Write here your dedication!

Cover

Acknowledgment

Write here the acknowledgments and grants, if any

Resumo

Write here your abstract in Portuguese

Abstract

Write here your abstract in English

Contents

CHAPTER 1

Introduction

1.1. Background & Context

The landscape of higher education has undergone significant transformation through the widespread adoption of digital technologies and learning management systems. This digital evolution has created unprecendented opportunities for institutions to leverage data analytics and innovative pegagogical approaches in order to enhance student learning outcomes and insitutional effectiveness. Within this context, the integration of gamification principles with educational analytics represents a promising avenue for addressing persistent challenges in student engagement, motivation and academic performance monitoring.

The concept of gamification, defined as the application of game design elements in non-game contexts [**1**], has gained considerable traction in educational settings as a mechanism for increasing student motivation and engagement. When combined with learning analytics, which involves the measurement, collection, analysis and reporting of data about students and their contexts for purposes of understanding and optimizing learning, gamification can prove to be a powerful tool for both students and teachers, both to monitor and improve the learning process.

The Learning Scorecard perfectly exemplifies this convergence of gamification and educational analytics. Originally conceived as an academic performance management system, the LS has evolved through multiple iterations to become a platform that merges business intelligence methodologies with educational gamification strategies. The platform's core objective is to provide the students and faculty with actionable insights into the learning progress while maintaining high levels of student engagement through game-like mechanics.

However, the evolution of the Learning Scorecard has revealed, throughout the years, significant architectural and operational limitations that constrain its effectiveness and adoption within institutional contexts. These limitations, particularly regarding system integration, data collection efficiency and user accessibility, have urged a fundamental reconceptualization of the platform's technical infrastructure and deployment strategy.

1.2. Statement of the Problem

The current implementation of the Learning Scorecard (version 2.0) platform presents several critical challenges that prevent its effectiveness and widespread adoption within higher education institutions. These challenges can be categorized into three primary domains: technical architecture, operational efficiency and institutional integration.

The existing Learning Scorecard operates as a standalone web application with external infrastructure dependencies, including cloud-hosted databases and third party service integrations. This architectural approach creates several complications: first, it requires a separate authentication system and user management processes that exist parallel to institutional systems; second, it requires ongoing maintenance of external infrastructure, increasing operational costs and security vulnerabilities; third, it creates data silos, isolated collections of data that prevent seamless integration and data sharing with existing institutional learning management systems.

The current system requires substancial manual data entry from both students and faculty in order to populate the platform with the necessary academic and performance information. This manual input requirement creates multiple friction points: Students must duplicate the effort of entering information that already exists in institutional systems; faculty members face additional administrative burdens in maintaining accurate records of data across multiple platforms, for example, the need to create another account, leading to another set of credentials and the requirement to input ther quiz and exercises grades manually into the platform; the potential for data inconsistencies and errors increases significantly due to human input mistakes.

Perhaps most critically, the standalone nature of the current Learning Scorecard creates barriers to institutional adoption and sustained usage. Students and faculty must actively choose to visit the external platform, creating an additional step in their academic workflow. The platform lacks integration with existing institutional systems, preventing it from leveraging the data already collected through learning management systems like Moodle.

The Learning Scorecard stemmed from an investigation project and had no institutional support in its development or maintenance. The development was made in ISCTE, by students but the goal was always to create a scalable learning tool that could be integrated seamlessly in other universities.

These challenges collectively result in a reduced user adoption, increased maintenance costs and compromised data accuracy.

1.3. Aims and Objectives

This dissertation addresses the identified limitations through the development and evaluation of a Learning Scorecard version 3.0, which fundamentally reconceptualizes the platform's architecture and deployment strategy by inverting it and explore the feasibility of integrating the Learning Scorecard with the moodle system and infrastructure.

From a technical perspective, this dissertation aims to: (1) design and implement a modular plugin architecture that could seamlessly integrate with ISCTE's Moodle infrastructure; (2) develop automated data collection mechanism that leverage Moodle's comprehensive student activity and performance data; (3) create user interfaces that maintain consistency with Moodle's design patterns while providing functionality of the

Learning Scorecard platform; (4) ensure data security and privacy protection through adherence to Moodle's established security frameworks.

This dissertation also seeks to validate the effectiveness of the integrated approach through: (1) demonstration of reduced manual input requirements compared to previous Learning Scorecard versions; (2) evaluation of user experience improvements resulting from seamless integration with existing academic workflows; (3) assessment of system performance and scalability within the Moodle environment; (4) analysis of the platform's potential for broader institutional adoption given its integration with existing infrastruture.

