

To cite this article: Dominic Dave B. Balino (2026). Development And Functional Testing Of A Web-Based Academic Grade Tracking Application. International Journal of Education, Business and Economics Research (IJEBER) 6 (5): 341-353

DEVELOPMENT AND FUNCTIONAL TESTING OF A WEB-BASED ACADEMIC GRADE TRACKING APPLICATION

Dominic Dave B. Balino

Mariners' Polytechnic Colleges Foundation,

Legazpi City, Albay, Philippines

Corresponding author: Dominic Dave B. Balino

<https://doi.org/10.59822/IJEBER.2026.6519>

ABSTRACT

This study reports the development and functional testing of a web-based academic grade tracking application designed to integrate core administrative, faculty, and student functions. The application was developed using a structured Waterfall process comprising requirements, analysis, design, coding, testing, operation, and maintenance. PHP, HTML, CSS, JavaScript, and XAMPP/php My Admin were used in implementation, with supporting authentication and web-access components. Functional verification was performed by the developer using dummy accounts and 28 predefined test cases covering account registration and authentication, profile management, administrative management, enrollment, grade encoding, grade viewing, and device access. All 28 test cases were recorded as successful under controlled testing conditions, yielding a 100% functional pass rate for the specified scenarios. The results indicate that the prototype executed its defined core functions in the tested environment. They do not establish usability, user acceptance, institutional effectiveness, long-term reliability, accessibility compliance, production-scale performance, or comprehensive cybersecurity assurance. Further independent and real-world evaluation is required before production use.

KEYWORDS: Web-based grade tracking, Academic information systems, Functional testing, Educational technology.

© The Authors 2026
Published Online: October
2026

Published by International Journal of Education, Business and Economics Research (IJEBER) (<https://ijeber.com/>) This article is published under the Creative Commons Attribution (CC BY 4.0) license. Anyone may reproduce, distribute, translate and create derivative works of this article (for both commercial and non-commercial purposes), subject to full attribution to the original publication and authors. The full terms of this license may be seen at: <http://creativecommons.org/licences/by/4.0/legalcode>

1.0 INTRODUCTION

The increasing use of information and communication technologies in education has expanded the role of web-based applications beyond instructional delivery to include academic administration and information management. Educational institutions increasingly depend on digital systems to

organize student information, communicate academic information, and support administrative processes. Technology-supported academic systems can improve the accessibility and management of institutional information by reducing reliance on fragmented paper-based processes and isolated digital records. Previous work on automated student academic-record management has likewise identified the importance of systematic, centralized handling of student information and grades [1]. Academic grades constitute an important component of institutional information management because they must be recorded, processed, stored, and communicated accurately to authorized users. Grade-management activities may involve several related processes, including student identification, enrollment, faculty grade encoding, academic-record storage, and student access to academic results. When these activities are handled through disconnected procedures, maintaining consistent information across administrative and academic functions can become difficult. Web applications provide an approach for integrating related functions through browser-accessible interfaces while allowing information to be maintained in a centralized database environment.

Web applications are particularly suitable for systems that require access from different user roles and devices. Unlike applications that require installation on individual computers, browser-based systems can provide a common interface through which authorized users can interact with centralized services. In educational settings, this architecture can support administrative information systems while allowing institutions to customize functions according to their operational requirements. Previous work on digital grading has similarly shown how software-supported workflows can automate repetitive grading activities and maintain digital records [2].

The development of an academic grade tracking application requires more than the creation of individual software pages. It requires the integration of authentication, role-based access, data management, enrollment-related functions, grade encoding, grade viewing, and administrative controls into a coherent system. The application must also provide appropriate mechanisms for protecting user accounts and restricting functions according to account roles. In the developed system, three principal account types were implemented: administrator, faculty member, and student. Each account type was provided with functions appropriate to its role within the academic information workflow.

The development process was structured using the Waterfall software-development model. The model used in the study consisted of seven sequential phases: requirements, analysis, design, coding, testing, operation, and maintenance. The sequential structure provided a defined progression from system requirements and technical design to implementation and functional verification.

