Jurnal E-Komtek
Vol. 10, No. 1. (2026) pp. 156-167
https://jurnal.politeknik-kebumen.ac.id/index.php/E-KOMTEK
p-ISSN : 2580-3719 e-ISSN : 2622-3066

Development of a Web-Based Graduation Document Completeness Information System

Annisa Gatri Zakinah[1], Cahya Vikasari[2], Nur Muniroh[3]

[1-3] Department of Informatics Engineering, Politeknik Negeri Cilacap, Indonesia, 53212

annisagatri@pnc.ac.id

https://doi.org/10.37339/e-komtek.v10i1.3189

Published by Politeknik Piksi Ganesha Indonesia

Artikel Info
Submitted:
21-05-2026
Revised:
11-06-2026
Accepted:
30-06-2026
Online first :
30-06-2026

Abstract

The manual graduation administration process causes data errors, incomplete documents, and legal risks such as diploma revocation. This study aims to develop a web-based graduation document information system to automate final project document verification and improve graduation administration accuracy and efficiency. The study used the waterfall method, including requirements analysis, database and interface design, implementation with PHP and MySQL, and black-box testing. The system features student data management, an 8-document checklist (including file upload), automation of eligibility status, a statistical dashboard, and three report levels. Testing on 47 scenarios achieved a 100% success rate. The web-based system effectively improves graduation data accuracy, eliminates human error, and supports digital transformation of academic services. Further development requires adding student roles and automatic notifications.

Keywords: *Graduation Information System, Graduation Administration, Web-Based, PHP MySQL*

Abstrak

Proses administrasi kelulusan yang dilakukan secara manual menyebabkan kesalahan data, ketidaklengkapan berkas, dan risiko hukum seperti pencabutan ijazah. Penelitian ini bertujuan mengembangkan sistem informasi kelengkapan dokumen yudisium berbasis web untuk mengotomatisasi verifikasi dokumen tugas akhir mahasiswa serta meningkatkan akurasi dan efisiensi administrasi kelulusan. Penelitian menggunakan metode *waterfall* yang meliputi analisis kebutuhan, perancangan basis data dan antarmuka, implementasi dengan PHP dan MySQL, serta pengujian *black-box*. Sistem memiliki fitur manajemen data mahasiswa, checklist 8 dokumen (termasuk *unggah file*), otomatisasi status *eligible* yudisium, *dashboard* statistik, dan tiga level laporan. Pengujian pada 47 skenario mencapai tingkat keberhasilan 100%. Sistem berbasis web ini efektif meningkatkan akurasi data kelulusan, meminimalisir kesalahan manusia, dan mendukung transformasi digital layanan akademik. Pengembangan lebih lanjut memerlukan penambahan *role* mahasiswa dan notifikasi otomatis.

Kata-kata kunci: *sistem informasi yudisium, administrasi kelulusan, berbasis web, PHP MySQL*

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

1. Introduction

In the university environment, the graduation administration process is a crucial stage that directly affects the final output of students and the institution's reputation. Several universities have been manually collecting final project (TA) documents. Students submit physical documents to the administrative office, where staff then check the completeness of each document one by one. Based on initial observations, several problems were identified: (1) frequent incomplete files; (2) instances where students with incomplete files were already reported to attend the yudisium (graduation council); (3) lengthy verification process; (4) no integrated system to ensure document authenticity. These issues pose significant risks, as they could lead to future legal problems, such as the revocation of diplomas or academic degrees **[1]**.

Several previous studies have developed yudisium information systems. Anggraeni **[2]** built a web-based final project information system using the waterfall method, but it lacks real-time automation of graduation status. Nurliana and Esabella **[3]** developed a yudisium registration application at Sumbawa University of Technology using the spiral method, but document completeness status is not updated immediately. Slam, Herikson, and Yandri **[4]** created a yudisium registration system at Raja Ali Haji Maritime University (UMRAH) which remains semi-manual (admin verification) and has not adopted eligible status automation, statistical dashboards, auto-save, or data export features. Ramadhan et al. **[5]** implemented an integrated yudisium grade processing system, but it does not provide a monitoring dashboard for department heads. Meanwhile, Fauzi and Kurniawan **[6]** developed an integrated academic system using PHP/MySQL, but it does not include automatic recording of document completeness dates as an audit basis.

