Journal of Students' Research in Computer Science

e-ISSN: 2722 – 290X
Vol. 4 No. 2 (November 2023), Page: 199 – 214
https://doi.org/10.31599/jsrcs.v4i2.2714
Available Online at http://ejurnal.ubharajaya.ac.id/index.php/JSRCS

Web Application for Providing Score Access to Facilitating Students' Performance

Kelvin Asclepius Minor [1,*]

* Corresponding Author Email: kelvin.minor@binus.ac.id

[1] Mathematics Department; Bina Nusantara University; Kebon Jeruk Raya No. 27, Jakarta Barat, 11530, (+6821) 21 534 5830; e-mail: kelvin.minor@binus.ac.id

Submitted	: **11 Agustus 2023**
Revised	: **14 September 2023**
Accepted	: **18 Oktober 2023**
Published	: **30 November 2023**

Abstract

Promoting student engagement is a primary goal in education, and educators have identified strategies to enhance student involvement. Among these strategies, providing access to scores has emerged as a vital component. Scores, as indicators of academic performance, hold a significant influence on student participation. When students work diligently to achieve good grades or scores, they not only experience a sense of accomplishment, but the scores also fuel their enthusiasm and commitment towards their studies. Providing students with access to their scores is significant for several reasons. It fosters a positive classroom environment and positively impacts student performance. Sharing scores on Microsoft Excel in the educational system offers convenience in organizing and sharing student scores, but may lead to negative consequences, including unhealthy competition and privacy concerns. The development of web applications to provide access to students' scores ensures improved data security and privacy through authentication measures, while also encouraging self-assessment, motivation, accountability, and communication.

***Keywords**: Application, Education, Performance, Score, Student*

Abstrak

Meningkatkan keterlibatan mahasiswa merupakan tujuan utama dalam pendidikan, dan para pendidik telah mengidentifikasi strategi untuk meningkatkan keterlibatan mahasiswa. Salah satu strategi tersebut adalah memberikan akses pada nilai yang merupakan komponen penting. Nilai, sebagai indikator kinerja akademik, memiliki pengaruh yang signifikan terhadap partisipasi mahasiswa. Ketika mahasiswa bekerja dengan tekun untuk mencapai nilai yang baik, mereka tidak hanya memiliki rasa pencapaian, tetapi nilai yang didapat juga membangkitkan semangat dan komitmen mereka terhadap studi mereka. Memberikan akses kepada mahasiswa untuk melihat nilai, memiliki signifikansi dalam beberapa hal. Hal ini dapat menciptakan lingkungan kelas yang positif dan berdampak positif terhadap kinerja siswa. Memperlihatkan nilai dengan Microsoft Excel dalam sistem pendidikan memang memudahkan dalam mengorganisir dan pembagian informasi, tetapi dapat menyebabkan konsekuensi negatif, termasuk kompetisi yang tidak sehat dan kekhawatiran akan privasi. Pengembangan aplikasi web untuk memberikan akses kepada nilai memastikan peningkatan keamanan dan privasi data melalui langkah-langkah autentikasi, dan juga mendorong penilaian diri, motivasi, akuntabilitas, dan komunikasi.

Kata kunci: Aplikasi, Edukasi, Mahasiswa, Nilai, Performa

1. Introduction

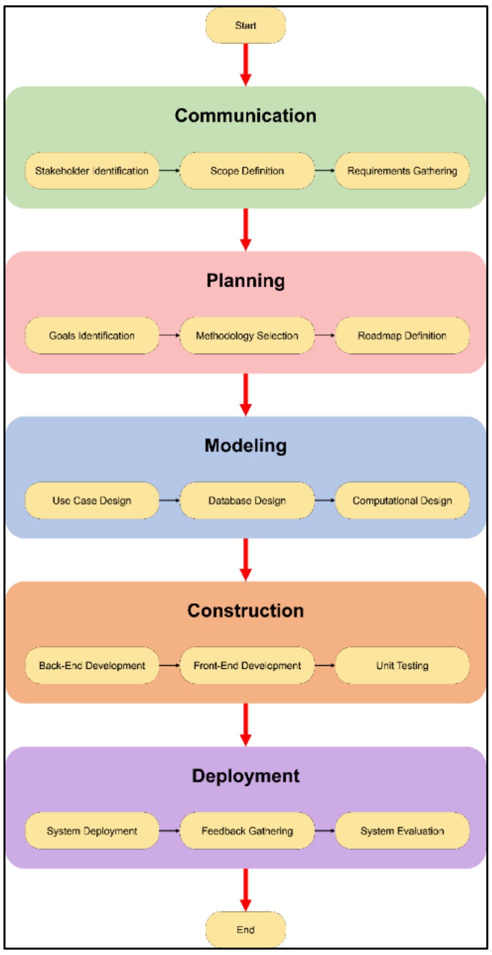
Scores play a crucial role in motivating students to participate actively in their studies (Delfino, 2019). Providing students with access to their scores fosters a positive classroom environment and positively impacts student motivation (Meşe & Sevilen, 2021). Lack of motivation in classroom activity is a common problem among today's generation of students (Briones, Dagamac, David, & Landerio, 2021). Allowing students to review their scores empowers them to engage in self-assessment and understand their academic performance better. Enabling students to access their scores provides valuable feedback, aiding their learning and understanding of the content (l-Darei & Ahmed, 2022). Feedback and communication are crucial in education, improving the learning process and boosting student motivation (Akhtar, Hussain, Afzal, & Gilani, 2019). Students perceive the information as positive news, potentially boosting their confidence and satisfaction with their academic performance (Azmat, Bagues, Cabrales, & Iriberri, 2019).