The application was implemented using commonly available web-development technologies, including PHP, HTML, CSS, and JavaScript, with XAMPP and phpMyAdmin used for server and database management. Composer and PHP Mailer were incorporated into authentication-related processes, while Cloud flare Tunnel and a registered domain were used to provide an alternative means of remote access to the application [3]. The system was designed to operate through a web browser and could also be configured for access through a local network environment.

An important aspect of software development is functional verification. Before a system is considered ready for further evaluation or deployment, its defined functions should be tested to determine whether they perform the intended operations. In this study, functional verification was conducted by the developer using dummy accounts and predefined test cases. The testing covered major workflows including registration, authentication, account management, administrative functions, enrollment, grade encoding, grade viewing, and access through mobile and other devices. The thesis documentation records 28 functional test cases, each marked as successful during the verification process.

The technical contribution of this study is therefore not the invention of a new programming language or underlying computing technology. The work instead demonstrates an application-development approach in which established web technologies are integrated into a single academic information system. This perspective is consistent with information-systems research emphasizing the relationship between technology capabilities and the tasks they are intended to support [4], [5]. The present study does not test task-technology fit empirically; rather, the principle provides context for the role-based organization of the application's functions.

Accordingly, this study focused on the development and functional verification of a web-based academic grade tracking application. Specifically, it aimed to develop the application's core administrative, faculty, and student functions and to determine whether the implemented functions operated successfully under predefined controlled test conditions.

The design also considered the need for differentiated privileges. Administrative users require broader management functions, faculty members require access to assigned academic records, and students require access to their own enrollment and grade information. This separation of functions reduces the need to expose the same interface and operations to all users and provides a structural basis for controlling which functions are available after authentication. The resulting architecture was implemented through separate role-specific dashboards within one web application.

Academic information systems typically combine multiple information flows rather than operating as isolated tools. In a grade-management environment, the accuracy of a displayed grade depends not only on the grade-viewing page but also on the integrity of the preceding processes for user identification, enrollment, course assignment, data storage, and grade encoding. The developed application was therefore organized as an integrated workflow in which the user account, academic program, course enrollment, faculty assignment, and grade information are related through the same application environment.

2.0 METHODS

2.1 Study design and development approach

The study employed a structured software-development approach based on the Waterfall model. The development process consisted of seven sequential phases: requirements, analysis, design, coding, testing, operation, and maintenance. The requirements phase established the functions and project specifications to be addressed by the application. The analysis phase examined system specifications and implementation constraints. During the design phase, the technical architecture

and application interfaces were developed. Coding and implementation converted the design specifications into functional software components, followed by testing and debugging. Operation and maintenance represented the subsequent stages for deployment, correction, adaptation, and future enhancement. The seven phases are summarized in Figure 1.

2.2 Development environment and technologies

The application was developed using PHP for server-side processing and HTML, CSS, and JavaScript for presentation and client-side functionality. XAMPP was used as the local server environment [6], while phpMyAdmin provided database-management functionality. Composer and PHP Mailer were incorporated into authentication-related processes. Cloud flare Tunnel and a registered domain were used to provide an Internet-access configuration for the application in addition to its local-server configuration. XAMPP was used as the development and local hosting environment, while the application architecture was designed to support local network or remote browser access.

2.3 Application architecture and functional modules

The developed application implemented three principal user roles: administrator, faculty member, and student. Role-based access was used to provide each account type with functions appropriate to its responsibilities. The administrator interface included functions for managing users, institutional information, curricula, programs, courses, teaching assignments, and enrollment-related requests. The faculty interface provided authenticated faculty members with access to assigned courses and grade-encoding functions. The student interface provided functions for profile management, enrollment, and viewing academic grades.

2.4 Authentication and access-control functions

Registration required users to provide account information, select an account type, and satisfy password requirements. Newly registered accounts were placed in a pending state until email verification was completed. Login was restricted to verified accounts with valid credentials and an appropriate account type. The application also implemented password verification for sensitive account and institutional-information changes. Session validation and role-based access controls were used to restrict unauthorized access to functions intended for particular account types. The system documentation also describes input validation, prepared statements, sanitization, and related data-handling measures.

