


Automating Internship Data Management Processes in a Web-Based Application: A Case Study of FTI UNTAR

^{1,*} Maria Rosa Gosal, ² Wasino, ³ Tony 

^{1,2} Department of Information Systems, Faculty of Information Technology, Universitas Tarumanagara, Jakarta, Indonesia

* Corresponding Author: maria.825210014@stu.untar.ac.id

Abstract: The Faculty of Information Technology at Universitas Tarumanagara (FTI UNTAR) supports the “Merdeka Belajar-Kampus Merdeka” (MBKM) industrial internship program, providing students with valuable work experiences through partnerships with companies and accessible internship listings. However, the existing management system lacks comprehensive digital support, leaving many processes manual and prone to inefficiencies, delays, and data errors. To address these challenges, this paper presents the design of a web-based application that aims to streamline internship data management by automating core processes and reducing administrative overhead. The application leverages the iterative methodology within the Software Development Life Cycle (SDLC), allowing for flexible adjustments and continuous refinement through repeated cycles of development and testing. Following development, the application successfully passed User Acceptance Testing (UAT) with all user types, including students, faculty advisors, company mentors, and administrators. Additionally, a System Usability Scale (SUS) survey conducted specifically for student users resulted in an excellent final score of 90.42, graded as A+. This outcome confirms the application’s effectiveness in enhancing operational accuracy and accessibility, elevating the overall quality of internship management at the faculty.



Citation: Gosal, M. R., & Wasino, T. (2024). Automating internship data management processes in a web-based application: Study case: FTI UNTAR. *Iota*, 4(4). ISSN 2774-4353. <https://doi.org/10.31763/iota.v4i4.827>

Academic Editor : Adi, P.D.P

Received : Oktober, 13 2024

Accepted : November, 05 2024

Published : November, 26 2024

Publisher’s Note: ASCEE stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2024 by authors. Licensee ASCEE, Indonesia. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution Share Alike (CC BY SA) license(<https://creativecommons.org/licenses/by-sa/4.0/>)

Keywords: MBKM Industrial Internship, Data Management Automation, Web-Based Application, Software Development Life Cycle, System Usability Scale.

1. Introduction

In today's rapidly evolving job market, employers increasingly value candidates who possess not only theoretical knowledge but also practical, real-world experience. For university students, gaining hands-on experience before graduation has become a critical step toward securing employment and succeeding in their chosen careers. This need has led many educational institutions and industries to promote internship programs as a bridge between academic learning and the workplace. Recognizing this shift, the Indonesian Ministry of Education, under the leadership of Nadiem Makarim, launched the “Merdeka Belajar-Kampus Merdeka” (MBKM) initiative. One key component of MBKM is the industrial internship program, which allows students to immerse themselves in industry settings relevant to their studies [1]. This experience enables students to acquire practical skills, build professional networks, and gain insights into their prospective careers, which are essential for succeeding in a competitive job market.

Universitas Tarumanagara, particularly its Faculty of Information Technology (FTI UNTAR), has actively embraced all types of programs under the “Merdeka Belajar Kampus Merdeka” (MBKM) initiative. Among these, the industrial internship program stands out as one of the most popular and sought-after options, attracting the majority of students [24]. FTI UNTAR demonstrates its commitment to MBKM through strategic partnerships with numerous companies and by providing students

with access to relevant and impactful internship opportunities. The faculty's proactive approach not only focuses on securing placements but also on delivering a well-rounded learning experience. This includes regular mentorship and constructive feedback from industry professionals. By immersing students in real-world projects and fostering active collaboration between academia and industry, FTI UNTAR aims to prepare its graduates with the skills, expertise, and confidence required to excel in their professional careers.

However, while the MBKM internship program provides valuable opportunities for students, the current internship management system at FTI UNTAR faces significant challenges due to its reliance on manual processes. The existing system has limitations in covering the entire operational scope of the internship program, as it lacks flexibility in accommodating varying internship durations, such as 6 months, 12 months, or extended internships. Furthermore, essential functions, such as distributing internship information, creating logbooks, maintaining mentoring attendance records, administering evaluation forms, and sending reminders to mentors, are still managed through traditional methods. These manual procedures not only slow down administrative operations but also increase the risk of human error, leading to potential data inaccuracies. Additionally, the system struggles with managing comprehensive mentoring data, including detailed records of each student's mentoring progress, which are crucial for monitoring performance and ensuring accountability. These limitations are further compounded by the need to efficiently handle large volumes of data, essential for tracking student progress, recording evaluation metrics, and maintaining timely communication with all stakeholders involved in the internship process [2].

To address these challenges, this study introduces a web-based application specifically designed to streamline the management of industrial internships at FTI UNTAR. This proposed system aims to replace the current manual processes with automated workflows that minimize administrative burdens, reduce data entry errors, and enhance the accessibility and accuracy of internship records. By centralizing and digitalizing key tasks, the application will facilitate a more efficient and organized approach to internship management. This digital transformation is expected to benefit not only administrators and faculty supervisors but also the students and company mentors involved in the MBKM program. Through this application, FTI UNTAR seeks to create a more supportive and efficient environment for its students, aligning with the broader objectives of the MBKM initiative.

2. Literature Review

In developing the MBKM data management application for industrial internships at the Faculty of Information Technology, Universitas Tarumanagara, various studies were used as references. Previous studies have highlighted the importance of managing MBKM internship activities within educational institutions, but many focus on broad systems for general program management rather than specialized tools tailored for industrial internships. While some research centers on creating web-based platforms, the approach to user roles and system capabilities varies widely. For instance, some studies are limited to specific user roles, such as administrators and students, while others focus on single-agency frameworks or lack scalability for multi-stakeholder environments. These limitations often leave gaps in addressing the unique needs of industrial internships, such as direct mentor evaluations, detailed logbook management, or flexible tracking of mentoring attendance. A detailed comparison of these studies, their limitations, and the novelty introduced by this research is presented in Table 1, which highlights how this study addresses these gaps by providing a more versatile and comprehensive solution tailored specifically for MBKM industrial internships.

Table 1. State of the Art				
Author(s) and Year	Actors	Scope	Limitation	Novelty of This Study
Sudargo and Tony [3]	Admin and students	MBKM activities at FTI UNTAR	Limited to admin and students; does not support specific internship needs like evaluations by mentors or supervisors.	Includes additional roles (faculty advisors and company mentors) and allows evaluations directly from mentors. Focuses specifically on MBKM industrial internships with more detailed features tailored to internships.
Edy [4]	Admin, students, and faculty advisors	Focused on students interning in a government agency	Limited to a single government agency; lacks scalability for multiple partnerships and broader academic needs.	Expands to handle multiple corporate partners and academic stakeholders, with features like logbooks, mentoring attendance, and direct mentor evaluations.
Marpaung et al. [5]	Admin, students, and mentors	Focused on company- level internship data	Restricted to a single- company framework; not designed for multi- stakeholder environments like universities.	Supports diverse partnerships and includes features for both academic and corporate collaboration in managing internships.
Rumbang and Tony [25]	Admin and intern	Focused on managing attendance records and intern employee data at PT Sembilan Pilar Semesta	Focused solely on attendance; lacks broader internship features	Offers mentorship tracking, logbooks, and evaluations for a comprehensive internship management solution

The system introduced in this study was designed to address the specific challenges highlighted in previous research. By incorporating a broader range of user roles—administrator, student, faculty advisor, and company mentor—the application bridges gaps in existing solutions and ensures functionality that is directly aligned with the needs of MBKM industrial internships. This approach not only supports collaboration between academic institutions and multiple industry partners but also enhances operational efficiency through streamlined data management and communication tools. Furthermore, the integration of features like automated evaluation processes, flexible internship tracking, and detailed mentoring oversight demonstrates the system’s capability to provide a scalable and user-centric platform tailored for industrial internships.

In addition to drawing insights from previous studies, having a comprehensive understanding of specific theories is crucial to ensure that the system’s development and implementation align seamlessly with the established requirements. These theories serve as the foundational pillars that guide the application’s design and functionality. Below are some key theoretical concepts that underpin the development of the Web Application for MBKM Data Management in Industrial Internships at FTI UNTAR.

2.1 Application

Applications are programs that are developed specifically to make it easier for users to complete certain jobs, including data processing, analysis, and information management [6]. With applications, various activities and tasks can be carried out more efficiently and structured, following the objectives and functions desired by its users.

2.2 Website

Elgamar describes a website as a digital platform made up of interlinked pages that deliver information in diverse and dynamic forms [7]. This concept traces back to Tim Berners-Lee, a British computer scientist credited with creating the world's first website. His innovation laid the foundation for the modern web, allowing for the seamless sharing and access of information across interconnected pages.

2.3 The Merdeka Belajar-Kampus Merdeka (MBKM) program

The “Merdeka Belajar-Kampus Merdeka” (MBKM) program, initiated by Indonesia’s Ministry of Education, aims to develop skilled human resources across the country [8]. A key feature of MBKM is the flexibility it offers students, allowing them to take courses outside their main study program for one semester and participate in activities beyond campus for up to two semesters [9]. Universitas Tarumanagara actively supports and implements the MBKM program, which includes activities such as internships, teaching, research, humanitarian projects, entrepreneurship, independent studies, community service in rural areas, and student exchange programs. Figure 1 outlines the eight activity types within MBKM.

2.4 Industrial Internship

The industrial internship program within “Merdeka Belajar-Kampus Merdeka” (MBKM) provides students with valuable hands-on experience in the professional world. Through this program, students work as trainees at partner organizations for a designated period, applying their academic knowledge in real-world contexts [10]. A key benefit of the program is the opportunity for students to engage directly in an institution’s internal operations, gaining practical insights that enrich their learning. Furthermore, successful internships can lead to full-time employment offers, significantly enhancing students’ job market prospects. Supported by collaborations among the Ministry of Education, government agencies, and private companies, this program serves as a crucial link between academic and professional spheres, helping students build connections within their chosen industries [11].



Figure 1. Illustration of MBKM Program Activities [8]

2.5 Framework

A framework is a structured toolkit consisting of scripts, such as classes, functions, and libraries, designed to streamline a developer's work [12]. By using a framework, application design and development become faster, as many code components are pre-built and reusable. This reuse saves time and reduces redundant coding, allowing developers to concentrate on crafting application logic and features. Consequently, frameworks support more efficient and organized application development.

2.6 Database

Connolly and Begg describe a database as a logically connected collection of data organized to meet an organization's informational needs [13]. Beyond simply storing data, a database is structured to manage and arrange data for fast and efficient access [14]. This system facilitates essential data operations, such as storage, retrieval, modification, and deletion, supporting a wide range of data processing activities [15].

2.7 Unified Modelling Language

The Unified Modelling Language (UML) is a standardized approach for creating meaningful, object-oriented documentation models applicable to any real-world software system [16]. UML enables the development of detailed and comprehensive models that clearly and systematically illustrate the workings of both software and hardware systems. To achieve this clarity, UML offers a range of diagrams that visually represent different facets of a system.

2.8 User Acceptance Testing

User Acceptance Testing (UAT) is a process intended to verify that an application meets the requirements and expectations of its end users. This testing emphasizes the application's ability to perform as expected in real-world scenarios, ensuring that all functions align with the practical needs and workflows of users. Unlike other testing stages that may delve into code or internal structure, UAT centers solely on the user experience and functional accuracy from the perspective of intended usage [17].

