produced the functional and non-functional requirements and the use cases.
- System design: We produced the architecture, database schema, use case, activity and entity-relationship diagrams.

- Implementation: Core modules were built first, then secondary features. Each module was unit-tested before integration.
- Testing and evaluation: Black-box testing was followed by user acceptance testing.
- Deployment and review: The system was deployed to a controlled server environment, and feedback shaped final refinements.

3.2 Technology stack

The stack was chosen for cost, maturity and community support. Development used Ubuntu 22.04 LTS and Windows 11 workstations, Apache HTTP Server, PHP 8.1, Laravel 10, MySQL 8.0, HTML5, CSS3, Bootstrap 5 and JavaScript. Visual Studio Code and the Laragon local stack supported day-to-day work, and Git with GitHub handled version control. The development machine was a laptop with an Intel Core i5 processor, 8 GB of RAM and 500 GB of storage. Nothing in this stack requires paid licences, which matters for institutions with tight budgets.

3.3 Evaluation approach

Two complementary methods were used. Black-box testing checked that each function behaves correctly from the user's perspective, without reference to internal code (Nidhra & Dondeti, 2012). Each test case recorded a scenario, expected output, actual output and a pass/fail verdict. User acceptance testing then asked real users to complete role-specific tasks and rate the system on a five-point scale across functionality, usability, reliability and overall satisfaction.

4 SYSTEM DESIGN

4.1 Architecture

The system follows a three-tier architecture. The presentation layer is a responsive Bootstrap interface. The application layer is a Laravel MVC application that handles routing, validation, authentication and business logic. The data layer is a MySQL relational database holding users, courses, questions, assessments, submissions and activity logs.

