JURNAL TEKNOLOGI DAN OPEN SOURCE
Vol. 9, No. 1, June 2026, pp. 01~17
e-ISSN: 2622-1659, accredited Four Grade by Kemenristekdikti, Decree No: 152/E/KPT/2023
DOI: https://doi.org/10.36378/jtos.v9i1.5188

Design and Development of a Website-Based Internship Information System Using the Personal Extreme Programming Method (Case Study: Dinas Komunikasi dan Informatika Kabupaten Sidoarjo)

Muhammad Faiq Al Abiyyi[1], Nur Cahyo Wibowo[2], Seftin Fitri Ana Wati [3]
[1,2,3]Department of Information System, Faculty of Computer Science, UPN "Veteran" Jawa Timur, Indonesia

Article Info

Article history:

Received 12 28, 2025
Revised 03 23, 2026
Accepted 05 15, 2026

Keywords:

Information System
Personal Extreme Programming
Internship
Website
Diskominfo Sidoarjo

ABSTRACT

The management of the internship program at the Department of Communication and Informatics (Diskominfo) of Sidoarjo Regency is currently conducted manually. This conventional approach faces significant challenges regarding information transparency, the ease of monitoring daily activities, and the administration of program completion, necessitating a digital transformation to enhance service quality. This research aims to design and develop a web-based internship information system to automate the entire administrative process at Diskominfo Sidoarjo. The research methodology employed is Personal Extreme Programming (PXP), chosen for its agility in individual development. The PXP phases conducted include requirements gathering via interviews, planning which resulted in 28 user stories, iterative design, implementation using Express.js and React.js, and system testing. The result is a functional information system that provides features for internship data management, daily activity monitoring, and program completion administration. System testing was conducted using the black-box testing method, which demonstrated that all functionalities met the functional requirements of the users. Overall, the web-based internship information system at the Department of Communication and Informatics of Sidoarjo Regency was successfully developed using the Personal Extreme Programming approach.

Corresponding Author:

Muhammad Faiq Al Abiyyi
Department of Information Systems
Universitas Pembangunan Nasional "Veteran" Jawa Timur
Surabaya, Indonesia
Email: abiyyiaja@gmail.com

1. Introduction

The Department of Communication and Informatics (Diskominfo) is a government agency responsible for managing information and communication technology across various regions in Indonesia. Diskominfo of Sidoarjo Regency plays a pivotal role in supporting the Regent in executing local government functions, particularly in the fields of communication, informatics, statistics, and encryption, including additional duties assigned to the regional government [1]. In the modern era, the implementation of electronic-based government systems (SPBE) is essential to improve public service efficiency and

transparency [2]. In an effort to enhance the quality of human resources, Diskominfo Sidoarjo opens opportunities for university and school students to participate in an internship program. This program aims to provide hands-on experience in communication, informatics, and statistics, thereby equipping participants with relevant skills and insights for the professional world [3]. Internship programs are crucial for bridging the gap between academic theory and industrial practice [4]. Approximately 25 interns participate annually, with an average duration of three months per period. The program accommodates both individual and group applicants. Furthermore, this internship program supports Diskominfo in realizing transparent governance and provides organizational benefits, such as improving its public image and reputation [5].

Although the internship program organized by Diskominfo Sidoarjo offers relevant skills and insights for the workforce, its management still faces numerous obstacles due to the absence of a supporting information system. This deficiency results in a lack of clarity regarding the registration process and internship quotas. Manual administrative processes often lead to data redundancy and inefficient information retrieval [6]. The lack of transparency in the workflow and quota information makes it difficult for prospective interns to understand the stages or availability of positions. On the other hand, staff across various divisions face difficulties in monitoring whether quotas are filled or vacant, which can lead to mismatches in intern placement. The absence of a system also prevents staff from monitoring the daily activities of interns practically, resulting in suboptimal guidance and evaluation. Digitalization of internship monitoring is proven to increase the effectiveness of student supervision and assessment [7]. These issues indicate that Diskominfo requires a systematic solution to address these challenges.

Previous studies have demonstrated that software development methods such as Extreme Programming (XP) are capable of producing information systems with shorter development times and iterative processes to achieve established goals [8]. Salim and Alijoyo (2021) conducted research using the Extreme Programming method to create a web-based internship service management information system at Diskominfo Purwakarta. The results showed that the system functioned well, passed black-box testing, and achieved a score of 80.8 on the System Usability Scale (SUS) test. This score indicates that users were satisfied with the developed system [9]. Another study by Saputra and Prihandani (2024) also demonstrated the success of the Extreme Programming method in developing a Node.js-based internship portal. System testing using black-box and white-box methods proved that the developed application ran as expected and met user needs [10]. Furthermore, web-based systems developed with Agile methods have shown superior adaptability to changing user requirements compared to traditional models [11] [12].

In addition to XP, the Personal Extreme Programming (PXP) approach offers significant advantages over traditional methods like Waterfall, particularly regarding flexibility and efficiency for solo developers. A comparative study conducted in research [13] on the development of a practical work monitoring application showed that PXP provides greater convenience and flexibility due to its iterative nature and responsiveness to changing user needs. Unlike Waterfall, which tends to be rigid and performs testing only at the end of the project cycle [14], the implementation phase in PXP integrates unit testing, coding, and refactoring simultaneously. This mechanism allows errors to be detected much earlier as soon as a new feature is completed, and it significantly reduces development time [15]. Recent studies also emphasize that PXP is highly suitable for academic projects or small-scale development where resources are limited but quality must be maintained [16].

Based on the advantages and the capability for early error adaptation demonstrated in previous studies, the development of the internship information system in this research applies the Personal Extreme Programming (PXP) method. Unlike previous studies that often focus on general internship management in the private sector, this research contributes to the specific context of local government bureaucracy in Indonesia. The primary novelty of this study lies in the system's design compliance with the Electronic-Based Government System (SPBE) standards as mandated by Presidential Regulation No. 95 of 2018 [17]. This system is expected to enhance information flow transparency, facilitate the monitoring of the internship process, and support staff in providing more optimal guidance and evaluation for interns, while serving as a reference for implementing agile methods in government agencies with limited technical resources.

2. Research Method

According to Dzhurov et al., as cited in the study “Pengembangan Aplikasi Perangkat Bergerak Pengobatan Tuberculosis Berbasis Android Menggunakan Personal Extreme Programming,” Personal Extreme Programming is a method designed to assist developers working independently in developing software effectively [18]. PXP integrates practices from Extreme Programming with principles adapted from the Personal Software Process (PSP) to streamline the software development process. This method is designed to enhance the efficiency and work quality of individual developers by automating daily activities and conducting periodic evaluations [19]. To achieve performance improvements, Personal Extreme

Programming adopts Extreme Programming principles that have been specifically tailored to the needs of autonomous developers. The stages of the Personal Extreme Programming method are presented in Figure 1.