2.9 System Usability Scale

The System Usability Scale (SUS) is a widely used standardized tool for assessing users' perceptions of an application's usability. Originally intended as a quick and simple measure, SUS has proven effective in delivering a comprehensive view of perceived usability without compromising measurement quality. Its relevance as a usability metric is well-established due to its popularity and sustained use in both research and practical usability assessments [18].

3. Method

In developing the MBKM data management application for industrial internships at the Faculty of Information Technology, Universitas Tarumanagara, the methodology used as a foundation is the System Development Life Cycle (SDLC). The SDLC offers a structured approach for managing projects at the conceptual level, guiding the progression from strategic planning to operational implementation [20]. This methodology is designed to facilitate the development, deployment, maintenance, and eventual decommissioning of information systems, following a well-defined set of stages to ensure an organized and efficient project flow.

The SDLC includes several models, each suited to different project needs. For this study, the iterative model was selected, as it emphasizes phased development that incrementally builds upon a basic version of the system [21]. Unlike linear models, the iterative approach does not require fully defined specifications from the outset. Instead, development begins with initial components, which are tested and refined based on feedback before proceeding to the next phase. Each iteration aims to produce a functional part of the application, gradually shaping it into a complete system. This cycle of repeated testing and refinement allows the project to adapt to evolving requirements, making it an effective methodology for creating applications

that benefit from continuous improvement and real-time feedback, as illustrated in Figure 2.

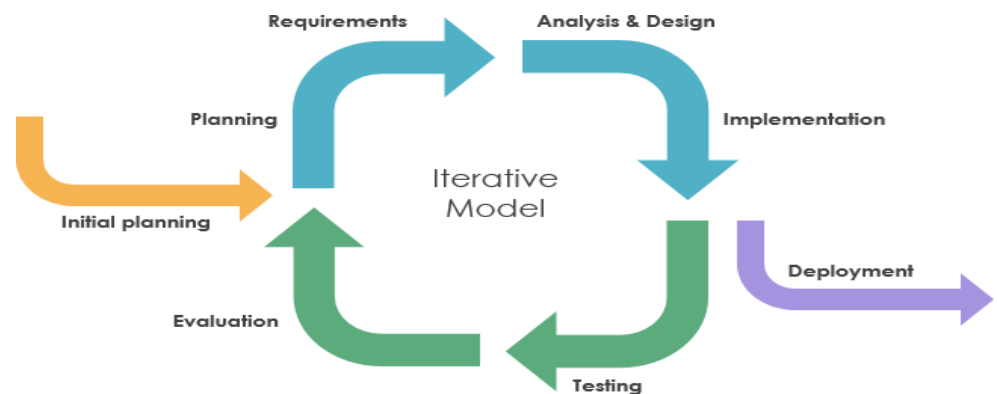


Figure 1. Illustration of the SDLC Iterative Model [19]

In the initial planning stage, initial planning is conducted to identify the general needs of the system. This process usually involves gaining a preliminary understanding of the project's objectives and the system's overall scope without detailing every specific requirement. The focus is on establishing a foundational direction for development, ensuring that all stakeholders share a common vision for the system's goals. Following initial planning, planning is undertaken for each iteration, focusing on specific plans for the features or modules to be developed in the current cycle. Each iteration requires a thorough plan outlining the features to be implemented, including task priorities, time estimates, and the necessary resources to achieve them.

After planning, the requirements stage involves detailed requirements gathering for the first iteration. This process includes interviews, surveys, and observations to identify both functional and non-functional requirements. The outcomes from this phase lay the groundwork for creating an effective and efficient system architecture that meets the needs of all stakeholders.

In the analysis and design phase, analysis and design are carried out based on the requirements gathered for each iteration. The system is designed on a smaller scale to allow flexibility for adding or modifying design elements in subsequent iterations. This modular approach to design enables updates in later iterations based on any evolving requirements or additional needs.

The Implementation stage in the iterative model is conducted in small units. Developers code the features designed in the previous iteration and integrate them into the system. This approach allows developers to focus on manageable portions of the project, reducing the complexity of each implementation phase. As each unit is completed and integrated, the system gradually evolves, making it easier to track progress and identify any issues early in the development process.

Each iteration concludes with a Testing phase, where newly implemented modules or features undergo a structured evaluation process. This phase ensures the system operates as expected, minimizing errors and maintaining stability and performance across iterations. Testing serves as a critical step in identifying potential issues before deployment, validating both functionality and reliability for end users [17]. Beyond identifying bugs, it also ensures that the application aligns with all specified functional requirements. The testing process in this project was structured into four levels: unit testing, integration testing, system testing, and User Acceptance Testing (UAT). Figure 3 illustrates the hierarchy and flow of these testing levels within the SDLC iterative model.

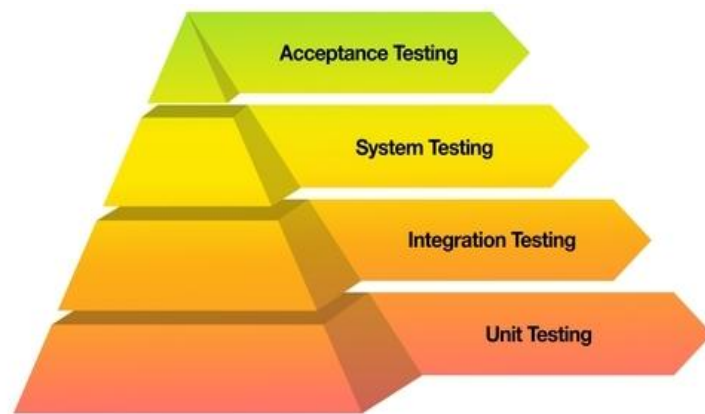


Figure 2. Levels of Testing [23]

The first three levels were performed by the developers to systematically verify that individual modules (unit testing) functioned correctly, that these modules interacted seamlessly with one another (integration testing), and that the overall system performed as intended (system testing). The final stage, UAT, involved real users, including students, faculty advisors, mentors, and administrators, testing the application to ensure that it met their expectations and was ready for deployment in live environments. For a clearer depiction of the testing processes carried out, refer to the detailed flowchart presented in Figure 4.

Finally, after completing multiple iterations and achieving sufficient system stability, the deployment phase takes place. In the Iterative Model, deployment can occur gradually, allowing completed portions of the system to be released to end users while additional features are developed in subsequent iterations. This staged deployment strategy facilitates ongoing improvements while providing users access to the most stable version of the system to date. The detailed activities within each SDLC stage are presented comprehensively in Figure 5, which illustrates the block diagram of the SDLC Iterative process for this study.

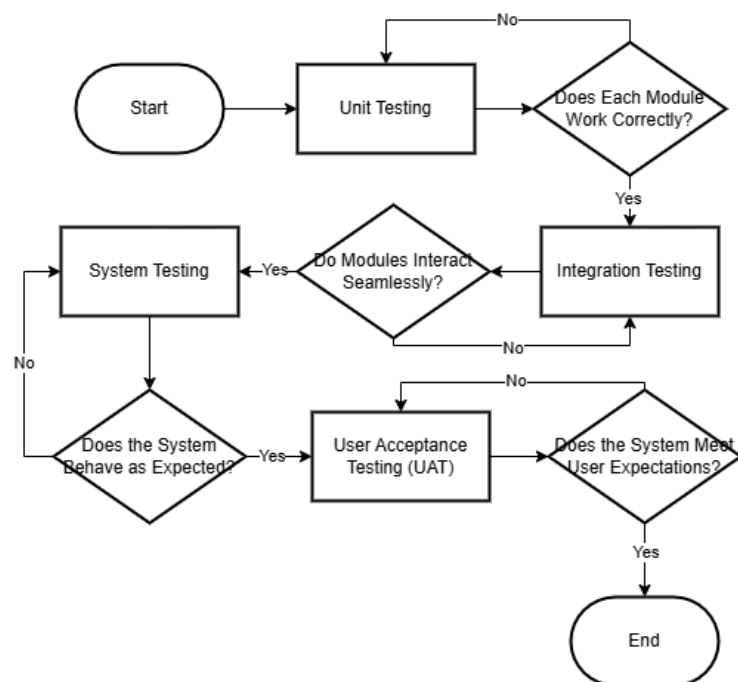


Figure 3. Detailed Flowchart of the Testing Processes. *Source:* Developed by this study.

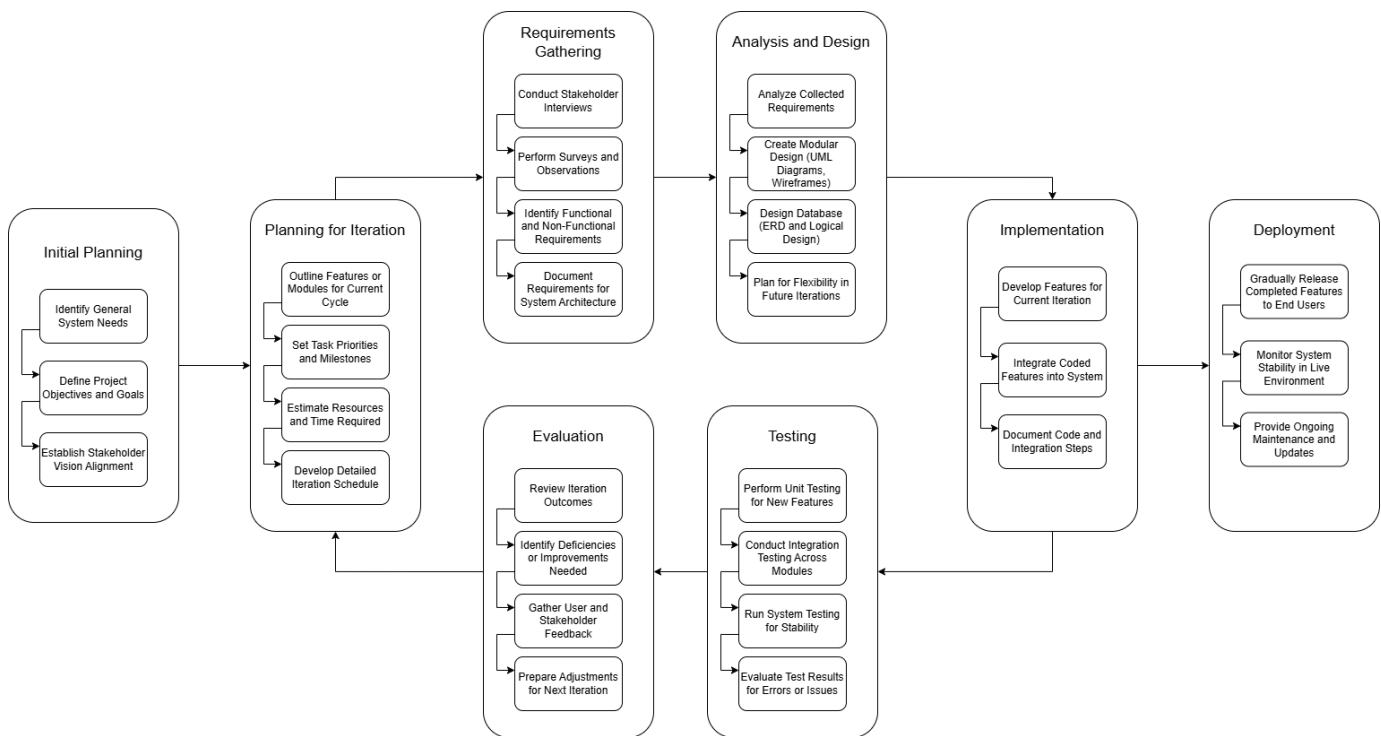


Figure 4. Block Diagram of the SDLC Iterative Process. *Source:* Developed by this study.