In the face of the COVID-19 pandemic, the education sector needs to embrace emerging digital technologies that have become more crucial than before (Onyema, et al., 2020). Sharing scores on Excel is a convenient way to utilize digital information in the context of academic performance. By using Excel, lecturers can easily input and organize students' scores, making it convenient to share. Sharing scores digitally enables students to receive feedback from lecturers or peers, facilitating transparency and competition among students. On the other hand, sharing scores openly can raise privacy concerns and may lead to feelings of inadequacy or unhealthy competition, and some students may prefer to keep their academic performance confidential. Unhealthy competition also creates a sense of pressure and may lead to stress, burnout, and a focus on external metrics rather than self development (Giselle, Terrell, David, & Ann-Gel, 2020).

A web application that provides students with access to their scores may offer several advantages compared to using Excel or other offline methods. The web application allows students to retrieve their information easily from any device with an internet connection. This eliminates the need for manual sharing of performance results and saving time for both students and lecturers. Besides that, web applications can ensure data security and privacy by implementing appropriate authentication. By allowing students to easily access their grades, the application can promote self-assessment, motivation, accountability, and communication to enhance the collaborative learning environment.

2. Research Methodology

In the research methodology chapter, the chosen approach for developing the application is the waterfall framework. The waterfall framework is a sequential and linear approach that guides the systematic development process (Pressman & Maxim, 2020). Since the project is not easily divisible into independent parts, agile approach may not be suitable to this project, and a classical waterfall approach is recommended (Thesing, Feldmann, & Burchardt, 2021).



Source: Research Results (2023)

Figure 1. Waterfall Framework in Scoreboard Web Application Development

The development process, as illustrated in Figure 1, follows the steps outlined in the waterfall framework. Each phase builds upon the outputs of the previous phase. Here are the steps involved in the waterfall framework of this research:

2.1. Communication

In this initial phase, effective communication is established to gather requirements from stakeholders. Identifying stakeholders is an important step in discussing focus on project requirements and constraints. Clear communication and documentation ensure shared understanding among all involved parties.

2.2. Planning

During this stage, the established requirements are allocated to a comprehensive project plan based on the gathered requirements. It involves identifying and defining project goals and objectives, selecting a suitable development methodology, and outlining key activities, tracks progress, and defines deliverables for the software development process.

2.3. Modeling

During this stage, stakeholder requirements are processed and translated into software systems. This includes allocating requirements to different parts of the software architecture and establishing user and system interactions using the Use Case Diagrams. For score management, a suitable database structure and data model are designed with an Entity Relationship Diagram (ERD). The stage also designs the algorithm that defines the logic and calculations to solve a specific problem and perform a specific task.

2.4. Construction

During this stage, the software design specifications is translated into program units. Back-End Development involves building server-side components using PHP, integrating with the MySQL database, and implementing required functionalities, such as Microsoft account integration for user authentication. Front-End Development focuses on creating a secure user interface user interface for both students and lecturers using HTML, CSS, and JavaScript, and integrating it with back-end APIs to access students' scores. Throughout the implementation stage, unit testing is conducted to ensure the application meets the requirements.

2.5. Deployment

During this stage, the system is deployed on the website for practical use, and maintenance activities are carried out to address any previously undetected errors and enhancing the system’s services as new requirements are discovered by gathering feedback from users.

3. Results and Discussions

This chapter utilizes the waterfall framework to present the results and discussion of the application development process. It focuses on providing students and lecturers access to students' grades through the Scoreboard application. The chapter describes the outcome of each phase in the waterfall framework.

3.1. Communication

In the communication phase, the main objective is to gather and document the necessary information and specifications for the application. The following points are included in the communication process.

a. Stakeholders Identification
   Stakeholder identification involves categorizing individuals based on their roles and responsibilities in the application. The outcome of this identification reveals three key stakeholders for the application: Students, Lecturers, and Administrator.

b. Scope Definition
   The scope of the application encompasses the utilization of the user's Microsoft account, which aligns with the account system utilized by Bina Nusantara University. This integration allows users to access the application using their existing Microsoft account credentials.

c. Functional Requirements
   After identifying the stakeholders and defining the scope, the comprehensive list of functionalities for the application is documented and organized in Table 1.

Table 1. Functionalities of the Application

No.	Functionality	Stakeholders	Description
1.	Add Period	Administrator	add new academic period
2.	Add Lecturer	Administrator	add new lecturer
3.	Add Class	Lecturer	add new class
4.	Add Student	Lecturer	add new student or update existing student
5.	Add Class Components	Lecturer	define the class's score structure
6.	Add Class Sub-Components	Lecturer	divide class components into sub-components
7.	Set Component Calculation	Lecturer	configure the calculation of the class component based on the sub-components
8.	Download Scores	Lecturer	download scores in CSV
9.	Upload Scores	Lecturer	upload scores into the system in bulk using CSV format
10.	View Scores	Lecturer & Student	view student scores

Source: Research Results (2023)

3.2. Planning

In the planning phase, the main objective is to establish the objectives of the application and determine the software development methodology. The following points are included in the planning process:

a. Goals Definition

In the goals definition phase, the primary objective is to understand the purpose and desired outcomes of the application and align them with the needs and expectations of the stakeholders. These objectives include: (1) Self-Assessment: The application enables students to assess their performance to make decisions about study habits and learning strategies; (2) Motivation: The application acts as a source of motivation by reinforcing efforts and providing a sense of accomplishment for positive grades and prompting students to seek support and work towards improvement for lower grades; (3) Accountability: The application fosters a sense of responsibility for learning outcomes, where students recognize that their efforts directly impact their grades; and (4) Communication: The application provides a means of communication between students and lecturers, allowing students to assess their performance and address any concerns about the accuracy or fairness of their grades.

b. Methodology Selection

In the Methodology Selection section, the waterfall framework is chosen for the software development process. The waterfall framework gives clear project structure because it follows a sequential and structured approach. This can provide clarity and make it easier to plan the development process, especially for projects with fixed requirements. This advantage is suitable for one man programmer when the project has well-defined and stable requirements.

c. Project Roadmap

In the waterfall framework, roadmap definition is integral phase that serves as a structured guide for the project execution as illustrated in Figure 1.

