Informatics and Software Engineering
p-ISSN/e-ISSN: 2988-2818/2988-2222
Homepage: https://sanscientific.com/journal/index.php/ise
4(1) 27-35 (2026)
https://doi.org/10.58777/ise.v4i1.627

Research Article

Web-Based COT Monitoring Application for Operating Room Service Optimization

Muhammad Iqbal Aditya[1*], Norma Yunita[2], Siti Marlina[3]
[1] Department of Information Technology, Faculty of Engineering and Informatics, Universitas Bina Sarana Informatika, Jakarta, Indonesia
[2] Department of Information Technology, Faculty of Engineering and Informatics, Universitas Bina Sarana Informatika, Jakarta, Indonesia
[3] Department of Information System, Faculty of Engineering and Informatics, Universitas Bina Sarana Informatika, Jakarta, Indonesia

Received: May 26, 2026; Revision: July 3, 2026;
Accepted: July 7, 2026; Available Online: July 11, 2026

Abstract
This study aims to develop a web-based COT Monitoring application that supports real-time operating room status monitoring at Universitas Indonesia Hospital. The problem addressed is the use of manual and spreadsheet-based monitoring, which caused information delays, input errors, and limited operational reporting. The study used a qualitative case study approach through observation, interviews, documentation study, and literature review. The application was developed using the Waterfall model, covering requirements analysis, system design, coding, and Black-Box Testing. The system provides login, real-time dashboard, user management, operation status management, operating room master data, and sequential status update features. Black Box Testing was conducted on the main system functions, and the results showed that all tested features produced outputs in accordance with the expected scenarios. This indicates that the application functions properly and meets the defined functional requirements. The system currently runs on an internal hospital network and is not integrated with external hospital information systems. This article contributes a practical web-based monitoring design that improves information flow, supports transparency, reduces delays in status updates, and enhances the efficiency of operating room services.

Keywords: COT Monitoring; Laravel; Operating room; Web application

***Corresponding author:** Muhammad Iqbal Aditya (iqbalmuhamad764@gmail.com)

1. Introduction
Operating room management needs accurate information because room status changes in line with clinical and operational workflows. In a hospital environment, operating rooms involve surgeons, anesthesiologists, nurses, administrative officers, information technology staff, and management. Each party needs updated status information to prepare resources, monitor case progress, and coordinate patient flow. Delayed information can slow communication and degrade the quality of operational decision-making. For this reason, health service organizations need information systems that can record, process, and present data in a timely and structured manner (Kim et al., 2021; Kraus et al., 2021).

At Universitas Indonesia Hospital, the previous monitoring and reporting process for operating room activities relied on manual work or spreadsheets. The primary case source identified several obstacles, including delayed data

processing, risk of input errors, limited analysis, weak accessibility, incomplete operating room status history, and limited reporting support. These conditions made it difficult for management to obtain timely, accurate information on operational schedules and room allocation. Similar problems have appeared in other health information system studies, where fragmented data and manual recording reduce the speed of service coordination (Ananda et al., 2023; Tristiyanto & Adelliani, 2021).

Previous studies have developed several hospital information systems, such as web-based medical record systems, patient visit reporting systems, clinic information systems, and inpatient service applications. These studies show that web-based systems can support data recording, service administration, and reporting in healthcare institutions. However, most of them focus on general hospital administration. They do not specifically address real-time monitoring of operating room status, sequential status validation, role-based status updates, and centralized status history in operating room operations. This limitation underscores the need for a monitoring system designed for the specific workflow of operating room services.

A web-based monitoring system can reduce these obstacles by enabling users to access updated information in a browser and to record operational data in a centralized system. Laravel and PostgreSQL were used as supporting technologies in this study. However, the main focus of this research is not the technical advantages of the tools but rather the design of a monitoring model that aligns with the operating room workflow, supports timely status updates, and improves operational visibility.

The research gap addressed in this article is the lack of a focused web-based monitoring application for the flow of operating room status in the observed hospital unit. Prior studies and related healthcare web applications discuss hospital information systems, medical records, patient visit reports, clinic services, or inpatient services. This study focuses on operating room status monitoring and the operational need to update room status sequentially (Ananda et al., 2023; Kadim et al., 2025; Sekarini & Widhiyanti, 2023).