The novelty of the system developed in this research is the automation of the eligible status, which is determined automatically by the system based on the completeness of 8 documents (7 checkboxes + 1 file upload), thereby minimizing human error. Additionally, the automatic recording feature of document completeness dates logs the first time a student achieves complete status, serving as an accountable audit trail. There is also a real-time monitoring dashboard that provides up-to-date graphs and statistical summaries (total students, number eligible, average grade), accessible to the Head of Study Program and Department Administrator with different roles. Every change to checkboxes and file uploads is recorded in the database along with timestamps, facilitating tracking of document completeness history.

This research aims to develop a web-based yudisium document completeness information system that automates document verification, improves data accuracy, and provides ease of access for the Head of Study Program and Department Administrator. This system is expected to be a solution to the ongoing graduation administration problems.

2. Method

This research uses the waterfall system development method, which consists of five stages: requirements analysis, design, implementation, testing, and maintenance [7] [8]. This method was chosen based on the relatively clear and stable system requirements from the outset [5].

2.1. Requirements Analysis

The requirements analysis was conducted through structured interviews with the Head of Study Program (Kaprodi) and direct observation of the yudisium administration workflow. The number of respondents involved in eliciting requirements was three end users, the Head of Study Program and two Department Administrator. The instrument used was an interview guide as listed in **Table 1**. The requirements validation technique involved document review by the Head of Study Program of the compiled list of functional requirements, ensuring alignment with applicable academic regulations.

Table 1. List of Agreed System Requirements

No	Question	Answer Notes
1	What documents must students submit to yudisium registration?	Final Project Report/Book, Final Project Journal, Final Project Poster, CD/Softcopy of Final Project, Library Clearance Letter, Final Project Equipment, Logbook, and Endorsement Sheet
2	Who are the staff involved in verification? How are tasks divided?	Students submit all required documents to the Head of Study Program, then checks and verifies the files. Students who meet the requirements will be registered for yudisium by the Department Administrator.
3	What are the most frequent obstacles in the current verification process?	Document verification takes a relatively long time and is error-prone. For example, there are students who have not submitted the Final Project Report but are already registered for

No	Question	Answer Notes
		graduation, and document recording is still manual, not automated.
4	What features do you expect in the yudisium information system?	A checklist feature for all documents except the Endorsement Sheet which requires a file upload, displaying the list of students who are eligible and non-eligible for yudisium, a document completeness date feature that automatically appears when all documents are complete.
5	How many access roles are there? Explain the access rights for each.	Yes, there are 2 access roles: Head of Study Program has full access to all features and Department Administrator can access the list of eligible and non-eligible students.

The identified functional requirements include the system can manage student data, provides a checklist for 8 types of yudisium documents, supports file upload (endorsement sheet), automates the eligible status (8/8 documents), records the completion date when a student becomes eligible, provides a statistical dashboard, provides eligible/non-eligible/all reports, supports data export to CSV, has two access roles (Head of Study Program and Department Administrator); and has a login and session management mechanism. Non-functional requirements include data security (password encryption), response time of less than 3 seconds, and a user-friendly interface.

2.2. System Design

Before building the system, the author first designed the database and user interface. Figure 1 shows the database design, which resulted in three main tables:

(1) *tabel_users* : stores user data (id, username, password, role),

(2) *tabel_mahasiswa* : stores student data (student_id, name, cohort year, grade, supervisor1, supervisor2, thesis title), and

(3) *tabel_dokumen* : stores document completion status per student (student_id, document1 to document7 (boolean), document8_file (varchar), completion_date).

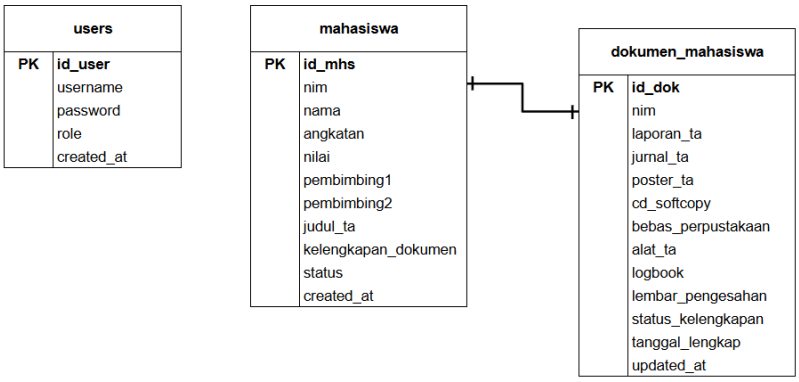

Figure 1. Entity Relationship Diagram (ERD)