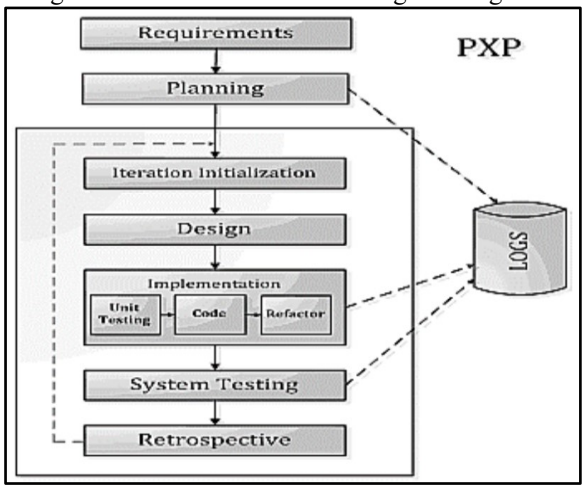


Figure 1. Stages in the Personal Extreme Programming Method

2.1. Requirements

In the requirements phase, the primary objective is to gather system requirements from prospective users. This process is typically conducted using user stories, which capture the "who", "what", and "why" of a requirement in a simple language [20]. While developers can document requirements based on user input, it is preferable to allow prospective users the freedom to write their requirements directly on a blank sheet. This approach enables users to express their needs and expectations of the system more freely.

2.2. Planning

In the planning phase, requirements gathered from user stories are analyzed and categorized into several stories. Each story may be decomposed into smaller segments. During this phase, time estimation for each iteration is established based on priority levels using story points. The magnitude of a story point is directly proportional to the development time [21]. The determination of story point values is performed by the developer by assessing the time required to complete a specific feature. Subsequently, the estimated duration per iteration is structured based on priority, referencing the velocity value. Velocity is a measure of the developer's capacity or speed in completing a number of story points within a single iteration cycle, calculated based on the total story points of one iteration [22].

2.3. Iteration Initialization

The iteration initialization phase marks the commencement of an iteration. This process begins with the selection of stories based on predetermined priorities. The duration of an iteration typically ranges from one to three weeks, depending on the complexity of the stories being addressed. If changes in requirements occur, those requirements are decomposed into several tasks, grouped by similarity, and then reviewed to re-establish priorities for each iteration.

2.4. Design

The design phase involves determining the design patterns to be utilized in program development, including the design of the user interface and database. Unified Modeling Language (UML) diagrams, such as Use Case and Activity Diagrams, are often employed to visualize the system architecture before coding begins [23]. In this phase, the principle of "simple design" is applied, focusing exclusively on the requirements currently communicated by the user. The system design does not encompass predictions for future requirements; thus, the design is created to be as simple and efficient as possible, aligned strictly with current needs.

2.5. Implementation

The implementation phase is the process of writing program code according to the previously created design, while simultaneously testing the code to ensure it functions correctly. This phase consists of three main activities: unit testing, coding, and refactoring (improving code structure without altering its functionality). The use of modern frameworks like React.js for the frontend and Express.js for the backend ensures the code is modular and maintainable [24].

2.6. System Testing

The system testing phase aims to ensure that the program code aligns with the specified requirements and meets user expectations. This research employs two testing approaches: Black Box Testing and User Acceptance Testing (UAT). Black Box Testing is crucial to validate the input-output functionality without ignoring the internal code structure [25]. Any errors in the code or discrepancies with the requirements must be documented and rectified; if no errors are found, the iteration is considered complete. Subsequently, UAT is conducted to evaluate the system's usability and user satisfaction from the end-user's perspective. This testing is carried out by distributing questionnaires to relevant stakeholders, including administrators, division sub-coordinators, and internship participants, using a Likert scale to quantify their feedback.

2.7. Retrospective

The retrospective phase aims to evaluate the outcomes of every process that has been executed. In this phase, a review is conducted to determine whether the development time aligned with the initial plan. If the development time exceeds the specified limits, the causes must be identified to serve as lessons for the next iteration or future projects.

3. Result and Discussion

3.1. Requirements

In the requirements phase, the researcher conducted interviews with the Public Relations division (Pranata Humas) of Diskominfo Sidoarjo to analyze the internship workflow. Observation results indicated that the current management process, ranging from checking quotas to document submission, is still conducted manually. This condition highlights the urgency of digital transformation to overcome operational constraints and clarify the roles of the involved actors, as the manual handling of data often leads to inefficiencies in monitoring and administration.

As a remedial solution, the researcher designed a proposed system workflow where the process begins with prospective participants accessing the website to check internship quotas in real-time. Registration and document submission are conducted online for verification by the administrator, and upon account activation, participants carry out their internship activities with the mandatory requirement to input daily logbooks directly through the system, replacing the previous manual recording method. The workflow concludes with the submission of activity reports verified by the divisional sub-coordinator, which then triggers the system to automatically issue a completion letter and an internship certificate that can be printed independently by the participant to ensure administrative effectiveness.

3.2. Planning

The planning process in this research commences with a comprehensive gathering of system requirements from prospective users. This requirement collection mechanism is documented in the form of user stories, where the developer records feature specifications based on narratives provided directly by the users. Subsequently, the collected requirements are analyzed and classified into several story groups. Each complex story is decomposed into smaller, specific tasks to ensure they are manageable.

Once the task list is established, the researcher performs time estimation using the story point method. Story point values are determined based on the researcher's experience in assessing feature complexity, where the magnitude of the story point is directly proportional to the estimated development time. The development strategy in this research deliberately divides the work into small-scale iterations. This approach aims to ensure a more targeted work focus, minimize the risk of cumulative errors, and enable faster feature evaluation.

The sequence of iteration execution is structured based on data flow dependencies. The first priority focuses on developing the authentication feature as the foundation for access security. Once access rights are established within the system, the iteration proceeds to the internship registration feature, serving as the entry point for the business process. The subsequent stage focuses on primary operational features, namely internship activity reporting (daily logbooks and final internship reports), and concludes with the internship completion feature (submission of internship reviews and certificates). Details of the user stories along with their estimated development times are presented in Table 1.

Table 1. User Story Information System Internship

Kode User Story	User Story	Story Point
US-01	User Story / Description	2
US-02	Interns can register an account.	1
US-03	Users can log in.	1

Kode User Story	User Story	Story Point
US-04	Users can reset their password.	1
US-05	Interns can edit their profile.	1
US-06	Interns can view internship quotas.	1
US-07	Interns can provide feedback and suggestions.	3
US-08	Interns can apply for an internship.	1
US-09	Interns can view their internship application status.	3
US-10	Interns can upload daily reports (logbooks).	2
US-11	Interns can upload final internship reports.	2
US-12	Interns can add internship reviews.	2
US-13	Interns can print internship certificates.	3
US-14	Admin and Division Sub-Coordinators can send certificates.	1
US-15	Admin can manage internship divisions/fields.	2
US-16	Admin and Division Sub-Coordinators can send internship acceptance letters.	2
US-17	Admin and Division Sub-Coordinators can view intern data.	1
US-18	Admin and Division Sub-Coordinators can view internship proposal data.	2
US-19	Admin and Division Sub-Coordinators can respond to internship proposals.	2
US-20	Admin can edit intern data.	1
US-21	Admin can manage Division Sub-Coordinator accounts.	2
US-22	Admin can verify intern accounts.	1
US-23	Admin can view feedback and suggestion data.	1
US-24	Admin and Division Sub-Coordinators can export internship data.	1
US-25	Admin and Division Sub-Coordinators can export daily report data.	1
US-26	Admin and Division Sub-Coordinators can view participants' daily report data.	1
US-27	Admin and Division Sub-Coordinators can view participants' final internship report data.	2
US-28	Admin and Division Sub-Coordinators can verify participants' final internship reports.	1

Following the time estimation phase for each user story, the subsequent step involves structuring the iterations. This iteration planning is determined based on the priority and dependencies among the user stories. This stage resulted in four iterations, with a velocity of 13 for Iteration 1, 10 for Iteration 2, 11 for Iteration 3, and 10 for Iteration 4. The velocity value indicates the duration required for each iteration in days; specifically, Iteration 1 will require 13 days, Iteration 2 will take 10 days, Iteration 3 will take 11 days, and Iteration 4 will take 10 days. The results of the iteration planning for this research are presented in Table 2.

