International Journal of Emerging Information Science and Technology
Vol. 6, No. 1, (2025), pp. 1-15

Web-Based E-Office System Design Using Yii2 Framework

Hery Siswanto[1,*], Siti Sofiatun[1]

[1]*Universitas Muhammadiyah PKU Surakarta, Jl. Tulang Bawang Sel. No.26, Kadipiro, Kec. Banjarsari, Kota Surakarta, Jawa Tengah 57136*

E-mail: hery@umpku.ac.id, shof_fiiya@umpku.ac.id

Abstract

Universitas Muhammadiyah PKU Surakarta in its efforts to improve the efficiency and effectiveness of office administration management faces challenges in document management and workflows that are still running conventionally. Manual processes often take longer, are prone to errors, and require large physical storage space. To overcome these problems, this study aims to design and build a web-based E-Office system by utilizing the Yii2 framework. The method used in this study is SDLC and the system design used includes the needs analysis stage, system design using the Unified Modeling Language (UML) consisting of use case diagrams, activity diagrams, and class diagrams, and system implementation using the Yii2 framework with the PHP programming language and MySQL database. Implementation using Devops. The Yii2 framework was chosen because of its robust structure, complete security features, SSO login with Google, and its ability to accelerate the application development process. The E-Office system developed will include various essential features such as incoming and outgoing mail management, electronic mail disposition, SPPS management, and document status tracking features. It is expected that with the implementation of this system, Muhammadiyah University of PKU Surakarta can realize a more modern, paperless, transparent, accountable office administration system, and is able to increase productivity and quality of service for all academicians. This system is also expected to facilitate the decision-making process of leaders through fast and accurate access to information.

Keywords E-Office, Yii2 Framework, Administrative Governance, Office Information System.

1. Introduction

The rapid development of information and communication technology has brought significant changes in various aspects of life, including in the higher education sector. Muhammadiyah University of PKU Surakarta, as an educational institution that continues to strive to improve the quality of its services and governance, faces the demands to adapt to this digital era. Efficient and effective office administration processes are one of the main keys to supporting smooth operations and decision-making in the university environment. However, administrative management that still relies on conventional methods is often faced with various obstacles, such as slow bureaucratic flows, the potential for loss or damage to physical documents, difficulties in tracking letters and dispositions, and the high use of paper that is less environmentally friendly and costly.

ImplementationElectronic office system or e-office becomes a strategic solution. E-office offers a transformation of administrative processes from manual to digital, which allows document management, workflow, and internal communication to be faster, transparent, accountable, and well-documented. By utilizing information technology, various office activities such as managing incoming and outgoing mail,

Copyright©2025 EIST

disposition, to document archiving can be done electronically, thus minimizing the obstacles that exist in conventional systems.. Therefore, the design of a comprehensive e-office system that is tailored to the specific needs of Muhammadiyah University of PKU Surakarta, using the Yii2 framework which is known to be reliable and flexible, is expected to be an important step towards modernizing superior and competitive administrative governance.

2. Method

The research method used in designing the e-Office system at Muhammadiyah University PKU Surakarta describes the systematic stages starting from the research approach and system design, to the evaluation of research results. The method used in this study is the SDLC (Software Development Life Cycle) Method.[1], [2]. SDLC method is a framework or structured process used in software development, which aims to produce high-quality software that meets or even exceeds customer expectations. SDLC includes a series of clear and sequential stages, starting from planning where project needs, scope, and resources are defined; continued with in-depth user needs analysis; then design of system architecture and interface; followed by the implementation stage or actual coding of the software; then the testing stage to ensure the system runs according to specifications and is error-free; and ends with the maintenance stage where the system is updated, repaired, and improved over time to ensure its functionality remains optimal. Each of these stages produces certain outputs that become input for the next stage, ensuring a systematic and controlled approach to software development. This study will apply Devops to SDLC.[3], [4]. Applying DevOps to the SDLC means integrating DevOps philosophy, practices, and tools into every stage of the software development life cycle. The goal is to accelerate the delivery of high-quality software by increasing collaboration between development (Dev) and operations (Ops) teams,[5]

Stages Study

In general, the stages of research carried out include:

1. **Preliminary Study:**Conducting literature studies related to e-Office systems, Yii2 framework technology, and similar research. Initial observations were also conducted to understand the general description of the administration system at Muhammadiyah University PKU Surakarta.
2. **Data collection:**Collect primary and secondary data relevant to system design needs.
3. **System Requirements Analysis:**Analyze the collected data to identify the functional and non-functional requirements of the e-Office system.
4. **System Design:**Designing the architecture, database, and user interface of an e-Office system using UML.
5. **System Implementation:**Developing an e-Office system using the Yii2 framework, PHP programming language, and MySQL database with DevOpas
6. **System Testing:**Conduct testing to ensure the system is running according to design and requirements.
7. **Evaluation of Results:**Assess the effectiveness and suitability of the system that has been built.

Method of collecting data

To obtain accurate and comprehensive data, several data collection techniques are used, namely:

1. **Observation:**Conducting direct observation of the current office administration process at Muhammadiyah University PKU Surakarta, including the flow of correspondence, disposition. This observation aims to understand the actual workflow and identify potential problems.
2. **Interview:**Conducting semi-structured interviews with related parties at Muhammadiyah University PKU Surakarta, such as administrative staff, administration department, lecturers, and leaders involved in the office process. The interviews aim to explore the needs, expectations, and obstacles faced related to the administration system.
3. **Document Study:**Analyze documents related to administrative governance at Muhammadiyah University PKU Surakarta, such as sample letter formats, existing manual disposition flows, Standard Operating Procedures (SOP) if available, and other relevant documents to understand the applicable rules and formats.
4. **Literature review:**Reviewing scientific literature, journals, articles, and technical documentation related to e-Office design, use of the Yii2 framework, and best practices in developing office information systems.

System Design Using UML (Unified Modeling Language)

The design of this e-Office system will be visualized using the Unified Modeling Language (UML) to model the system comprehensively from various perspectives. The UML diagrams that will be used include: Use Case Diagram: Describes the functionality of the system from the user's (actor's) perspective. This diagram will show the interaction between actors (eg, admin, staff, leaders) with the main features of the e-Office system such as login, incoming mail management, outgoing mail management, disposition. Activity Diagram: Models the workflow of various business processes in the e-Office system. For example, the process flow from receiving incoming mail to disposition, or the flow of creating and sending outgoing mail. This diagram helps understand the sequence of activities and decisions in the system[6]. Class Diagram: Describes the static structure of the system by showing object classes, their attributes, methods, and relationships between classes (e.g., association, aggregation, generalization). This diagram is the basis for database design and program code structure. Sequence Diagram (if needed): Describes the interaction between objects in a certain time sequence for a specific scenario. This diagram can be used to detail how objects collaborate to run a use case.[7], [8], [9].