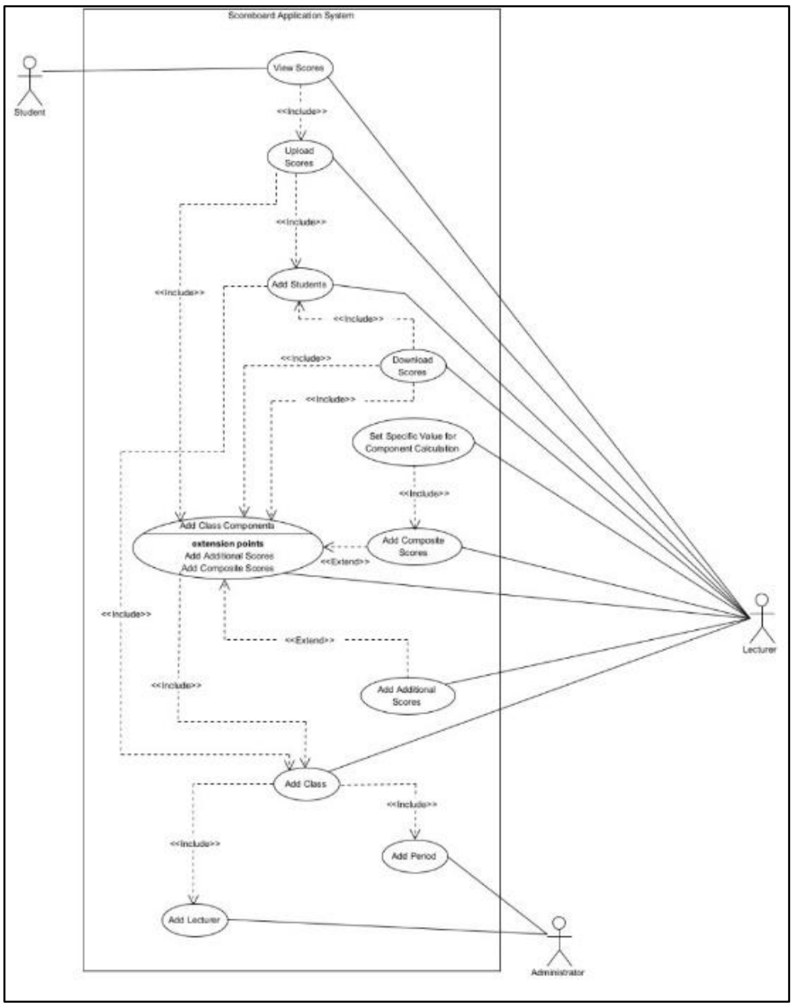
3.3. Modeling

In the modeling phase, the main objective is to convert and assign established requirements to software systems. The following points are included in the modeling process:

a. Use Case Diagram

The Use Case Diagram of Scoreboard Application can be shown in Figure 2. In the Use Case Diagram, the actors represent the different roles interacting with the system. The actors in the Use Case Diagram are Administrator, Lecturer, and Student.

Administrators have two main responsibilities: adding new academic periods (referring to academic terms) and adding new lecturers who will be responsible for teaching classes. Lecturers have multiple actions they can perform, including adding new classes with specified, adding class components like exams and assignments, calculating composite scores based on class components with specific values, adding supplementary scores, enrolling students in their classes, and managing scores by downloading, uploading, and viewing them. The Students have the capability to view scores, which scores are uploaded by the Lecturer.

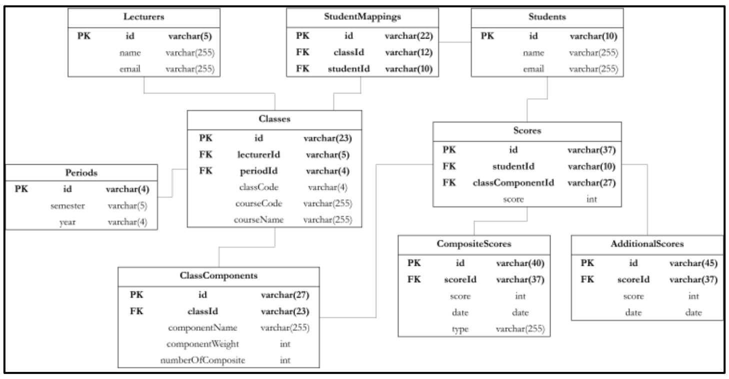


Source: Research Results (2023)

Figure 2. Use Case Diagram of Scoreboard Application

b. Entity Relational Diagram

The Entity Relational Diagram of Scoreboard Application can be shown in Figure 3. The Entity Relational Diagram reveals the presence of nine tables that play a crucial role in storing and organizing the data within the Scoreboard application.