Table 2. Iteration Compilation Results

Iteration 1		
Kode User Story	User Story	Story Point
US-01	User Story / Description	2
US-02	Interns can register an account.	1
US-03	Users can log in.	1
US-04	Users can reset their password.	1
US-21	Admin can manage Division Sub-Coordinator accounts.	2
US-14	Admin and Division Sub-Coordinators can send certificates.	1
US-20	Admin can edit intern data.	1
US-16	Admin and Division Sub-Coordinators can send internship acceptance letters.	2
US-19	Admin and Division Sub-Coordinators can respond to internship proposals.	2
Velocity		13
Iteration 2		
Kode User	User Story	Story

Story		Point
US-05	Interns can edit their profile.	1
US-07	Interns can provide feedback and suggestions.	3
US-08	Interns can apply for an internship.	1
US-17	Admin and Division Sub-Coordinators can view intern data.	1
US-18	Admin and Division Sub-Coordinators can view internship proposal data.	2
US-15	Admin can manage internship divisions/fields.	2
Velocity		10
	Iteration 3	
Kode User Story	User Story	Story Point
US-09	Interns can view their internship application status.	3
US-10	Interns can upload daily reports (logbooks).	2
US-23	Admin can view feedback and suggestion data.	1
US-24	Admin and Division Sub-Coordinators can export internship data.	1
US-25	Admin and Division Sub-Coordinators can export daily report data.	1
US-26	Admin and Division Sub-Coordinators can view participants' daily report data.	1
US-27	Admin and Division Sub-Coordinators can view participants' final internship report data.	2
Velocity		11
	Iteration 4	
Kode User Story	User Story	Story Point
US-06	Interns can view internship quotas.	1
US-11	Interns can upload final internship reports.	2
US-12	Interns can add internship reviews.	2
US-13	Interns can print internship certificates.	3
US-22	Admin can verify intern accounts.	1
US-28	Admin and Division Sub-Coordinators can verify participants' final internship reports.	1
Velocity		10

3.3. Iteration Initialization

The iteration initialization phase marks the execution of system development based on the iterations defined in the preceding stage. Each iteration encompasses a detailed description of the tasks derived from every user story, aligned with the planned iteration structure.

3.4. Design

In this phase, the researcher translates system requirements into a comprehensive visual design before commencing the coding process. The objective is to establish a clear guideline for each iteration. In the execution of this phase, the researcher utilizes the LucidChart platform specifically to design activity diagrams that illustrate user activity flows. Meanwhile, for other technical diagrams such as use case, sequence, and class diagrams, the researcher employs StarUML software. Finally, to provide a concrete representation of the website interface layout that users will interact with, the researcher constructs initial design sketches, or wireframes, using the Figma design application.

3.4.1. Use Case Diagram

In this design phase, a comprehensive Use Case Diagram was constructed to provide an overview of the interactions between actors and the system. The Use Case Diagram in Figure 4 illustrates how each actor—namely the Admin, Division Sub-Coordinator, and Intern, interacts with the various features available in the Internship Information System. Each actor has access to specific features corresponding to their respective roles. For instance, the Admin has full access to master data management and system configuration features, such as managing Division Sub-Coordinator accounts, verifying intern accounts, and managing internship quotas. Interns interact with features supporting the registration workflow and internship activities, ranging from account registration, internship application, and daily report submission, to certificate printing. Meanwhile, the Division Sub-Coordinator is responsible for managing the internship application process, including responding to proposals, issuing acceptance letters, and sending certificates. The overall Use Case Diagram is presented in Figure 2.

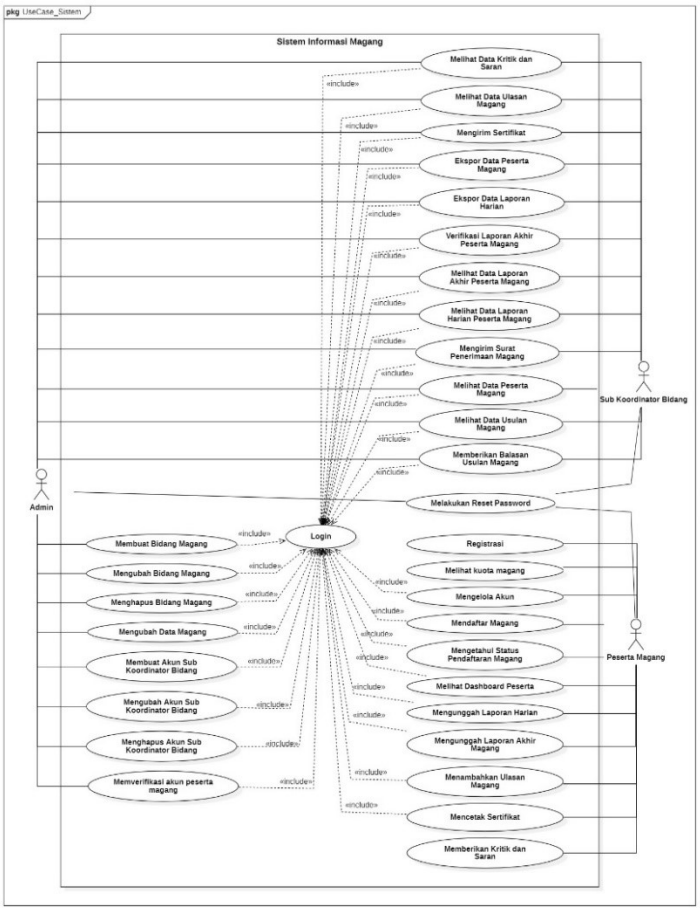


Figure 2. Use Case Diagram of the Overall Internship Information System

3.4.2. Activity Diagram

a. *Activity Diagram Input Daily Report*

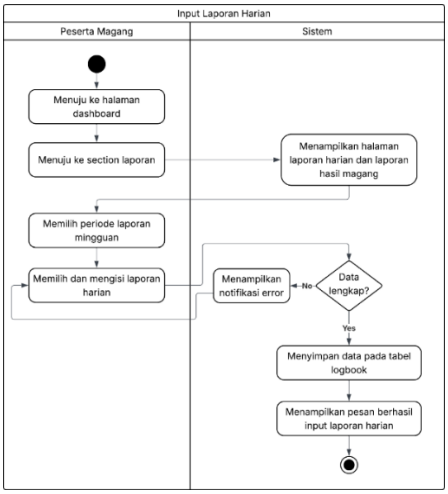


Figure 3. Activity Diagram Input Daily Report

The daily report upload workflow, as illustrated in Figure 3, initiates when the Intern accesses the daily report menu and clicks the 'Add Report' button. Subsequently, the system displays the entry form, requiring the Intern to input the activity description corresponding to the specific date. Upon submission, the system validates the input for completeness; if the data is valid, the report is stored in the *Logbook* table and reflected in the activity history list.