Beyond technical and empirical considerations, this dissertation aims to contribute to the broader understanding of how gamification platforms can be effectively integrated within established educational technology ecosystems. The findings will provide insights into best practices for developing educational technology solutions that complement rather than compete with existing institutional systems.

1.4. Methodological Approach

This dissertation employs a design science methodology, combining systematic design and development processes with empirical evaluation. The methodology encompasses three primary phases: analysis and design, implementation and development and evaluation and validation.

The initial phase involves an analysis of existing Learning Scorecard implementations, detailed examination of Moodle's plugin architecture and Service API capabilities and systematic requirements gathering. This phase includes mapping the Learning Scorecard concepts to Moodle's existing data structures and functionality, identifying integration opportunities and constraints and developing a system architecture that addresses the identified limitations while preserving core platform capabilities.

The development phase follows an agile methodology with iterative development cycles focused on delivering functional components that can be tested and refined throughout the process. The development approach emphasizes modular design and adherence to Moodle coding standards to ensure compatibility with Moodle's ecosystem. Key deliverables include a block plugin to exemplify a user interface component of the student interface and a local plugin for main functionality and who's responsible for the addition of database schemas that integrate with Moodle's existing data structures.

More about the evaluation and validation

1.5. Structure of the Dissertation

This dissertation is organized into seven chapters that systematically address the objectives and present the development, implementation and evaluation of Learning Scorecard 3.0.

Chapter 2 provides comprehensive background on learning management systems, with particular focus on Moodle's architecture and capabilities and examines the theoretical foundations and practical applications of gamification in educational contexts. This chapter establishes the theoretical framework that suplements the design decisions throughout the dissertation.

In chapter 3 the historical development of the Learning Scorecard is traced through its various iterations, analyzing the lessons learned from previous implementations and establishing the conceptual framework of the Learning Scorecard.

Chapter 4 presents the technical architecture of the Learning Scorecard 3.0, including detailed requirements analysis, system design decisions, plugin architecture specification and database design. It's also presented a detailed mapping of Learning Scorecard concept to Moodle's existing functionality and its mapping feasibility. This chapter provides the technical foundation for understanding the implementation approach and design rationale.

Chapter 5 describes the key deliverables and presents the proposed solution and serves as a demonstration of some functionality from the Learning Scorecard and its feasibility as Moodle integrated plugins.

Chapter 6

Chapter 7

CHAPTER 2

Literature Review

2.1. Learning Management Systems

Learning Management Systems (LMS) represent sophisticated software applications engineered to administer, track, report, automate and deliver educational courses, training programs and comprehensive learning materials. Following their initial emergence in the late 1990s, these platforms have achieved significant adoption across higher education institutions globally. Recently, the unprecedented expansion of LMS usage has been significantly accelerated by the institutional shift toward remote learning by the COVID-19 pandemic. [**2**]

There are several LMS available for educational institutions, with commercial and open-source platforms representing the two primary categories of LMS offerings [**3**]. Commercial LMS are either available in the market for a price or, although very rarely, proprietary and developped by the institution. In contrast, the open source alternative is available at no cost to the institution, although technical expertise is required as well as regular functional assessments. Open-source systems provide publicly accessible source code that institutions can modify and improve to meet their specific requirements and needs [**4**].

Moodle (Modular Object-Oriented Dynamic Learning Environment) stands as one of the most widely adopted open-source learning management systems globally. According to a recent systematic review on tendencies in the use of LMS, Moodle is the most popular and preferred open-source LMS [**5**]. The platform's extensive reach is evidenced by its current deployment across 238 countries, serving approximately 468 million users worldwide [**6**].

From a technical perspective, Moodle is built on a layered architecture (Figure 1) that ensures flexibility and scalability. The presentation layer handles the user interface and allows users to interact both on desktop and mobile devices. The application layer accommodates core functionalities and acts as an intermediary between the presentation and data layer, supporting the installation of plugins and modules to customize features. The data layer includes the database management system, typically SQL-based, which stores and manages all the data. This modular design allows institutions to customize the functionality and appearance through an extensive plugin ecosystem.