The relationship between the tables is that each student has exactly one document record (one-to-one relationship). The interface design (mockup) resulted in 6 main pages: login, dashboard, student data, document checklist, eligible yudisium, and logout. The design prioritizes ease of navigation with a sidebar menu and responsive tables.

2.3. Implementation

The system implementation uses a backend (PHP with a simple MVC architecture), database (MySQL), frontend (HTML5, CSS3 (Bootstrap 5), JavaScript), and local server (XAMPP (Apache + MySQL)). PHP and MySQL were chosen for their flexibility and ease of integration [5][6]. The key features implemented are:

(1) Auto-save document checklist (without page reload) using AJAX,

(2) File upload with extension validation (PDF, JPG, PNG) and size limit (max. 2 MB),

(3) Eligible status automation, where the system changes a student's status to "eligible" if all seven checkboxes are ticked and the endorsement sheet file has been uploaded,

(4) Automatic completion date recording when the status changes from "incomplete" to "complete",

(5) Automatic logout idle timer after 30 minutes of inactivity.

2.4. Testing

Testing was conducted using the black-box method with 47 test scenarios covering all functional features **[9] [10]**. The testing involved end users, namely the Head of Study Program and 2 Department Administrator as respondents. The testing scope covered each module, as follows:

(1) Login → Authentication for Head of Study Program and Department Administrator, password validation, toggle password visibility

(2) Dashboard → Display of statistics, graphs, and data according to role

(3) Student Data → CRUD, import/export CSV, bulk delete, template download

(4) Document Checklist → Auto-save checkboxes, upload/replace/view endorsement sheet file, automatic eligible status

(5) Eligible Yudisium → Tab display, progress bar, CSV export per tab

(6) Security → Idle auto-logout, role-based menu

Every bug found was recorded and fixed iteratively until all scenarios passed. **Table 2** presents an example test scenario for the Login module, containing the scenario type, test steps, test data, expected result, and actual result/status. The success indicator is stated as "Passed" if the actual output matches the expected result.

Table 2. Black-Box Test Scenarios for the Login Module

Scenario	Steps	Test Data	Expected Result	Status (Pass/Fail)	Remarks
User login with correct credentials	1. Open login page 2. Enter username and password 3. Click LOGIN	username: **kaprodi** password: **password**	Redirect to dashboard page, display full menu (Dashboard, Student Data, Checklist, Eligible)		
Login with incorrect username or password	1. Open login page 2. Enter username and password 3. Click LOGIN	username: **wrong** password: **password**	Show alert/notification "Username not found!"		
Toggle password visibility	1. On the login page, fill in the password 2. Click the eye icon	-	Password text becomes visible; icon changes to eye with slash		

Note: The "Status" and "Remarks" columns are left empty as they would be filled during actual testing.

3. Results and Discussion

3.1. System Implementation Results

The yudisium document completeness information system has been successfully developed under the name "Sistem Informasi Yudisium (SIYUDIS). The login page verifies user credentials and directs users according to their role. If the username and/or password are incorrect, the user cannot access the system. There is also a toggle password feature to view or recheck the entered password. The dashboard page displays a summary of total students, number of eligible students, number of incomplete students, average grade, as well as a pie chart and bar chart. The student data page supports CRUD (Create, Read, Update, Delete), CSV import/export, and search functionality. This page can only be accessed by the Head of Study Program to manage student data for those submitting yudisium documents. The Head of Study Program can add student data either directly through the system or by importing data in CSV format. The Head of Study Program can also export all student data, which will be downloaded in CSV format. Student data deletion can be done in two ways: by editing each student and clicking the delete icon, or by bulk deletion by checking the checkboxes of the students to be deleted and then clicking the Delete button. The displays of these three pages are shown in **Figure 2.**