Source: Research Results (2023)

Figure 3. Entity Relational Diagram of Scoreboard Application

The database consists of several tables, including Lecturers, Students, Periods, Classes, ClassComponents, StudentMappings, Scores, CompositeScores, and AdditionalScores. The Lecturers table stores information about lecturers, while the Students table stores information about students. The Periods table contains information about academic semesters. It represents different periods during which classes are conducted. The Classes table contains information about individual classes. It represents different classes that are offered, typically associated with specific courses. The ClassComponents table represents the components of a class. It stores information about various components that contribute to the overall evaluation of a class. The StudentMappings table represents the mappings between students and classes. It associates students with the classes they are enrolled in. The Scores table stores individual scores associated with class components. It represents the scores obtained by students for different class components. The CompositeScores table stores composite scores associated with a specific score entry. It contains entries representing composite scores that are calculated based on multiple class components. The AdditionalScores table stores additional scores associated with a specific score entry. Each entry in this table represents an additional score given to a student for a particular class component.

c. Mathematical Computation

In this section, the focus is on designing and implementing algorithms that perform specific calculations or solve mathematical problems within the software application. This section involves the following calculation.

Calculating the score for a Class Component consists of only Composite Scores: The computation is performed when the Class Component consists of only Composite Scores. The calculation can be expressed mathematically as shown in Equation 1. The first step is to identify and select the top total_composites highest scores from the Composite Scores list for further processing. These selected scores are summed up, and the resulting sum is divided by total_composites to obtain the Class Component Score.

$$Component_{Composites} = \frac{\sum_{n=1}^{total_{composites}} \max\left(Composites_{Array}\right)[n]}{total\_composites} \quad (1)$$

where:

$Component_{Composites} = Class\ Component\ that\ contains\ Composite\ Scores\ only$

$Composites_{Array} = List\ of\ Composite\ Scores$

$total_{composites} = number\ of\ Composite\ Scores\ to\ be\ calculated$

Calculating the score for a Class Component consists of only Additional Scores: The computation is performed when the Class Component consists of only Additional Scores. The calculation can be expressed mathematically as shown in Equation 2. The first step involves summing all scores in the Additional Scores list. The resulting sum is then added to the Class Component Score. However, if the sum exceeds 100, the computation caps the value at 100.

$$Component_{Additionals} = \min\left(Component_{Score} + \sum_{n=1}^{total_{additional}} Additional_{Array}\,[n], 100\right) \quad (2)$$

where:

$Component_{Additionals} = Class\ Component\ that\ contains\ Additional\ Scores\ only$

$Composites_{Score} = the\ Class\ Component\ Score$

$Additional_{Array} = List\ of\ Additional\ Scores$

$total_{additional} = size\ of\ the\ list\ of\ Additional\ Scores$

Calculating the score for a Class Component consists of Composite Scores and Additional Scores: The computation is performed when the Class Component consists of Composite Scores and Additional Scores. The calculation can be expressed mathematically as shown in Equation 3. The first step involves identifying the highest scores from the Composite Scores list. Then, the top total_composites scores are selected for further processing. These selected scores are summed up, and the resulting sum is added by the sum of all scores in the Additional Scores list. Then the resulting sum is divided by total_composites to obtain the Class Component Score. However, if the sum exceeds 100, the computation caps the value at 100.

$$Component_{\substack{Composites\\Additionals}} = \min\left(\frac{\sum_{n=1}^{total_{composites}} \max(Composites_{Array})[n] + \sum_{n=1}^{total_{additional}} Additional_{Array}\,[n]}{total\_composites}, 100\right) \quad (3)$$

where:

$Component_{\substack{Composites\\Additionals}} =$

$Class\ Component\ that\ contains\ Composite\ Scores\ and\ Additional\ Scores$

$Composites_{Array} = List\ of\ Composite\ Scores$

$total_{composites} = number\ of\ Composite\ Scores\ to\ be\ calculated$

$Additional_{Array} = List\ of\ Additional\ Scores$

$total_{additional} = size\ of\ the\ list\ of\ Additional\ Scores$

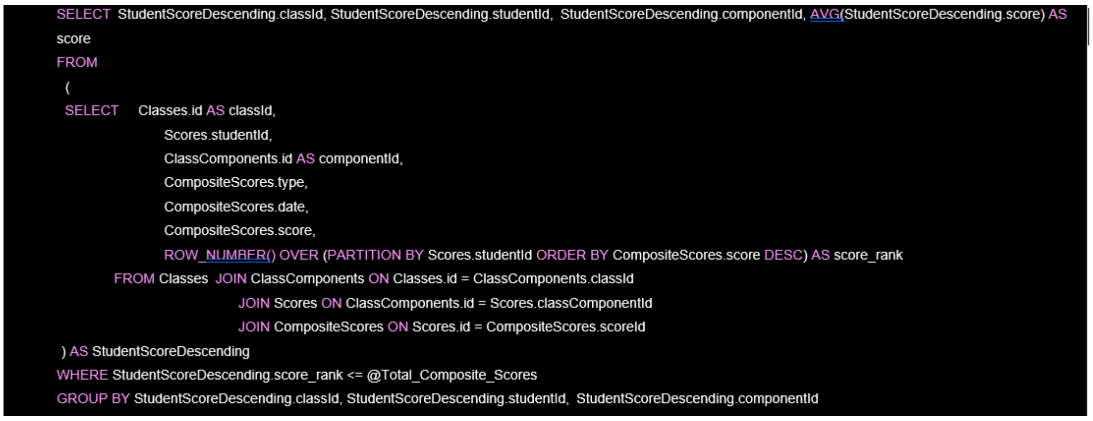
3.4. Construction

In the construction phase, the main objective is to translate the software design specification to the program unit. The following points are included in the construction process:

a. Back-end Development

Back-End Development for the Scoreboard application involves using PHP programming language to create server-side logic and handle operations. This includes integrating the application with a MySQL database to store and manage data related to student scores. The process translates mathematical computations in Chapter 3.3.3. into queries to perform calculations and retrieve relevant data from the database. By utilizing SQL, the process writes queries that perform calculations based on mathematical equations.