4. Result and Discussion

This research resulted in a web-based application developed using the iterative SDLC methodology. Through repeated cycles of planning, design, implementation, testing, and evaluation, each iteration contributed to refining and enhancing the application to meet the specific requirements of the MBKM industrial internship program at the Faculty of Information Technology, Universitas Tarumanagara. This approach allowed for continuous adjustments and improvements, ensuring that the final application effectively addressed user needs and supported efficient data management throughout the internship process. The following sections provide a detailed explanation of the research results, organized according to each stage that was undertaken.

4.1 Initial Planning

In the initial planning stage, a general identification of the objectives and scope for the application to be developed is conducted. For the MBKM data management application tailored to support industrial internships at the Faculty of Information Technology, Universitas Tarumanagara, the author carried out observations on the current internship system to gain a clear understanding of the fundamental needs for the application design. This observational approach provided insight into existing processes and highlighted essential features that the new application would need to address, establishing a baseline for development.

4.2 Planning

In the subsequent planning phase, the author focused on planning for resources, timelines, and potential risks associated with the development of each module. This stage involved careful consideration of the tools and technologies required, estimated durations for each development task, and an assessment of challenges that could arise throughout the iterative cycles. By outlining these factors, a structured roadmap was created to ensure efficient resource allocation and minimize disruptions, guiding the project's progression through each phase of development.

4.3 Requirements

In the requirements gathering stage, the author collected and identified detailed requirements specific to the current iteration. This process involved conducting interviews with a few lecturers at the Faculty of Information Technology, Universitas Tarumanagara, which provided valuable insights into essential features needed for the application. Key requirements identified included functionalities for managing data related to students, faculty advisors, and company mentors, each critical for effectively supporting the MBKM industrial internship program. These interviews helped clarify the core elements of the system to meet user expectations and operational needs. Any additional requirements or changes that arise will be addressed in subsequent iterations, aligning with the application's evolving development and continuous refinement based on user feedback.

4.4 Analysis and Design

In the analysis and design stage, a comprehensive and structured design process was executed to ensure the system's alignment with both functional and non-functional requirements. Unified Modelling Language (UML) diagrams played a pivotal role in representing the structure, behavior, and interactions within the system. UML provides a standardized approach for modeling complex software systems, enabling clear documentation of processes and interactions. Through the use of object-oriented models, UML facilitates enhanced understanding and communication among stakeholders, making it an essential tool for both developers and project managers [22].

Among the UML diagrams developed, use case diagrams were extensively used to capture and visualize the system's functional requirements from the perspectives of different user roles. These diagrams delineate the interactions between actors—such as students, faculty advisors, mentors, and administrators—and the system's functionalities. Figures 6, 7, and 8 illustrate the use case diagrams for students, faculty advisors company mentors, and administrators, respectively, showcasing the diverse features tailored to each role. Students can manage their internship data, logbooks, mentoring attendance, and reports, while also accessing vacancies and scores. Faculty advisors and mentors focus on evaluating student performance, managing scores, and monitoring mentoring sessions. Meanwhile, administrators oversee all aspects of the system, including user accounts, vacancies, deadlines, and reporting. Together, these diagrams provide a comprehensive representation of the system's scope, ensuring that all user requirements are thoroughly addressed and streamlined.

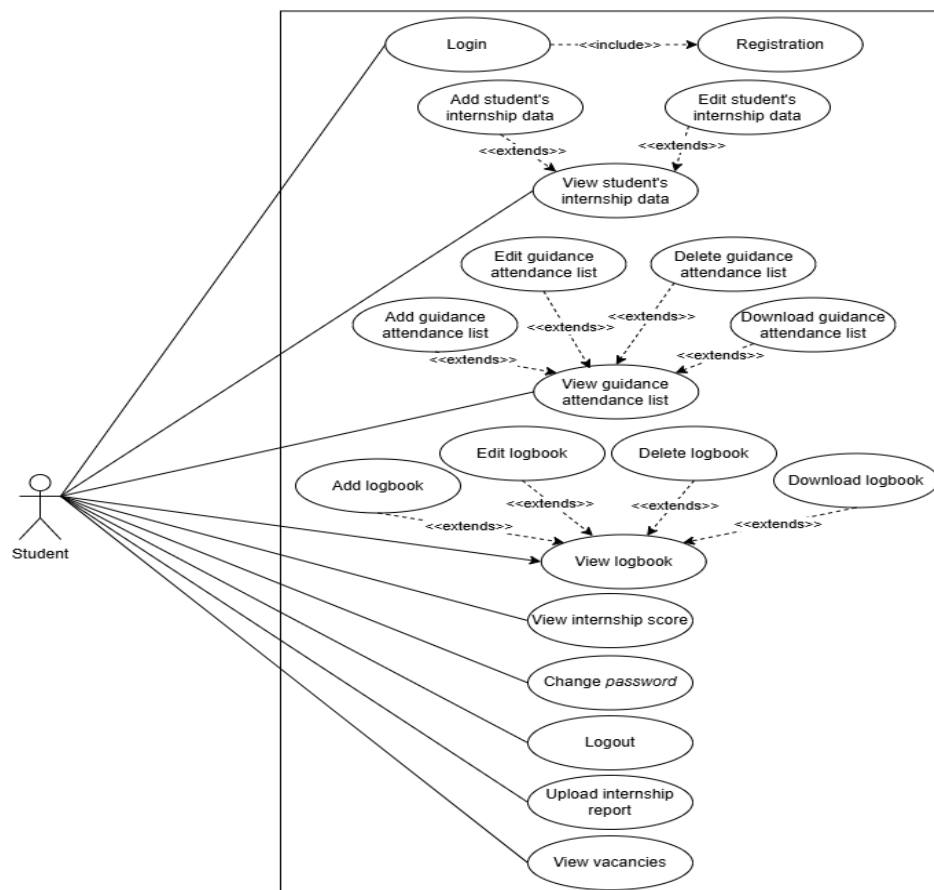


Figure 5. Use Case Diagram for Student. *Source:* Developed by this study.

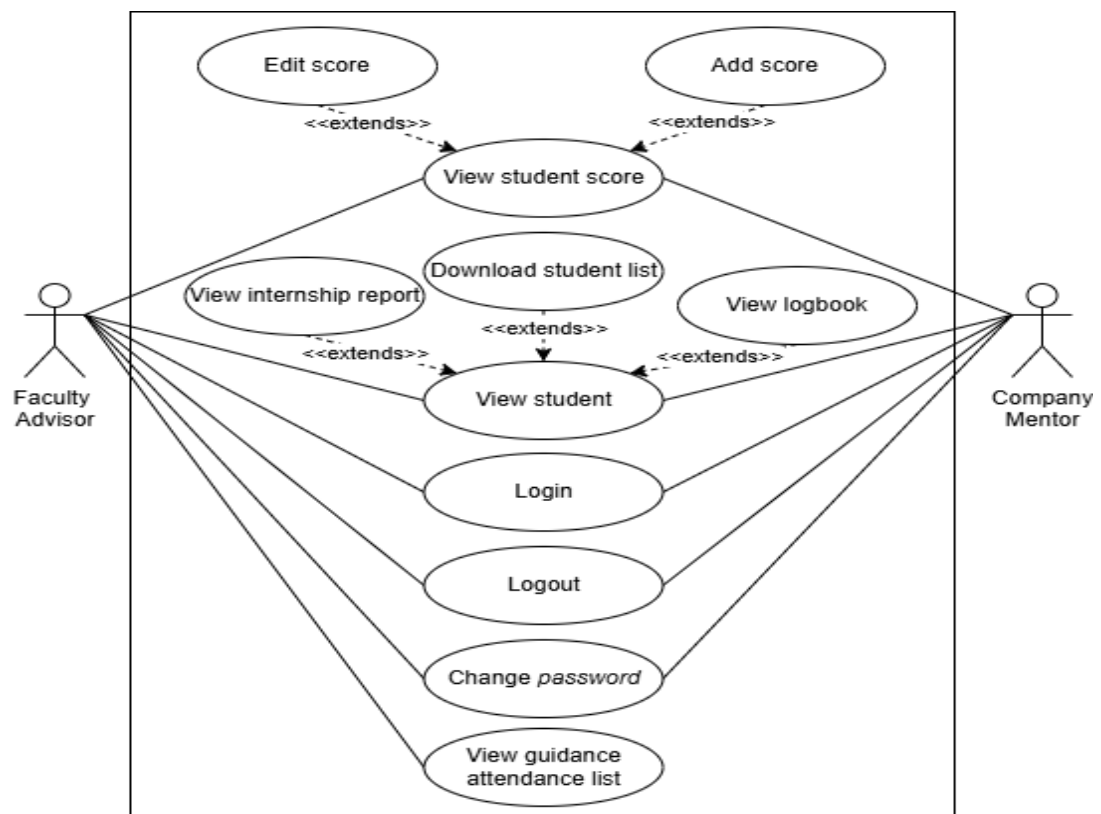


Figure 6. Use Case Diagram for Faculty Advisor and Company Mentor. *Source:* Developed by this study.

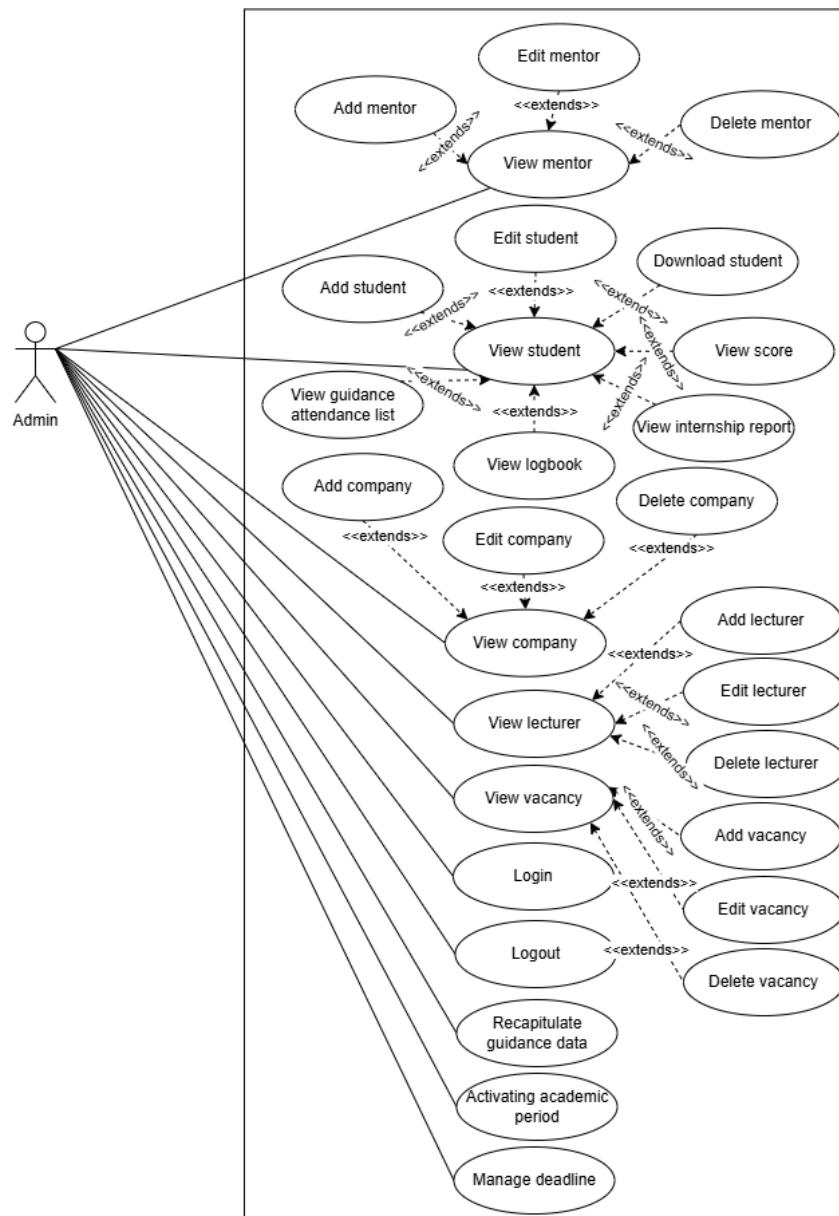


Figure 7. Use Case Diagram for Administrator. *Source:* Developed by this study.