How to Process Data and System Development.

System Development with Yii2 Framework: The system implementation will be done using the Model-View-Controller (MVC) approach which is the basic architecture of the Yii2 framework. Model: Represents data and business logic. This will include interaction with the MySQL database for CRUD (Create, Read, Update, Delete) operations on entities such as letters, users, dispositions and other features. View: Responsible for displaying data to the user and capturing input from the user. This will be built using HTML, CSS, and JavaScript, with the help of Yii2's template engine. Controller: Acts as an intermediary between the Model and the View, managing user requests, calling business logic in the Model, and selecting the right View to display. The programming language used is PHP, with a MySQL database.[10], [11], [12].

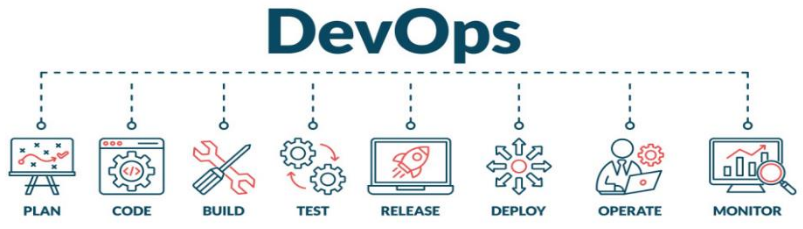


Figure1DevOps Model

Implementing DevOps as a system developer is very possible and very beneficial. The main focus is on automating the process (build, test, deploy) and using the right tools to support the creation of information systems. How to adopt a mindset and practices that will help in system development to become a much more efficient and effective developer. For example, automating testing or setting up a simple CI/CD pipeline.[5]

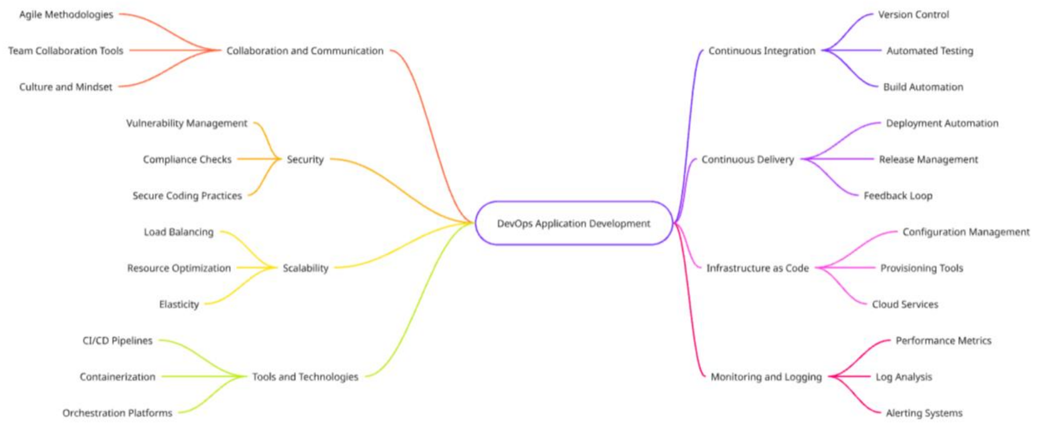


Figure 2. DevOps Application Development

DevOps is a trend in software engineering that aims to bridge the gap between development and operations teams through a set of principles, practices, and cultural changes that emphasize collaboration and automation. Although there is no universally agreed definition, DevOps is generally understood as an effort to improve the speed of delivery of software products or services by applying Agile and Lean principles to operational teams as well. Core practices in DevOps include various Continuous Software Engineering (CSE) activities such as Continuous Integration (CI), Continuous Delivery (CDE), and Continuous Deployment (CD), as well as strong cultural and human resource aspects such as responsibility and knowledge sharing. [13], [14].

In developing eOffice with a DevOps approach, each component has an important role to ensure the process runs smoothly, quickly and reliably, without having to discuss the technical details of its installation:

1. Ubuntu Server 24.04: This is the operating system foundation on which all eOffice applications and supporting services will run. Think of it as the

land you build your house on. Ubuntu was chosen for its stability and security.

2. Nginx Webserver: Acts as the main gateway that receives requests from users (for example, when opening eOffice in a browser) and delivers the eOffice application view to them. Nginx is known for its speed in handling many requests at once.
3. MySQL Database: This is the data vault for all important eOffice information, such as mail, user data, dispositions, etc. MySQL ensures that the data is stored securely and in a structured manner.
4. Git Version Control: Think of Git as a time machine and notebook for your eOffice code. Every time there is a change in the code, Git records it. This allows teams of developers to work together, track change history, and revert to previous versions if something goes wrong.
5. GitHub: This is an online collaboration platform that uses Git. GitHub is the central repository for eOffice code. Teams can share code, review each other's work, and manage eOffice development projects in a more organized manner. GitHub is also often used to automate the testing and release process of applications.

In simple terms, with DevOps, the eOffice development team uses Ubuntu as the foundation, Nginx to serve users, MySQL to store data, Git to manage code changes, and GitHub to collaborate and automate workflows. All of this aims to make eOffice development faster, more efficient, and produce high-quality applications.

How to Evaluate/Assess Research Results

Evaluation of research results will be conducted through several testing methods to assess the functionality, usability, and suitability of the system to user needs, Functional Testing (Black Box Testing): This testing is conducted to ensure that each function of the e-Office system runs according to the specified requirements specifications. Testing is conducted by providing certain inputs and examining the output produced without considering the internal structure of the code. Testing scenarios will be created based on the use cases that have been designed.[15]

3. Results

This section presents the results of the design and initial implementation of the e-Office system of Muhammadiyah University of PKU Surakarta based on the methodology described previously. The discussion will focus on the main features of the system, the architecture used, and the user interface that has been developed.

General Description of E-Office System

Based on the needs analysis and design process that has been carried out, a prototype of a web-based E-Office system for Muhammadiyah University PKU Surakarta has been successfully designed and implemented. This system is named "SIPEKU E-Office" (Electronic Office Information System of Muhammadiyah University PKU Surakarta) and was developed using the Yii2 framework with the PHP programming language and MySQL database management system. The main objective of this system is to digitize the office administration process, especially in the management of correspondence, disposition, so as to improve efficiency, transparency, and accountability.

