**Quality Management System Admin Module Development: A Study in the Department of Computer Science**

<table>
<thead>
<tr>
<th>Anggie Tamara¹</th>
<th>Didik Kurniawan²</th>
<th>Tristiyanoy³</th>
<th>Dwi Sakethiy⁴</th>
</tr>
</thead>
<tbody>
<tr>
<td>Universitas Lampung, INDONESIA</td>
<td>Universitas Lampung, INDONESIA</td>
<td>Universitas Lampung, INDONESIA</td>
<td>Universitas Lampung, INDONESIA</td>
</tr>
</tbody>
</table>

**Article Info**

**Article history:**
Received: April 6, 2023  
Revised: May 30, 2023  
Accepted: June 28, 2023

**Keywords:**
Admin Module, Computer science, Rapid Test Development, Quality management systems.

**Abstract**

Elevating the quality of learning is central to ensuring academic programs align with national standards. One crucial aspect of this process is the semester-based learning planning. Traditionally, the Department of Computer Science orchestrates this planning by creating separate documents for various data points, including courses, CPL (Graduate Learning Outcomes), and CPMK (Course Learning Outcomes), which are then integrated into the RPS (Semester Learning Plan) document each term. This study seeks a solution to enhance the efficiency of such data management through the development of an admin module. The system, designed using the Rapid Test Development (RAD) approach and the Laravel framework, offers a more effective data management solution, subsequently elevating the pedagogical quality of the Department of Computer Science. In conclusion, an integrated approach to data management can make a significant contribution to educational quality. Through this research, the scholarly contribution provided is the introduction of a systematic model in educational data management to boost the efficiency and effectiveness of the learning process in higher education institutions.


**INTRODUCTION**

Quality Management Systems (QMS) stand as a cornerstone in the development and sustenance of procedures that ensure the unwavering delivery of superior products and services [1]. Such systems, bolstered by comprehensive methodologies like ISO 9001, lay down the blueprint for organizations to integrate QMS seamlessly, enhancing their operational efficiency and reliability [2], [3]. At the heart of these systems lie fundamental pillars, including thorough documentation, vigilant process supervision, insightful performance assessments, and a culture deeply rooted in the pursuit of continuous improvement and excellence [4]. By adhering to these principles, organizations can foster a proactive environment where quality remains at the forefront of every endeavor.

High-quality higher education not only molds graduates to tap into and broaden their innate potential but also spurs the development of knowledge or technology that can profoundly benefit society, the nation, and even on a global scale [5]–[11]. However, a notable challenge emerges when observing the current state of higher education in Indonesia. Accreditation results indicate that out of 4,472 universities in Indonesia, a mere 50 have achieved an 'A' accreditation. Additionally, from the myriad of study programs on offer, only 2,512 have secured this prestigious 'A' rating [12].

Building on the broader landscape of higher education in Indonesia, the trajectory of the Computer Science Department at the renowned Faculty of Mathematics and Sciences in Lampung University is a noteworthy example. Initially embedded within the Mathematics Department, the journey began with the initiation of an undergraduate program. By 2005, recognizing the growing importance and distinct nature of computer science, the institution formally established the

*Corresponding author:*
Didik Kurniawan, Universitas Lampung, INDONESIA. didikkurniawan@fmipa.unila.ac.id
© 2023 The Author(s). Open Access. This article is under the CC BY SA license (https://creativecommons.org/licenses/by-sa/4.0/).
Computer Science Department. It then became an integral part of the Faculty of Mathematics and Sciences at the University of Lampung, highlighting the university's commitment to adapting and evolving with the demands of modern education. This evolution not only underscores the importance of specialized fields of study but also mirrors the broader national ambition to elevate the standards of higher education.

Building on the transformative journey of the Computer Science Department at Lampung University, maintaining and elevating its educational quality remains paramount. Adhering rigorously to program benchmarks ensures consistent evolution in line with global standards [13], [14]. A critical aspect of this is refining the effectiveness of individual academic courses, which in turn, shapes the collective learning experience [15]–[18]. Yet, challenges persist. Processes such as crafting the Learning Plan (RPS) – which consolidates information on courses, Graduate Learning Outcomes (CPL), and Course Learning Outcomes (CPMK) – are currently hindered by inefficiencies. The fragmented nature of course data and CPL complicates their integration into the RPS, making it a tedious manual task. To combat these obstacles and streamline operations, this research endeavors to introduce a comprehensive system specifically designed to mitigate the challenges encountered by the Computer Science Department.