Calculating the score for a Class Component consists of only Composite Scores: To calculate the score for a Class Component consisting of only Composite Scores, the system utilizes the query shown in Figure 4. The query assigns a rank to each score within a specific student, ordering them in descending order. The outer query filters the results to include only scores with ranks up to a specified value. The query provides a way to obtain the average scores for students in each class and component, based on their highest ranked scores.

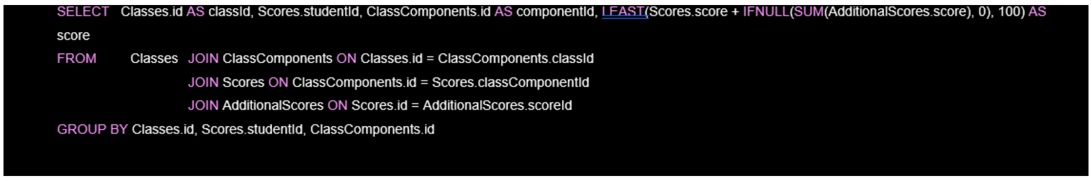


```
SELECT  StudentScoreDescending.classId, StudentScoreDescending.studentId,  StudentScoreDescending.componentId, AVG(StudentScoreDescending.score) AS
score
FROM
  (
  SELECT    Classes.id AS classId,
                    Scores.studentId,
                    ClassComponents.id AS componentId,
                    CompositeScores.type,
                    CompositeScores.date,
                    CompositeScores.score,
                    ROW_NUMBER() OVER (PARTITION BY Scores.studentId ORDER BY CompositeScores.score DESC) AS score_rank
          FROM Classes  JOIN ClassComponents ON Classes.id = ClassComponents.classId
                                JOIN Scores ON ClassComponents.id = Scores.classComponentId
                                JOIN CompositeScores ON Scores.id = CompositeScores.scoreId
 ) AS StudentScoreDescending
WHERE StudentScoreDescending.score_rank <= @Total_Composite_Scores
GROUP BY StudentScoreDescending.classId, StudentScoreDescending.studentId,  StudentScoreDescending.componentId
```

Source: Research Results (2023)

Figure 4. Query to Calculate the Class Component consists of only Composite Scores

Calculating the score for a Class Component consists of only Additional Scores: To calculate the score for a Class Component consisting of only Additional Scores, the system utilizes the query shown in Figure 5. This query retrieves the calculated scores for each student in a class for the Class Components, considering both the primary scores and any additional scores. The query also ensures that the final score does not exceed 100. The results are grouped by the class ID, student ID, and component ID.

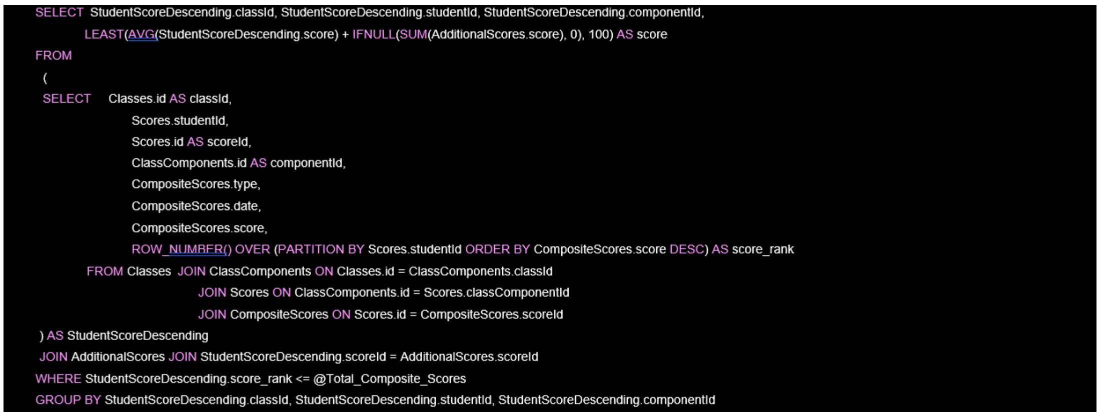


```
SELECT   Classes.id AS classId, Scores.studentId, ClassComponents.id AS componentId, LEAST(Scores.score + IFNULL(SUM(AdditionalScores.score), 0), 100) AS
score
FROM       Classes  JOIN ClassComponents ON Classes.id = ClassComponents.classId
                    JOIN Scores ON ClassComponents.id = Scores.classComponentId
                    JOIN AdditionalScores ON Scores.id = AdditionalScores.scoreId
GROUP BY Classes.id, Scores.studentId, ClassComponents.id
```

Source: Research Results (2023)

Figure 5. Query to Calculate the Class Component consists of only Additional Scores