System Architecture

Model-View-Controller (MVC) is described as a well-known and used design pattern for interactive software system architecture. The way the MVC method works is by separating the main components of the system such as data manipulation (Model), display or user interface (View), and process logic (Controller), which aims to make the development more neat, structured, and easy to develop further. In this research, MVC architecture is implemented to build a web-based freelancer project monitoring system using the Laravel and Slim PHP frameworks. Laravel is a full MVC framework, while Slim is a micro framework that basically only has a View component, but the MVC method can be added by including the Controller and Model, and in this research both frameworks are implemented using the MVC method.[16], [17].

Main Features Design Results of E-Office System

The Unified Modeling Language (UML) is leveraged as the basis for automated visualization of software ecosystem architecture, with a particular focus on component and package diagrams. The proposed approach uses UML to model the relationships between components, stack dependencies, and hierarchical structures in software systems, whose data is extracted from release notes. UML was chosen because it is an industry standard, supports dynamic diagramming from textual descriptions through APIs (particularly with PlantUML), and is extensible for specific visualization needs.[18], [19], [20]. Use Case Diagram of eOffice System includes Administrative Admin (TU): Administrative staff responsible for managing correspondence and documents in general. Leader: University structural officials who have the authority to provide disposition, approval, and monitoring (for example: Rector, Vice Rector, Dean, Vice Dean, Head of Study Program). Lecturers & Staff: General users of the system (Lecturers, Education Personnel/non-TU Staff) who create, receive, and follow up on letters/documents.

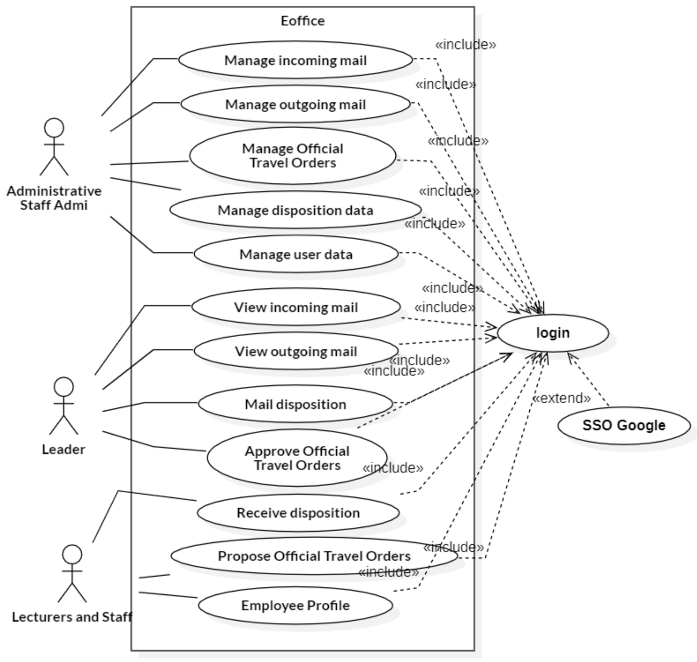


Figure 2. E-Office Use Case

In designing the university eOffice, activity diagrams are used to visualize the dynamic workflow of various electronic office administration and correspondence management processes. The activity diagram for the user login process will start when the user accesses the login page, then the system will display two options:

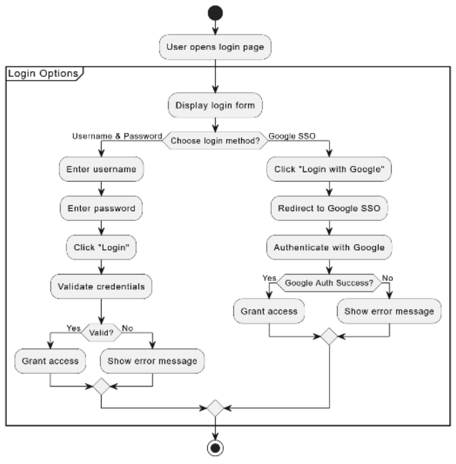
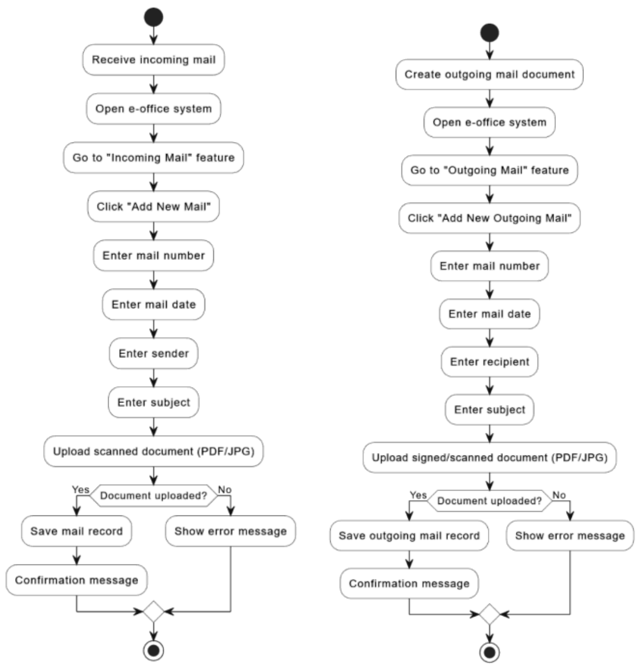


Figure 4. User login activity diagram

Login using username and password or login via Google Single Sign-On (SSO). If the user chooses username and password, the next activity is the user enters the credentials, followed by validation by the system; if valid, the user successfully logs in, otherwise an error message will appear and return to the login option. If the user chooses Google SSO, the system will redirect the user to the Google authentication page; after successful authentication on the Google side, the system will receive a verification token, validate it, and if successful, the user is immediately redirected to the application dashboard as a sign of successful login, or return to the login option if Google authentication fails or is canceled. The activity diagram for the incoming mail input process in the eOffice system will illustrate the workflow that begins when the administrative staff (Administration) receives mail, either physical or digital.



\

Figure 3. Activity Diagram of Incoming and Outgoing Mail

This diagram illustrates the sequential steps of an activity, such as the process of creating an incoming letter, disposition by management, follow-up by staff, and digital archiving, including decision points (e.g., whether a letter requires further approval) and parallel flows, if any. By mapping the interactions between users (such as TU staff, lecturers, deans, or rectors) and the eOffice system, the activity diagram helps developers and stakeholders clearly understand the workflow, identify potential inefficiencies, and ensure that all the functionality needed for digital office operations in a university environment is included in the system design. The activity diagram of the disposition process in a university eOffice illustrates the workflow that begins when a leader (e.g., Rector, Dean, or Unit Head) receives a notification or opens a new incoming letter that requires follow-up in the system. The activity continues with the leader studying the contents of the letter and its attachments, which then leads to a decision point.