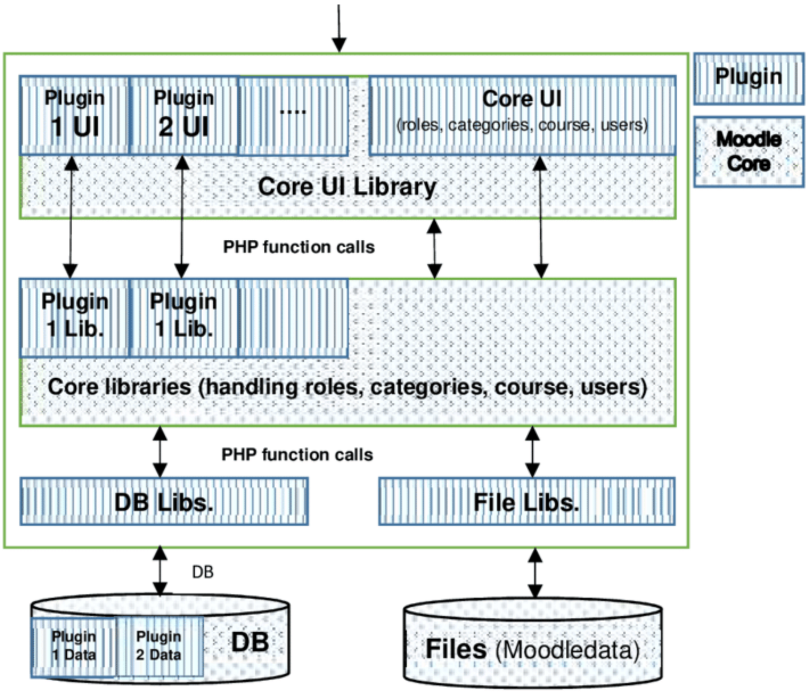

FIGURE 1. Moodle three tier architecture. Source: Dolawattha et al. [7].

Moodle offers many plugins that allow for functionality expansion. There are multiple types of plugins from local plugins, block plugins all the way to antivirus plugins. Each with its own purpose and place in the moodle folder structure.

2.2. Gamification and Motivation in Education

2.2.1. Theoretical Perspectives on Gamification

2.2.2. Gamification and Student Engagement

CHAPTER 3

The Learning Scorecard: Concept and Evolution

3.1. Genesis and Conceptualization of the Learning Scorecard

The Learning Scorecard emerged from a fundamental pedagogical challenge identified within higher educational institutions: the lack of understanding by faculty members regarding the difficulties experienced by students in relations to the content taught in curricular units aswell as the students disinterest and lack motivation. Initially, this platform was conceptualized as an academic performance management system designed to support both students and faculty through data-driven insights that assist teachers in continuously monitoring the curricular units they teach while enabling students to visualize their performance in each course in which they are enrolled. Their main mission was to offer higher education students an analytic environment that allows progress monitorization through a familiar game-like scope, contributing for a better learning experience.

The platform was originally conceptualized with three distinct user perspectives: student view, teacher view and administrative view. The student view provides essential visualizations that enable students to understand their learning trajectory in a specific curricular unit while offering tools that provide important data for continuous monitoring by faculty. The teacher view provides management, evaluations and visualization tools for educators regarding each curricular unit they teach, offering various visualizations to track each students progress through all phases of the curricular unit along with assessment tools.

3.2. Evolution Through Iterative Development

3.2.1. Historical Overview of Versions

The Learning Scorecard development began in the 2015/16 academic year as a collaborative project project undertaken by a group of Computer Science masters students enrolled in the Information Systems of Decision Support II curricular unit. This class established a foundation of rigorous analysis and systematic development that has characterized the platform's evolution throughout subsequent iterations.

The development timeline reflects the progression of features and refinements with each major version addressing specific limitations and opportunities identified through previous implementations. This iterative approach has enabled the platform to evolve in response to changing technological landscapes, pedagogical insights and institutional requirements while maintaining consistency in core design principles.

Learning Scorecard Version 1 (2016)

Learning Scorecard Version 2 (2017)
Learning Scorecard Version 3 (2018)

3.2.2. Lessons Learned from Prior Iterations

The evolutionary development of the Learning Scorecard has yielded important insights into the challenges and opportunities associated with educational technology implementation in higher education contexts. These lessons have directly informed the design decisions underlying Learning Scorecard 3.0 and provide valuable guidance for future LS development efforts.

Previous iterations consistently revealed the critical importance of seamless integration with existing institutional systems and workflows. Standalone implementations, while providing flexibility and autonomy, created barries to adoption through the requirement for separate authentication, manual data entry and navigation between multiple platforms. These experiences highlighted the need for deeper integration with institutional learning management systems to achieve sustainable adoption and effectiveness.