The novelty of this study lies in the development of a focused COT Monitoring application that combines real-time dashboard monitoring, role-based access, centralized status recording, and sequential room status validation. This system differs from previous hospital monitoring applications because it is designed specifically for operating room workflow control. The application not only displays but also stores operational data. It also guides status changes in accordance with the defined sequence of operating room activities. This feature helps prevent inconsistent status updates and supports more reliable monitoring information.

The specific objective of this study is to design, implement, and test a COT Monitoring application for internal operating room monitoring at Universitas Indonesia Hospital. Beyond system implementation, this study provides a scientific contribution by proposing a practical monitoring model that integrates workflow analysis, role separation, centralized recording, sequential status validation, and functional testing. This contribution can be used as a reference for developing internal hospital monitoring systems that require real-time visibility, data consistency, and operational transparency. Recent operating room literature shows that operating room planning and surgical scheduling remain complex because time, resources, staff, and uncertainty factors affect utilization, delays, and service performance (Dodaro et al., 2022; Rahimi & Gandomi, 2021; Taaffe et al., 2021). Digital health implementation research also shows that technology success depends on workflow fit, user adoption, organizational readiness, and sustainability of use (Braa et al., 2023; Butcher & Hussain, 2022; Haleem et al., 2021; Kim et al., 2021; Kraus et al., 2021; Makumbani & Tsibolane, 2025; Wienert et al., 2022).

2. Methods

This study used a qualitative case study approach to understand the actual workflow, user needs, and problems in the operating room management process at Universitas Indonesia Hospital. Qualitative inquiry was suitable because the system requirements were derived from field conditions rather than from a purely numerical experiment. The data collection techniques consisted of observation, interviews, document analysis, and literature review. An observation was conducted to understand the current workflow, the previous manual or semi-digital process, and the obstacles hospital staff face in managing operating room activities. Interviews were conducted with five respondents, selected purposively because they were directly involved in operating room monitoring and information system support. The respondents consisted of one hospital information technology staff member, one operating room administrative officer, two operating room operators, and one management representative. These respondents were selected based on three criteria: direct involvement in operating room monitoring, knowledge of the previous reporting process, and ability to explain user needs or system constraints. The interviews focused on workflow monitoring, data input issues, room status reporting, user roles, and expectations for the proposed application. The documentation study used internal records, such as operation schedules, operating room status notes, room usage reports, and standard operating procedures. Literature review supported the theoretical basis for information systems, web applications, real-time monitoring, operating room management, software development, and software testing.

The software development method followed the Waterfall model. This model organizes development into sequential phases: requirements analysis, system design, coding, and testing (Maspupah, 2024; Subecz, 2021). The Waterfall model was selected because the main requirements were identified before implementation, the system scope was limited to internal hospital monitoring, and the development process needed clear documentation through diagrams, database design, interface design, and testing tables. Compared with Agile or iterative methods, Waterfall was more suitable for this study because the functional requirements were relatively stable and the project followed a structured academic development process. Current references on software quality, software testing, and web application security strengthen the development stages. These references support the use of structured requirements, interface design, functional validation, access control, and traceable testing in application development (Sarwosri et al., 2023; Maspupah, 2024; Kamal et al., 2024; Perdana et al., 2024).

The requirements analysis phase identified functional requirements for two main roles. The administrator requires login, dashboard access, operation status management, operating room master data management, and user management. The operator requires a responsive web interface and limited authority to update operating room status. Non-functional requirements include availability on the internal hospital network, simple browser-based access, clear validation to reduce input errors, and access control based on user roles.
The design phase translated requirements into a database structure, use-case flows, activity logic, interface layouts, and status transition logic. Visual modeling was used to organize interactions and workflows because current software-quality references emphasize clear requirements, functional suitability, maintainability, and traceable validation (Salsabila et al., 2025). The implementation phase used Laravel, PHP, PostgreSQL, HTML, CSS, JavaScript, Bootstrap, Laragon, Composer, Visual Studio Code, and common web browsers. The testing phase used Black Box Testing to verify functional behavior from given input without inspecting internal code structure (Salsabila et al., 2025).

Figure 1 presents the research method and development flow used in this study. The figure should be provided at a higher resolution in the final publication to improve the readability of each development stage.