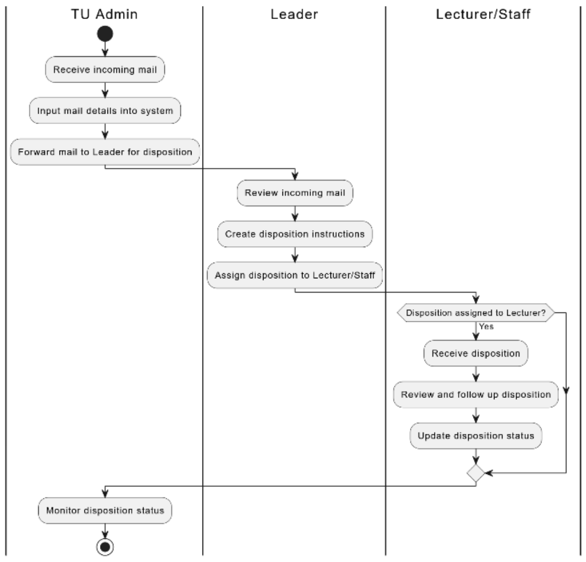

TU Admin
Leader
Lecturer/Staff
Receive incoming mail
Input mail details into system
Forward mail to Leader for disposition
Review incoming mail
Create disposition instructions
Assign disposition to Lecturer/Staff
Disposition assigned to Lecturer?
Yes
Receive disposition
Review and follow up disposition
Update disposition status
Monitor disposition status

Figure 4. Activity Diagram Disposition of Letters

The sequence diagram for the flow of incoming mail to the disposition process in the eOffice system will visualize the sequence of message interactions and data exchange between lifelines (objects or actors) such as Administrative Staff, the eOffice System itself, and Leaders (for example, the Rector or Dean) over time.

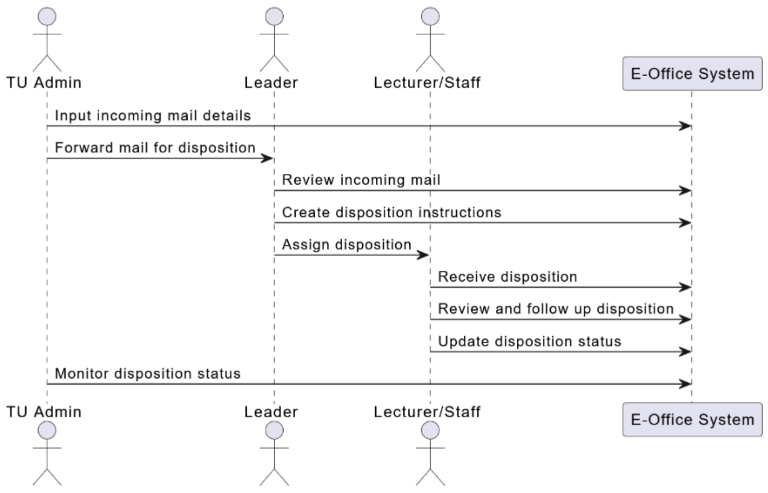

TU Admin
Leader
Lecturer/Staff
E-Office System
Input incoming mail details
Forward mail for disposition
Review incoming mail
Create disposition instructions
Assign disposition
Receive disposition
Review and follow up disposition
Update disposition status
Monitor disposition status

Figure 5. Sequence diagram of incoming mail up to disposition

This diagram will begin with the Administrative Staff sending a message to the eOffice System to input the details of the incoming letter and its attachments; the system will then process, store the letter data, and send a notification of a new letter to the Manager. The Manager will then interact with the eOffice System to view the details of the letter, then send a message with disposition instructions (such as disposition purpose, notes, and deadline) back to the eOffice System, which will then record this disposition and forward the notification with the letter details and instructions to the designated Disposition Recipient staff or unit, clearly showing how the flow of control and information moves chronologically between the entities involved. Sequence diagramfor the outgoing mail process in the eOffice system, it will visualize the chronological sequence of message interactions between various lifelines such as users (conceptualizing staff).

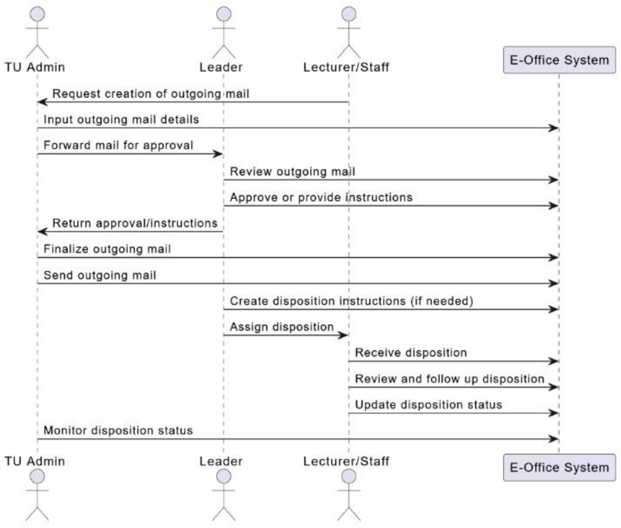

TU Admin
Leader
Lecturer/Staff
E-Office System
Request creation of outgoing mail
Input outgoing mail details
Forward mail for approval
Review outgoing mail
Approve or provide instructions
Return approval/instructions
Finalize outgoing mail
Send outgoing mail
Create disposition instructions (if needed)
Assign disposition
Receive disposition
Review and follow up disposition
Update disposition status
Monitor disposition status

Figure 6. Outgoing mail flow diagram

The eOffice System itself, the Manager (as the approver), and the Administrative Staff (for finalization and sending). This diagram generally begins when a user drafts an outgoing letter and sends it to the eOffice System for approval; the system then forwards the notification and the draft to the Manager. The sequence diagram for the process of submitting a Travel Order Letter (SPPD) in the eOffice system will describe the sequence of message interactions between key lifelines, such as employees submitting a Travel Order Letter (SPPD).