b. *Activity Diagram for Final Report Input and Final Report Verification*

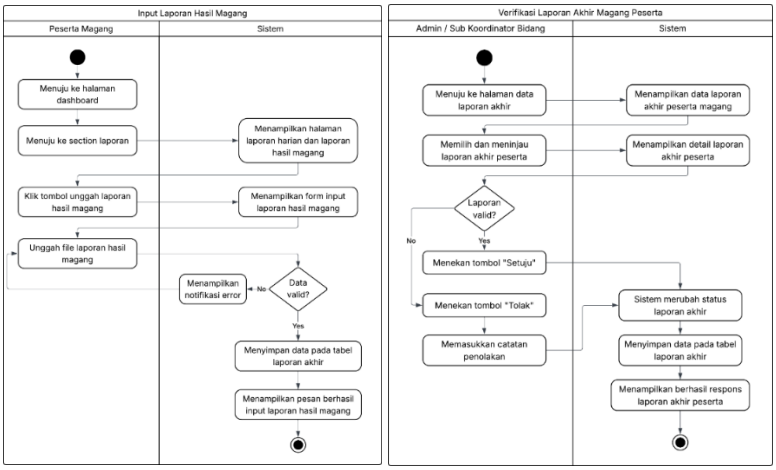


Figure 4. Activity Diagram for Final Report Input and Final Report Verification

The activity diagrams presented in Figure 4 illustrate the comprehensive lifecycle of the final internship report, encompassing both the submission process by the Intern and the subsequent verification by the administration. The workflow initiates when the Intern accesses the report menu, triggering an automatic system validation of graduation prerequisites—specifically the completion of the internship period and the fulfillment of daily logbook entries. Upon satisfying these conditions, the Intern uploads the final report in PDF format, which is then validated and stored in the *Laporan Akhir* table. Following the submission, the workflow shifts to the verification phase, where the Admin or Division Sub-Coordinator reviews the document. When the actor selects a decision (Approve or Reject), the interface dispatches a request to the LaporanController, which then processes the data, updates the status in the *Laporan Akhir* table, and returns a confirmation response to the user.

c. *Activity Diagram Send Certificate*

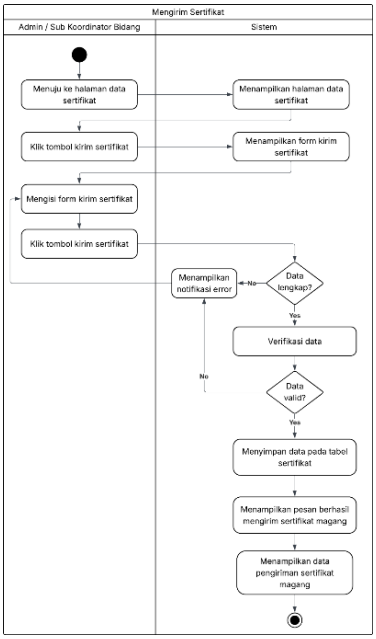


Figure 5. Activity Diagram Send Certificate

The activity diagram in Figure 5 illustrates the final workflow for the Admin or Division Sub-Coordinator regarding the issuance of completion certificates (US-13). The process initiates when the actor

accesses the certificate issuance page. The system interacts with the *Sertifikat* table to retrieve and filter interns who have fulfilled all completion prerequisites: the submission of complete daily reports, approval of the final report, conclusion of the internship period, and submission of the internship review. Only interns satisfying these criteria are populated in the list. Subsequently, the Admin selects an intern and clicks the send button. The process concludes when the system successfully updates the delivery status in the *Sertifikat* table, thereby automatically enabling the certificate printing functionality for the respective intern.

3.4.3. Sequence Diagram

a. *Daily Report Input Sequence Diagram*

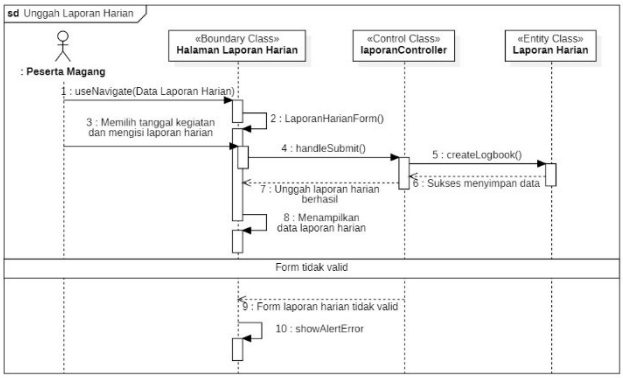


Figure 6. Daily Report Input Sequence Diagram

The Daily Internship Report Input workflow, as depicted in Figure 6, details the technical interactions involved when an Intern logs their daily activities. The process begins when the actor completes the activity form corresponding to the specific date and selects the save button, prompting the report page to dispatch a request to the laporanController. Subsequently, the controller validates the input and persists the report data into the *LaporanHarian* table. Once the data is successfully stored, the controller returns a success response to the daily report interface, triggering a confirmation notification for the participant.

b. *Final Report Input Sequence Diagram and Final Report Verification*

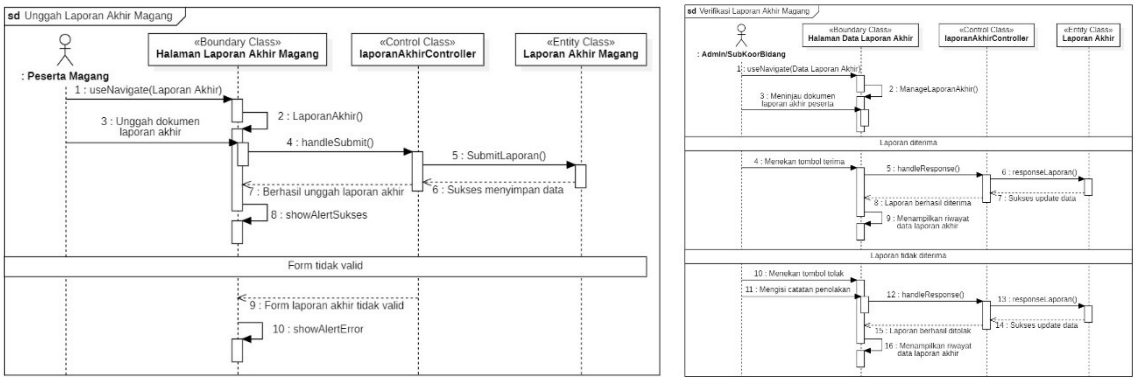


Figure 7. Final Report Input Sequence Diagram and Final Report Verification

The final internship report upload workflow, as illustrated in Figure 7, details the technical interactions involved in the Intern's submission of the program completion document. The process is initiated on the final report interface, where the Intern uploads the document and selects the save button, thereby dispatching a request to the laporanController. Subsequently, the controller validates the graduation prerequisites—specifically the completeness of the daily logs—and persists the document data into the *LaporanAkhir* table. Once the data is successfully stored, the controller returns a positive response to the interface, triggering a success notification for the participant.

c. Send Certificate Sequence Diagram

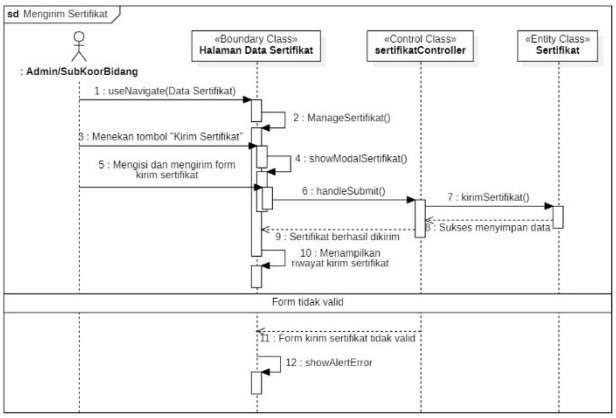


Figure 8. Send Certificate Sequence Diagram

The certificate issuance workflow, as depicted in Figure 8, details the technical interactions involved when the Admin or Division Sub-Coordinator issues the internship program completion document. The process initiates when the actor triggers the certificate request, prompting the interface to transmit the delivery data to the SertifikatController. Subsequently, the controller verifies the intern's graduation prerequisites within the *Sertifikat* table, ensuring that daily reports are complete, the final report is approved, the internship period has concluded, and the mandatory internship review has been submitted. If all criteria are satisfied, the controller executes a status update for certificate delivery in the database. Once the update is confirmed, the controller returns a success response to the interface, displaying a notification to the Admin or Division Sub-Coordinator.