In the realm of system development, prior studies have delved deeply into the creation and optimization of admin modules across various sectors. These range from applications in e-grocery platforms [19], intricate academic procedures [20], [21], healthcare administration in hospitals [22], oversight of campus transportation [23], security infrastructures [24], [25], to the management of financial systems [26]. However, a discerning examination of the literature reveals a notable gap: dedicated research centered purely on the intricacies of admin modules remains elusive. Consequently, this study aims to bridge this lacuna, focusing specifically on the development and refinement of the admin module.

In light of the previously outlined challenges and the apparent dearth of integrated solutions custom-built for specialized academic sectors such as Computer Science, this research introduces an innovative system. Designed with precision, this system endeavors to simplify and expedite the RPS formulation process by consolidating vital components into a cohesive unit. The objective is to not only alleviate the complexities inherent in the task but also to amplify its efficacy. In essence, this initiative seeks to fortify the existing quality management endeavors within the Computer Science Department, offering a comprehensive remedy to the identified issues. Rooted in the foundational principles of QMS, our novel approach is meticulously crafted to resonate with the nuanced demands and hurdles intrinsic to academic environments.

**METHOD**

The Rapid Application Development (RAD) method, a cornerstone in system development, is recognized for its adeptness in swiftly delivering software solutions. Unlike traditional models, RAD embodies a sequential and linear strategy complemented by succinct development phases [27]–[29]. Adopting RAD equips development squads with a toolkit of distinctive principles: active user engagement, iterative cycles, timeboxing, and rapid prototyping techniques [30].

Active user involvement ensures that the end users actively contribute to the software's evolution, providing input that shapes its features and functionality. On the other hand, Iterative development offers a framework for continuous refinement of the software, ensuring user feedback is assimilated throughout its development. Timeboxing introduces a systematic approach, segmenting the overall development process into manageable milestones, ensuring that projects remain on track. Lastly, rapid prototyping techniques are instrumental in swiftly producing initial models of the software, facilitating prompt user feedback and validation.

An insightful comparative study contrasting RAD and Agile methodologies, especially in supervising computing students' graduation projects, highlighted RAD's edge in producing comprehensive documentation, outpacing Agile by a notable 13.33% [31]. Such detailed documentation is paramount for significant projects, serving as a linchpin for...
understanding, maintenance, and knowledge transfer. RAD’s meticulous approach ensures that every nuance of the software is documented, reducing potential information gaps and streamlining future modifications.

However, the merits of RAD’s documentation-centric approach extend beyond the academic realm. In diverse software development contexts, well-structured documentation facilitates a seamless transition from the creation phase to the subsequent maintenance and support stages. Comprehensive documentation aids developers and stakeholders alike in grasping the intricacies of the software, from its architecture and design rationale to its core functionality. This repository of information becomes invaluable during software enhancement, debugging, or when integrating new team members.

Additionally, the study underscores the importance of aligning the chosen methodology with the specific demands of a project. While RAD stands out with its emphasis on documentation, Agile, with its inherent flexibility and iterative nature, might be a better fit for projects prioritizing rapid releases and frequent updates. Every methodology has its strengths and potential pitfalls, and it is incumbent upon project managers to evaluate a project’s unique requirements and constraints before making a methodological commitment.

The structural framework of the RAD method is visually represented in Figure 1.

![Figure 1. RAD flow](image)

The testing conducted in this research follows the black box testing procedure. Black-box testing aims to determine the accuracy of the system’s functionality according to the desired specifications. This testing phase, known as alpha testing, will be performed by the development team to ensure that the system operates successfully according to the specified scenarios.

**RESULTS AND DISCUSSION**

The research culminated in creating the Quality Management System for Teaching, explicitly designed to manage teaching data for the Computer Science Department at Lampung University. This system aims to streamline and enhance the efficiency of formulating teaching plans tailored for each semester and every course in the department.

The system was brought to life using the Laravel framework, providing the backbone for a PHP-based website. It amalgamates the Bootstrap interface, leveraging the Star Admin template and employing JavaScript. The primary functionality centers around an admin module. An individual granted the "admin" status can onboard other users, encompassing professors and quality assurance staff.

**Planning**

The Quality Management System for Teaching bifurcates into two core modules: the admin and the lecturer modules. This phase crystallized user requirements, depicted vividly in Figure 2.
1st System Design

The initial design aligns with the user requirements outlined in Figure 2. In this design framework, the Semester Learning Plan, or RPS, serves as an input and directly relates to the Course's Curriculum. The RPS incorporates the Graduate Learning Outcomes (CPL) which are segmented into four key aspects: attitude, knowledge, general aptitude, and skills. Additionally, it encompasses Course Learning Outcomes (CPMK), provided by the creator of the RPS and details the primary topics. The system is designed to cater to users with varying permissions and includes a versatile question bank to accommodate an array of topics or CPMKs.

The Entity Relationship Diagram (ERD) acts as the data model for this initial design, and it's based on the pre-established requirements. This diagram can be referenced in Figure 3.