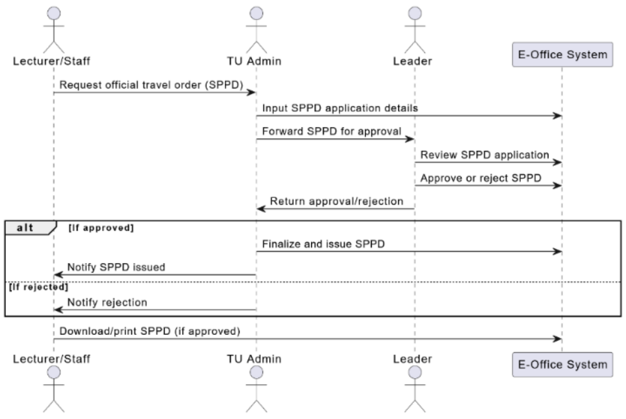


Figure 7. Sequence diagram of SPPD submission

The eOffice system itself, and the Leader or Superior who is authorized to give approval. This diagram begins when the Employee sends SPPD application data—including travel purpose, date, activity details, cost estimates, and supporting documents—to the eOffice System. The system will then process the request, perhaps perform initial validation, save it, and automatically forward a notification along with application details to the Direct Superior for review. The Superior will then interact with the system to view the SPPD application and send a decision message (e.g., approved, requires revision, or rejected); the eOffice System will then record this decision, update the SPPD status, and send a notification of the approval results back to the requesting Employee, and if approved, can forward the information to other units such as finance or HR for follow-up, all visualized as a chronologically structured message flow.

Database Design Results

Database design produces a normalized table structure to store data efficiently and consistently. Some of the main tables designed include: users (for user data), roles (for user roles), incoming_letter (for incoming mail data), outgoing_letter (for outgoing mail data), disposition (for disposition data), attachment_letter (for attachment files). Relationships between tables have been defined to maintain data integrity, for example the relationship between the incoming_letter table and the disposition table.

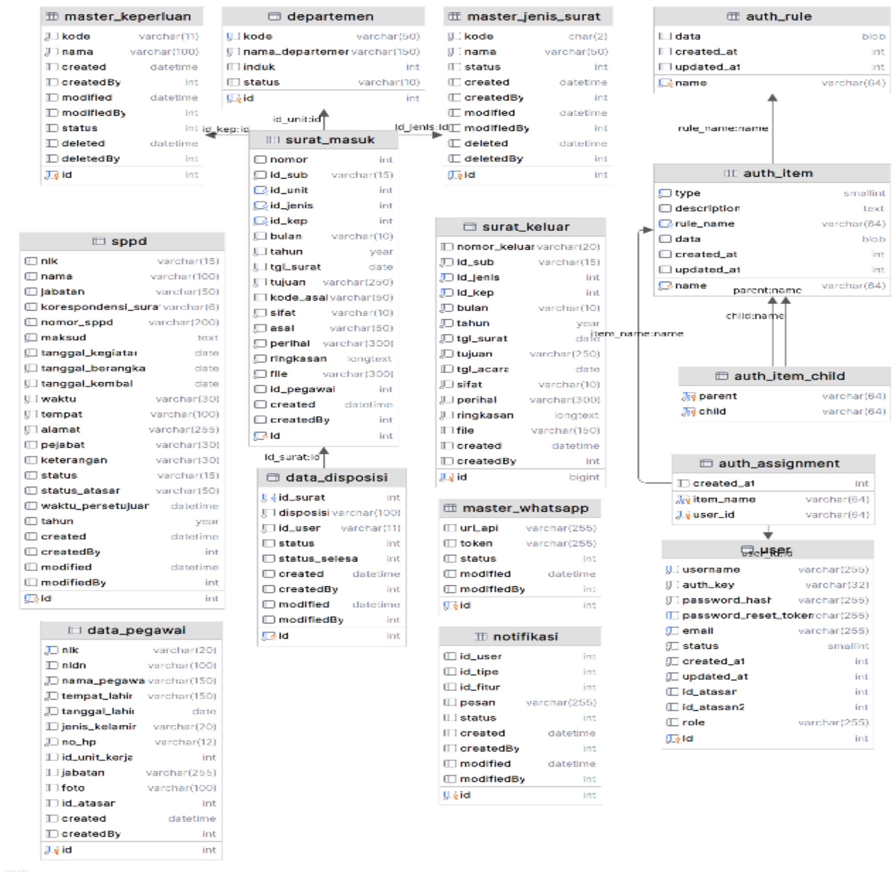
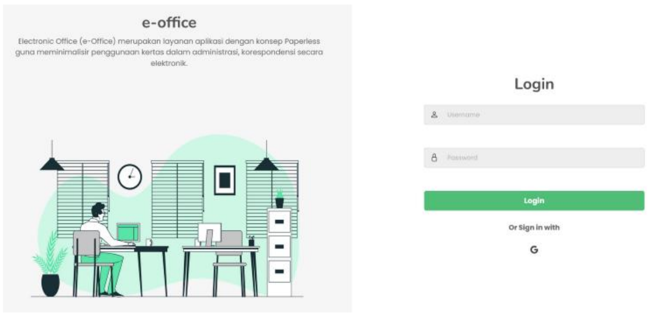


Figure8. E-office ERD system

User Interface (UI) Design Results

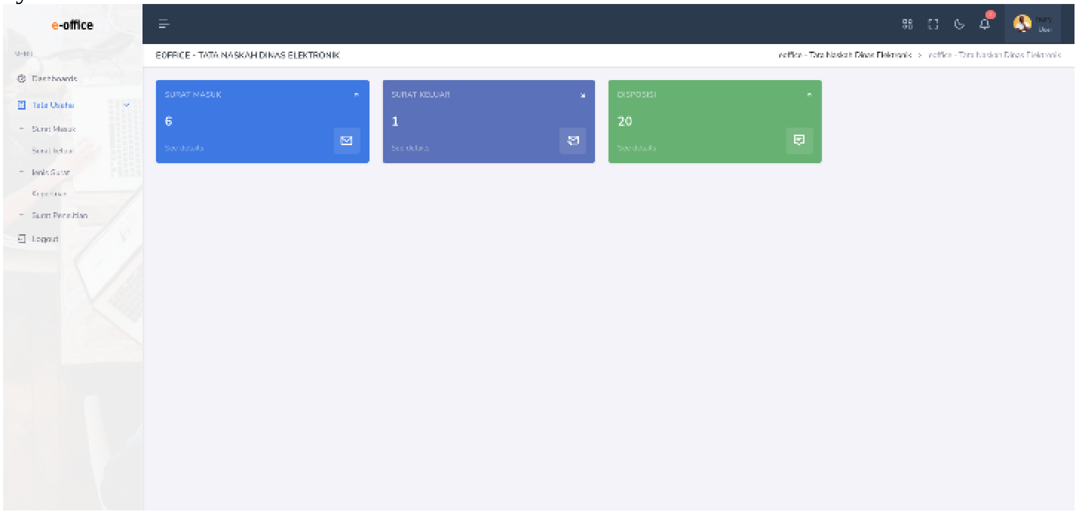
The E-Office system user interface is designed with user-friendliness, intuitiveness, and responsiveness in mind so that it can be accessed through various devices. Some design principles applied:


e-office

Electronic Office (e-Office) merupakan layanan aplikasi dengan konsep Paperless guna meminimalisir penggunaan kertas dalam administrasi, korespondensi secara elektronik.