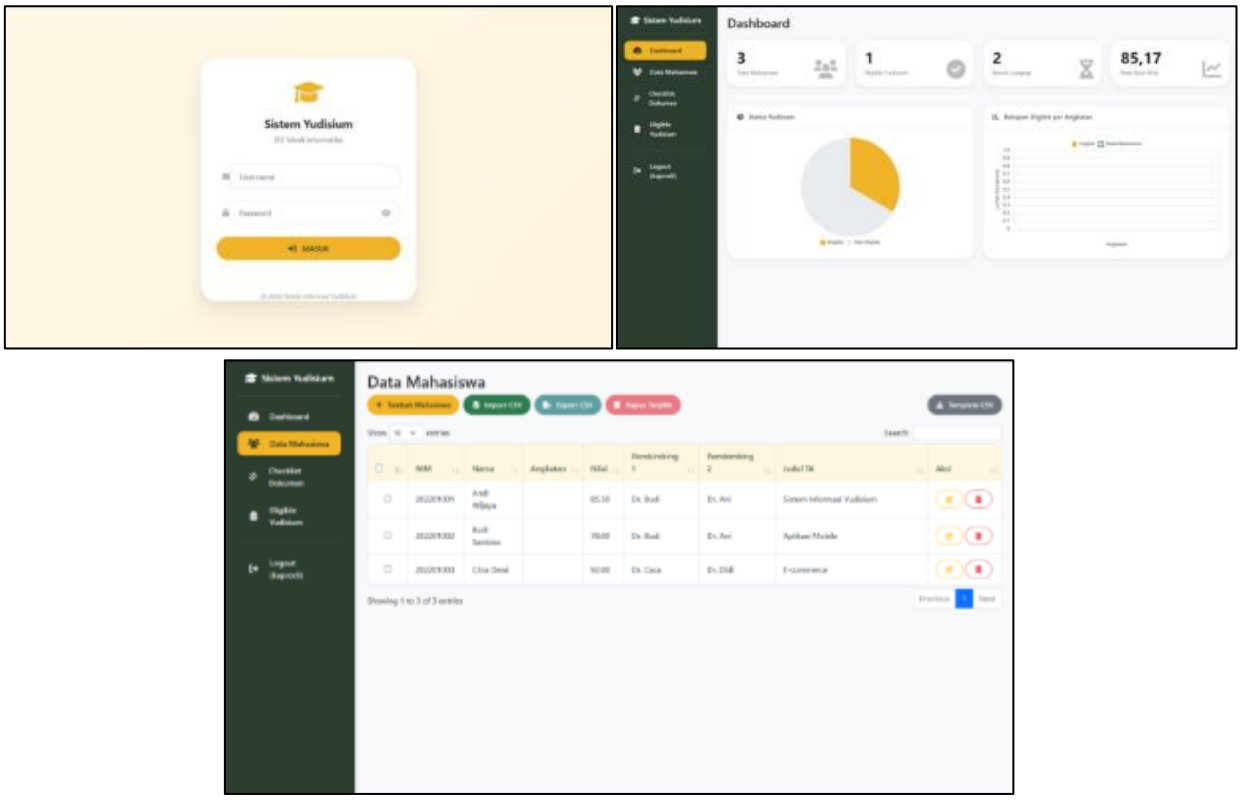

Figure 2. Display of the Login, Dashboard, and Student Data Pages

The document checklist page displays 8 document columns consisting of 7 auto-save checkboxes (AJAX) and 1 upload button for the endorsement sheet. The auto-save feature works with every change to the checklist columns for documents 1 to 7, whether checked or unchecked. The auto-save feature on the checklist columns makes the document verification process easier and faster without having to press a Save button. Additionally, there is an automation feature in the form of a completion date, where the document completion date is automatically filled when the 7 documents are checked and the 1 document (the Endorsement Sheet) has been uploaded. The date listed corresponds to the date when all 8 documents are complete. This reduces the possibility of human error in recording the yudisium date for each student. The display of the checklist page can be seen in **Figure 3.**