2.5 Functional verification procedure

The predefined tests were organized around the major transactions supported by the application. The registration and authentication group examined whether an account could be created, verified, authenticated, and recovered. Profile-management tests examined whether authorized account information could be updated. Administrative tests examined the management of school information, programs, courses, teaching loads, course status, users, and user statistics. Enrollment tests examined the movement of course information from student selection to administrative approval and subsequent appearance in academic records. Grade-management tests examined the link between assigned courses, faculty grade entry, grade storage, and student grade viewing. The final tests examined interface transitions and access through mobile or other devices.

Functional verification was performed by the developer after completion of the application and before the expert-validation stage documented in the original thesis. Dummy accounts were created to test registration and login processes for different user roles. Each functional component was checked against predefined requirements, and errors identified during testing were corrected through code revisions.

The functional checklist consisted of 28 test cases covering registration, account authentication, profile management, administrative functions, enrollment, grade management, interface behavior, and access through mobile and other devices. Each test case was classified according to whether the intended function was successfully performed. The original testing record marked all 28 test cases as Successful.

2.6 Data analysis

Because the present article reports developer-performed functional verification rather than human-participant evaluation, no participant-based statistical analysis was applied. Functional results were summarized descriptively according to the number and proportion of predefined test cases that successfully executed their intended functions. The functional pass rate was calculated as the proportion of successful test cases relative to the total number of predefined test cases, multiplied by 100.

2.7 Ethical considerations

The present manuscript reports only software development and developer-performed functional verification using dummy accounts and simulated system transactions. No human participants, participant responses, expert ratings, or participant-derived research data are included in the reported analysis.

Waterfall Development Process

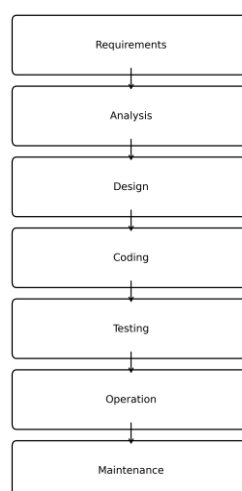


Figure 1: Waterfall development process used for the application

From a deployment perspective, the prototype supported more than one operating configuration. The documented local configuration used a Windows computer running the local server and

database environment, while networked computers could access the application through a local area network or Wi-Fi. A remote-access configuration used Cloudflare Tunnel and a registered domain. This arrangement provided an implementation path in which an institution could begin with a local deployment and later transfer the application to a cloud-hosted environment, subject to the institution's infrastructure and security requirements.

The system design also reflected the need for a continuous data path between account creation and academic information. A registered account first passed through verification before access was granted. After authentication, the available dashboard and transactions depended on the assigned account type. Academic actions then used information associated with the authenticated user, such as the student's program and enrollment or the faculty member's assigned courses. This sequence reduced the number of independent interfaces that a user needed to access for common academic transactions.

3.0 RESULTS

3.1 Developed web-based academic grade tracking application

The completed application provided integrated functions for administrator, faculty, and student users. The three-role architecture allowed the system to distinguish administrative functions from faculty grade-management functions and student academic-information functions. The administrator component provided functions for user management, institutional information, curriculum management, programs, courses, teaching assignments, and enrollment-related requests. The faculty component provided authenticated access to assigned courses and grade encoding. The student component provided profile management, enrollment, and academic grade viewing.

The application also incorporated account registration and email verification before login access. The registration workflow included input validation, password confirmation, account-type selection, and verification of the user's email address. The system database and server functions were implemented through XAMPP/phpMyAdmin. The application could operate locally through a configured Windows computer and local network and could also be configured for remote browser access using Cloudflare Tunnel and a registered domain.

3.2 Functional verification results

The functional verification covered 28 predefined test cases. These tests were grouped into six functional categories for presentation in this article.