3.4.4. Class Diagram

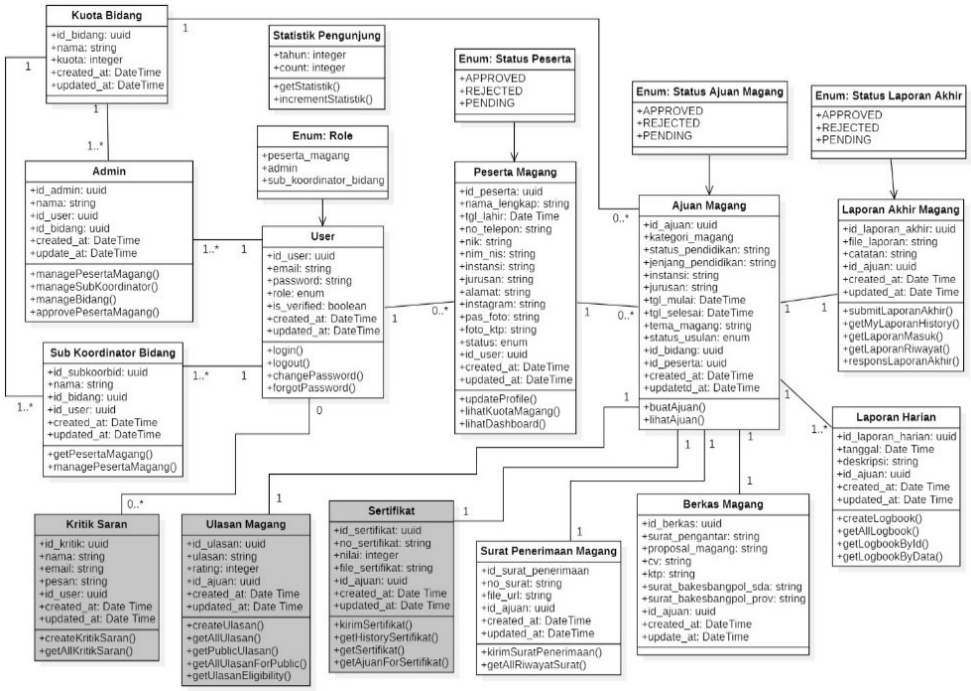


Figure 9. Class Diagram of the Overall Internship Information System

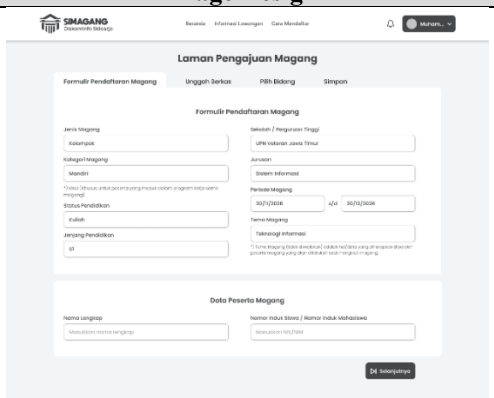
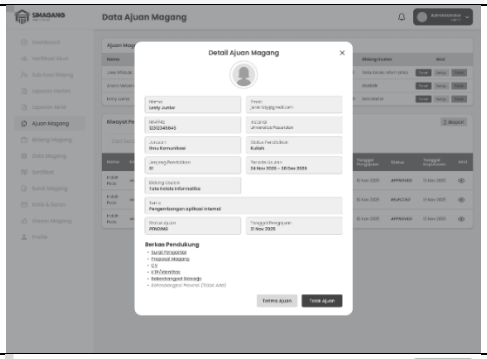
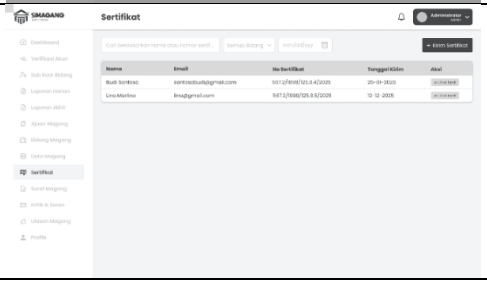
The Class Diagram in Figure 9 illustrates the database structure updates designed to support feedback management and internship program completion administration during the fourth iteration. The shaded classes (highlighted in gray) within the diagram indicate the functionality newly built or developed in

this iteration. This update involves the addition of two main related entities: KritikSaran, which functions to store input from general users, and UlasanMagang, which stores reviews of internship activities completed by the Interns. This data structure enhancement is designed to facilitate the separate collection and archiving of feedback, while simultaneously serving as the technical foundation for program performance evaluation features (US-22, US-28) and certificate issuance (US-13).

3.4.5. Wireframe

The wireframe design phase focuses on establishing the visual structure for the essential user interfaces across the entire internship management lifecycle. This comprehensive design covers the initial access phase, represented by the Registration Page and the Account Verification interface. It proceeds to the core operational workflows, including the Internship Application Page and the Application Verification interface. Additionally, the design addresses the monitoring and program completion stages by detailing the interfaces for Daily and Final Report inputs, as well as the administrative dashboards for Report Verification and Certificate Issuance. The complete list of designed wireframes is presented in Table 4.

Table 4. Wireframe Design Results

No	Page Design	Information	Role
1		Internship Application Page	Intern
2		Internship Application Verification Page	Admin and Sub-Coordinator
3		Send Certificate Page	Admin and Sub-Coordinator

3.5. Implementation

In this phase, the researcher initiates the implementation of the system design adhering to the Test-Driven Development (TDD) principle, where tests are constructed prior to the development of functional code. The researcher utilizes the JavaScript framework, Jest, to establish unit tests that serve as the initial reference point. Once the tests are defined, the researcher proceeds with coding using the JavaScript programming language, implementing Express.js for the server-side (backend) and React.js for the client-side (frontend), with MySQL serving as the database. If the written code fails to pass the tests or encounters

technical constraints, the researcher executes a refactoring step to improve the code structure. This process ensures that every developed feature functions according to the plan before proceeding to the development of subsequent features.

3.5.1. Unit Testing

Figure 10. Internship Registration Unit Testing

The unit testing for the Intern registration user story was designed to validate the entire business logic within the controller. These tests encompass both success scenarios and various failure scenarios to ensure that error handling mechanisms function effectively. The tested failure scenarios include the submission of incomplete data, the use of duplicate emails, invalid date formats, and simulated database errors. The test results, as presented in Figure 10, demonstrate that all five test scenarios (5 passed) were executed successfully. This confirms that the registration controller logic functions in accordance with the specified requirements.