In addition to UML diagrams, the database design process involved the creation of a detailed Entity-Relationship Diagram (ERD) at the logical level. The ERD was essential for mapping out the relationships between different data entities, ensuring efficient data organization, and supporting seamless access and management. Figure 9 presents the ERD for the industrial internship management system at the Faculty of Information Technology, Universitas Tarumanagara. This ERD reflects the integration of key system components, such as internship details, logbooks, mentoring attendance, and evaluation records, into a cohesive database structure.

The ERD demonstrates how different entities—such as Students, Lecturers, Mentors, Companies, and Vacancies—are interconnected to support core functionalities. For instance, the *Student_Internship_Log* entity serves as a central hub for tracking internship progress, linking students to their assigned mentors, vacancies, and academic periods. Similarly, the *Logbook*, *Guidance_Attendance*, and *Internship_Report* entities ensure that students' daily activities, mentoring sessions, and progress reports are properly recorded and easily accessible. Additionally, the *Score_Detail* and *Score_Category* entities enable detailed evaluation of students'

performance by mentors and faculty advisors. The integration of the Deadline entity ensures that critical dates, such as evaluation deadlines and academic period changes, are systematically managed. The inclusion of status flags, such as *is_active* in entities like Company and Lecturer, allows the system to maintain up-to-date records without permanently removing data, preserving historical information for reference.

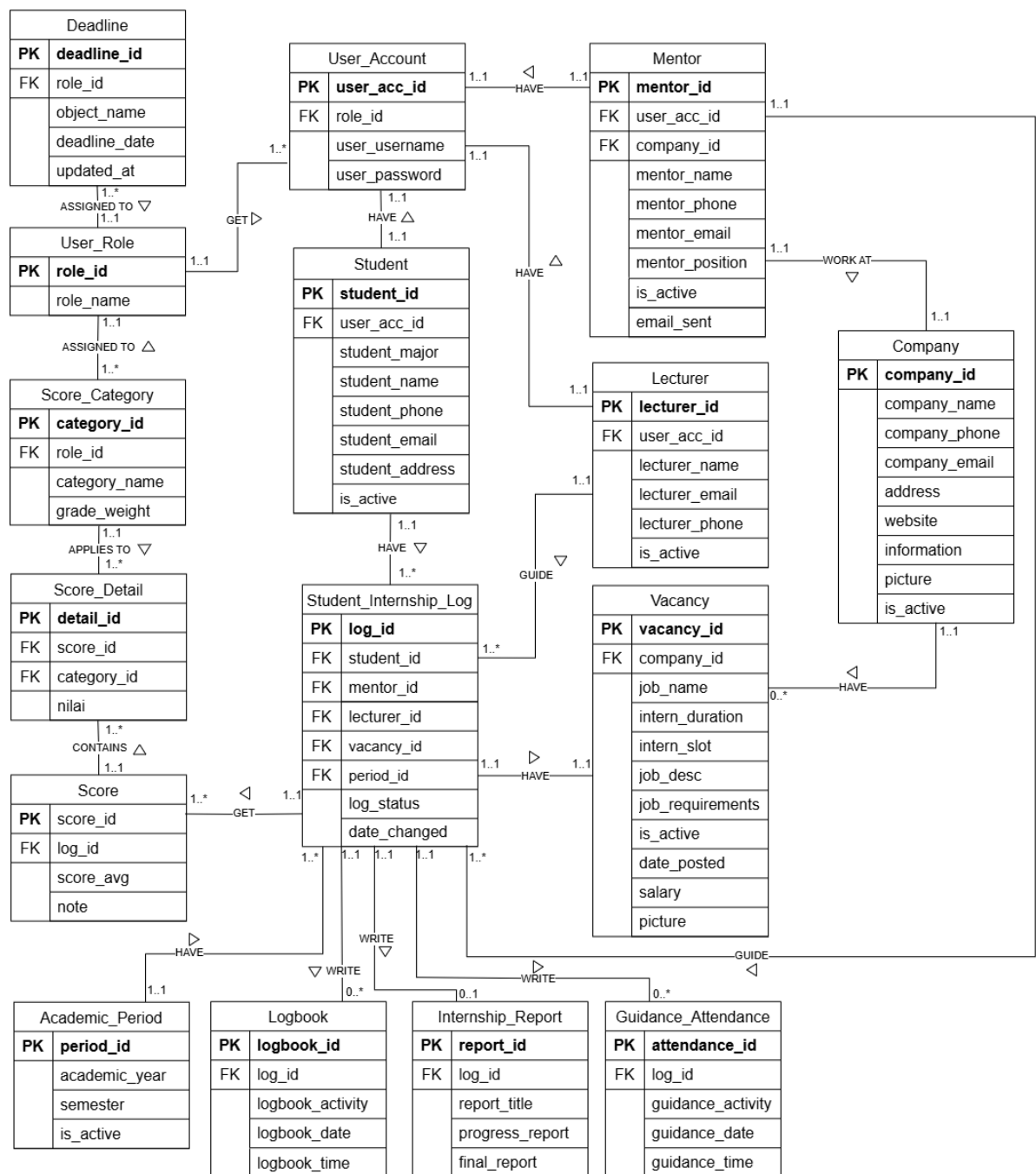


Figure 8. Logical Database Design. Source: Developed by this study.

4.5 Implementation

To develop the MBKM data management web application, a diverse set of tools and technologies was employed to support both backend functionality and an intuitive user interface. C# served as the primary programming language, integrated with ASP.NET to build a dynamic, object-oriented application framework. Visual Studio was selected as the main Integrated Development Environment (IDE) due to its seamless integration with ASP.NET and advanced features that streamline code development and debugging. For database management, SQL Server Management Studio (SSMS) was utilized to ensure efficient data handling and support for the complex relationships among entities as defined in the ERD. On the front end, HTML and CSS established the structure and styling of the web pages, while JavaScript was incorporated to add interactivity and responsiveness. To further enhance the design, Bootstrap was used as a frontend framework, enabling the rapid development of a responsive and user-friendly interface across various devices.

The following sections provide a detailed breakdown of the features developed to expand and automate key components of the existing system, including the scope of the internship program, distribution of internship information, logbook creation, mentoring attendance tracking, mentor evaluation forms, and automated reminder notifications.

4.5.1 Scope of Internship Program Duration

In the existing internship management system, data tracking is limited to students who undertake a 6-month internship, restricting its flexibility and usability for various internship scenarios. However, the internship program at the Faculty of Information Technology, Universitas Tarumanagara, requires a more adaptable approach, as students may also engage in 12-month internships or, in special cases, may need to repeat their internships if they do not meet passing criteria. The newly developed application addresses these limitations by allowing flexible configuration of internship durations. Administrators can now set and track internships for 6 months, or 12 months, or even handle cases where students need to retake their internships.

This enhanced functionality ensures that the application can comprehensively support and manage all operational aspects of the internship program, meeting the diverse needs of students and faculty. For example, in Figure 10, the "Student Detail" page is shown, which can be accessed by the administrator for students undertaking a 6-month internship. This page includes a "Student Internship History" table that logs the student's internship activities. In the table, it can be seen that the student was initially registered in the system, then accepted for the internship and assigned a faculty advisor. Finally, when the academic period is about to change, the system automatically sets the student's status to "Exit," indicating the internship has concluded.

For students enrolled in a 12-month internship, as illustrated in Figure 11 the workflow follows a similar process to the 6-month internship. However, a key difference appears before the academic period changes. At this stage, the student's status is marked as "Extended," signifying that the first 6-month phase has been completed and the internship will continue into the second 6-month period. Before the next academic period change occurs, the system automatically updates the student's status to "Exit," indicating the conclusion of the 12-month internship. This automation provides administrators with a clear and efficient process to manage internships of varying durations, accommodating unique scenarios as needed. Below is the pseudocode that outlines the logic implemented in the system to handle this workflow efficiently.

```
START
FUNCTION UpdateInternshipStatus(studentId, academicPeriod)
    student = GetStudentDetails(studentId)
    IF academicPeriod.isEnding() THEN
        SWITCH student.duration
            CASE 6:
                UpdateStatus(studentId, "Exit")
            CASE 12:
                IF student.currentPhase == 1 THEN
                    UpdateStatus(studentId, "Extended")
                    IncrementPhase(studentId)
                ELSE
                    UpdateStatus(studentId, "Exit")
                END IF
            CASE "Repeat":
                UpdateStatus(studentId, "Exit")
        END SWITCH
    END IF
END FUNCTION
FOR EACH student IN GetAllActiveInternships()
    UpdateInternshipStatus(student.id, GetCurrentAcademicPeriod())
END FOR
END
```