Lastly, the use-case diagram, illustrating the process modeling for the initial design, is available in Figure 4.

1st Development

The system's development employs the Rapid Application Development (RAD) approach. The principles of RAD are exemplified in recent methodologies, including the open standard referred to as the Dynamic Systems Development Method [30].

The "Login" page is depicted in Figure 5. It features fields for both email and password input. If the provided email and password
correspond to an admin account, the user will be navigated to the dashboard. Conversely, if the credentials match other roles, the user will be directed to the dashboard specific to that role.

In Figure 6, the "Add User" page features multiple input fields to be completed. At the top are two buttons: "Template" and "Import." The "Template" button allows for the download of an Excel template to facilitate user data import in the specified format. Conversely, the "Import" button enables uploading an Excel file with the necessary user data.
In Figure 7, the "User List" page showcases the roster of registered users, with options to edit, delete, and reset their passwords. Figure 8’s "Profile" page allows to modify the account name, profile picture, and password. The "Course List" page, as seen in Figure 9, presents the catalog of added courses with editing and deletion capabilities. Lastly, the "Add Course" page in Figure 10 provides input fields necessary for introducing a new course.

![Figure 7. User list page](image1)

![Figure 8. Profile page](image2)
The "Graduate Learning Outcomes List" page showcases a categorized list of added graduate learning outcomes based on their aspects. For instance, Figure 11 presents a list where attitude is one of the emphasized aspects. The "Add Graduate Learning Outcome" page, depicted in Figure 12, contains several input fields. To accommodate multiple graduate learning outcomes, these fields can be augmented by pressing the "Add Field" button in the Action column. Figure 13's "Dashboard" card page exhibits an aggregate count of users and curricula, among other elements, and features a graph that is adjustable based on the curriculum. This visual representation illustrates the quantity...
of Graduate Learning Outcomes (CPL), Courses (MK), Semester Learning Plans (RPS), and available questions. Notably, it highlights the frequency of CPL integrated within each course. Lastly, the “Curriculum” page in Figure 14 facilitates the addition and review of previously incorporated curricula.

Figure 11. Graduate Learning Outcome list page

Figure 12. Add Graduate Learning Outcome page

Figure 13. Dashboard page
2nd System Design

The design was revised based on feedback from the initial development phase, leading to alterations in the Entity Relationship Diagram (ERD) depicted in Figure 15. Moreover, the use-case diagram underwent adjustments to include the newly added features within the admin module, as illustrated in Figure 16.

![Figure 14. Curriculum page](image1)

![Figure 15. 2nd Entity Relationship Diagram](image2)

2nd Development

The development of this system employs the RAD method, emphasizing the reuse of pre-existing components. For instance, the RPS pages in the lecturer module come equipped with create, edit, and delete features. These components will be integrated into the admin module, supplemented with additional components. The "Add RPS" page within the admin module is illustrated in Figure 17.

![Figure 16. 2nd use-case diagram](image3)

Similarly, the CPMK page, initially found in the lecturer module, has been incorporated into the admin module. The "Add CPMK" page and the "CPMK List" page in the admin module are depicted in Figures 18 and 19, respectively.

The CPLMK page from the lecturer module has also been adapted for use in the admin module. The "Add CPLMK" page and the "CPLMK List" page are presented in Figures 20 and 21, respectively.
Figure 17. Add RPS page

Figure 18. Add CPMK page

Figure 19. CPMK list page
Implementation

The system has fulfilled the user requirements based on the planned needs and feedback during the development process. The next step is to conduct testing to identify any errors or discrepancies in the system.

System Testing

The system was tested using the black-box testing method to ensure that the developed system aligns with the expected outcomes. This alpha testing was conducted internally by the system developers. The testing results for the lecturer module are presented in Table 1, Table 2, Table 3, and Table 4.

<table>
<thead>
<tr>
<th>Test List</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login Page</td>
<td>Display login page</td>
<td>Display login page</td>
<td>Valid</td>
</tr>
<tr>
<td>Login Function</td>
<td>Log into the system</td>
<td>Log into the system</td>
<td>Valid</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Test List</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>List User Page</td>
<td>Display a list of users with the authority of dosen and penjamin mutu</td>
<td>Display a list of users with the authority of dosen and penjamin mutu</td>
<td>Valid</td>
</tr>
<tr>
<td>Edit Button</td>
<td>Redirect to the user edit form page for the selected user</td>
<td>Redirect to the user edit form page for the selected user</td>
<td>Valid</td>
</tr>
<tr>
<td>Delete Button</td>
<td>Delete the selected user</td>
<td>Delete the selected user</td>
<td>Valid</td>
</tr>
<tr>
<td>Add User Page</td>
<td>Display the user addition form</td>
<td>Display the user addition form</td>
<td>Valid</td>
</tr>
<tr>
<td>Kurikulum Page</td>
<td>Display the curriculum addition form and a list of added curricula</td>
<td>Display the curriculum addition form and a list of added curricula</td>
<td>Valid</td>
</tr>
<tr>
<td>Edit Button</td>
<td>Transform the curriculum list column into an edit-form field</td>
<td>Transform the curriculum list column into an edit-form field</td>
<td>Valid</td>
</tr>
</tbody>
</table>