Figure 3. Display of the Document Checklist Page

The eligible yudisium page provides three tabs (Eligible, Non-Eligible, All) with a completion progress bar, completion date column, date filter, and CSV export button. The Head of Study Program and Department Administrator can access all features on this page. Each tab displays student data according to the tab name, along with a CSV data export feature. In the Eligible and All tabs, there is a filter and completion date feature. This feature functions to view student data based on when the documents were completed within a specified date range. The completion date will appear when all 8 documents have been fulfilled on the Document Checklist page. This page also has an automation feature where, when there is an update to eligible student data, it will automatically appear in the appropriate tab. The display of the eligible yudisium page can be seen in **Figure 4.**

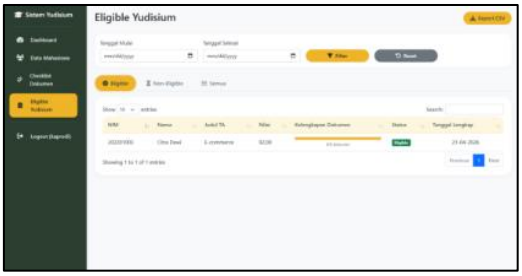

Figure 4. Display of the Eligible Yudisium Page

3.2. Functional Testing Results

Of the 47 test scenarios executed, 25 scenarios were tested by the Head of Study Program, 3 scenarios were tested by the Department Administrator, and 19 scenarios were tested by both (Head of Study Program and Department Administrator). All scenarios were declared 100% passed, the calculation of which can be seen in **Table 3**.

Table 3. Summary of Test Results

Role	Total Test Scenarios	Passed	Failed	Pass Percentage
Head of Study Program	25	25	0	100.00%
Department Administrator	3	3	0	100.00%
General (Both)	19	19	0	100.00%
Overall Total	47	47	0	100.00%
Conclusion				**feasible**

These results align with previous research findings showing that the black-box testing method is effective for identifying functional errors in academic information systems **[9]** **[10]**. Detailed test results per module can be seen in **Table 4.**

Table 4. Summary of Test Results per Module

Module	Number of Scenarios	Passed	Remarks
Login	5	5	Includes idle timer
Dashboard	2	2	Data synchronized
Student Data	9	9	Import/export works
Document Checklist	14	14	Response < 1 second, extension & size validation
Eligible Yudisium	13	13	Status changes correctly, file format correct
Security	4	4	Department Administrator role cannot access student data and document checklist modules
Total	**47**	**47**	**100%**

Bugs found during testing and fixed:

1. array_map error on mass delete feature (PHP 8.x compatibility)
2. Inconsistent file extension validation
3. Session not readable on certain pages due to path error

3.2. Discussion

The system successfully minimizes incidents where students are declared eligible even though their documents are incomplete, because the eligible status is purely determined by the system, calculated automatically based on the fulfillment of the 7-document checklist and 1 uploaded file (Endorsement Sheet). A comparison of conditions before and after system implementation can be seen in **Table 5.**

Table 5. Comparison of Conditions Before and After System Implementation

Aspect	Before (Manual)	After (System)
Risk of human error (fictive completeness)	High	None
Speed of eligible data summary	2-3 days	Real-time
History documentation of completeness	None	Stored (completion date)
Documentation of completeness history	None	Saved (completion date)
Access by Department	Must request data from Study Program	Direct (limited role)
Ease of reporting (export)	Manual retyping	One-click CSV

This research aligns with findings from previous studies on the digitization of academic administration **[2] [11] [12]**. The implementation of auto-save using AJAX has been shown to improve user experience because users do not need to repeatedly press a save button **[1]**.

This system implements eligible status automation, auto-save checklists, automatic date recording, and a real-time dashboard. This makes it possible to minimize human error in determining graduation status by up to 100%, as there is no longer any subjective manual intervention. As stated by Lestari and Wijaya **[1]**, automating graduation rules eliminates interpretation ambiguity and ensures policy consistency.

The implementation of auto-save with AJAX reduces the number of clicks from 8 clicks (per student) to 0 clicks for the saving process, while also preventing data loss due to forgetting to save. Previous research by Nurliana and Esabella **[3]** still used a separate save button, which posed a risk of incomplete data entry. The completion date feature automatically records the first time a student reaches complete status, which is not found in the research by Anggraeni **[2]**, Fauzi

and Kurniawan [6], or Slam, Herikson, and Yandri **[4]**. The presence of the completion date increases the transparency and accountability of the yudisium process, especially for academic audit purposes.