Analysis of user behavior accross previous versions revealed important patterns regarding the factors that drive sustained engagement with educational technology platforms. Students demonstrated higher levels of engagement when gamification elements were seamlessly integrated with existing academic workflows rather than requiring separate platform visits or additional effort investment. Faculty adoption correlated strongly with the availability of actionable insights that directly supported pedagogical decision-making without creating additional workload or administrative burden.

Scalability and Maintenance Considerations The operational experience of maintaining standalone Learning Scorecard implementations revealed significant challenges related to infrastructure costs, security management and version compatibility. These experiences demonstrated the advantages of leveraging existing institutional infrastructure and support systems rather than maintaining separate technological stacks that require independent updates, security monitoring and user support.

3.3. Learning Scorecard Conceptual Framework

The Learning Scorecard (LS) conceptual framework encompasses a comprehensive set of interconnected concepts designed to support gamified learning experiences in higher education. This framework is organized around four primary fomains that reflect different aspects of the educational experience: Academic Structure and Organization, Gamification Mechanics, Social LearningThese concepts can be organized into four primary domains: Academic Structure, Gamification Mechanics, Social Learning and Recognition and Achievement Systems. Each domain contains interconnected concepts that work together to create a comprehensive educational technology ecosystem that supports both individual learning, cooperation and institutional objectives.

3.3.1. Academic Structure and Organization

The academic foundation of the Learning Scorecard is built upon traditional higher education organizational structures, adapted to support digital learning environments.

Courses and Curricular Units. The system recognizes a two-tier academic structure where *Courses* represent overarching academic programs (e.g., "Computer Science") that encompass multiple *Curricular Units*. Each Curricular Unit corresponds to an individual subject with specific learning objectives, content and assessments that students must complete as part of their academic progression. This hierarchical organization maintains consistency with traditional academic frameworks while enabling digital tracking and gamification.

Students and Faculty. The system accommodates two primary user roles: *Students*, who are enrolled in one or more Curricular Units and participate in gamified learning activities and *Teachers* or Faculty members, who are responsible for creating and managing curricular content, designing learning activities and monitoring student progress within their assigned Curricular Units.

Syllabus Contents, Calendar and Timeline. *Syllabus Contents* encompass all learning materials, resources and educational content within a Curricular Unit. The organizational framework includes both *Calendar* and *Timeline* concepts that serve as complementary planning systems, enabling students and faculty to organize learning activities and maintain temporal awareness of curricular progression.

3.3.2. Gamification Mechanics

The core gamification elements provide the foundation for student engagement and progress tracking through game-like mechanics adapted to educational contexts.

Experience Points and Ranks. *Experience Points* (XP) serve as the fundamental currency of progress within the platform, earned through completion of various educational activities. The *Rank* system establishes hierarchical progression levels based on accumulated XP, including ranks such as Newbie (entry level at 0 XP), Rookie, Skilled Expert, Master and Legendary. Faculty members can configure rank thresholds dynamically, with each rank unlocking new privileges and recognition within the platform.

Quests. *Quests* represent the core educational activities that students complete to earn XP and demonstrate learning. The system categorizes quests into five distinct types: Class Attendance, Practical Assignment, Quiz, Exercise and Event. Each quest can be designated as mandatory (milestone quests) or optional, providing flexibility in curriculum design while maintaining educational objectives.

Last Chances. The *Last Chance* mechanism provides opportunities for recovery and continued engagement when students fall behind or make mistakes, preventing permanent failure states that could lead to disengagement and ensuring continued participation throughout the academic term.

3.3.3. Social Learning

The social components of the Learning Scorecard foster collaboration and healthy competition among students through structured group dynamics.

Alliances. *Alliances* function as large-scale organizational units that typically correspond to academic classes or programs (e.g., LEI or ETI). These structures facilitate competition and comparison at the class level while maintaining educational coherence, enabling students to compare their progress through dedicated leaderboards and participate in alliance-specific activities.

Guilds. *Guilds* represent smaller collaborative units within alliances, typically corresponding to project teams or study groups. The guild system promotes cooperative learning by encouraging mutual assistance among members, with guild-specific achievements, badges and leaderboards fostering team spirit while maintaining individual accountability within the collaborative framework.

3.3.4. Recognition and Achievement System

The recognition framework provides multiple pathways for acknowledging student accomplishments and maintaining motivation through visible achievements.