----- Pseudocode 1 -----

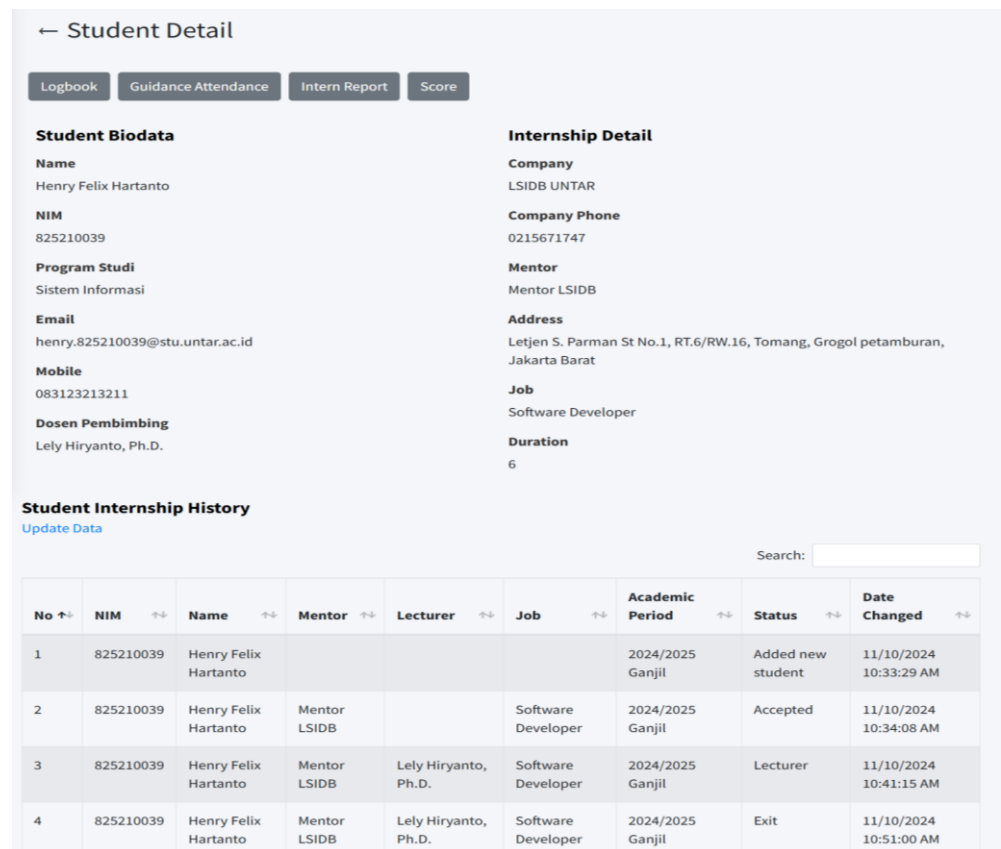


Figure 9. Student Detail Page for a 6-month Internship. Source: Screenshot from the developed application.

← Student Detail

Logbook
Guidance Attendance
Intern Report
Score

Student Biodata

Name
Maria Rosa Gosal hehe hehe

NIM
825210014

Program Studi
Sistem Informasi

Email
maria.825210014@stu.untar.ac.id

Mobile
089696895332

Dosen Pembimbing
Prof. Lina, Ph.D.

Internship Detail

Company
Kawan Lama Solution

Company Phone
081510009997

Mentor
Mentor Kawan Lama

Address
Jl. Puri Kencana No.1, Kembangan Sel., Kec. Kembangan, Jakarta Barat

Job
Data Analyst

Duration
12

Student Internship History
[Update Data](#)

Search:

No	NIM	Name	Mentor	Lecturer	Job	Academic Period	Status	Date Changed
1	825210014	Maria Rosa Gosal hehe hehe				2023/2024 Genap	Added new student	11/3/2024 11:20:36 AM
2	825210014	Maria Rosa Gosal hehe hehe	Mentor Kawan Lama	Prof. Lina, Ph.D.	Data Analyst	2023/2024 Genap	Lecturer	11/3/2024 11:37:37 AM
3	825210014	Maria Rosa Gosal hehe hehe	Mentor Kawan Lama	Prof. Lina, Ph.D.	Data Analyst	2023/2024 Ganjil	Extended	11/3/2024 11:53:48 AM
4	825210014	Maria Rosa Gosal hehe hehe	Mentor Kawan Lama	Prof. Lina, Ph.D.	Data Analyst	2024/2025 Genap	Exit	11/3/2024 12:02:20 PM

Figure 10. Student Detail Page for a 12-month Internship. *Source:* Screenshot from the developed application.

4.5.2 Distribution of Internship Information

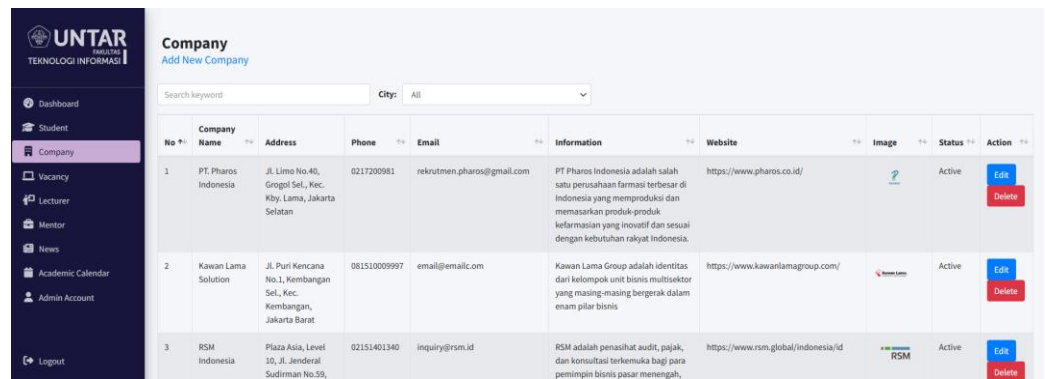
In the current system, internship information distribution remains uncomputerized, with opportunities being shared manually via Microsoft Teams or Instagram posts. In the newly developed application, this process is centralized into a single platform, allowing for streamlined management and access. Administrators can easily add company information along with available internship positions, as shown in Figure 12, which displays the "Company" page, and Figure 13, which shows the "Vacancy" page. In both sections, administrators can perform Create, Read, Update, and Delete (CRUD) operations. These internship listings are then displayed on the student page (see Figure 14), where students can filter internship opportunities according to their needs, a feature that is unavailable in the current system. This centralized and searchable platform simplifies operations and significantly enhances efficiency in managing and accessing internship information. Below is the pseudocode that outlines the workflow for managing and distributing internship information in the newly developed application.


```

START
FUNCTION ManageInternship(action, data)
  SWITCH action
    CASE "Create":
      Add("Companies", data.companyDetails)
      Add("Vacancies", data.vacancyDetails, data.companyId)
    CASE "Read":
      RETURN GetAll("Companies"), GetAll("Vacancies")
    CASE "Update":
      Update("Companies", data.companyId, data.updatedDetails)
      Update("Vacancies", data.vacancyId, data.updatedDetails)
    CASE "Delete":
      Delete("Vacancies", data.vacancyId)
      IF CheckNoVacancies(data.companyId) THEN
        Delete("Companies", data.companyId)
      END IF
  END SWITCH
END FUNCTION
FUNCTION FilterVacancies(criteria)
  query = BuildQuery(criteria)
  RETURN ExecuteQuery(query)
END FUNCTION
FUNCTION Add(table, details, optionalId = NULL)
  ExecuteQuery("INSERT INTO " + table + " VALUES (?, ?)", details, optionalId)
END FUNCTION
FUNCTION Update(table, id, details)
  ExecuteQuery("UPDATE " + table + " SET details = ? WHERE id = ?", details, id)
END FUNCTION
FUNCTION Delete(table, id)
  ExecuteQuery("DELETE FROM " + table + " WHERE id = ?", id)
END FUNCTION
FUNCTION GetAll(table)
  RETURN ExecuteQuery("SELECT * FROM " + table)
END FUNCTION
FUNCTION CheckNoVacancies(companyId)
  RETURN ExecuteQuery("SELECT COUNT(*) FROM Vacancies WHERE companyId = ?", companyId) == 0
END FUNCTION
ADMIN_ACTION(action, data)
  CALL ManageInternship(action, data)
END ADMIN_ACTION
STUDENT_ACTION(criteria)
  DISPLAY FilterVacancies(criteria)
END STUDENT_ACTION
END

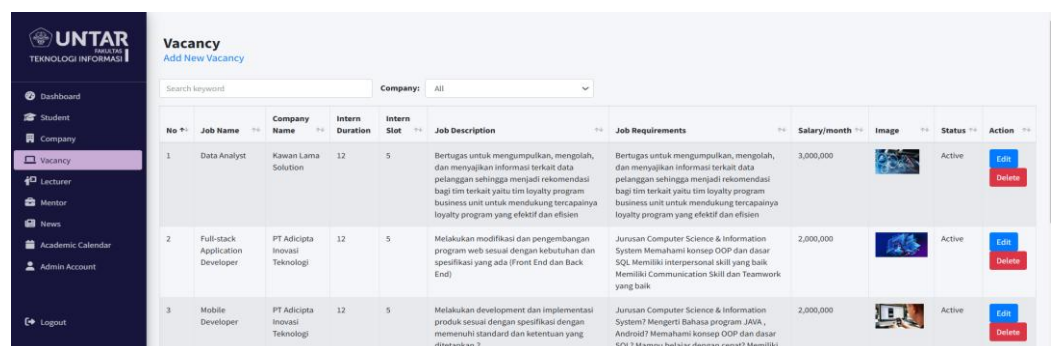
```

----- Pseudocode 2 -----



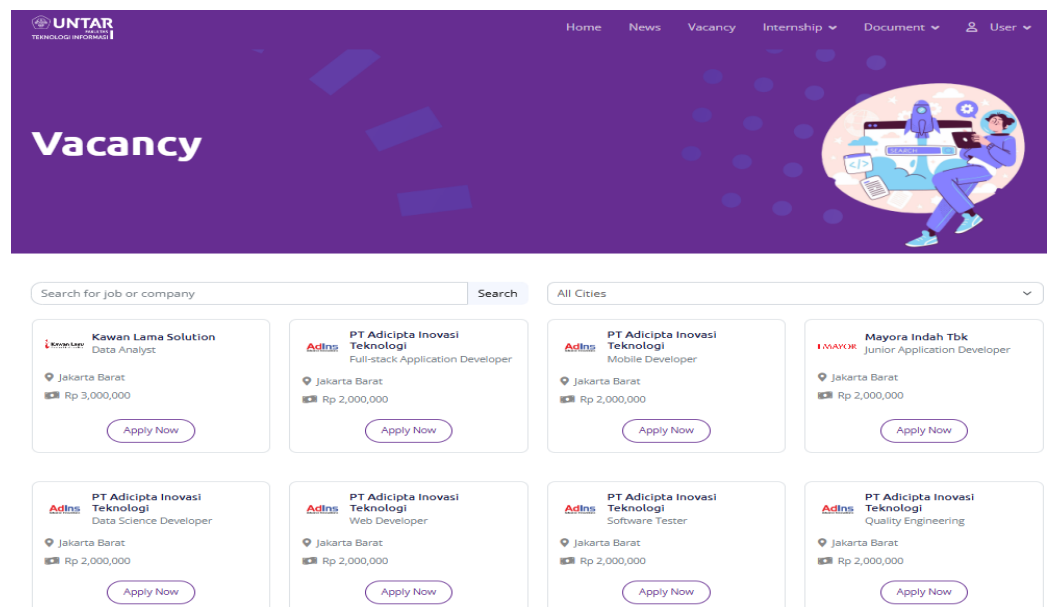
No	Company Name	Address	Phone	Email	Information	Website	Image	Status	Action
1	PT. Pharos Indonesia	Jl. Limo No.40, Grogol Sel., Kec. Kby. Lama, Jakarta Selatan	0217200981	rekrutmen.pharos@gmail.com	PT Pharos Indonesia adalah salah satu perusahaan farmasi terbesar di Indonesia yang memproduksi dan memasarkan produk-produk kefarmasian yang inovatif dan sesuai dengan kebutuhan rakyat Indonesia.	https://www.pharos.co.id/		Active	Edit Delete
2	Kawan Lama Solution	Jl. Puri Kencana No.1, Kembangan Sel., Kec. Kembangan, Jakarta Barat	06151009997	email@emalic.com	Kawan Lama Group adalah identitas dari kelompok unit bisnis multisektor yang masing-masing bergerak dalam enam pilar bisnis	https://www.kawanlamagroup.com/		Active	Edit Delete
3	RSM Indonesia	Plaza Asia, Level 10, Jl. Jenderal Sudirman No.59,	02151401340	inquiry@rsm.id	RSM adalah penasihat audit, pajak, dan konsultasi terkemuka bagi para pemimpin bisnis pasar menengah,	https://www.rsm.global/indonesia/id		Active	Edit Delete

Figure 11. Company Management Page for Administrators. Source: Screenshot from the developed application.