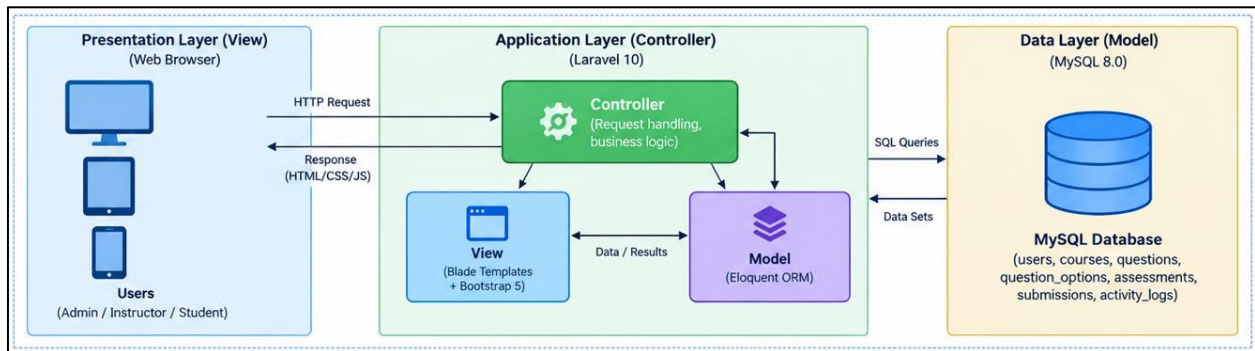


Figure 1: Three-tier architecture diagram of the system

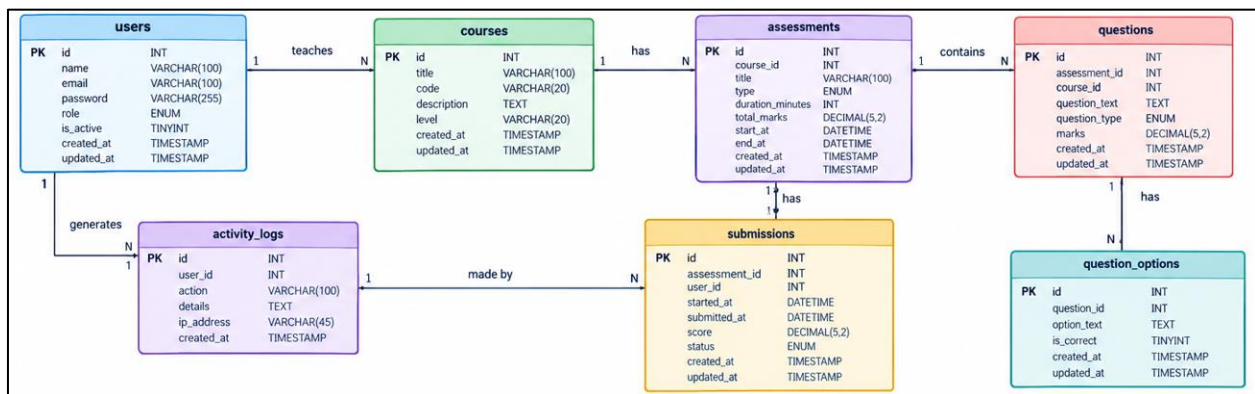


Figure 2: Entity-relationship diagram of the database schema (users, courses, questions, question options, assessments, submissions, activity logs)

4.2 Roles

Every account carries one role. Administrators manage users and courses and view institution-wide reports. Instructors create courses, build question banks and assessments, and view class analytics. Students take assessments assigned to them and see their own results. Role checks are enforced by middleware on every protected route, so a student who types an administrator URL is turned away at the server and not merely hidden from a menu.

4.3 Modules

- Module 1: Authentication and role management. Handles registration, login, session handling and role assignment. Passwords are hashed with Bcrypt before storage, so plaintext credentials are never kept (Provos & Mazières, 1999).
- Module 2: Course and assessment management. Instructors create courses and attach assessments with a duration, an availability window and a maximum number of attempts. Administrators can oversee all courses from one place.
- Module 3: Question bank. Instructors build a persistent bank of multiple-choice and true/false questions, edit and categorise them, and either pick items manually or switch on randomisation so each student receives a different arrangement. Reuse is the point: a good question written once keeps paying off.
- Module 4: Online test delivery and automated grading. Students open an assessment only inside its time window. A countdown timer runs during the session, and if it reaches zero the recorded answers are submitted automatically. Objective responses are marked instantly, and unanswered items score zero.
- Module 5: Results and reporting. Students see a score summary with a question-by-question breakdown. Instructors see class average, highest and lowest scores, a score distribution and average-score trends across assessments, and can export results to CSV. Administrators see institution-wide summaries that support audit and accreditation reporting.

4.4 Security by design

Security features were built into the framework layer rather than bolted on. Laravel's authentication scaffolding manages sessions and expiry; CSRF tokens protect state-changing requests; Eloquent and the query builder issue parameterized queries; Blade escapes output by default; and server-side validation checks all inputs. The security behaviour of the system is evaluated in more detail in a companion paper.

5 IMPLEMENTATION AND RESULTS

5.1 User interface

The interfaces are deliberately plain. The login page is a single centred card with email and password fields, and after authentication each user is routed to a dashboard matched to their role. Administrators see summary cards for users, courses, assessments and submissions together with a six-month chart of assessment activity. Instructors see their courses in a table with view and delete actions and a button to create new ones. During a test, students see a numbered question navigator (answered items in green, unanswered in grey), one question at a time, and a timer that turns red in the final five minutes.