Table 1: Functional verification results of the web-based academic grade tracking application

Functional category	Test cases	Successful	Pass rate
Registration and authentication	1-8	8/8	100%
Profile and account management	9-10	2/2	100%
Administrative management	11-19	9/9	100%
Enrollment and academic records	20-23	4/4	100%

Functional category	Test cases	Successful	Pass rate
Grade management	24-26	3/3	100%
Interface and device access	27-28	2/2	100%
Total	1-28	28/28	100%

** Functional results are based on developer-performed testing using dummy accounts and predefined test cases.*

The registration and authentication tests covered account creation, password requirements, terms-and-conditions confirmation, password confirmation, email verification, login, and password-reset functionality. All eight tests were recorded as successful. Profile and account-management tests involving information updates and avatar upload were also recorded as successful. Administrative tests covering school information, program and course management, teaching assignments, course status, user search and filtering, and user statistics were likewise successful.

The enrollment and academic-record tests verified program searching, course enrollment, administrator approval or disapproval, and the appearance of enrolled courses in the student grade interface. All four tests were marked successful. Grade-management tests verified that assigned courses appeared in the faculty grade-encoding interface, faculty members could encode and save grades, and students could view grades entered by faculty members. All three tests were successful. The final two tests evaluated interface behavior and access through mobile devices and other devices using the registered domain. Both were recorded as successful. Overall, the application achieved a 28/28 functional pass result, corresponding to a 100% functional pass rate under the predefined controlled testing conditions.

The functional pass result demonstrates that the tested application workflows operated as intended within the test environment. It should not, however, be interpreted as evidence of usability, institutional effectiveness, large-scale performance, accessibility compliance, or comprehensive cybersecurity performance. These dimensions were outside the scope of the reported functional verification. The sequence of core transactions is summarized in Figure 2.

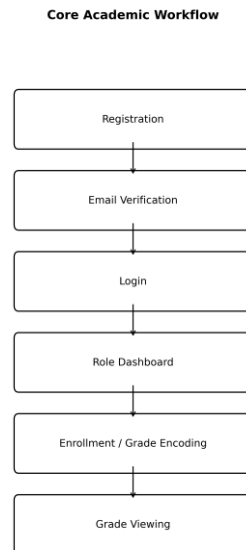


Figure 2: Core academic workflow represented by the tested application functions

4.0 DISCUSSION

The developed application demonstrates how commonly available web technologies can be combined to integrate several academic information-management functions within a single browser-based system. Rather than treating registration, enrollment, grade encoding, and grade viewing as independent activities, the application connects these functions through role-based interfaces for administrators, faculty members, and students. This architecture allows each account type to interact with functions relevant to its role while maintaining a common application and database environment. The resulting design is consistent with the broader direction of automated academic-record systems, which have been developed to centralize student records, grade submission, assessment, and related registrar functions [1].

The role-based structure is also relevant to academic information systems because different users have different responsibilities and information-access requirements. The administrator manages institutional and curricular information, faculty members manage grades associated with assigned courses, and students access enrollment and academic information. This arrangement provides a logical mechanism for separating tasks and permissions. The design is conceptually consistent with the task-technology fit perspective, which emphasizes alignment between technology capabilities and the tasks they support [4]. The present study does not empirically test task-technology fit; it uses the principle as a design rationale rather than as a measured outcome.

The functional verification results showed that all 28 predefined test cases were successfully completed under the controlled testing conditions. The tests included account registration, authentication, profile management, administrative management, enrollment, grade encoding, grade viewing, and access through mobile and other devices. The successful execution of these tests provides evidence that the implemented application workflows corresponded to the functions specified in the developer's test checklist. In particular, successful testing of the complete sequence from registration and verification to enrollment and grade viewing demonstrates that the

application's major functional components were able to operate together within the implemented environment.

The results are consistent with the role of functional testing in software development: determining whether specified operations execute according to predefined requirements. Previous work on digital grading and automated academic-record management has likewise emphasized the use of software to automate repetitive academic processes and preserve digital records [1], [2]. The present study contributes a smaller-scale integrated implementation that combines administrative, faculty, and student workflows in a single browser-based application.

The application's authentication and access-control mechanisms also illustrate an important consideration in academic information-system development. Email verification, password requirements, session validation, role-based authentication, and controlled modification of sensitive information were included as implementation features. These mechanisms establish architectural controls for restricting access, but the present functional tests cannot establish comprehensive security assurance. Security quality would require additional activities such as penetration testing, vulnerability assessment, secure-code review, and monitoring under realistic deployment conditions.