Table 1. Normal testing on authentication testing

Table 2. Normal testing on functionality testing
### Table 3. Abnormal testing on authentication testing

<table>
<thead>
<tr>
<th>Test List</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empty Email and Password</td>
<td>Failed login attempt</td>
<td>Failed login attempt</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect Email and Password</td>
<td>Failed login attempt</td>
<td>Failed login attempt</td>
<td>Valid</td>
</tr>
</tbody>
</table>

### Table 4. Abnormal testing on functionality testing

<table>
<thead>
<tr>
<th>Test List</th>
<th>Expected Result</th>
<th>Actual Result</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect format in adding user data</td>
<td>Failed to add user</td>
<td>Failed to add user</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in editing user data</td>
<td>Failed to edit user</td>
<td>Failed to edit user</td>
<td>Valid</td>
</tr>
<tr>
<td>Deleting a curriculum with associated CPL/MK</td>
<td>Failed to delete curriculum</td>
<td>Failed to delete curriculum</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in adding curriculum data</td>
<td>Failed to add curriculum</td>
<td>Failed to add curriculum</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in adding CPL data</td>
<td>Failed to add CPL</td>
<td>Failed to add CPL</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in editing CPL data</td>
<td>Failed to edit CPL</td>
<td>Failed to edit CPL</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in adding MK data</td>
<td>Failed to add MK</td>
<td>Failed to add MK</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in editing MK data</td>
<td>Failed to edit MK</td>
<td>Failed to edit MK</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in adding CPMK data</td>
<td>Failed to add CPMK</td>
<td>Failed to add CPMK</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in editing CPMK data</td>
<td>Failed to edit CPMK</td>
<td>Failed to edit CPMK</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in adding CPLMK data</td>
<td>Failed to add CPLMK</td>
<td>Failed to add CPLMK</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in editing CPLMK data</td>
<td>Failed to edit CPLMK</td>
<td>Failed to edit CPLMK</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in adding RPS data</td>
<td>Failed to add RPS</td>
<td>Failed to add RPS</td>
<td>Valid</td>
</tr>
<tr>
<td>Incorrect format in editing RPS data</td>
<td>Failed to edit RPS</td>
<td>Failed to edit RPS</td>
<td>Valid</td>
</tr>
</tbody>
</table>
Based on the test results, we can conclude that the system meets the expected scenarios. The Teaching Quality Management System, equipped with the Admin Module, effectively addresses the outlined challenges and aligns with the set requirements. The internal development team conducted the testing, and the system was found to be free of bugs or errors.

The Admin Module fortifies administrators’ capability to oversee accounts with diverse privileges, guaranteeing effective access control and user governance within the Quality Management System in Teaching [32]–[34]. This system permits role and permission allocation, fostering a safeguarded and systematized environment [35], [36]. Furthermore, curriculum management becomes seamless, ensuring its pertinence and synchronization with pedagogical objectives. The module simplifies the management of Graduate Learning Outcomes (CPL) and Course Learning Outcomes (CPMK), confirming that academic programs align with requisite benchmarks. Additionally, it facilitates the crafting and supervision of Semester Learning Plans (RPS), enriching the academic journey. Including CPL in Courses (CPLMK) lets administrators oversee the realization of educational aims across diverse courses.

The Admin Module’s development hinged on the Rapid Application Development (RAD) methodology, prioritizing teamwork, cyclical development, and swift prototyping. This modus operandi fostered efficient creation, punctuated by continuous stakeholder engagement and feedback. Comprehensive black-box testing was undertaken to vouch for the module’s robustness and dependability [37]. Assessing the system’s external functionality without delving into its inner workings led to discovering and rectifying potential performance disparities [38], [39].

To sum up, this research heralded the successful inception of the Admin Module for the Quality Management System in Teaching. Crafted as a website via the Laravel framework, this module equips administrators with the tools required to oversee various system components proficiently. Through applying the Rapid Application Development (RAD) technique, coupled with extensive black-box testing, the module consistently aligns with the anticipated functionality based on predefined scenarios. This enables administrators to adeptly manage accounts with varying privileges, curricula, CPL, CPMK, RPS, and CPLMK.

REFERENCES