No	Job Name	Company Name	Intern Duration	Intern Slot	Job Description	Job Requirements	Salary/month	Image	Status	Action
1	Data Analyst	Kawan Lama Solution	12	5	Bertugas untuk mengumpulkan, mengolah, dan menyajikan informasi terkait data pelanggan sehingga menjadi rekomendasi bagi tim terkait yaitu tim loyalty program business unit untuk mendukung tercapainya loyalty program yang efektif dan efisien	Bertugas untuk mengumpulkan, mengolah, dan menyajikan informasi terkait data pelanggan sehingga menjadi rekomendasi bagi tim terkait yaitu tim loyalty program business unit untuk mendukung tercapainya loyalty program yang efektif dan efisien	3,000,000		Active	Edit Delete
2	Full-stack Application Developer	PT Adicpta Inovasi Teknologi	12	5	Melakukan modifikasi dan pengembangan program web sesuai dengan kebutuhan dan spesifikasi yang ada (Front End dan Back End)	Jurusan Computer Science & Information System Memahami konsep OOP dan dasar SQL. Memiliki interpersonal skill yang baik. Memiliki Communication Skill dan Teamwork yang baik	2,000,000		Active	Edit Delete
3	Mobile Developer	PT Adicpta Inovasi Teknologi	12	5	Melakukan development dan implementasi produk sesuai dengan spesifikasi dengan memenuhi standard dan ketentuan yang ditetapkan.	Jurusan Computer Science & Information System? Mengerti Bahasa program JAVA, Android? Memahami konsep OOP dan dasar SQL? Mampu belajar dengan cepat? Memiliki	2,000,000		Active	Edit Delete

Figure 12. Vacancy Management Page for Administrators. Source: Screenshot from the developed application.



Search for job or company All Cities

Kawan Lama Solution
Data Analyst

Jakarta Barat
Rp 3,000,000

[Apply Now](#)

PT Adicpta Inovasi Teknologi
Full-stack Application Developer

Jakarta Barat
Rp 2,000,000

[Apply Now](#)

PT Adicpta Inovasi Teknologi
Mobile Developer

Jakarta Barat
Rp 2,000,000

[Apply Now](#)

Mayora Indah Tbk
Junior Application Developer

Jakarta Barat
Rp 2,000,000

[Apply Now](#)

PT Adicpta Inovasi Teknologi
Data Science Developer

Jakarta Barat
Rp 2,000,000

[Apply Now](#)

PT Adicpta Inovasi Teknologi
Web Developer

Jakarta Barat
Rp 2,000,000

[Apply Now](#)

PT Adicpta Inovasi Teknologi
Software Tester

Jakarta Barat
Rp 2,000,000

[Apply Now](#)

PT Adicpta Inovasi Teknologi
Quality Engineering

Jakarta Barat
Rp 2,000,000

[Apply Now](#)

Figure 13. Internship Listings Page for Students. Source: Screenshot from the developed application.

4.6 Creating Logbooks

At the Faculty of Information Technology, Universitas Tarumanagara, students participating in internships are required to create a logbook, which records their daily activities from the start to the end of the internship. In the current system, students must create this logbook entirely on their own, including setting up the template, creating the table, and manually entering their data. In the newly developed application, however, this process is fully computerized. As shown in Figure 15, students simply click the "Add New Log" button, enter their activity details, and submit the entry. Additionally, the logbook table can be downloaded in either Excel or PDF format, offering flexibility based on the student's needs. Students can also edit entries if they make a mistake or delete them if necessary, streamlining the logbook creation process and ensuring ease of use. Below is the pseudocode that represents the functionality for creating, editing, deleting, and downloading logbook entries within the developed application.

```

START
FUNCTION AddLogEntry(studentId, date, activityDetails)
    INSERT INTO Logbook (studentId, date, activityDetails)
    VALUES (studentId, date, activityDetails)
END FUNCTION
FUNCTION EditLogEntry(logId, updatedDate, updatedActivityDetails)
    UPDATE Logbook
    SET date = updatedDate, activityDetails = updatedActivityDetails
    WHERE logId = logId
END FUNCTION
FUNCTION DeleteLogEntry(logId)
    DELETE FROM Logbook WHERE logId = logId
END FUNCTION
FUNCTION GetStudentLogEntries(studentId)
    RETURN ExecuteQuery("SELECT * FROM Logbook WHERE studentId = ?",
studentId)
END FUNCTION
FUNCTION DownloadLogbook(studentId, format)
    logEntries = GetStudentLogEntries(studentId)
    IF format == "Excel" THEN
        GenerateExcel(logEntries)
    ELSE IF format == "PDF" THEN
        GeneratePDF(logEntries)
    END IF
    RETURN DownloadFile()
END FUNCTION
STUDENT_ACTION(action, data)
    SWITCH action
        CASE "Add":
            CALL AddLogEntry(data.studentId, data.date, data.activityDetails)
        CASE "Edit":
            CALL EditLogEntry(data.logId, data.updatedDate,
data.updatedActivityDetails)
        CASE "Delete":
            CALL DeleteLogEntry(data.logId)
        CASE "Download":
            CALL DownloadLogbook(data.studentId, data.format)
    END SWITCH
END STUDENT_ACTION
END

```

----- Pseudocode 3 -----

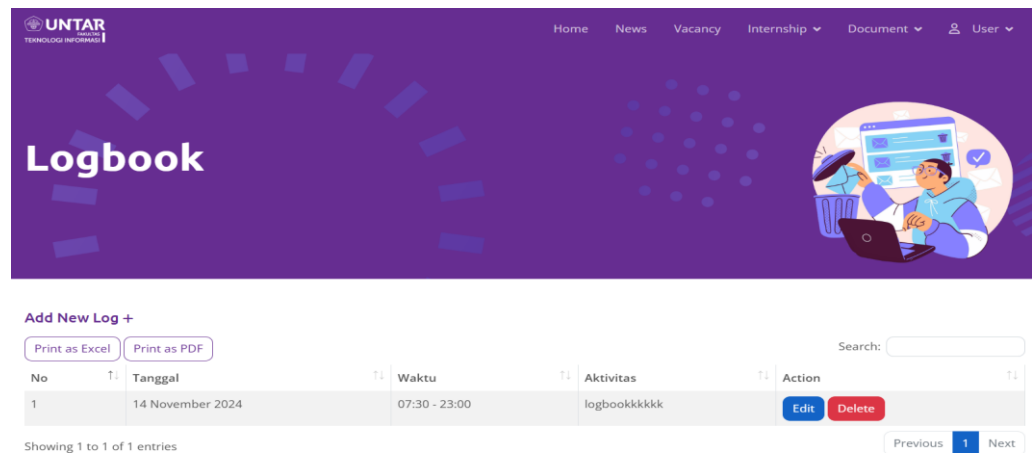


Figure 14. Student Logbook Page. *Source:* Screenshot from the developed application.

4.7 Tracking Mentoring Attendances with Faculty Advisors

Similar to the logbook feature, this component records the mentoring sessions conducted with faculty advisors throughout the internship. As shown in Figure 16, students can log each mentoring session, creating a record of their interactions and progress. These entries can also be downloaded in Excel or PDF format, providing flexibility for reporting purposes. Additionally, faculty advisors have access to view their students' mentoring attendance records, as seen in Figure 17, allowing them to monitor the frequency and consistency of mentoring sessions, ensuring effective guidance and support during the internship. Below is the pseudocode that represents the functionality implemented for tracking mentoring attendances.

```

START
FUNCTION LogMentoringSession(studentId, advisorId, sessionDetails)
    INSERT INTO MentoringSessions (studentId, advisorId, sessionDetails) VALUES
    (studentId, advisorId, sessionDetails)
END FUNCTION
FUNCTION DownloadMentoringSessions(studentId, format)
    records = SELECT * FROM MentoringSessions WHERE studentId = studentId
    IF format == "Excel" THEN
        GenerateExcel(records)
    ELSE IF format == "PDF" THEN
        GeneratePDF(records)
    END IF
END FUNCTION
FUNCTION ViewMentoringRecords(advisorId)
    RETURN SELECT * FROM MentoringSessions WHERE advisorId = advisorId
END FUNCTION
STUDENT_ACTION(studentId, sessionDetails, format)
    LogMentoringSession(studentId, advisorId, sessionDetails)
    DownloadMentoringSessions(studentId, format)
END STUDENT_ACTION
ADVISOR_ACTION(advisorId)
    mentoringRecords = ViewMentoringRecords(advisorId)
    DISPLAY mentoringRecords
END ADVISOR_ACTION
END

```

----- Pseudocode 4 -----

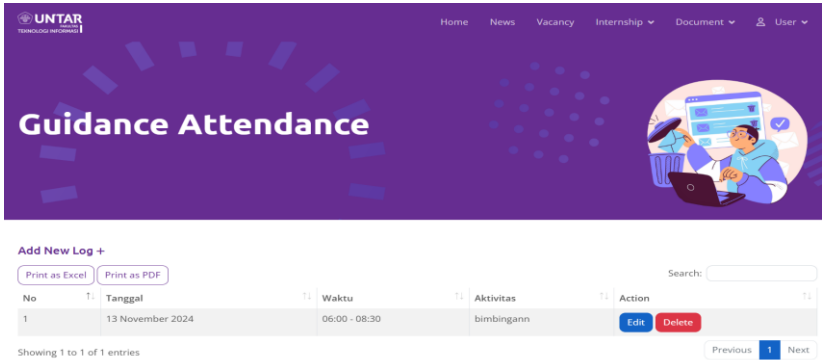


Figure 15. Student Mentoring Attendance Page. Source: Screenshot from the developed application.

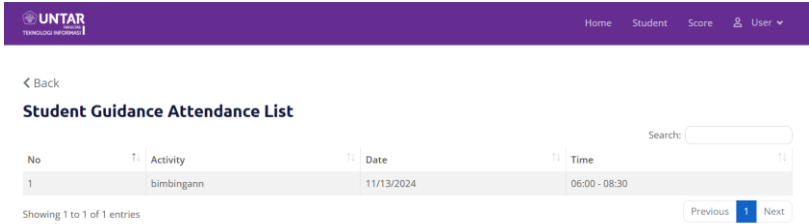


Figure 16. Faculty Advisor Mentoring Attendance Page. Source: Screenshot from the developed application.