Figure 9. User Login

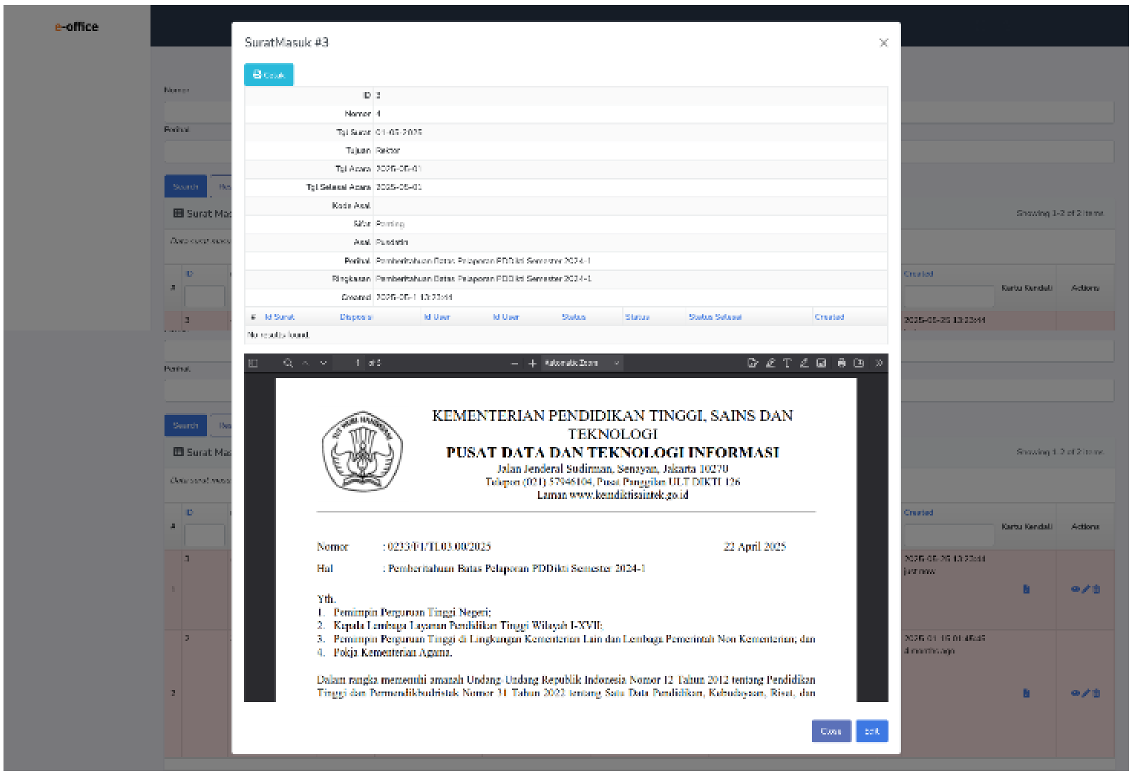
Main Dashboard: After logging in, users are presented with a dashboard that displays a summary of important information such as the number of new incoming mails, dispositions awaiting follow-up. Clear Navigation: The main navigation menu is laid out consistently and is easily accessible, grouping the main features of the system.


EOFFICE - TATA NASKAH DINAS ELEKTRONIK

SURAT MASUK 6
SURAT KELUAR 1
DISPOSISI 20

Figure 10. Eoffice Dashboard

Consistent Layout: Use of a uniform layout, iconography, and color scheme across all system pages to provide a cohesive user experience. Structured Input Forms: Data input forms are designed to be clear, with informative labels and input validation to minimize errors. Responsive Display: The interface design strives to adapt to a variety of device screen sizes (desktop, tablet, smartphone). The user interface (UI) of the eOffice information system is designed to be intuitive and efficient, especially in key modules such as disposition, view incoming mail, and SPPD submission. For disposition, the UI displays the details of the letter to be disposed of clearly, accompanied by a structured form for selecting the destination official or unit, typing instructions, and specifying a deadline, often with easily recognizable action buttons. In the view incoming mail feature, users are presented with a list of letters in an informative table format (e.g. sender, subject, date received, status) that can be sorted or searched, and when a letter is selected, the full details along with attachments are neatly displayed with options for further action. Meanwhile,


SuratMasuk #3

ID	3
Nomor	4
Tgl Surat	01-05-2025
Tujuan	Rektor
Tgl Acara	2025-05-01
Tgl Selesai Acara	2025-05-01
Kode Asal	
Sifat	Penting
Asal	Pusdatin
Perihal	Pemberitahuan Batas Pelaporan PDDikti Semester 2024-1
Ringkasan	Pemberitahuan Batas Pelaporan PDDikti Semester 2024-1
Created	2025-05-1 13:23:44