3.5.2. Code

a. Daily Report Input Page

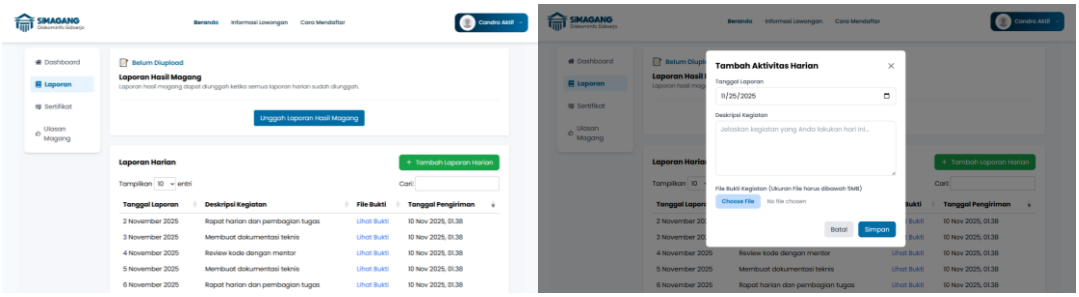


Figure 11. Daily Report Input Page

The Daily Report interface presented in Figure 11 reflects the UI refinements implemented following the previous iteration's retrospective. A notable feature is the conditional link to the Final Internship Report upload located at the top of the page, which becomes accessible only upon the completion of all daily reporting requirements. The intern's activity history is efficiently organized within a datatable, displaying essential attributes such as the report date, activity description, proof files, and submission timestamp in a unified view. To record new activities (US-09), the intern selects the 'Add Daily Report' button, which triggers a modal form. Designed for both simplicity and comprehensiveness, this modal requires inputs for the date and activity description, along with an upload field for supporting evidence that enforces a 5MB size limit to ensure optimal system performance.

b. Final Report Input and Final Report Verification Pages

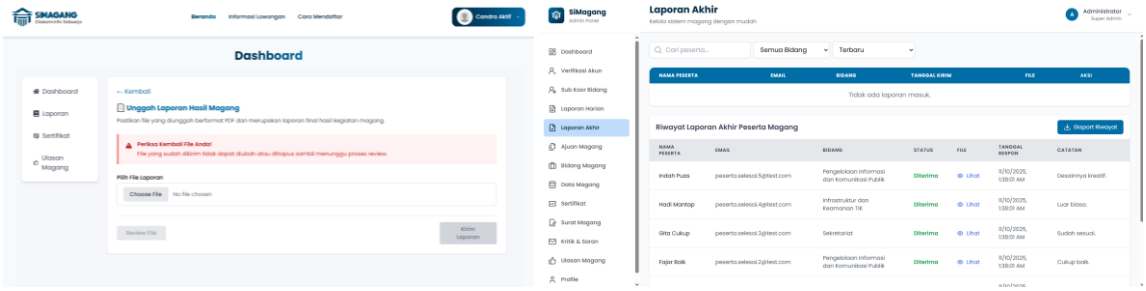


Figure 12. Final Report Input and Final Report Verification Pages

The user interfaces presented in Figure 12 illustrate the comprehensive workflow for the final internship report, encompassing both the submission by the intern (US-10) and the subsequent administrative verification. The intern's interface features an intuitive drag-and-drop upload form for submitting PDF documents, accompanied by transparent status indicators—such as 'Not Submitted' or 'Pending Verification'—that track the progress of the review. Upon submission via the 'Save' button, the workflow transitions to the administrative dashboard, which serves as the central hub for verifying program graduation. In this view, Admins and Division Sub-Coordinators access a detailed roster of submissions, including internship periods, report files, and current verification statuses (Pending, Approved, or Rejected). The interface empowers administrators to validate documents and issue decisions through the 'Action' column, while also providing an 'Export Internship Data' function (US-23) to generate a complete recapitulation of intern performance data.

c. Print Certificate and Send Certificate Page

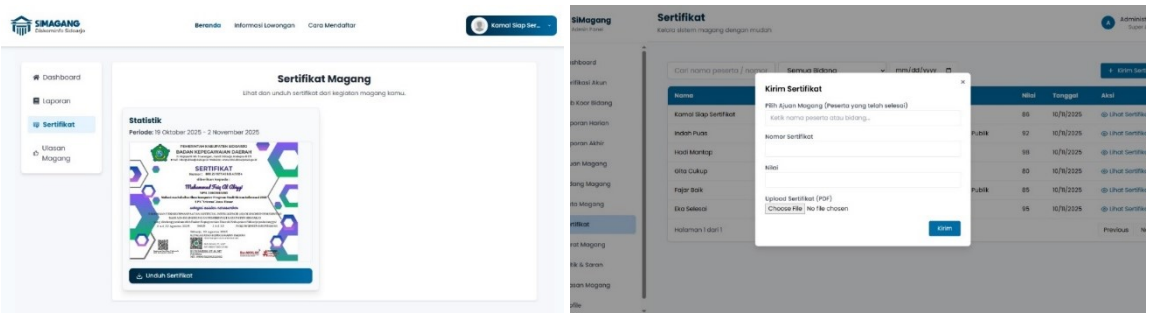


Figure 13. Print Certificate and Send Certificate Page

The user interfaces presented in Figure 13 illustrate the certificate management workflow from both the participant and administrative perspectives. For the Intern, the interface provides a visual preview of the completion document alongside internship period statistics, allowing for immediate retrieval via the 'Download Certificate' button. However, this feature is conditionally restricted, becoming accessible only after the internship status is finalized and the certificate has been formally issued. On the administrative side, the process is executed through the certificate issuance modal. This interface is triggered when an Admin or Division Sub-Coordinator selects an eligible candidate to finalize the issuance. The modal facilitates the uploading or confirmation of the certificate file; once the 'Send' action is confirmed, the system updates the status, automatically enabling the certificate printing functionality for the respective intern.