4.8 Administering Evaluation Forms to Company Mentors

In the current system, administrators manually send student evaluation forms to company mentors, which involves a complex and time-consuming process. First, administrators create a spreadsheet where students fill in their mentor details, such as name, phone number, and email. The administrator then emails the evaluation form individually to each mentor, a process prone to inefficiencies and errors. To streamline this, the new application allows mentors to directly access and submit evaluations for their interns. The workflow begins with students entering their internship details, as shown in Figure 18. Once submitted, the mentor information provided by students is automatically displayed on the administrator’s page, as illustrated in Figure 19. Here, administrators can select the mentors to whom they wish to send login credentials by checking the appropriate boxes and clicking the "Send User Credential to Email" button. Mentors then receive an email with their account credentials, as shown in Figure 20. Once mentors receive their login credentials, they can access their dedicated portal to submit evaluations for the students they supervise. Figure 21 shows the "Company Mentor Evaluation Page," where mentors can directly enter performance scores and feedback for each student. This streamlined process not only saves time for administrators but also ensures that evaluations are systematically recorded in one centralized platform, reducing the risk of errors and providing easy access for administrators and faculty to review mentor feedback. Below is the pseudocode for this feature.

UNTAR
TEKNOLOGI INFORMASI

Home News Vacancy Internship Document User

< Back

Change Internship Detail

Company
Company not listed in the dropdown list below? [Add new company](#)

Company
PT Adicpta Inovasi Teknologi

Job
Full-stack Application Developer

Mentor
Mentor not listed in table below? [Add new mentor](#)

Search:

No	Name	Email	Phone	Position	Action
1	Mentor Adins	mentor@adins.com	08979979781	Developer	Selected

Showing 1 to 1 of 1 entries

Previous 1 Next

Submit Cancel

Figure 17. Internship Detail Entry Page. *Source:* Screenshot from the developed application.

```

START
FUNCTION SubmitInternshipDetails(studentId, mentorName, mentorEmail,
internshipDetails)
    INSERT INTO InternshipDetails (studentId, mentorName, mentorEmail, details)
    VALUES (studentId, mentorName, mentorEmail, internshipDetails)
END FUNCTION
FUNCTION SendCredentialsToMentors(mentorIds)
    FOR EACH mentorId IN mentorIds
        mentorDetails = SELECT * FROM Mentors WHERE mentorId = mentorId
        credentials = GenerateCredentials(mentorId)
        emailContent = CreateCredentialEmail(mentorDetails.email, credentials)
        SendEmail(emailContent)
        UpdateMentorStatus(mentorId, "Credentials Sent")
    END FOR
END FUNCTION
FUNCTION SubmitMentorEvaluation(mentorId, studentId, performanceScore, feedback)
    INSERT INTO Evaluations (mentorId, studentId, performanceScore, feedback)
    VALUES (mentorId, studentId, performanceScore, feedback)
END FUNCTION
FUNCTION AdminAction(action, data)
    SWITCH action
        CASE "ViewMentorDetails":
            RETURN GetAllMentorDetails()
        CASE "SendCredentials":
            CALL SendCredentialsToMentors(data.mentorIds)
    END SWITCH
END FUNCTION
FUNCTION MentorAction(action, data)
    SWITCH action
        CASE "SubmitEvaluation":
            CALL SubmitMentorEvaluation(data.mentorId, data.studentId,
data.performanceScore, data.feedback)
    END SWITCH
END FUNCTION
FUNCTION ViewMentorEvaluations(adminId)
    evaluations = SELECT * FROM Evaluations WHERE adminId = adminId ORDER BY studentId
    RETURN evaluations
END FUNCTION
END

```

----- Pseudocode 5 -----

No	Checklist	Name	Company	Email	Phone	Position	Status	Email Sent
1	<input type="checkbox"/>	mentor 6	company 6	lilacdlwlrma@gmail.com	123	tes	Inactive	False
2	<input type="checkbox"/>	Mentor Kawan Lama	Kawan Lama Solution	lilacdlwlrma@gmail.com	123	Senior Developer	Active	True
3	<input type="checkbox"/>	Mentor Adins	PT Adicpta Inovasi Teknologi	mentor@adins.com	08979979781	Developer	Active	False
4	<input type="checkbox"/>	Mentor LSIDB	LSIDB UNTAR	lsidb@mentor.com	08979979781	Mentor	Inactive	False

Figure 18. Administrator Mentor Management Page. *Source:* Screenshot from the developed application.

Akun Mentor - Sistem Magang

mariarosagosalswork@gmail.com

to me

[Translate to English](#)

Kepada Yth. Bapak/Ibu Mentor Kawan Lama,

Berikut adalah informasi akun Anda untuk login ke website Magang FTI UNTAR:

Username: He90J1uz

Password: He90J1uzn1

Silakan login ke sistem dengan menggunakan kredensial ini. Terima kasih.

Regards,

Admin FTI UNTAR

Figure 19. Mentor Credential Email. *Source:* Developed by this study.

Kriteria	Score
Kedisiplinan kehadiran kerja	80
Etika, Komunikasi dan Kerjasama tim	81
Dedikasi dan tanggung jawab	99
Inisiatif dan respon terhadap instruksi kerja	90
Kemampuan dalam memanfaatkan berbagai Teknologi informasi	100
Kompleksitas pekerjaan	98
Kemampuan menyelesaikan masalah / pekerjaan	88
Capaian hasil kerja	89
Average Score	90

Notes: ww

[Submit](#)

Figure 20. Company Mentor Evaluation Page. *Source:* Screenshot from the developed application.

4.9 Automated Reminder Notifications to Company Mentors

In the current system, administrators manually remind company mentors to complete student evaluations, which requires multiple steps and communication through Microsoft Teams. Administrators typically post announcements asking students who have not yet received evaluations to remind their mentors as the deadline approaches. This process is time-consuming and inefficient, as it relies on three parties—administrators reminding students, students reminding mentors, and mentors finally completing the evaluations.

In the new application, automated reminders simplify this workflow significantly. On the mentor dashboard, as shown in Figure 22, important dates—including the evaluation submission deadline—are displayed to ensure mentors are aware of upcoming deadlines. If mentors are not logged into the system, students are notified on their "Intern Score" page, as seen in Figure 23, with an alert indicating that their mentor has yet to submit their evaluation. This alert includes an option to send a reminder directly to the mentor's email. Once clicked, the system automatically sends a notification to the mentor, prompting them to complete the evaluation, as illustrated in Figure 24. This feature reduces the process from three parties to just two, allowing students to directly prompt their mentors, saving time and improving efficiency in managing evaluation deadlines. The following section presents the pseudocode that outlines the implementation of this feature.

```

START
FUNCTION DisplayMentorDashboard(mentorId)
    importantDates = SELECT * FROM Deadlines WHERE mentorId = mentorId
    RETURN importantDates
END FUNCTION

FUNCTION CheckEvaluationStatus(studentId)
    evaluationStatus = SELECT status FROM Evaluations WHERE studentId =
studentId
    IF evaluationStatus == "Pending" THEN
        DISPLAY Alert("Mentor has not submitted the evaluation.")
    END IF
END FUNCTION

FUNCTION SendReminderToMentor(studentId, mentorId)
    IF IsReminderAllowed(studentId, mentorId) THEN
        mentorEmail = SELECT email FROM Mentors WHERE mentorId = mentorId
        emailContent = CreateReminderEmail(mentorEmail)
        SendEmail(emailContent)
        LogReminder(studentId, mentorId)
    ELSE
        DISPLAY Error("Cannot send another reminder now.")
    END IF
END FUNCTION

FUNCTION IsReminderAllowed(studentId, mentorId)
    lastReminderDate = SELECT lastSent FROM Reminders WHERE studentId =
studentId AND mentorId = mentorId
    currentDate = GET_CURRENT_DATE()
    IF currentDate >= lastReminderDate + 2 DAYS THEN
        RETURN TRUE
    ELSE
        RETURN FALSE
    END IF
END FUNCTION

```

```

STUDENT_ACTION(studentId)
    CALL CheckEvaluationStatus(studentId)
    IF UserClicksReminder THEN
        CALL SendReminderToMentor(studentId, mentorId)
    END IF
END STUDENT_ACTION
END

```

----- Pseudocode 6 -----

Dashboard
List supervised students based on the active academic period

No	Student ID	Name	Internship Detail
1	535000000	Mahasiswa 12 bulan	Back End Developer at Kawan Lama Solution
2	535970010	Lely Hiryanto	Back End Developer at Kawan Lama Solution
3	825111111	tes	Data Analyst at Kawan Lama Solution
4	825210014	Maria Rosa Gosal hehe hehe	Data Analyst at Kawan Lama Solution
5	825210054	Clive Riady	Data Analyst at Kawan Lama Solution

15 November Deadline for students assessment grade from mentor

08 November Deadline for students second assessment grade from mentor (1 year internship)

Students who have not been assessed

tes 825111111	Give Score
Mahasiswa 12 bulan 535000000	Give Score
Maria Rosa Gosal hehe hehe 825210014	Give Score

Figure 21. Mentor Dashboard Page. *Source:* Screenshot from the developed application.

Mentor belum memberikan nilai. Deadline pemberian nilai oleh mentor adalah tanggal 15 November 2024. Ingatkan Mentor

Score from Mentor

Criteria	Score
Capaian hasil kerja	
Dedikasi dan tanggung jawab	
Etika, Komunikasi dan Kerjasama tim	
Inisiatif dan respon terhadap instruksi kerja	

Figure 22. Intern Score Page for Students. *Source:* Screenshot from the developed application.

Reminder Pemberian Nilai untuk Mahasiswa 6 bulan Inbox

mariarosagosalwork@gmail.com Sun, Nov 3, 11:06AM (12 days ago) ☆ 😊 ↶ ⋮

🗣️ Translate to English ×

Kepada Yth: Bapak/Ibu mentor 6,

Kami ingin mengingatkan kembali batas waktu pengisian penilaian magang untuk mahasiswa yang saat ini berada di bawah bimbingan Bapak/Ibu, yaitu **Mahasiswa 6 bulan** adalah tanggal **08 November 2024**. Mohon kesediaan Bapak/Ibu untuk segera memberikan penilaian sebelum batas waktu tersebut.

Terima kasih atas kerja samanya.

Regards,
Admin FTI UNTAR

Figure 23. Automated Reminder Notification to Mentor. *Source:* Developed by this study.

4.10 Testing

After completing the application design and coding, a thorough testing phase was initiated to confirm the system's quality and robustness.

4.10.1 Unit Testing

Unit testing is the initial stage in the testing process, where each unit or smallest component of an application is tested independently to ensure it functions correctly. A unit can be a function, procedure, or small module within the code. Unit testing is conducted to identify bugs or errors at the most fundamental level [17]. This testing is essential as it verifies that each part of the application works individually before being integrated with other units.

4.10.2 Integration Testing

After completing the unit testing phase, the authors proceed with integration testing. Integration testing is the stage where all units that have passed unit testing are combined and tested to ensure they function correctly when working together. This process helps identify any issues that may arise from interactions between different units, which might not have been apparent during individual unit testing. By thoroughly checking the integrated components, developers can verify that the application meets compatibility and performance standards outlined in the specifications, ensuring a smoother transition to subsequent testing phases [17].