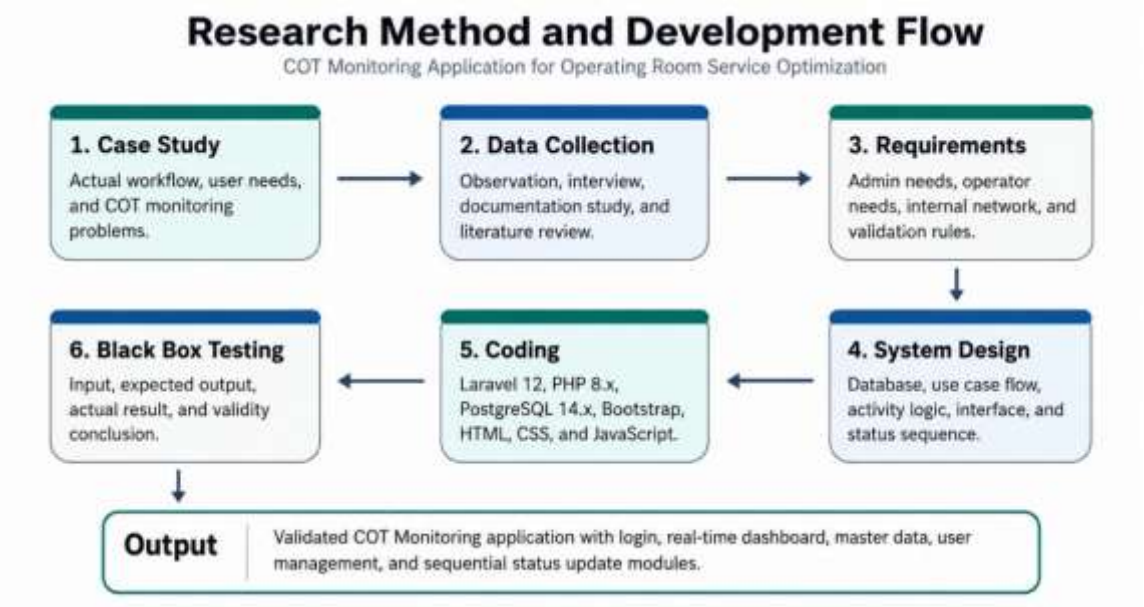

Figure 1. Research method and development flow of the COT Monitoring application

3. Results and Discussion

3.1 Problem analysis and system needs

The analysis found that the main operational problems were delayed operational schedule information, a lack of real-time room status notifications, and difficulty preparing structured COT activity reports. The previous manual or semi-digital process created inefficient communication between units. These findings show that the core issue was not only data entry but also the absence of a centralized monitoring mechanism that could update room status and present it to the right users quickly.

The system needs were grouped into administrator and operator needs. The administrator needs account management, operation status management, room master data, and dashboard monitoring. The operator needs a responsive interface and controlled access for status updates. This role separation is important because hospital information systems must limit user actions based on their responsibilities. It also supports accountability because

users can perform only the functions related to their operational role. From the information system perspective, the proposed application changes the workflow from fragmented recording into centralized recording. Centralized recording supports faster data retrieval and reduces the duplication of manual notes. From the hospital management perspective, this change helps staff and management obtain the same operational view through the dashboard.

3.2 System design and operating room status flow

The database was designed to support efficient storage, processing, and retrieval. The main data objects include users, operating rooms, operation statuses, and status history. Entity-Relationship Diagrams and Logical Record Structures were used to model the relationships among these objects. This design supports integrity because each update can be linked to a room, a status, and a user. The system uses a sequential status update approach. This approach prevents status changes that do not follow the operational process. For example, an operator should not update a room directly from Sterile to Incision without passing the anesthesia preparation status. This rule supports process discipline and reduces inconsistent data. The application also applies status ordering, so active, empty, or cleaning rooms are displayed in a more useful sequence for medical staff. The interface design uses a dashboard as the central monitoring page. The dashboard displays operating room status and allows authorized users to update data. Bootstrap supports responsive interface development, while HTML, CSS, and JavaScript support structure, presentation, and user interaction (Santoso, 2019).

3.3 Application implementation

The application was implemented as a web-based system for the internal hospital network. It is not published online because the monitored data relates to internal operating room operations. This deployment decision supports access control and reduces unnecessary exposure to public networks. Users access the application through a browser on client computers connected to the local hospital network. The main features include login, real-time dashboard, user management, master operation status, master operating room data, and operating room status updates. The administrator can add, edit, delete, and reorder operation statuses. The administrator can also manage users and room master data. The operator can update room status according to the permitted workflow. These features directly address the earlier problems of slow status updates, weak accessibility, and limited reporting structure.