Calculating the score for a Class Component consists of Composite Scores and Additional Scores: To calculate the score for a Class Component consisting of Composite Scores and Additional Scores, the system utilizes the query shown in Figure 6. This query combines the functionality of the previous two queries into a single system, with a slight change in adding Additional Scores. The query also ensures that the final score does not exceed 100. The results are grouped by the class ID, student ID, and component ID. The query includes only scores with ranks up to a specified value.



```
SELECT  StudentScoreDescending.classId, StudentScoreDescending.studentId, StudentScoreDescending.componentId,
          LEAST(AVG(StudentScoreDescending.score) + IFNULL(SUM(AdditionalScores.score), 0), 100) AS score
FROM
  (
  SELECT    Classes.id AS classId,
                 Scores.studentId,
                 Scores.id AS scoreId,
                 ClassComponents.id AS componentId,
                 CompositeScores.type,
                 CompositeScores.date,
                 CompositeScores.score,
                 ROW_NUMBER() OVER (PARTITION BY Scores.studentId ORDER BY CompositeScores.score DESC) AS score_rank
          FROM Classes  JOIN ClassComponents ON Classes.id = ClassComponents.classId
                              JOIN Scores ON ClassComponents.id = Scores.classComponentId
                              JOIN CompositeScores ON Scores.id = CompositeScores.scoreId
 ) AS StudentScoreDescending
 JOIN AdditionalScores JOIN StudentScoreDescending.scoreId = AdditionalScores.scoreId
WHERE StudentScoreDescending.score_rank <= @Total_Composite_Scores
GROUP BY StudentScoreDescending.classId, StudentScoreDescending.studentId, StudentScoreDescending.componentId
```

Source: Research Results (2023)

Figure 6. Query to Calculate the Class Component consists of Composite and Additional Scores

Additionally, one of the key functionalities to be implemented in Scoreboard application is the integration of Microsoft account authentication. This involves utilizing APIs or libraries provided by Microsoft to enable users to authenticate and log in to the application using their Microsoft account credentials.

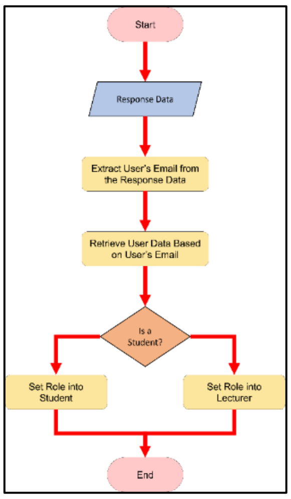
The process begins with Microsoft Application Registration on the Azure Portal, where details such as application name, platform, and redirect URL are specified, generating unique app_id and tenant_id for authentication. In the subsequent Microsoft Application Initialization step, PHP variables like application ID, tentant ID, redirection URL, and login URL are configured, forming essential components for authentication.

Following the setup, Authentication URL Building involves creating a parameter array defining crucial parameters for authentication requests, facilitating the interaction with Microsoft's authentication service. After redirection, the code checks for URL fragments and handles them with JavaScript, ensuring PHP can extract the access token from the URL.

Subsequently, the Access Token Validation phase verifies the presence of the access token in the URL parameters. If successful, the script proceeds to Access Token Processing, where the token is stored for future use, and a cURL request is made to the Microsoft Graph API for authenticated actions.

The last step encompasses the process of integrating user data from a database into the application by retrieving and processing the relevant information based on the user's email as

Illustrated in Figure 7. The process extracts the user's email from the response data and execute a database query to retrieve user data from the "Students" and "Lecturers" tables. If a matching row is found, the user's ID, name, and email are extracted and stored. The role of the user is determined by checking if the user's email exists in either the "Lecturers" or "Students" table in the database. The user's data is then stored in the session variable, allowing it to be accessed across the application.


Start
Response Data
Extract User's Email from the Response Data
Retrieve User Data Based on User's Email
Is a Student?
Set Role into Student
Set Role into Lecturer
End

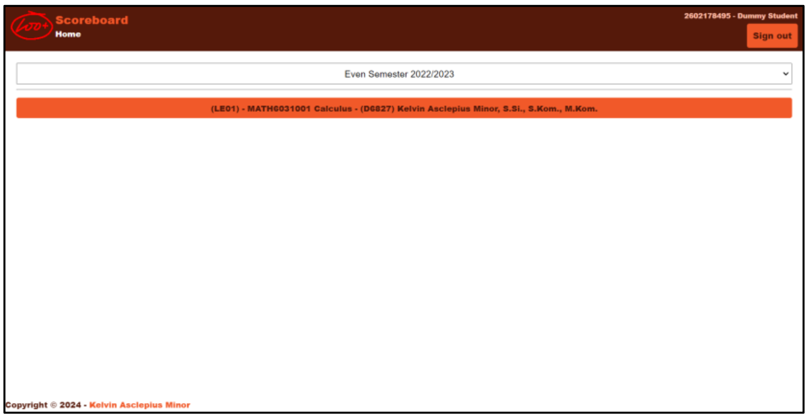
Source: Research Results (2023)

Figure 7. Flowchart of Database User Integration

b. Front-end Development

Front-End Development encompasses the creation of a secure and user-friendly interface for both students and lecturers. This involves utilizing technologies such as HTML, CSS, and JavaScript to integrate the front-end interface with back-end APIs to access and display students' scores.

The design for each page of the website includes a header section at the top, featuring clickable buttons for easy navigation to different sections or pages, as shown in Figure 8. The header also displays the user's name, adding a personalized touch. Additionally, the design includes a footer at the bottom, providing information about the creator for acknowledgment.



2602178495 - Dummy Student

Even Semester 2022/2023

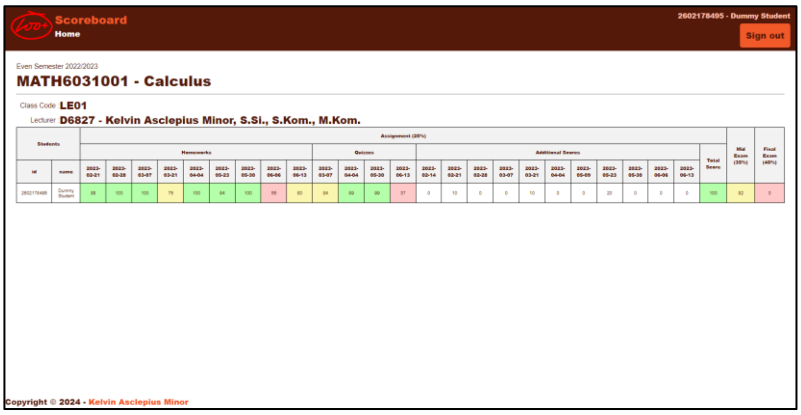
(LE01) - MATH6031001 Calculus - (D6827) Kelvin Asclepius Minor, S.Si., S.Kom., M.Kom.