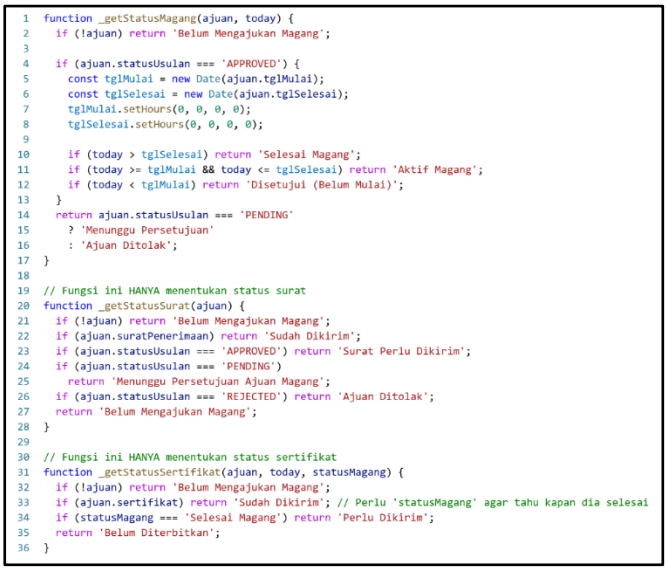
3.5.3. Refactor

In the refactoring phase, improvements were made to the internal code structure to enhance readability and maintainability. One specific section of the code subjected to refactoring was the getAllPesertaMagang function, which is responsible for handling the retrieval and transformation of data for all interns.

```
module.exports = {
  getAllPesertaMagang: async (req, res, next) => {
    const allPeserta = users.map((user) => {
      const profile = user.pesertaMagang || {};
      const ajuan =
        profile.ajuan && profile.ajuan.length > 0 ? profile.ajuan[0] : null;

      let statusSuratMagang = 'Belum Mengajukan Magang';
      let statusMagang = 'Belum Mengajukan Magang';
      let statusSertifikat = 'Belum Mengajukan Magang';
      let tglMulai = null;
      let tglSelesai = null;

      if (ajuan) {
        if (ajuan.suratPenerimaan) {
          statusSuratMagang = 'Sudah Dikirim';
        } else if (ajuan.statusUsulan === 'APPROVED') {
          statusSuratMagang = 'Surat Perlu Dikirim';
        } else if (ajuan.statusUsulan === 'PENDING') {
          statusSuratMagang = 'Menunggu Persetujuan Ajuan Magang';
        } else if (ajuan.statusUsulan === 'REJECTED') {
          statusSuratMagang = 'Ajuan Ditolak';
        }

        tglMulai = new Date(ajuan.tglMulai);
        tglSelesai = new Date(ajuan.tglSelesai);
        tglMulai.setHours(0, 0, 0, 0);
        tglSelesai.setHours(0, 0, 0, 0);

        if (ajuan.statusUsulan === 'APPROVED') {
          if (today > tglSelesai) {
            statusMagang = 'Selesai Magang';
```

Figure 14. Internship Registration Unit Testing

Prior to refactoring, as shown in Figure 14, this function was excessively long. The complex business logic required to determine participant statuses (such as statusMagang, statusSuratMagang, and statusSertifikat) was intertwined directly with the data retrieval (querying) and transformation logic. This resulted in a bloated and illegible main function, thereby complicating debugging efforts whenever business rule modifications were necessary.



```
function _getStatusMagang(ajuan, today) {
  if (!ajuan) return 'Belum Mengajukan Magang';

  if (ajuan.statusUsulan === 'APPROVED') {
    const tglMulai = new Date(ajuan.tglMulai);
    const tglSelesai = new Date(ajuan.tglSelesai);
    tglMulai.setHours(0, 0, 0, 0);
    tglSelesai.setHours(0, 0, 0, 0);

    if (today > tglSelesai) return 'Selesai Magang';
    if (today >= tglMulai && today <= tglSelesai) return 'Aktif Magang';
    if (today < tglMulai) return 'Disetujui (Belum Mulai)';
  }
  return ajuan.statusUsulan === 'PENDING'
    ? 'Menunggu Persetujuan'
    : 'Ajuan Ditolak';
}

// Fungsi ini HANYA menentukan status surat
function _getStatusSurat(ajuan) {
  if (!ajuan) return 'Belum Mengajukan Magang';
  if (ajuan.suratPenerimaan) return 'Sudah Dikirim';
  if (ajuan.statusUsulan === 'APPROVED') return 'Surat Perlu Dikirim';
  if (ajuan.statusUsulan === 'PENDING')
    return 'Menunggu Persetujuan Ajuan Magang';
  if (ajuan.statusUsulan === 'REJECTED') return 'Ajuan Ditolak';
  return 'Belum Mengajukan Magang';
}

// Fungsi ini HANYA menentukan status sertifikat
function _getStatusSertifikat(ajuan, today, statusMagang) {
  if (!ajuan) return 'Belum Mengajukan Magang';
  if (ajuan.sertifikat) return 'Sudah Dikirim'; // Perlu 'statusMagang' agar tahu kapan dia selesai
  if (statusMagang === 'Selesai Magang') return 'Perlu Dikirim';
  return 'Belum Diterbitkan';
}
```

Figure 15. Internship Registration Unit Testing

Consequently, refactoring was executed using the Extract Method technique, as illustrated in Figure 15. The complex business logic for status determination was extracted into three distinct helper functions: _getStatusMagang(), _getStatusSurat(), and _getStatusSertifikat(). Following the refactoring process, the main getAllPesertaMagang function became significantly cleaner and more readable. The function now serves solely as a controller that invokes these helper functions, resulting in simplified code that is much easier to maintain or test independently, as demonstrated in Figure 16.

Figure 16. Internship Registration Unit Testing

3.6. *System Testing*

The system testing phase was executed using the blackbox testing method to ensure that all system functionalities met the specifications defined in the user stories. As presented in Table 4, the testing results highlight the validation of key reporting and completion features. Specifically, the testing covered the uploading of daily logs (US-09) and final internship reports (US-10), followed by the verification process conducted by the Admin or Division Sub-Coordinator (US-27). Furthermore, the testing included the administrative functionality for certificate issuance (US-13) to ensure the internship cycle concludes effectively. The results indicate that all tested scenarios were declared successful, confirming the system's readiness for deployment.

Table 4. System Test Results

User Story	Scenario	Expected Results	Test Results	Status
US-09	Intern uploads daily report	The Intern can upload a daily report.	The Intern successfully uploaded a daily report.	Success
US-10	Intern uploads final internship report	The Intern can upload a final internship report.	The Intern successfully uploaded a final internship report.	Success
US-27	Admin or Sub-Coordinator verifies Intern's final report	The Admin or Division Sub-Coordinator can verify the Intern's final report.	The Admin or Division Sub-Coordinator successfully verified the Intern's final report.	Success
US-13	Admin and Sub-Coordinator send certificate	The Admin and Division Sub-Coordinator can send the certificate.	The Admin and Division Sub-Coordinator successfully sent the certificate.	Success

In addition to functional testing, non-functional evaluation was conducted to measure user acceptance and system usability through User Acceptance Testing (UAT). Questionnaires were distributed to 26 respondents, consisting of 1 Admin from the IT Governance Division (Tata Kelola Informatika), 5 Division Sub-Coordinators representing all agency divisions (Secretariat, Statistics, Public Information and Communication Management, ICT Infrastructure and Security, and IT Governance), and 20 internship participants. The questionnaire aimed to evaluate the system based on aspects of ease of use, interface design, and information clarity. The UAT results were calculated using a Likert scale analysis. The evaluation showed that the system achieved an average user approval rating of 87%. This score falls into the "Very Good" category, indicating that the system is not only functionally correct but also user-friendly and meets the operational needs of the Diskominfo Sidoarjo internship environment. The high acceptance rate confirms that the transition from manual to digital processes was well-received by the stakeholders.

3.7. *Retrospective*

The retrospective summary presented in Table 5 provides an evaluation of the development execution for the critical post-internship workflows. The scope of this evaluation focuses on the management of daily and final reporting (US-09, US-10, US-27) and the final administrative function of certificate issuance (US-13). A critical observation from the data is the precise alignment between the Job Estimate and Job Realization for these stories. This consistency indicates that the story point estimation method was accurate and the development process was executed efficiently without significant deviations. Consequently, all targeted functionalities were successfully completed ("Finished") within the allocated timeframes.

Table 5. Retrospective Summary Results

User Story	Story Point	Job Estimate	Job Realization	Status
US-09	3	3 Day	3 Day	Finished
US-10	2	2 Day	2 Day	Finished
US-27	2	2 Day	2 Day	Finished
US-13	3	3 Day	3 Day	Finished

4. Conclusion

Based on the problem formulation established in the introduction, this study successfully designed and developed a web-based internship information system at the Department of Communication and Informatics (Diskominfo) of Sidoarjo Regency using the Personal Extreme Programming (PXP) method. This research demonstrates that PXP is highly effective for accelerating digital transformation in a government environment with limited technical resources. The resulting system not only addresses manual operational inefficiencies but also aligns with the Electronic-Based Government System (SPBE) standards, ensuring compliance with national administrative regulations. The development process, executed through four structured iterations involving 28 user stories, produced a robust solution ranging from registration to certificate issuance.