The technology stack supports modular development. Laravel structures the backend through controllers, routes, middleware, and models. PostgreSQL stores structured operating room data. HTML, CSS, JavaScript, and Bootstrap provide the interface layer. This combination is aligned with current web application development practices and supports maintainability for a small internal system (Subecz, 2021; Sinlae et al., 2024; Salunke & Ouda, 2024). The technical references were also updated to support the selected tools. Laravel was used as the web framework, PostgreSQL as the relational database, Bootstrap as the interface toolkit, and OWASP Top 10 as the security reference for common web application risks (Subecz, 2021; Siregar & Yahfizham, 2024; Salunke & Ouda, 2024; Perdana et al., 2024).

3.3.1 Implementation visualization

To strengthen the implementation section, visual evidence is presented in five parts: user interface, implemented features, application usage flow, implementation outcomes within the institution, and Black-Box Testing evidence. These figures were prepared from the thesis documentation and are intended to make the implementation process easier to understand.

Figure 2 shows the main interface of the application. The login page controls access for administrators and operators, while the dashboard presents real-time operating room status information in a simple layout.

a. Main user interface of the COT Monitoring application

Figure 2. Main user interface of the COT Monitoring application

Figure 3 summarizes the main features that were implemented in the system. These include user management, operation status management, and operating room master data management. Together, these modules support operational control and data consistency in the monitoring process.

b. Implemented features in the COT Monitoring application

Figure 4 illustrates the basic use flow. The operator first selects an operating room, then updates the status according

to the defined sequence, and finally monitors the result on the dashboard. This sequence reflects the intended operating procedure of the application.

Figure 3. Implemented features in the COT Monitoring application

c. Basic usage flow of the application

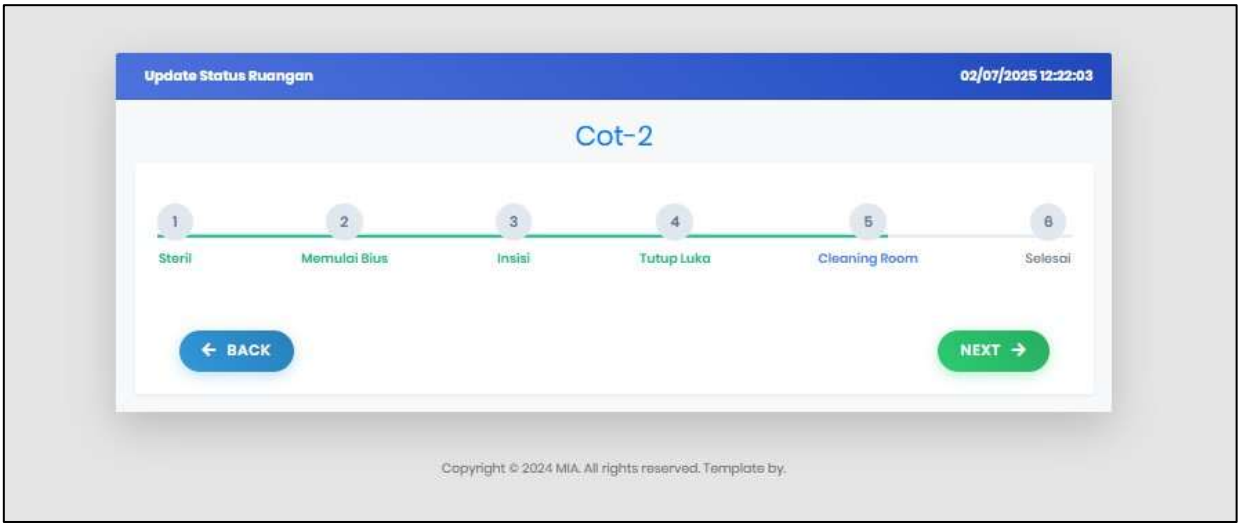

Figure 4. Basic usage flow of the application

Figure 5 explains the implementation context and the practical results observed in the institution. The application is used internally through the hospital intranet by administrators and operators. In practice, the system enables faster access to room status, greater transparency, and reduced communication delays during operating room monitoring.

d. Institutional implementation context and observed results

Figure 5. Institutional implementation context and observed results

3.3.2. System testing

Figure 6 presents the Black Box Testing evidence and a concise testing summary. The login, master status, master operating room, and update status modules all produced valid results in accordance with the expected functional requirements. These findings indicate that the application aligns with user needs.