#	Id Surat	Disposisi	Id User	Id User	Status	Status	Status Selesai	Created
No results found.								

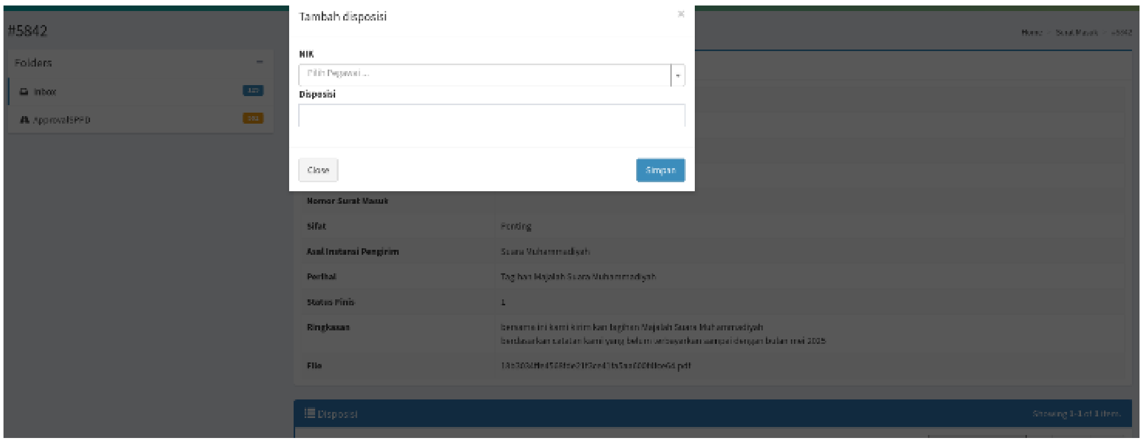
KEMENTERIAN PENDIDIKAN TINGGI, SAINS DAN TEKNOLOGI
PUSAT DATA DAN TEKNOLOGI INFORMASI
Jalan Jenderal Sudirman, Senayan, Jakarta 10270
Telepon (021) 57946104, Pusat Panggilan ULT DIKTI 126
Laman www.kemdiktisaintek.go.id

Nomor : 0233/F1/TI.03.00/2025 22 April 2025
Hal : Pemberitahuan Batas Pelaporan PDDikti Semester 2024-1

Yth.
1. Pemimpin Perguruan Tinggi Negeri;
2. Kepala Lembaga Layanan Pendidikan Tinggi Wilayah I-XVII;
3. Pemimpin Perguruan Tinggi di Lingkungan Kementerian Lain dan Lembaga Pemerintah Non Kementerian; dan
4. Pokja Kementerian Agama.

Dalam rangka memenuhi amanah Undang-Undang Republik Indonesia Nomor 12 Tahun 2012 tentang Pendidikan Tinggi dan Permendikbudristek Nomor 31 Tahun 2022 tentang Satu Data Pendidikan, Kebudayaan, Riset, dan

Figure 13. View Incoming Mail Document


Tambah disposisi

NIK
Pilih Pegawai ...
Disposisi

Nomor Surat Masuk	
Sifat	Penting
Asal Instansi Pengirim	Suara Muhammadiyah
Perihal	Tagihan Majalah Suara Muhammadiyah
Status Finis	1
Ringkasan	bersama ini kami kirim kan tagihan Majalah Suara Muhammadiyah berdasarkan catatan kami yang belum terbayarkan sampai dengan bulan mei 2025
File	[illegible].pdf

Figure 11. Disposition form

The SPPD submission UI will be an electronic form with guided input columns to fill in the purpose of the trip, date, activity details, cost estimates, and supporting document upload facilities, often accompanied by visualization of the approval flow so that users understand the stages of the process.

Black-box testing is described as an approach to software testing where the tester does not have access to the internal source code of the system, but instead relies solely on available interfaces to access the API. This method is often the only viable option, especially when source code is not available or in microservices architectures that make white-box testing difficult. Black-box techniques rely heavily on API specifications, often in OpenAPI Specification (OAS) format, to automatically generate test cases. Although existing tools with black-box approaches show promising results in error detection, especially through fuzzing, they have significant limitations due to their focus on random inputs only and the difficulty in describing personalized and specific test case workflows using only API specifications. This paper introduces RapiTest, an open-source continuous black-box testing application for RESTful web APIs, which utilizes API specifications for automatic test generation but also uses a new Domain Specific Language (DSL) called Test Specification Language (TSL) to create richer and more specific test cases.[15], [21]

The following are the results of testing with the blackbox method for the eoffice system developed

Table 1. Blackbox Test

NO	Feature	Test Case Description	Expected results	Actual Results	Conclusion
1	Login	Login with valid username and password	User successfully logged in and was redirected to the dashboard.	As Expected	Valid
2	Login	Login with valid username and invalid password	User failed to login and an error message "Incorrect username or password" appears.	As Expected	Valid
3	Incoming mail	View the list of incoming mail	The list of incoming mails is successfully displayed with relevant information (Mail No., Sender, Subject, Date).	As Expected	Valid
4	Incoming mail	Search for incoming mail by mail number	Incoming mail with number "SM/2024/001" was successfully found and displayed.	As Expected	Valid
5	Incoming mail	Download incoming mail attachments (if any)	The letter attachment was downloaded successfully.	As Expected	Valid
6	Outgoing mail	View the list of outgoing mail	The list of outgoing mails is successfully displayed with relevant information.	As Expected	Valid
7	Outgoing mail	Search for outgoing mail by mail number	The outgoing letter with number "SK/2024/001" was successfully found and displayed.	As Expected	Valid
8	Outgoing mail	Printing outgoing mail	The print preview of the outgoing letter is displayed correctly and can be printed.	As Expected	Valid
9	Dispositio n	Create a new disposition for incoming mail	Disposition successfully created and sent to destination. Notification received by disposition recipient.	As Expected	Valid
10	Dispositio n	Forwarding disposition	Disposition successfully forwarded to new destination.	As Expected	Valid
11	Submissio n of SPPD	Create a new SPPD application with valid data	SPPD submission has been successfully created and awaiting approval.	As Expected	Valid

Based on testing with blackbox from the table above all features can run well for the eoffice system developed at Universitas Muhammadiyah PKU Surakarta.

4. Conclusion

This research has successfully designed and built a web-based E-Office system by utilizing the Yii2 framework as a solution to overcome the challenges of inefficiency and effectiveness of office administration governance that is still running conventionally at Muhammadiyah University PKU Surakarta. The system developed includes essential features such as incoming and outgoing mail management, electronic mail disposition, SPPD management, and document status tracking, which are designed using the needs analysis method, UML design, and implementation with PHP and MySQL. With the implementation of this E-Office system, it is expected that Muhammadiyah University of PKU Surakarta can achieve significant transformation towards a more modern, paperless, transparent, and accountable office administration system. Furthermore, this system is projected to be able to increase work productivity, service quality for all academicians, and support the leadership decision-making process through faster and more accurate information access, while also showing that the Yii2 framework is the right choice for developing similar applications thanks to its structure, security (including the potential for Google SSO), and speed of development. Implementing DevOps as a system developer is very beneficial. The main focus is on automation (build, test, deploy) and using the right tools to support system development, adopting mindsets and practices that will make system development more efficient and effective.

References

[1] M. Yan Handita, D. E. Setiawan, and S. Kom, "SISTEM INFORMASI PENERIMAAN PEGAWAI DENGAN METODE AGILE EXTREME PROGRAMMING," *Jurnal Pendidikan Teknologi Informasi (JUKANTI)*, vol. 5, no. 2, pp. 154–164, Nov. 2022, doi: 10.37792/JUKANTI.V5I2.571.