The local and remote deployment configurations provide a degree of technical flexibility. The local configuration allows use within a networked environment, while the documented Cloudflare Tunnel configuration provides a mechanism for remote browser access without requiring a publicly routable origin address [3]. This flexibility may be relevant to institutions that differ in network availability and server infrastructure. However, actual scalability, reliability under concurrent use, and network performance were not experimentally established in this study.

The development approach also shows that a functional academic information system can be constructed by integrating established technologies rather than introducing a new programming language or computing paradigm. The technical contribution should therefore be understood as an application-development and systems-integration contribution. The design demonstrates one way of organizing common web technologies into an academic grade-management workflow, with functional verification providing evidence that the implemented modules operate within the defined test scenarios.

Several limitations must be considered. Functional verification was performed by the developer using dummy accounts and simulated transactions, so the findings do not represent actual user experience or institutional use. The study did not conduct independent usability testing, user-acceptance testing, longitudinal implementation, large-scale load testing, formal penetration testing, or systematic accessibility assessment. The results therefore establish functional feasibility within controlled test conditions rather than overall software quality or institutional effectiveness. Future work should subject the system to independent security assessment, usability evaluation, accessibility testing, and performance and load testing, and pilot implementation under real academic operating conditions.

From a development perspective, the study demonstrates that the core academic workflow can be represented as a sequence of connected transactions: registration, verification, authentication, role-specific activity, academic enrollment or grade encoding, and grade viewing. The successful completion of these transactions during functional verification indicates that the prototype achieved internal functional integration across the tested modules. The result provides a technical baseline from which subsequent evaluations can be conducted without treating the present functional pass result as a substitute for those evaluations.

Another implication concerns the relationship between functional integration and future extensibility. Because the application separates major user roles while maintaining shared academic data, additional modules can be introduced without changing the basic account structure. Potential extensions include reporting, analytics, data import, notification services, and integration with other institutional systems. Such extensions should nevertheless undergo the same cycle of requirement definition, implementation, functional verification, security review, and user evaluation rather than being assumed to work solely because the current modules passed their predefined tests.

The same principle applies to deployment decisions. Functional completion can establish that a defined workflow operates in the development environment, but production readiness also depends on infrastructure capacity, backup procedures, security controls, maintenance responsibilities, and institutional policies. Documenting these boundaries is important because it prevents a functional prototype from being interpreted as a complete institutional information-system solution without the additional evaluation required for real-world adoption.

The implementation also illustrates the practical value of using established technologies when the objective is to produce an integrated information system rather than a new underlying computing technology. PHP, HTML, CSS, JavaScript, XAMPP, and phpMyAdmin are established technologies with widely available development resources. Their use can simplify maintenance and future modification compared with creating a proprietary technical stack, although the suitability of any technology combination depends on the institution's personnel, infrastructure, security requirements, and long-term support capacity.

The maintenance implications are also significant. A web-based grade tracking application is not a finished product merely because its initial functions pass a test checklist. The thesis documentation identifies regular database backups, software updates, access-right reviews, code reviews, network checks, user support, and continuing testing as maintenance considerations. These practices are consistent with the broader software-engineering requirement that information systems be monitored and updated as technologies, security threats, infrastructure, and institutional requirements change. The present article therefore treats maintenance as part of the system lifecycle rather than as an optional activity after deployment.

The application is also comparable in purpose to earlier student-record and digital-grading systems documented in the literature. A prior study [1] described a centralized academic-record system incorporating grade submission, assessment, security, and reporting functions, while Another study [2] described automation of grading activities and retention of digital records. A related study [7]

also discussed automated student information management using XAMPP and access control. The present system shares the general objective of centralizing academic information but emphasizes a compact web-based architecture that combines administrator, faculty, and student workflows and can operate in local or remote configurations.

The functional results also provide a useful way of distinguishing system completeness from system effectiveness. The 28 successful tests show that the developer's predefined scenarios were executable and that the principal modules were connected sufficiently for the intended test transactions. However, a successful transaction does not indicate how quickly users could complete it, how understandable the interface would be to first-time users, or how reliably the system would behave under heavy concurrent activity. The result is therefore best interpreted as a verification of functional behavior rather than a comprehensive assessment of software quality.