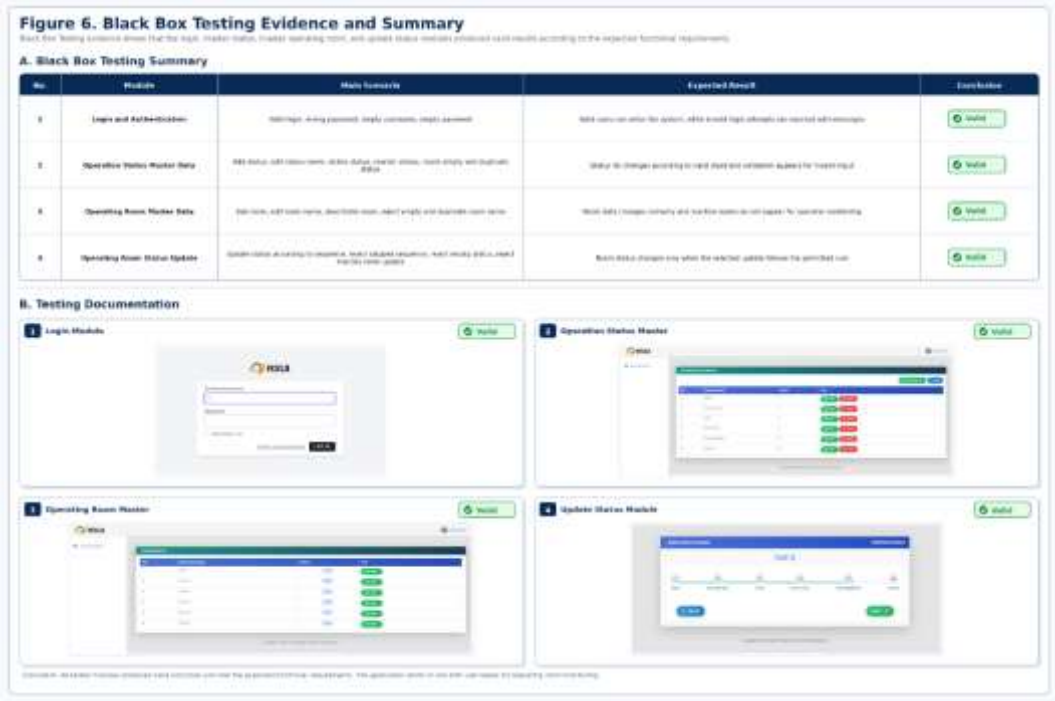

Figure 6. Black Box Testing summary and documentation

3.4 Testing results

Testing used Black Box Testing scenarios for the main system modules. The scenarios compared test input, expected output, actual result, and validity conclusion. The testing scope was expanded to cover login and authentication, user management, operation status master data, operating room master data, operating room status updates, dashboard display, validation rules, access control, and error handling. This wider scope was used to ensure that the application not only produced correct outputs but also handled invalid input, unauthorized access, duplicate data, inactive rooms, and incorrect status sequences.

Table 1. Summarizes the Black-Box Testing results for the system implementation.

Module	Main Scenario	Expected Result	Conclusion
Login and authentication	Valid login, wrong password, empty username, empty password	Valid users can enter the system, while invalid login attempts are rejected with messages	Valid
User management	Add user, edit user, delete user, reject empty input, reject duplicate username.	System saves valid data and rejects invalid or duplicate data	Valid
Operation status master	Add status, edit status name, delete status, reorder status, reject empty and duplicate status.	Status list changes according to valid input, and validation appears for invalid input.	Valid
Operating room master	Add room, edit room name, deactivate room, reject empty and duplicate room names.	Room data is updated correctly, and inactive rooms do not appear in operator monitoring.	Valid
Operating room status update	Update status according to sequence; reject skipped sequence; reject empty status; reject inactive room updates.	Room status changes only when the selected update follows the permitted rule.	Valid
Dashboard monitoring	Display the active room status, refresh the updated status and show the room condition clearly.	Dashboard presents updated operating room status for authorized users	Valid
Access control	Administrator accesses master data, operator accesses status update only	User access follows the assigned role, and unauthorized functions are restricted	Valid
Error handling and validation	Submit empty forms, duplicate data, and invalid status transitions	The system displays validation messages and prevents incorrect data storage	Valid