Badges. The *Badge* system provides targeted recognition for specific accomplishments and behaviors, featuring 39 different badges organized into four categories: individual achievements, guild-based accomplishments, forum participation and final questionnaire completion. Each category includes multiple tiers (bronze, silver, gold and platinum) corresponding to increasing levels of difficulty and commitment.

Trophies. *Trophies* represent the highest level of achievement, awarded to top performers in various leaderboard categories including overall rankings (Best Score Player), guild performance (Best Guild) and combined metrics (Best Triathlon Player). Each trophy provides substantial XP rewards (typically 1000 XP) and serves as a highly visible symbol of excellence.

Avatars. The *Avatar* system enables profile customization while serving as a visual indicator of progression. Students unlock new avatar options as they advance through rank levels, with both male and female options available at each tier, allowing for personal expression while maintaining the connection between visual representation and academic achievement.

Leaderboards. Multiple *Leaderboard* systems enable various forms of competition and comparison through five distinct categories: overall individual rankings, guild-based comparisons, exercise-specific performance, quiz results and combined metrics. Each leaderboard displays position, username, rank, avatar and XP totals, creating transparency in academic performance while fostering healthy competition among participants.

CHAPTER 4

System Architecture and Design

4.1. Requirements

An imperative aspect in any application conceptualization is a thorough requirement analysis. These requirements can be functional (FR) or non functional (NFR) and shape the architectural decisions that are made during its development as well as assessing whether a system will succeed.

Functional Requirements (FR)

It's important to define what an engineer should strive for when determining functional requirements as it will ultimately be what defines an application, molds its functionality and consequently decides it's architecture. Functional requirements are the essential characteristics that the system should deliver that the end-user specifically demands. [**8**] A simpler way to perceive it is that they define what the system should perform. It's important with any version increment of the LS to maintain fidelity to the previous requirements and adjust them when appropriate. Costa [**9**] identified base functional requirements which I will evaluate and integrate into my system architecture, but also define some myself.

- FR1 The LS platform must use Moodle's existing authentication and user management systems.
- FR2 The LS platform must be integrated with the e-learning system, providing seamless access to student data, course data and activities.
- FR3 The LS Platform must use Moodle's role-based access control mechanisms, distinguishing between student and teacher user types with corresponding permissions and interfaces.
- FR4 The LS platform must have a dashboard for individual student performance monitorization (students view)
- FR5 The LS platform must have a dashboard for class performance monitorization (teacher view)
- FR6 The LS platform must have a course chronogram with deadlines and study guidelines, personalized by the teacher/faculty
- FR7 The LS platform must include gamification elements (score, ranks, quests, leaderboards)

Non-Functional Requirements (NFR)

Another important aspect of the requirement analysis is to define the non-functional requirements. These, differently from their functional counterpart must define how the system should perform, rather than what. While not directly visible as features they are vital in shaping the user experience and ensuring the system's long-term success. A few examples of these requirements could be performance, security, reliability, etc. Costa [**9**] also layed out base non-functional requirements which I will also evaluate, adapt and integrate into my system architecture, in addition to some identified by myself. From Costa's non-functional requirements:

NFR1 Portability for the different web browsers, including mobile devices
NFR2 Intuitive and user-friendly interface (least possible manual input by students)
NFR3 The user identification data must be private (ethical requirement), the teacher/faculty must only have access to the class agregated data, even for at risk students.

This last non-functional requirement, as Costa correctly identified is a missed opportunity as the faculty will have no way of knowing which student is at risk and ultimately take advantage of such introspective metrics on student performance. As the system is integrated into moodles learning environment and all this information is already available to the faculty I believe this NFR should be altered into a more transparent requirement and only require anonymization of students the data if the Learning Scorecard extended its functionality to allow for a predictive model to be set in place to predict the outcome of the students grade based on current effort. Therefore, the corrected NFR3 is:

NFR3 The user identification data must be private (ethical requirement) in case it's used in colaboration with a predictive model for the grades. The teacher must only have access to the class agregated data.

Throughout the study and conceptualization of the system a few more non-functional requirements were identified:

NFR4 The system must be installable as standard Moodle plugin(s) without requiring server modification.
NFR5 The system must maintain visual consistency with the Moodle theme.
NFR6 The system must store all data within the Moodle database structure.