5.0 CONCLUSION

The study developed a web-based academic grade tracking application using a structured Waterfall software-development process. The resulting system integrated administrator, faculty, and student functions covering account registration and authentication, profile management, administrative information management, enrollment, grade encoding, and grade viewing. The application was implemented using PHP, HTML, CSS, JavaScript, XAMPP/phpMyAdmin, and supporting authentication and web-access technologies.

Developer-performed functional verification was conducted using dummy accounts and a predefined 28-item test checklist. The tests covered the major workflows of the application, and all 28 test cases were recorded as successful under the controlled testing conditions. The findings provide technical evidence that the developed prototype was able to execute its specified core functions within the tested environment.

The results do not establish usability, user acceptance, institutional effectiveness, long-term reliability, production-scale performance, accessibility compliance, or comprehensive cyber security assurance. Further evaluation under actual operational conditions is therefore required before the application can be treated as an institutionally validated production system. The principal contribution of the study is the development and documented functional verification of an integrated web-based academic grade tracking prototype that can serve as a foundation for subsequent technical and user-centered evaluation.

Author Contributions

The author was responsible for conceptualization, methodology, software development, investigation, functional testing, data curation, visualization, project administration, and manuscript preparation, including writing the original draft and reviewing and editing the final manuscript.

Funding

This research received no external funding.

Ethical Approval

Not applicable to the present article. The manuscript reports software development and developer-performed functional verification using dummy accounts and simulated transactions. No human participants, participant responses, expert ratings, or participant-derived research data are included in the reported analysis.

Competing Interest

The author declares no conflicts of interest.

Data Availability

The functional testing records supporting the findings of this study are available from the corresponding author upon reasonable request.

Declaration of Artificial Intelligence Use

In this work, the author utilized artificial intelligence (AI)-assisted technology, ChatGPT by OpenAI, to assist with manuscript organization, language refinement, restructuring, and drafting based on the author's original research materials. The author reviewed, evaluated, revised, and verified the resulting content and takes full responsibility for the accuracy, integrity, and final published content.

Acknowledgement

The author acknowledges the institutional and technical support that contributed to the development and completion of the work reported in this manuscript.

REFERENCES

- [1] M. D. C. Saquin and D. E. Marcial, "Developing an automated student academic record management with business intelligence approach," *Jurnal Informatika*, vol. 12, no. 2, pp. 111–123, 2016. doi: <https://doi.org/10.21460/inf.2016.122.453>.
- [2] A. Bloomfield, "Evolution of a digital paper exam grading system," in *Proc. 2010 IEEE Frontiers in Education Conference (FIE)*, 2010. doi: <https://doi.org/10.1109/FIE.2010.5673292>.
- [3] Cloudflare, "Cloudflare Tunnel," Cloudflare Zero Trust documentation, Aug. 27, 2024. [Online]. Available: <https://developers.cloudflare.com/cloudflare-one/connections/connect-networks/>. [Accessed: Oct. 15, 2026].
- [4] D. L. Goodhue and R. L. Thompson, "Task-technology fit and individual performance," *MIS Quarterly*, vol. 19, no. 2, pp. 213–236, 1995. doi: <https://doi.org/10.2307/249689>.
- [5] M. Aljukhadar, S. Senecal, and J. Nantel, "Is more always better? Investigating the task-technology fit theory in an online user context," *Information & Management*, vol. 51, no. 4, pp. 391–397, 2014. doi: <https://doi.org/10.1016/j.im.2013.10.003>.
- [6] Apache Friends, "XAMPP." [Online]. Available: <https://www.apachefriends.org/>. [Accessed: Oct. 15, 2026].
- [7] N. Ranjith, "Automation of student information using XAMPP with multiple access control," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 5, no. 3, 2023. doi: <https://doi.org/10.56726/IRJMETS35087>.

Author Profile

Dominic Dave B. Balino is an educator whose academic and professional interests include educational technology, web-based academic information systems, software development, and digital tools for academic record management. His work focuses on the design and development of technology-supported applications for educational settings.