The system's effectiveness is validated through comprehensive testing. Black Box testing confirmed that all functional requirements were met without errors. Furthermore, non-functional evaluation through User Acceptance Testing (UAT) involving 26 respondents yielded an average approval rating of 87%. This "Very Good" score indicates that the system is well-accepted by stakeholders and significantly improves service quality compared to the previous manual workflow. For future development, further research should focus on creating a mobile-responsive interface or dedicated application to improve accessibility. Additionally, future iterations could aim for deeper integration with central government data centers to support broader data synchronization and advanced analytics.

References

[1] Dinas Komunikasi dan Informatika Sidoarjo, “Tugas dan Fungsi Dinas Komunikasi dan Informatika Kabupaten Sidoarjo,” [Online]. Available: https://diskominfo.sidoarjokab.go.id/007/1717339580. [Accessed: Nov. 6, 2024].

[2] M. R. A. Nasution and M. Mesran, “Sistem Layanan Pengaduan Masyarakat (SIPEMAS) Berbasis Web pada Dinas Komunikasi dan Informatika Kota Pematangsiantar,” *Jurnal Teknologi dan Open Source (JTOS)*, vol. 5, no. 2, pp. 248-256, 2022.

[3] D. Nugraheni and L. S. Wijaya, “Pelaksanaan Program Internship dalam Upaya Meningkatkan Citra Lembaga Pendidikan (Studi Kasus: Fakultas Teknologi Informasi – Universitas Kristen Satya Wacana),” *Scriptura*, vol. 7, no. 2, pp. 47–56, 2017. doi: 10.9744/scriptura.7.2.47-56.

[4] A. S. Putra and O. M. Febriani, “Sistem Informasi Monitoring Akademik Siswa Berbasis Web,” *Jurnal Teknologi dan Open Source (JTOS)*, vol. 2, no. 1, pp. 31-40, 2019.

[5] D. D. Lutfia and D. R. Rahadi, “Analisis Internship bagi peningkatan Kompetensi mahasiswa,” *Jurnal Ilmiah Manajemen Kesatuan*, vol. 8, no. 3, pp. 199–204, 2020. doi: 10.37641/jimkes.v8i3.340.

[6] F. N. Hasanah and D. R. Untari, “Rekayasa Perangkat Lunak Sistem Informasi Pendaftaran Mahasiswa Baru Berbasis Web,” *Jurnal Manajemen Informatika dan Sistem Informasi*, vol. 3, no. 2, pp. 121-128, 2020.

[7] R. A. S. Arifin and D. P. Rakhmadani, “Rancang Bangun Sistem Informasi Monitoring Praktik Kerja Lapangan (PKL) Berbasis Web,” *Jurnal Teknologi Informasi dan Terapan*, vol. 7, no. 1, pp. 25-30, 2020.

[8] D. A. P. P. Sari and U. Hayati, “Penerapan Metode Extreme Programming Pada Sistem Informasi Penjualan Berbasis Web,” *Jurnal Teknologi dan Open Source (JTOS)*, vol. 4, no. 2, pp. 182-190, 2021.

[9] A. Y. Salim and F. A. Alijoyo, “Rancang Bangun Sistem Informasi Manajemen Layanan Magang di Diskominfo Kabupaten Purwakarta Berbasis Web dengan Menggunakan Metode Extreme Programming,” *Kudus Journal of Technology*, vol. 4, pp. 45–53, 2021.

[10] W. Saputra and K. Prihandani, “Rancang Bangun Portal Magang Berbasis Website Menggunakan Node.js (Studi Kasus: Fasilkom Unsika),” *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 8, 2024. doi: 10.36040/jati.v8i4.10012.

[11] I. K. A. G. Wiguna and I. P. Satwika, “Rancang Bangun Sistem Informasi Point of Sales Distro Management System dengan Menggunakan Framework React Native,” *Jurnal Teknologi dan Open Source (JTOS)*, vol. 5, no. 1, pp. 56-65, 2022.

[12] T. Hidayat and M. Muttaqin, "Pengujian Sistem Informasi Pendaftaran dan Pembayaran Wisuda Online menggunakan Black Box Testing dengan Metode Equivalence Partitioning dan Boundary Value Analysis," *Jurnal Teknik Informatika (JUTIF)*, vol. 1, no. 1, pp. 25-29, 2020.

[13] S. A. Asri, I. G. M. A. Sunaya, P. M. Prihatini, and W. Setiawan, "Comparing traditional and agile software development approaches: case of personal extreme programming," *International Conference on Science and Technology (ICST)*, 2018. doi: 10.2991/icst-18.2018.116.

[14] S. Rahayu and R. S. Dewi, "Analisis Perbandingan Metode Waterfall dan Agile Dalam Pengembangan Sistem Informasi," *Jurnal Teknologi Informasi*, vol. 5, no. 2, pp. 115-122, 2021.

[15] Y. Dzhurov, I. Krasteva, and S. Ilieva, "Personal Extreme Programming – An Agile Process for Autonomous Developers," *Sofia University*, 2009.

[16] A. R. T. H. Ririh, N. Lutfiani, and A. F. I. S. Manik, "Penerapan Metode Personal Extreme Programming pada Pengembangan Sistem Informasi Akademik Sekolah," *Technomedia Journal*, vol. 6, no. 1, pp. 12-23, 2021.

[17] B. A. S. A. Ginting, "Implementasi E-Government Melalui Website Dinas Komunikasi Dan Informatika," *Jurnal Teknologi dan Open Source (JTOS)*, vol. 3, no. 2, pp. 202-214, 2020.

[18] M. I. Wibowo, A. H. Brata, and K. C. Brata, "Pengembangan Aplikasi Perangkat Bergerak Pengobatan Tuberculosis Berbasis Android Menggunakan Personal Extreme Programming (Studi Kasus: Puskesmas Polowijen)," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 3, no. 1, 2019.

[19] Y. A. P. S. E. Pradana, "Rancang Bangun Sistem Informasi Manajemen Tugas Akhir Menggunakan Metode Extreme Programming," *Jurnal Repositor*, vol. 2, no. 4, pp. 121-130, 2020.

[20] M. Cohn, User Stories Applied: For Agile Software Development. Boston: Addison-Wesley, 2004.

[21] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 8th ed. New York: McGraw-Hill Education, 2015.

[22] K. Beck and C. Andres, *Extreme Programming Explained: Embrace Change*, 2nd ed. Boston: Addison-Wesley, 2004.

[23] R. A. Sukamto and M. Shalahuddin, *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika, 2018.

[24] A. Juansyah, "Pembangunan Aplikasi Child Tracker Berbasis Assisted – Global Positioning System (A-GPS) Dengan Platform Android," *Jurnal Ilmiah Komputer dan Informatika (KOMPUTA)*, vol. 1, no. 1, pp. 1-8, 2015.

[25] S. Nidhra and J. Dondeti, "Black Box and White Box Testing Techniques - A Literature Review," *International Journal of Embedded Systems and Applications (IJESA)*, vol. 2, no. 2, pp. 29-50, 2012.