4.2. Overview of LS Architecture

4.2.1. Evolution from Standalone Architecture

The LS architecture has withstood many iterations throughout the years and has become more complex and polished through them, as new features have been added, alternatives and changes in architecture have been thought as most adequate. Let's suppose we take the latest version as a baseline and dive into what can be improved.

In the latest version of the Learning Scorecard, version 2.0, as documented by [**17**] the technological stack consisted of React [**10**] frontend components communicating with Express [**11**] (NodeJS [**12**]) backend services, which in turn interfaced through REST with a cloud-hosted NoSQL MongoDB database [**14**] and GraphDB [**15**] system deployed in Azure infrastructure [**13**] primarily hosted in Heroku [**16**].

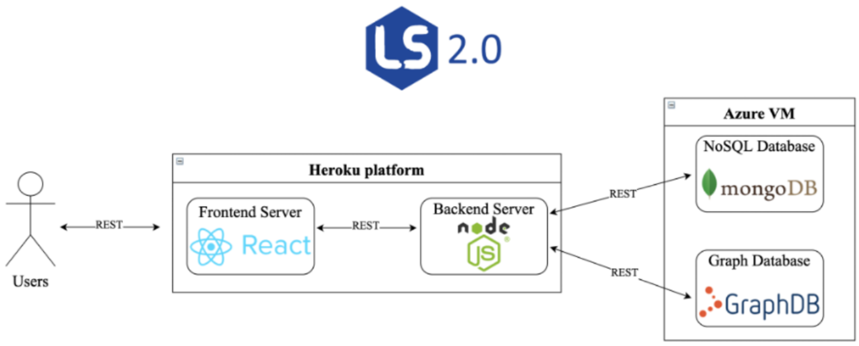

FIGURE 1. LS2.0 Technology Stack. Source: Lopes [**17**].

While this architecture demonstrated scalabilty, modularity and adherence to modern web development practices, it presented several critical limitations, mainly in the fact that it is not integrated with the ISCTE Moodle environment (FR2), relying on manual input by the faculty and students making it not an ideal system and one that would become more of a chore for teachers who have most commonly multiple classes to teach, register and manage and students who also have multiple classes, quizes and exercises to keep track of.

We must then conjure a system that is integrated with the ISCTE moodle environment and satisfies all functional and non-functional requirements set previously.

At first glance, it is possible to achieve a more integrated system making use of the platform developed previously with the use of the moodle native web services, although the standalone nature necessitated separate authentication systems, creating friction for users who must maintain multiple credentials and navigate between distinct platforms. The external infrastructure dependencies resulted in ongoing operational costs, particularly following Heroku's discontinuation of free tier services [**18**], and introduced security vulnerabilities through external data hosting that may not align with institutional data governance policies.

4.2.2. Plugin-Based Architecture Benefits

The Learning Scorecard 3.0 plugin-based architecture approach addresses these limitations through direct integration with Moodle's comprehensive infrastructure and Service API ecosystem. This approach provides several critical advantages that enhance both technical capabilities and institutional viability.

Infrastructure Integration By operating within Moodle's establish infrastructure, LS 3.0 eliminates external hsting requirements and associated costs while ensuring compliance with institutional security and data governance policies. The platform leverages existing ISCTE server infrastructure, Apache web servers and database systems without requiring additional resources or maintenance overhead.

Authentication and User Management Integration with Moodle's authentication system erases the need for separate user credentials and login processes, reducing friction for students and faculty while ensuring consistency with institutional access policies. The platform automatically inherits Moodle's sophisticated role and permission systems, enabling fine-grained access control without additional implemenetation complexity.

Data Ecosystem Access Perhaps most critically, the plugin arhitecture provides direct access to Moodle's comprehensive student activity and performance data, enabling automated data collection and real-time analytics without manual input requirements. This access includes quiz results, assignment submissions, forum participation, resource access patterns and completion tracking data that can be seamlessly integrated into gamification and analytics workflows.

User Experience Consistency The plugin approach ensures visual and functional consistency with existing institutional interfaces, reducing learning curves for users and promoting adoption through familiar interaction patterns. Students and faculty can access Learning Scorecard fuctionality without leaving their established academic workflows.

4.2.3. Concept Mapping & Integration Analysis

The successful integration of Learning Scorecard functionality within Moodle's existing architecture requires comprehensive analysis of conceptial alignments and implementation feasibilities. This mapping process identifies opportunities for leveraging existing Moodle capabilities while determining areas that require custom implementation to preserve Learning Scorecard legacy features.