4.10.3 System Testing

After all components are integrated, the application undergoes system testing, where it is evaluated as a complete entity. This stage ensures that the application functions according to the specified requirements and meets both functional and non-functional criteria. System testing is only performed once all application components have been successfully integrated, to validate that the entire system can fulfill the needs and expectations outlined in the specifications [17]. At this phase, comprehensive testing is conducted to observe how the application operates as a unified system rather than as separate units or modules. This involves verifying the entire application process flow, ensuring that each module interacts correctly, and confirming that the application performs reliably under various operational conditions. If the application passes system testing with satisfactory results, it is then ready for User Acceptance Testing (UAT), where end users further validate it to ensure it meets their needs.

4.10.4 User Acceptance Testing (UAT)

User Acceptance Testing (UAT) phase is the final step in the testing process before the application is officially deployed. UAT is designed to ensure that the application meets the needs and expectations of end users, focusing on its functionality as it relates to real-world usage scenarios, without regard to the internal code or technical implementation [17]. In this phase, four types of users participate according to their roles within the application: students, mentors, faculty advisors, and administrators. Each user role tests the application based on the specific functionalities required for their responsibilities, ensuring that all features work as expected and support the necessary processes for each role. Black box testing was used in this phase to assess the application solely from an end-user perspective, focusing on inputs and outputs rather than the internal workings of the code. Table 2 shows the results of black box testing, specifically examining the features developed to expand and automate key components of the existing system.

Table 1. Black Box Testing Result

Module	Test Case	Expected Result	Status
Intern Program Duration	Status will be updated to indicate that the student has completed the internship (for students with 6-month internships or in the second period of a 12-month internship)	The status of the student changes to “Exit” which indicates that the internship has been completed.	Passed
	Status will be updated to indicate that the student has extended the internship (for students in the first period of a 12-month internship)	The status of the student changes to “Extended” which indicates that the internship has been completed.	Passed
Intern Info Distribution	Admin add a new company	New company data will be saved	Passed
	Admin edit existing company	Updated company data will be saved	Passed
	Admin delete existing company	Company data will be deleted	Passed
	Admin add a new vacancy	New vacancy data will be saved	Passed
	Admin edit existing vacancy	Updated vacancy data will be saved	Passed
	Admin delete existing vacancy	Selected vacancy data will be deleted	Passed
	Open the Vacancy page as student	Display vacancies data that have been added by admin	Passed
Logbook	Add a new logbook data as student	New logbook data will be saved	Passed
	Edit existing logbook data as student	Updated logbook data will be saved	Passed
	Delete existing logbook data as student	Selected logbook data will be deleted	Passed
	Download logbook as PDF or Excel	Logbook data will be downloaded	Passed
Mentoring Attendance	Add a new attendance data as student	New attendance data will be saved	Passed
	Edit existing attendance data as student	Updated attendance data will be saved	Passed
	Delete existing attendance data as student	Selected attendance data will be deleted	Passed
	Download attendance as PDF or Excel	Attendance data will be downloaded	Passed
Students Assessment from Company Mentors	Students add their internship detail data, including mentor data	Internship data will be saved, mentor data will be created	Passed
	Admin opens Mentor Management page	Shows a list of mentors added by students	Passed
	Admin select a few mentors to send their account credential by email, then click the send email button	Emails containing the company mentors’ account credential will be sent to selected mentors	Passed
	Mentor login using the credential that was sent by email	Logins succeed	Passed
	Mentors give grades to the students their supervise	Score will be saved	Passed
	Open the intern score page for students who have not received a score 7 days before the mentor's scoring deadline.	A notification alert appears to remind the mentor.	Passed

Table 3. SUS Score Rating Scale Interpretation [26]

Letter grade	Numerical score range
A+	84.1 - 100
A	80.8 – 84.0
A-	78.9 – 80.7
B+	77.2 – 78.8
B	74.1 – 77.1
B-	72.6 – 74.0
C+	71.1 – 72.5
C	65.0 – 71.0
C-	62.7 – 64.9
D	51.7 – 62.6
F	0 – 51.6

5. Conclusion

This application has successfully computerized numerous processes that were previously conducted manually, including managing the scope of the internship program, distributing internship information, creating logbooks, tracking mentoring attendance, handling mentor evaluation forms, and automating reminder notifications. By fully digitizing these tasks, the application not only streamlines the management of student participation data for industrial internships but also reduces the risk of recording errors caused by human error. The application has been rigorously tested by four different user types through a black box testing approach, with results confirming that all testing scenarios were successfully met. Furthermore, a System Usability Scale (SUS) questionnaire assessment yielded a final score of 90.42, placing the application in the A+ grade category and reflecting an exceptionally high level of usability.

Acknowledgments: The successful completion of this project was made possible through the support and assistance of many individuals and organizations. Deep gratitude is extended to the Faculty of Information Technology, Universitas Tarumanagara, for their invaluable support throughout the development of this application. Their guidance, resources, and insights contributed significantly to the successful execution of this project. Special appreciation goes to the faculty advisors, administrators, students, and company mentors who provided their expertise, feedback, and time, ensuring that the project aligned with the needs and expectations of all users involved in the MBKM industrial internship program.

Author contributions: The authors were responsible for building Conceptualization, Methodology, analysis, investigation, data curation, writing—original draft preparation, writing—review and editing, visualization, supervision of project administration, funding acquisition, and have read and agreed to the published version of the manuscript.

Funding: The study was conducted without any financial support from external sources.

Availability of data and Materials: All data are available from the authors.

Conflicts of Interest: The authors declare no conflict of interest.

Additional Information: No Additional Information from the authors.

References

1. Kemendikbud, "Mendikbudristek: Kampus Merdeka untuk Pembelajaran yang Lebih Menyenangkan dan Relevan," Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, 6 October 2021. [Online]. Available: <https://www.kemdikbud.go.id/main/blog/2021/10/mendikbudristek-kampus-merdeka-untuk-pembelajaran-yang-lebih-menyenangkan-dan-relevan>. [Accessed 15 November 2024].
2. A. Haviz and M. I. P. Nasution, "Analisis Dampak Implementasi Sistem Informasi Manajemen Pada Efisiensi Proses Bisnis," Jurnal Ilmiah Ekonomi Dan Manajemen, 2024.
3. T. Sudargo and Tony, "Implementasi Framework Laravel Dalam Perancangan Website Manajemen Kegiatan MBKM Pada IBIKFTI," Journal of Information Technology and Computer Science (INTECOMS), 2023.
4. M. Edy, "Aplikasi Manajemen Data Anak Magang Berbasis Web pada Dinas Kebudayaan dan Pariwisata Kota Banjarmasin," Universitas Islam Kalimantan MAB, Banjarmasin, 2024.
5. P. H. Marpaung, N. Dahri dan W. Yahyan, "Sistem Informasi Pendataan Magang MBKM Berbasis Web," Jurnal Manajemen Teknologi Informatika, vol. 1, no. 2, pp. 109-116, 2023.
6. E. Turban, C. Pollard dan G. Wood, Information Technology for Management: On-Demand Strategies for Performance, Growth and Sustainability, Australia and New Zealand Edition, United Kingdom: John Wiley & Sons, Limited, 2019.
7. Elgamar, Buku Ajar Konsep Dasar Pemrograman Website dengan PHP, Ahlimedia Book, 2020.
8. Admin Universitas Tarumanagara, "8 Program MBKM yang Dapat Diketahui Mahasiswa," Universitas Tarumanagara, 5 December 2023. [Online]. Available: <https://untar.ac.id/2023/12/05/8-program-mbkm-yang-dapat-diketahui-mahasiswa/>. [Accessed 15 November 2024].
9. Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, "Apa itu Kampus Merdeka?," Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, 2022. [Online]. Available: <https://pusatinformasi.kampusmerdeka.kemdikbud.go.id/hc/en-us/articles/4417185050777-Apa-itu-Kampus-Merdeka>. [Diakses 21 Agustus 2024].
10. Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, "Apa itu Program Magang & Studi Independen Bersertifikat?," Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, 2022. [Online]. Available: <https://pusatinformasi.kampusmerdeka.kemdikbud.go.id/hc/en-us/articles/4416927940377-Apa-itu-Program-Magang-Studi-Independen-Bersertifikat>. [Accessed 15 November 2024].
11. Direktorat Jendral Pendidikan Tinggi, Riset, dan Teknologi, "Program MBKM Kemendikbudristek Membentuk Lulusan Siap Industri melalui Magang Bersertifikat," Direktorat Jendral Pendidikan Tinggi, Riset, dan Teknologi Kementrian Pendidikan, Kebudayaan, Riset, dan Teknologi, 27 November 2023. [Online]. Available: <https://dikti.kemdikbud.go.id/kabar-dikti/program-mbkm-kemendikbudristek-membentuk-lulusan-siap-industri-melalui-magang-bersertifikat/>. [Accessed 15 November 2024].
12. Y. Yudhanto dan H. A. Prasetyo, Panduan Mudah Belajar Framework Laravel, Elex Media Komputindo, 2018.
13. T. M. Conolly dan C. E. Begg, Database Systems: A Practical Approach to Design, Implementation, and Management Sixth Edition, United Kingdom: Pearson, 2015.
14. R. Fitri, Pemrograman Basis Data Menggunakan MySQL, Deepublish, 2022.
15. R. Yanto, Manajemen Basis Data Menggunakan Mysql, Yogyakarta: Deepublish, 2016.
16. A. Nordeen, Learn UML in 24 Hours, Guru99, 2020.
17. A. Nayyar, Instant Approach to Software Testing: Principles, Applications, Techniques, and Practices (English Edition), Germany: Walter de Gruyter GmbH, 2019.
18. B. Blažica dan J. R. Lewis, "The System Usability Scale: Past, Present, and Future," International Journal of Human-Computer Interaction, vol. 34, no. 7, pp. 577-590, 2018.
19. Visual Paradigm, "What is a Software Process Model?," Visual Paradigm, [Online]. Available: <https://www.visual-paradigm.com/guide/software-development-process/what-is-a-software-process-model/>. [Accessed 15 November 2024].
20. J. T. Finnell dan B. E. Dixon, Clinical Informatics Study Guide: Text and Review, Switzerland: Springer Nature, 2022.
21. A. Kumaresan, S. Sivaprakash, V. P. dan C. Madhuri, A Textbook of Software Engineering, Academic Guru Publishing House, 2024.
22. S. Samsudin, N. Nurhalizah dan U. Fadilah, "Sistem Informasi Pendaftaran Magang Dinas Pemuda Dan Olahraga Provinsi Sumatera Utara," Jurnal Teknologi Dan Sistem Informasi Bisnis, vol. 4, no. 2, pp. 324-332, 2022.
23. D. Paspelava, "What is unit testing in software testing and why is it important?," Exposit, 30 March 2023. [Online]. Available: <https://www.exposit.com/blog/what-unit-testing-software-testing-and-why-it-important/>. [Accessed 15 November 2024].
24. J. Phang, M. Bagus and N. J. Perdana, "Analisis Capaian Program Merdeka Belajar Kampus Merdeka Menggunakan Metode K-Means," Computatio: Journal of Computer Science and Information Systems, pp. 172 - 183, 2024.
25. M. A. Rumbang and Tony, "Perancangan Aplikasi Presensi Berbasis Web untuk Karyawan Magang di PT. Sembilan Pilar Semesta," Jurnal Ilmu Komputer dan Sistem Informasi, vol. 12, no. 2, 2024.
26. B. Klug, "An Overview of the System Usability Scale in Library Website and System Usability Testing," Journal of Library User Experience, vol. 1, no. 6, 2017.