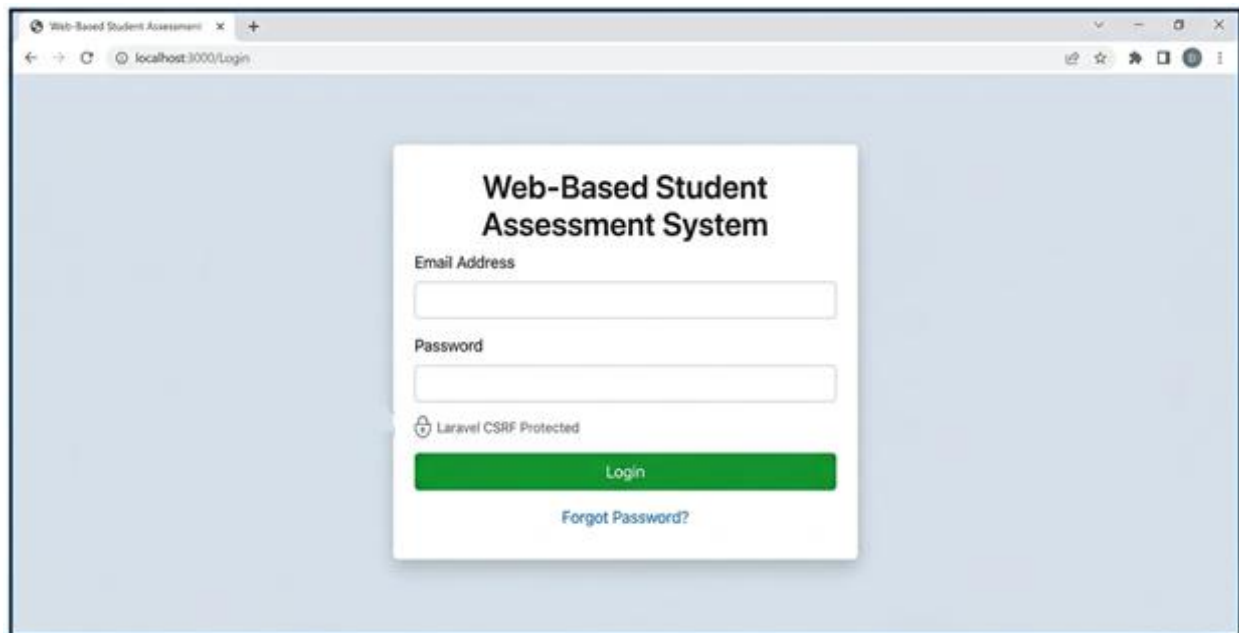


Figure 3: Screenshot of Student Assessment System Login Page

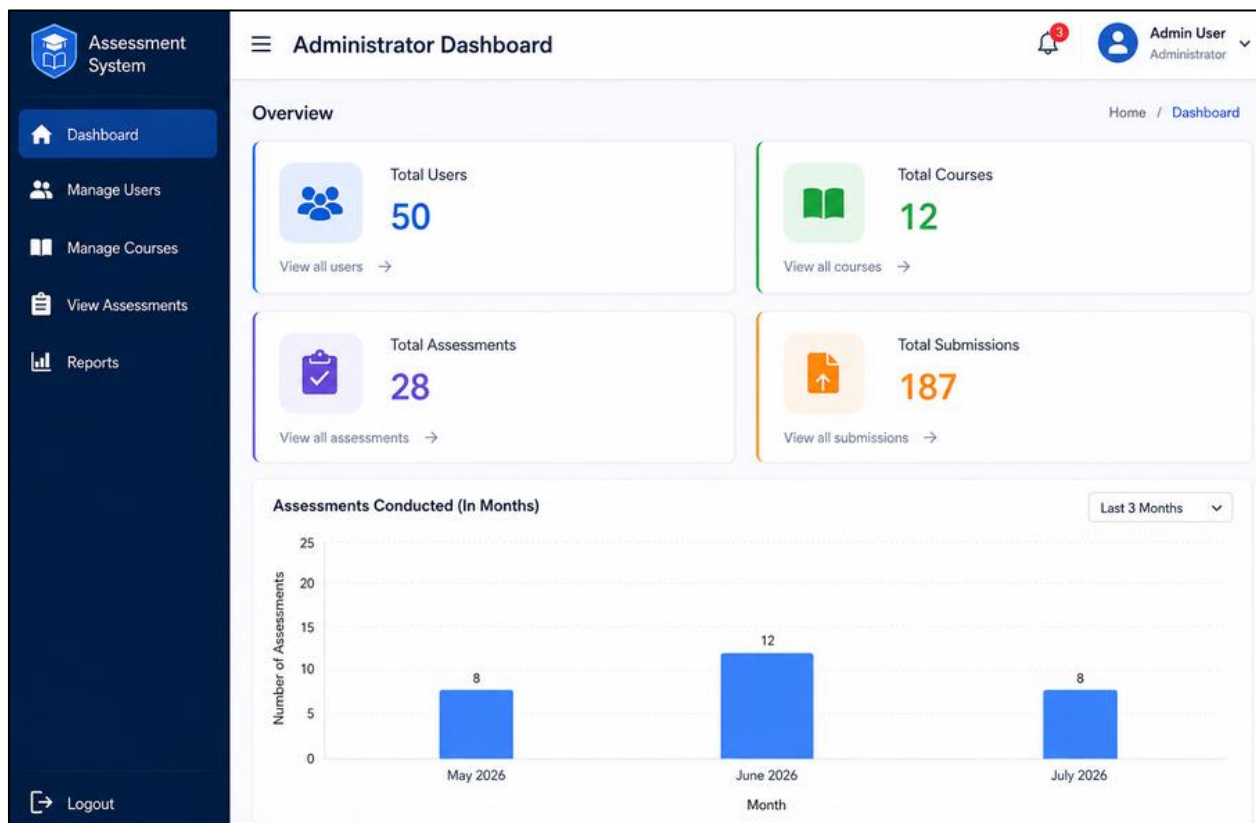


Figure 4: Screenshot of Administrator Dashboard

The screenshot displays the 'Instructor Dashboard' for an 'Assessment System'. The left sidebar contains navigation links: Dashboard, My Courses, Create Assessment, Question Bank, Reports, and Logout. The main content area is titled 'My Courses' and includes a 'Create New Course' button. Below this is a table listing five courses with their IDs, names, number of assessments, creation dates, and actions (View/Delete).

Course ID	Course Name	Number of Assessments	Date Created	Actions
CPE 311	Computer Organization and Architecture	6	May 10, 2026	View Delete
CPE 312	Digital Electronic Circuit	4	May 12, 2026	View Delete
CPE 313	Measurement and Instrumentation	5	May 14, 2026	View Delete
SDU-CPE 314	Numerical Computation Analysis I	3	May 16, 2026	View Delete
SDU-CPE 315	Computer Laboratory I	4	May 18, 2026	View Delete

Showing 1 to 5 of 5 courses

Figure 5: Instructor Dashboard and Course Management Interface

The screenshot shows the 'Online Test Interface (Student View)' for the course 'CPE 312: Digital Electronic Circuit — continuous assessment'. The left sidebar has links: Dashboard, My courses, Take assessment (highlighted), Question bank, Reports, and Logout. The main area displays the test title, time remaining (00:04:45), and two questions. Question 15 is a multiple-choice question about logic gates, and Question 16 is a true/false question about flip-flops.

CPE 312: Digital Electronic Circuit — continuous assessment
Time remaining: 00:04:45

Question 15 of 25
Which logic gate produces a HIGH output only when all its inputs are HIGH?

☐ A. OR gate
☐ B. NOR gate
☐ C. AND gate
☐ D. XOR gate

Question 16 of 25
A flip-flop is a sequential circuit element that can store one bit of data.

☐ True
☐ False

[Next question](#) [Submit assessment](#)

Figure 6: Online Test Interface (Student View)