Table 1. Learning Scorecard vs Moodle Concept Mapping

Concept	Moodle	Feasibility
	Academic	
Course	Course Categories	T
Curricular Unit	Course	T
Student	Users & Roles	T
Teacher / Faculty	Users & Roles	T
Syllabus Contents	Course Modules & Tags	T
Calendar	Calendar Events	P
Timeline	Activity Completion & Progress	P
	Gamification	
Experience Points	Grade Items & Custom Fields	P
Ranks	Custom Implementation	P
Quests	Activities	T
Alliances	Groups & Cohorts	T
Guilds	Groups	T
Badges	Badges System	T
Trophies	Badges System & Custom Impl.	P
Avatars	User Profiles, Files & Custom Impl.	P
Leaderboards	Gradebook & Custom Views	P
Last Chance System	Conditional Activities	P

T - Total Mapping P - Partial Mapping

4.3. Development Environment and Technical Infrastructure

The establishment of a robust, consistent and production-like development environment proved critical for the successful implementation of LS 3.0. The complexity of Moodle plugin development, combined with the need for the system to mimick the production environment as much as possible, required careful consideration of development infrastructure and tooling decisions.

4.3.1. Development Environment Evolution

The development environment configuration process revealed important insights into the requirements and challenges of Moodle plugin development, ultimately leading to the identification of optimal approaches for maintaining development productivity and most importantly, system reliability.

The initial development approach employed XAMPP [**20**] to provide a local PHP environment, apache server and Mysql database infrastructure, imperative for standalone Moodle instance execution. However, this approach demonstrated significant limitation that compromised development reliability and consistency. File persistence issues following server restarts and unpredicatable system behavior created an unstable foundation that was incompatible with the production-like development environment required for plugin implementation.

The second development approach investigated Docker containerization [**21**] with orchestrated Bitnami Moodle image [**22**] to achieve improved consistency and reproducibility. While this approach initially showed promise through improved container persistence and more predictable behavier, plugin integration revealed critical limitations. The Bitnami image implemented included automated Moodle configuration processes that incorrectly identified plugin additions as corrupted system files, triggering unwanted fresh installations that compromised the continuity of the plugin's development and was therefore excluded from consideration for the final Moodle development environment.

The final development environment employed custom docker container orchestration with separate Apache server and MariaDB database [**23**] images, enabling manual Moodle configuration that avoided the automated configuration issues encountered with pre-built images (like Bitnami's). Although with the increased effort of configuring the apache server and installing the same Moodle version as the production ISCTE 2024/2025 Moodle instance (version 4.4.1) and all the necessary dependencies this approach provided the stability, consistency and production-like behavior expected and required for reliable plugin development.

4.4. Plugin Integration with Moodle

There are multiple types of moodle plugins and depending on their purpose we should select the most adequate. According to the specification of the Learning Scorecard we would need to evaluate what plugin type is better suited.

Moodle's comprehensive plugin architecture [**24**] provides multiple plugin types, each optimized for specific functionality categories and integration patterns. The Learning Scorecard implementation requirements necessitated evaluation of plugin types to ensure appropriate functionality distribution and optimal integration with Moodle's existing systems.

Block plugins emerged as the optimal choice for user-facing dashboard and metric displays. The block architecure's deisgn for small, configurable information displays that can be positioned throughout Moodle's interface aligns perfectly with Learning Scorecard's requirement for accessible gamification metrics within existing workflows. block plugins enable experience point displays, ranking information and progress metrics to be seamlessly integrated into student and faculty dashboards without requiring navigation to separate interfaces.

Local plugins generic specification provide the appropriate architecture for additional LS functionality, including leaderboards, Learning Scorecard configuration, database table creation, etc.

The modular plugin approach requires appropriate directory organization within Moodle's established file structure. Block plugins are installed within the *"/blocks"* directory, while local plugins reside in the *"/local"* directory, ensuring compliance with Moodle's architectural standards and enabling proper plugin discovery and management through Moodle's administrative interfaces.

4.5. Database Design and Data Integration

CHAPTER 5

MVP

5.1. System Components and Modular Design

5.1.1. Student Interface and Experience

5.1.2. Teacher Dashboard and Analytics

5.1.3. Administrator Functions and Control

CHAPTER 6

Evaluation and Results

6.1. Demonstration of Key Features

6.2. Feedback and Empirical Results

6.3. Comparative Analysis with Previous Approaches

6.4. Discussion of Evaluation Metrics

CHAPTER 7

Conclusions

7.1. Summary of Contributions

7.2. Limitations and Challenges

7.3. Future Directions for LS

References

S. Deterding, R. Khaled, L. Nacke and D. Dixon (2011), "Gamification: Toward a definition", *Proceedings of CHI 2011 Workshop Gamification: Using Game Design Elements in Non-Game Contexts*, 6–9.