Copyright © 2024 - Kelvin Asclepius Minor

Source: Research Results (2023)

Figure 8. Basic Elements Design on Scoreboard Application

The website includes a feature that displays the scores of students, and these scores are represented using a color-coded system, as shown in Figure 9. This system utilizes different colors to indicate the level of safety associated with each score. The use of colors allows for the efficient identification of performance levels (Good, Average, and Poor), encouraging timely action or recognition of achievement.



Even Semester 2022/2023

MATH6031001 - Calculus

Class Code : **LE01**

Lecturer : **D6827 - Kelvin Asclepius Minor, S.Si., S.Kom., M.Kom.**

Students		Assignment (25%)																								Mid Exam (35%)	Final Exam (40%)
		Homeworks									Quizzes				Additional Scores											Total Score		
id	name	2023-02-21	2023-02-28	2023-03-07	2023-03-21	2023-04-04	2023-05-23	2023-05-30	2023-06-06	2023-06-13	2023-03-07	2023-04-04	2023-05-30	2023-06-13	2023-02-14	2023-02-21	2023-02-28	2023-03-07	2023-03-21	2023-04-04	2023-05-09	2023-05-23	2023-05-30	2023-06-06	2023-06-13			
2602178495	Dummy Student	88	100	100	75	100	94	100	55	80	84	89	99	37	0	10	0	0	10	0	0	20	0	0	0	100	82	0

Copyright © 2024 - Kelvin Asclepius Minor

Source: Research Results (2023)

Figure 9. Color-Coded System on Scoreboard Application

c. Unit Testing

The unit testing process for the Scoreboard Application involves implementing several test cases that must be successfully passed. The test cases presented in Table 2 are designed to evaluate the application's functionality and performance in providing access to scores.

Table 2. Functionalities of the Application

Units	Step Actions	Verifications	Status
Sign in	Sign in using Administrator credential	Verify that Home menu, Periods menu, and Lecturers menu are present	PASSED
	Sign in using Lecturer credential	Verify that Home menu and Classes menu is present	PASSED
	Sign in using Student credential	Verify that Home menu is present	PASSED
	Sign in using unknown credential	Show the error page	PASSED
Sign out	Click Sign out button	Redirect to Login Page	PASSED
Home Menu	Access Home Menu using Administrator, Lecturer, or Student role	Redirect to Home Page and verify that Semester Selection dropdown is present	PASSED
	Access Home Menu using unknown role	Redirect to Login Page	PASSED
Semester Selection Dropdown	Select Semester using Administrator role	Display all the class transactions in the semester	PASSED
	Select Semester using Lecturer role	Display all the class transactions in the semester that the Lecturer teaches	PASSED
	Select Semester using Student role	Display all the class transactions in the semester in which the student is enrolled	PASSED
	Select Semester using unknown role	Redirect to Login Page	PASSED

Units	Step Actions	Verifications	Status
Class Transaction Selection	Select Class Transaction using Administrator, Lecturer, or Student role	Redirect to View Class's Scores Page	PASSED
	Select Class Transaction using unknown role	Redirect to Login Page	PASSED
View Class's Scores Page	Access Page using Administrator role	Display all the class components and student's score	PASSED
	Access Page using Lecturer role	Display all the class components, student's score, and all buttons related to score management	PASSED
	Access Page using Student role	Display the class components of the students along with their respective scores.	PASSED
Add Students	Insert a nonexistent student	Insert the new student and insert a score of 0 for all the scores of each component.	PASSED
	Insert to an existing student id	Update the existing student's email.	PASSED
Add Sub-component Score	Selecting Composite Sub-Component, Input Composite Type Name, and Input Date	Insert a score of 0 for all students of the inserted Composite Sub-Component	PASSED
	Selecting some of data	Stay on Add Sub-component Score	PASSED
	Selecting Additional Sub-Component and Input Date	Insert a score of 0 for all students of the inserted Additional Sub-Component	PASSED
	Selecting Additional Sub-Component	Stay on Add Sub-component Score	PASSED
Add Composite Component Score	Input Date	Insert a score of 0 for all students of the inserted Composite Sub-Component	PASSED
	Leave the Date empty	Stay on Add Composite Component Score	PASSED
Add Additional Component Score	Input Date	Insert a score of 0 for all students of the inserted Additional Sub-Component	PASSED
	Leave the Date empty	Stay on Add Additional Component Score	PASSED
Download Scores Template	Click Download Template Button for each Sub-Components	Retrieve a csv file with the existing scores	PASSED
Upload Scores	Upload Scores	Update all scores and redirect to View Class's Scores Page	PASSED
Periods Menu	Access Periods menu using Administrator role	Redirect to Add Periods Menu and verify that Semester type dropdown and Year selection are present	PASSED
	Access Periods menu using non Administrator role	Redirect to Home Page	PASSED
Add Period	Select Semester Type (Odd, Even, or Short) and Select Year	Redirect to Home Page and verify that new Semester is available in the Semester Selection dropdown	PASSED
	Select uncomplet data	Stay on Add Period Menu	PASSED
Lecturers Menu	Access Lecturers menu using Administrator role	Redirect to Add Lecturers Menu and verify that text fields are present for entering the Lecturer's ID, Lecturer's Name, and Lecturer's Email	PASSED
	Access Lecturers menu using non Administrator role	Redirect to Home Page	PASSED