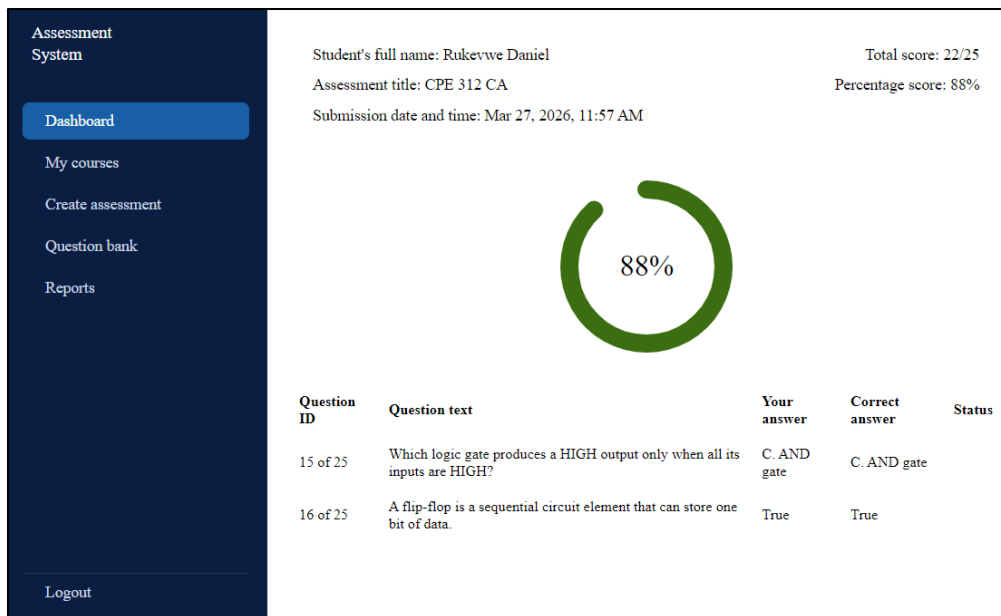


Figure 7: Automated Result Display (Student View)

5.2 Functional verification

Thirty black-box test cases were executed across the system. Table 1 groups them by area. All 30 passed.

Table 1: Black-box test cases by functional area

Functional area	Test cases	Representative scenarios	Passed
Authentication and access control	5	Valid registration; wrong password; student attempting admin or instructor routes; session expiry	5
Course management	3	Create course; delete course with its assessments; admin views all courses	3
Question bank	4	Add MCQ; add true/false; edit; delete	4
Assessment management	4	Timed assessment creation; randomization; access before start or after end	4
Test administration	4	Begin test; submit before timer ends; automatic submission at zero; attempts exceeded	4
Automated grading	4	Correct and incorrect MCQ; true/false; unanswered item	4
Result generation	1	Score, percentage and answer breakdown shown	1
Performance reporting	3	Class report; institution-wide report; CSV export	3
Security	2	SQL injection attempt; script injection attempt	2
Total	30		30 (100%)

The tests that matter most in practice are the boundary ones: access one minute before opening, access after closing, a further attempt beyond the permitted limit, and a timer that expires mid-answer. In each case the system behaved as specified.

5.3 User acceptance

Thirty participants from the target institution took part: 5 administrators, 10 instructors and 15 students. Each completed role-specific tasks and then rated eight statements on a five-point agreement scale covering four dimensions. Table 2 shows the mean ratings.

Table 2: Mean user acceptance ratings (out of 5)

Dimension	Administrators (n=5)	Instructors (n=10)	Students (n=15)	Overall
Functionality	4.80	4.70	4.60	4.68
Usability	4.60	4.50	4.73	4.63
Reliability	4.80	4.60	4.67	4.67
Overall satisfaction	4.80	4.70	4.80	4.78
Overall mean	4.75	4.63	4.70	4.69

All three groups rated the system highly and fairly consistently. Instructors gave the lowest usability score (4.50), which is unsurprising since they use the most complex screens, and is a pointer for future interface refinement. A fuller analysis of acceptance and performance is reported in a companion paper.

6 DISCUSSION

Three design decisions are worth highlighting for anyone building something similar.

Lean on the framework: Because Laravel already provides routing, authentication, validation and templating, development effort went into assessment logic instead of plumbing. This keeps the codebase small enough for a university department to maintain, which is often the real test of whether a system survives past its first semester.

Enforce roles on the server: Hiding a button is not access control. Putting the role check in middleware means that every route is protected regardless of how the user arrives at it.

Design for ordinary hardware: The whole system was built and evaluated on commodity equipment with free software. That is a deliberate response to the cost barriers highlighted in the literature on online examinations in developing regions (Muzaffar et al., 2021).

For lecturers, the practical gains are immediate scoring, no hand-tallying and ready-made class statistics. For students, they are faster feedback and a transparent breakdown of every answer, both of which support learning (Hattie & Timperley, 2007). For administrators, they are consistent records and exportable reports.

Limitations and Future Work

The results should be read with these limits in mind. First, the evaluation took place in a controlled academic setting with 30 pilot users, so large multi-institution deployments may reveal scaling and interoperability issues. Second, automated grading covers only multiple-choice and true/false items; open-ended answers still need a human marker. Third, the system depends on dependable internet or intranet access at the point of use, which is not guaranteed everywhere. Fourth, proctoring tools such as browser lockdown and webcam monitoring are not included, and the effectiveness of such tools in practice is itself debated (Dendir & Maxwell, 2020).

Future work will focus on natural-language-processing support for short-answer grading, integration with learning management systems, adaptive testing, a native mobile application and cloud deployment for scalability.