The statistical dashboard, with graphs and figures, allows the Head of Study Program and Department Administrator to make data-driven decisions without waiting for periodic reports. Fauzi and Kurniawan **[6]** show that real-time access to performance indicators accelerates problem detection (e.g., accumulation of students with incomplete documents) and aids yudisium planning.

Compared to typical graduation administration systems, the advantages of this system lie in full automation of eligible status without manual intervention, automatic recording of the completion date for graduation audits, role-based access restrictions (Head of Study Program vs. Department Administrator), and an idle timer for data security. However, this system still requires further development, particularly in terms of data security (HTTPS connection encryption if deployed to a public server) and resilience against attacks.

4. Conclusion

Based on the research results, this system was successfully developed with advantages including student data management features, an 8-document checklist with auto-save and file upload, eligible status automation, completion date recording, a real-time statistical dashboard, and reports exportable to CSV. Black-box testing on 47 scenarios achieved a 100% success rate.

This research contributes through the integration of four advanced features not commonly found in similar systems: eligible status automation without manual intervention which eliminates human error, automatic completion date recording as a graduation audit trail, auto-save checklist using AJAX which reduces repetitive actions, and a real-time dashboard with role-based access (Head of Study Program and Department Administrator). These results reinforce the evidence that digitization of graduation administration can significantly improve the accuracy, efficiency, and accountability of yudisium services.

Recommendations for future development include adding a student role so they can independently monitor their document status, integration with the main academic system, implementation of HTTPS encryption and scheduled backups for data security, and the

development of automatic notifications (email/WhatsApp) to remind students whose documents are incomplete.

References

[1] Lestari, P., & Wijaya, H. (2021). Otomatisasi verifikasi dokumen kelulusan berbasis web. Jurnal Sistem Informasi, 9(3), 201-210.

[2] Anggraeni, D. Y. (2023). Sistem informasi tugas akhir mahasiswa S1 Universitas Ahmad Dahlan berbasis web menggunakan metode waterfall [Skripsi]. Universitas Ahmad Dahlan.

[3] N. Nurliana and S. Esabella, "Rancang Bangun Aplikasi Pendaftaran Yudisium pada Fakultas Teknik Universitas Teknologi Sumbawa Berbasis Web," Skripsi, Universitas Teknologi Sumbawa, Sumbawa, Indonesia, 2020.

[4] B. E. Slam, R. Herikson, and S. Yandri, "Development of Academic Administration and Student Services System at the Faculty of Engineering and Maritime Technology, Raja Ali Haji Maritime University," J. Sist. Inf. dan Inform., vol. 8, no. 1, pp. 77–88, 2025.

[5] Ramadhan, A., Zulkifli, Z., Kurniawan, R., & Widianto, S. (2021). Pengembangan sistem informasi akademik untuk bagian keuangan dan bagian pengolahan nilai yudisium. JOISIE: Journal of Information System and Informatics Engineering, 5(2), 40-48.

[6] Fauzi, A., & Kurniawan, D. (2023). Implementasi PHP dan MySQL pada sistem informasi akademik terpadu. Jurnal Ilmiah Komputer, 12(1), 45-54.

[7] Kurniyanti, V. A., & Murdiani, D. (2022). Perbandingan model waterfall dengan prototype pada pengembangan sistem informasi berbasis website. Jurnal Syntax Fusion, 2(8), 511-520.

[8] Nugroho, S. (2024). Waterfall model untuk pengembangan aplikasi yudisium. Jurnal Rekayasa Perangkat Lunak, 7(1), 33-41.

[9] Putri, A. (2023). Pengujian black box website Kominfotik menggunakan teknik equivalence partitioning di Sudin Kominfotik Jakarta Barat [Skripsi]. Universitas Pembangunan Jaya.

[10] Saputra, R., & Hidayat, T. (2022). Pengujian fungsional sistem informasi menggunakan metode black box. Jurnal Teknologi Informasi, 8(2), 112-120.

[11] Anonim. (2021a). Aplikasi pemeriksaan persyaratan yudisium pada program studi teknik informatika Universitas Kristen Petra. Surabaya: Fakultas Teknologi Industri Universitas Kristen Petra.

[12] Anonim. (2021b). Sistem informasi pendaftaran yudisium di Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Gadjah Mada. Yogyakarta: UGM Press.