Units	Step Actions	Verifications	Status
Add Lecturer	Input Lecturer's ID, Lecturer's Name, and Lecturer's Email	Redirect to Home Page	PASSED
	Input uncompleted data	Stay on Add Lecturer Menu	PASSED
Classes Menu	Access Classes menu using Administrator role	Redirect to Home Page	PASSED
	Access Classes menu using Lecturer role	Redirect to Add Classes Menu and verify that Semester Selection dropdown, text fields for Class Code, Course Code, Course Name, and minimum score are present	PASSED
	Access Classes menu using Student or unknown role	Redirect to Home Page	PASSED
Add Class	Select Semester, Input Class Code, input Course Code, input Course Name, and input minimum score	Redirect to Class Components Menu	PASSED
	Input uncomplete data	Stay on Add Class Page	PASSED
Class Components Menu	Access Class Components menu using Administrator role	Redirect to Home Page	PASSED
	Access Class Components menu using Lecturer role	Redirect to Add Class Components Menu and verify that text fields are present for entering the Class Component's Name and Class Component's Weight and Submit button	PASSED
	Access Class Components menu using Student role	Redirect to Home Page	PASSED
	Access Class Components menu using unknown role	Redirect to Login Page	PASSED
Add Class Components	Input Class Component's Name and Class Component's Weight	The inputed Class Component's Name and Class Component's Weight is displayed on the page	PASSED
	Input Class Component's Name only	The inputed Class Component's Name is not displayed on the page	PASSED
	Input Class Component's Weight only	The inputed Class Component's Weight is not displayed on the page	PASSED
Add Class Components Submission	Click Submit Button when total of Component's Weight equals to 100	Redirect to View Class's Scores Page	PASSED
	Click Submit Button when total of Component's Weight does not equal to 100	Stay on the Class Components Page	PASSED

Source: Research Results (2023)

3.5. Deployment

In the deployment phase, the main objective is to deploy the system on the website. The following points are included in the deployment process:

a. Project Deployment

The Scoreboard project has been deployed using the Hostinger hosting service and currently accessible at the URL "https://minorkelvin.com/scoreboard".

b. Feedback Result

The evaluations for the application are gathered through a questionnaire-based approach. The questionnaire consists of a set of carefully crafted questions related to various aspects of the goals of the application, such as Self-Assessment, Motivation, Accountability, and Communication.

The feedback from the questionnaire indicates that most enrolled students agree on the benefits of checking grades in the course. The feedback reveals that 94% of students find checking grades valuable for self-assessment, 88% see grades as a motivational source, and 100% acknowledge the role of grades in fostering accountability. Moreover, 82% agree that grades facilitate communication with lecturers.

c. System Evaluation

The feedback collected from the questionnaire indicates that the users have positively evaluated the system. The high agreement percentages in areas such as self-assessment, motivation, accountability, and communication indicate that the system effectively meets the needs and expectations of the students. These findings suggest that the system has achieved its intended goal of providing students with a valuable tool for assessing their performance. The high agreement percentages indicate that the system has effectively addressed these aspects, enhancing the overall user experience.

4. Conclusion

Providing students with web-based access to their scores has proven to be a valuable strategy for promoting student engagement in education. The positive evaluation and feedback from users confirm the effectiveness of the system in meeting students' needs and expectations. The high agreement percentages in self-assessment, motivation, accountability, and communication highlight the system's success in fostering these crucial aspects of student engagement. Additionally, the positive feedback on user satisfaction with the application's speed and navigability further emphasize the system's positive impact on the overall user experience.

References

Akhtar, S., Hussain, M., Afzal, M., & Gilani, S. A. (2019). The Impact of Teacher-Student Interaction on Student Motivation and Achievement. *European Academic Research*.

Azmat, G., Bagues, M., Cabrales, A., & Iriberri, N. (2019). What You Don't Know…Can't Hurt You? A Natural Field Experiment on Relative Performance Feedback in Higher Education. *Management Science*.

Briones, S. K., Dagamac, R. J., David, J. D., & Landerio, C. A. (2021). Factors Affecting the Students' Scholastic Performance: A Survey Study. *Indonesian Journal of Educational Research and Technology*.

Delfino, A. P. (2019). Student Engagement and Academic Performance of Students of Partido State University. *Asian Journal of University Education*.

Giselle, L., Terrell, H., David, M., & Ann-Gel, P. (2020). Suspending Student Selections to Alpha Omega Alpha Honor Medical Society: How One School Is Navigating the Intersection of Equity and Wellness. *Academic Medicine*.

l-Darei, I. S., & Ahmed, A. M. (2022). The effect of feedback type in the e-learning environment on students' achievement and motivation. *Journal of Educational Technology & Online Learning*.

Meşe, E., & Sevilen, Ç. (2021). Factors influencing EFL students' motivation in online learning: A qualitative case study. *Journal of Educational Technology & Online Learning*.

Onyema, E. M., Eucheria, N. C., Obafemi, F. A., S. S., Atonye, F. G., Sharma, A., & Alsayed, A. O. (2020). Impact of Coronavirus Pandemic on Education. *Journal of Education and Practice*.

Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: a practitioner's approach, Ninth Edition.* New York: McGraw-Hill Education.

Thesing, T., Feldmann, C., & Burchardt, M. (2021). *Agile versus Waterfall Project Management: Decision Model for Selecting the Appropriate Approach to a Project.* The Netherlands: Elsevier.