J. H. L. Koh and B. K. Daniel (2022), "Shifting online during COVID-19: A systematic review of teaching and learning strategies and their outcomes", *International Journal of Educational Technology in Higher Education*, 19(1), 56.

N.N. Mohd Kasim and F. Khalid (2016), "Choosing the Right Learning Management System (LMS) for the Higher Education Institution Context: A Systematic Review", *International Journal of Emerging Technologies in Learning (iJET)*, 11(6), 55.

N. Cavus and T. Zabadi (2014), "A Comparison of Open Source Learning Management Systems", *Procedia - Social and Behavioral Sciences*, 143, 521–526.

H. Altinpulluk and M. Kesim (2021), "A Systematic Review of the Tendencies in the Use of Learning Management Systems", *Turkish Online Journal of Distance Education*, 22(3), 40–54.

Moodle (2025), "Moodle Statistics", Available at: https://stats.moodle.org/ (Accessed: 6 August 2025).

D.D.M. Dolawattha, H.K.S. Pramadasa and P.M. Jayaweera (2019), "The Impact Model: Teachers' Mobile Learning Adoption in Higher Education", *International Journal of Education and Development using Information and Communication Technology (IJEDICT)*, 15(4), 71–88.

S. Saroja and S. Haseena (2023), "Functional and Non-Functional Requirements in Agile Software Development", in *Agile Software Development*, John Wiley & Sons, Ltd, pp. 71–86.

D.S. Costa (2017), "Learning Scorecard: Plataforma para a Monitorização da Experiência de Aprendizagem de Alunos no Ensino Superior Aplicando Técnicas de Business Intelligence e Gamificação", Master's Thesis, ISCTE-IUL, Lisbon, Portugal.

Meta Platforms, Inc. (2024), "React: The Library for Web and Native User Interfaces", Available at: https://react.dev/ (Accessed: 28 August 2025).

OpenJS Foundation (2024), "Express: Fast, Unopinionated, Minimalist Web Framework for Node.js", Available at: https://expressjs.com/ (Accessed: 28 August 2025).

OpenJS Foundation (2024), "Node.js: JavaScript Runtime Built on Chrome's V8 JavaScript Engine", Available at: https://nodejs.org/en/ (Accessed: 28 August 2025).

Microsoft Corporation (2024), "Microsoft Azure: Cloud Computing Platform and Services", Available at: https://azure.microsoft.com/en-us/ (Accessed: 28 August 2025).

MongoDB, Inc. (2024), "MongoDB: The Developer Data Platform", Available at: https://www.mongodb.com/ (Accessed: 28 August 2025).

Ontotext (2024), "GraphDB: Enterprise Ready RDF Database", Available at: https://graphdb.ontotext.com/ (Accessed: 28 August 2025).

Salesforce, Inc. (2024), "Heroku: Cloud Application Platform", Available at: https://www.heroku.com/ (Accessed: 28 August 2025).

M.Â.D. Lopes (2021), "Utilização de ontologias para a gestão de dados educacionais: caso de estudo Learning Scorecard", Master's Thesis, ISCTE - Instituto Universitário de Lisboa, Lisbon, Portugal.

Salesforce, Inc. (2024), “Heroku Pricing”, Available at: https://www.heroku.com/pricing/ (Accessed: 28 August 2025).

Git Community (2024), “Git: Distributed Version Control System”, Available at: https://git-scm.com/.

Apache Friends (2024), “XAMPP Apache + MariaDB + PHP + Perl”, Available at: https://www.apachefriends.org/.

Docker Inc. (2024), “Docker: Accelerated Container Application Development”, Available at: https://www.docker.com/.

Bitnami (2024), “Bitnami: Packaged Applications for Any Platform”, Available at: https://bitnami.com/.

MariaDB Foundation (2024), “MariaDB Server: The Open Source Relational Database”, Available at: https://mariadb.org/.

Moodle Pty Ltd (2024), “Plugin Types API Documentation”, Moodle Developer Resources, Available at: https://moodledev.io/docs/4.1/apis/plugintypes.