7 CONCLUSION

This paper presented a role-based web assessment platform that automates test administration, grading and reporting using freely available technology. The Laravel MVC implementation passed all 30 functional test cases and earned a mean acceptance rating of 4.69 out of 5 from 30 pilot users. The work offers universities, particularly those with limited budgets, a documented and tested starting point for moving assessment online, and it identifies clear next steps for extending it.

STATEMENTS ON COMPLIANCE WITH ETHICAL STANDARDS

Acknowledgments

The authors sincerely appreciate the Department of Computer Engineering, Southern Delta University, for its academic support, guidance, and encouragement throughout the preparation of this paper. Special appreciation goes to the lecturers and colleagues whose contributions and constructive suggestions helped improve the quality of this work.

Disclosure of conflict of interest

The authors declare no conflict of interest.

REFERENCES

- [1] Adedoyin, O. B., & Soykan, E. (2020). Covid-19 pandemic and online learning: The challenges and opportunities. *Interactive Learning Environments*, 31(2), 863–875. <https://doi.org/10.1080/10494820.2020.1813180>
- [2] Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). *Manifesto for Agile Software Development*. <https://agilemanifesto.org>
- [3] Black, P., & Wiliam, D. (1998). Assessment and classroom learning. *Assessment in Education: Principles, Policy & Practice*, 5(1), 7–74. <https://doi.org/10.1080/0969595980050102>
- [4] Dendir, S., & Maxwell, R. S. (2020). Cheating in online courses: Evidence from online proctoring and lockdown browser. *Computers in Human Behavior Reports*, 2, 100033. <https://doi.org/10.1016/j.chbr.2020.100033>
- [5] Garg, M., & Goel, A. (2022). A systematic literature review on online assessment security: Current challenges and integrity strategies. *Computers & Security*, 113, 102544. <https://doi.org/10.1016/j.cose.2021.102544>
- [6] Haladyna, T. M., Downing, S. M., & Rodriguez, M. C. (2002). A review of multiple-choice item-writing guidelines for classroom assessment. *Applied Measurement in Education*, 15(3), 309–334. https://doi.org/10.1207/S15324818AME1503_5
- [7] Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of Educational Research*, 77(1), 81–112. <https://doi.org/10.3102/003465430298487>
- [8] Heil, J., & Ifenthaler, D. (2023). Online assessment in higher education: A systematic review. *Online Learning*, 27(1), 187–218. <https://doi.org/10.24059/olj.v27i1.3398>
- [9] Hodges, C., Moore, S., Lockee, B., Trust, T., & Bond, A. (2020). The difference between emergency remote teaching and online learning. *EDUCAUSE Review*, March 27, 2020. <https://er.educause.edu/articles/2020/3/the-difference-between-emergency-remote-teaching-and-online-learning>
- [10] Krasner, G. E., & Pope, S. T. (1988). A cookbook for using the model-view-controller user interface paradigm in Smalltalk-80. *Journal of Object-Oriented Programming*, 1(3), 26–49.
- [11] Mishra, A., & Alzoubi, Y. I. (2023). Structured software development versus agile software development: A comparative analysis. *International Journal of System Assurance Engineering and Management*, 14(4), 1504–1522. <https://doi.org/10.1007/s13198-023-01958-5>
- [12] Muzaffar, A. W., Tahir, M., Anwar, M. W., Chaudry, Q., Mir, S. R., & Rasheed, Y. (2021). A systematic review of online exams solutions in e-learning: Techniques, tools, and global adoption. *IEEE Access*, 9, 32689–32712. <https://doi.org/10.1109/ACCESS.2021.3060192>
- [13] Nidhra, S., & Dondeti, J. (2012). Black box and white box testing techniques – A literature review. *International Journal of Embedded Systems and Applications*, 2(2), 29–50.
- [14] Provos, N., & Mazières, D. (1999). A future-adaptable password scheme. In *Proceedings of the USENIX Annual Technical Conference* (pp. 81–91). USENIX Association.
- [15] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., & Youman, C. E. (1996). Role-based access control models. *Computer*, 29(2), 38–47. <https://doi.org/10.1109/2.485845>