[2] F. Erich, "Devops is simply interaction between development and operations," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11350 LNCS, pp. 89–99, 2019, doi: 10.1007/978-3-030-06019-0_7.

[3] K. S. Węgrzecki and M. Dzieńkowski, "Performance analysis of Laravel and Yii2 frameworks based on the MVC architectural pattern and PHP language," *Journal of Computer Sciences Institute*, vol. 24, pp. 265–272, Sep. 2022, doi: 10.35784/JCSI.3002.

[4] P. P. dan Pembuatan Web ERP untuk Cipta Tekno Mandiri Menggunakan Framework Yii and T. Hidayat, "Perancangan dan Pembuatan Web ERP untuk PT Cipta Tekno Mandiri Menggunakan Framework Yii 2," *JITSI : Jurnal Ilmiah Teknologi Sistem Informasi*, vol. 2, no. 3, pp. 85–89, Sep. 2021, doi: 10.62527/JITSI.2.3.43.

[5] M. Tri Indah Rahmayani *et al.*, "Implementasi Metode DevOps pada Sistem Informasi Pengelolaan Uang Kas Prodi Ekonomi Syari'ah STAIN Bengkalis," *Jurnal Nasional Komputasi dan Teknologi Informasi (JNKTI)*, vol. 7, no. 3, pp. 421–427, Jul. 2024, doi: 10.32672/jnkti.v7i3.7633.

[6] M. A. Londa, Y. A. Wee, and M. Radja, "Implementasi Sistem Informasi Monitoring Disposisi Surat Masuk dan Surat Keluar Berbasis Website," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 21, no. 2, 2022, doi: 10.30812/matrik.v21i2.1443.

[7] F. Kurniawan, E. A. Khrisnawati, R. Hadiwiyanti, and A. S. Fitri, "PENGUJIAN SISTEM INFORMASI MANAJEMEN SISWA BERBASIS WEBSITE MENGGUNAKAN METODE BLACK BOX DAN WHITE BOX," *Prosiding Seminar Nasional Teknologi dan Sistem Informasi*, vol. 2, no. 1, pp. 249–261, Sep. 2022, doi: 10.33005/SITASI.V2I1.306.

[8] S. Brinkkemper, "Method engineering: Engineering of information systems development methods and tools," *Inf Softw Technol*, vol. 38, no. 4 SPEC. ISS., pp. 275–280, 1996, doi: 10.1016/0950-5849(95)01059-9.

[9] O. H. Plant, A. Aldea, and J. van Hillegersberg, "Improving DevOps team performance through context-capability coalignment: Towards a profile for public sector organizations," *Inf Softw Technol*, vol. 178, p. 107585, Feb. 2025, doi: 10.1016/J.INFSOF.2024.107585.

[10] S. Tria Siska, L. Suhery, S. Rizki Ananda, S. Tinggi Teknologi Payakumbuh, and J. Khatib Sulaiman Sawah, "Application for Recording Marriage Events at KUA: Design and Implementation Using Visual Studio 2012 and MySQL," *Emerging Information Science and Technology*, vol. 5, no. 2, pp. 80–88, Nov. 2024, doi: 10.18196/EIST.V5I2.24846.

[11] A. Wampler, A. Charny, L. DeHaven, and C. Banta, "Implementing a Paperless Initiative in a Dietetic Internship Program," *J Am Diet Assoc*, vol. 111, no. 9, p. A20, Sep. 2011, doi: 10.1016/J.JADA.2011.06.068.

[12] P. K. Senyo, J. Effah, and E. L. C. Osabutey, "Digital platformisation as public sector transformation strategy: A case of Ghana's paperless port," *Technol Forecast Soc Change*, vol. 162, p. 120387, Jan. 2021, doi: 10.1016/J.TECHFORE.2020.120387.

[13] S. AL-Zahran, "How DevOps Practices Support Digital Transformation," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 3, pp. 2780–2788, Jun. 2020, doi: 10.30534/IJATCSE/2020/46932020.

[14] R. Grande, A. Vizcaíno, and F. O. García, "Is it worth adopting DevOps practices in Global Software Engineering? Possible challenges and benefits," *Comput Stand Interfaces*, vol. 87, p. 103767, Jan. 2024, doi: 10.1016/J.CSI.2023.103767.

[15] T. Yumoto, T. Matsuodani, and K. Tsuda, "A Test Analysis Method for Black Box Testing Using AUT and Fault Knowledge," *Procedia Comput Sci*, vol. 22, pp. 551–560, Jan. 2013, doi: 10.1016/J.PROCS.2013.09.135.

[16] A. Sunardi and Suharjito, "MVC Architecture: A Comparative Study Between Laravel Framework and Slim Framework in Freelancer Project Monitoring System Web Based," *Procedia Comput Sci*, vol. 157, pp. 134–141, Jan. 2019, doi: 10.1016/J.PROCS.2019.08.150.

[17] J. Hirai, S. Raghavan, H. Garcia-Molina, and A. Paepcke, "WebBase: a repository of Web pages," *Computer Networks*, vol. 33, no. 1–6, pp. 277–293, Jun. 2000, doi: 10.1016/S1389-1286(00)00063-3.

[18] J. Kniežová and P. Hrkút, "UML-Based Modeling for Effective Implementation of Constraint-Type Business Rules in Software Development," *Procedia Comput Sci*, vol. 257, pp. 1060–1066, Jan. 2025, doi: 10.1016/J.PROCS.2025.03.138.

[19] V. Kan, M. P. Lnu, S. Berhe, C. El Kari, M. Maynard, and F. Khomh, "Automated UML Visualization of Software Ecosystems: Tracking Versions, Dependencies, and Security Updates," *Procedia Comput Sci*, vol. 257, pp. 834–841, Jan. 2025, doi: 10.1016/J.PROCS.2025.03.107.

[20] Y. R. Kirschner, M. Gstür, T. Sağlam, S. Weber, and A. Koziolek, "Retriever: A view-based approach to reverse engineering software architecture models," *Journal of Systems and Software*, vol. 220, p. 112277, Feb. 2025, doi: 10.1016/J.JSS.2024.112277.

[21] D. Felicio, J. Simao, and N. Datia, "RapiTest: Continuous Black-Box Testing of RESTful Web APIs," *Procedia Comput Sci*, vol. 219, pp. 537–545, Jan. 2023, doi: 10.1016/J.PROCS.2023.01.322.