The test results in table 1 show that the system exhibited the expected behavior across all tested modules. The login and authentication module accepted valid users and rejected incorrect credentials. The user management module

handled valid user additions, edits, and deletions, while also rejecting empty input and duplicate usernames. The operation status module allowed valid status changes and prevented empty or duplicate status entries. The operating room master module supported room activation control, which is important because inactive rooms should not appear in operator monitoring. The most important result appears in the operating room status update module. The system accepted status changes that followed the defined sequence and rejected those that skipped it. This behavior is relevant to operating room monitoring because the status sequence represents the actual operational process. If users can skip status stages at will, the dashboard may display inaccurate clinical workflow information. The validation therefore improves data consistency and helps management better understand the operational situation. From a service-optimization perspective, the application offers three benefits. First, it reduces manual recording by moving the update process to a structured web interface. Second, it improves transparency by allowing updated room status to be viewed on the dashboard. Third, it supports data-driven decision-making because room status records can serve as a foundation for future reporting. These benefits match the broader role of digital information systems in supporting operations and management (Kraus et al., 2021; Alfarisi et al., 2023).

3.5 Discussion

The findings confirm that a web-based monitoring system can address a practical operational gap in a hospital unit. The application does not replace clinical judgment or surgical scheduling policy. Instead, it strengthens the information layer that connects staff activity, room status, and management visibility. This role is consistent with recent health information system studies that view digital systems as tools for improving coordination, documentation, and operational control (Ananda et al., 2023; Chen et al., 2021; Fahrudin et al., 2024).

The findings are consistent with operating room management studies that emphasize the importance of timely data, schedule visibility, uncertainty management, and structured rescheduling in improving operating room utilization and operational decision-making (Dodaro et al., 2022; Rahimi & Gandomi, 2021; Taaffe et al., 2021).
However, the current deployment has several limitations. First, the application runs only on the hospital intranet, so access is limited to users connected to the internal network. Second, the system is not yet integrated with the hospital information system, electronic medical records, surgical scheduling system, or notification platform. This condition may still require duplicate data entry when users need to update information in different systems. Third, the application has been tested only through functional Black-Box Testing and has not yet been evaluated through long-term operational metrics, such as time reduction, error rate, room utilization, user satisfaction, and reduced communication delays. Fourth, the system does not yet include advanced features such as an audit trail, automatic alerts, graphical reporting, and mobile device integration.

These limitations show that the current application is suitable as an initial internal monitoring system, but further development is needed before wider implementation. Future integration with hospital information systems should consider data security, privacy, access control, audit trail, and interoperability. Web applications that handle operational hospital data should apply secure configuration, validation, and role-based access control to reduce operational and security risks (Siregar & Yahfizham, 2024). The article also shows that system development in healthcare should combine technical design with an understanding of workflows. Without observation and interview, the application could miss important operational rules such as status sequence and role separation. For this reason, requirements analysis is a critical phase in hospital software development, especially when the system affects real-time coordination among staff (Makumbani & Tsibolane, 2025; Wienert et al., 2022).

4. Conclusion

This study developed a web-based COT Monitoring application for operating room status monitoring at Universitas Indonesia Hospital. The main features include login, real-time dashboard, user management, operation status management, operating room master data, and sequential room status updates. Black Box Testing showed that the tested modules produced the expected outputs. The system validated input, managed master data, supported dashboard monitoring, restricted user access based on roles, and prevented status updates that skipped the permitted sequence. These results indicate that the application can support more reliable operating room monitoring by improving data consistency, accelerating information flow, and strengthening transparency in internal operating room management.

The scientific contribution of this study is the proposed monitoring model that integrates workflow analysis, role separation, centralized status recording, sequential status validation, and functional testing to optimize operating room service. This model can become a reference for hospitals that still use manual or spreadsheet-based monitoring and need a structured internal system for real-time operational visibility. The system has the potential to be adopted in other hospitals with similar operating room monitoring problems. However, implementation in other institutions should account for differences in workflow, user roles, infrastructure, data governance, and integration requirements. Future research should measure quantitative impacts before and after implementation, such as time reduction, error rate, room utilization, reporting speed, and user satisfaction. Future development should also add automatic notifications, graphical reports, an audit trail, stronger role-based access control, and integration with

hospital information systems.

Recommendation

The hospital should provide user training for operators and administrators before full operational use. Future development should add automatic notifications, graphical reports, an audit trail, integration with the hospital information system, and stronger role-based access control. Cloud access can be considered only when security, privacy, and data governance requirements are clearly defined.

Limitations and avenues for future research

This article is limited to application development and functional testing in an internal hospital context. It does not measure quantitative impacts such as time reduction, error rate, room utilization, or user satisfaction after long-term deployment. Future research should use operational data before and after implementation, involve more user groups, and compare the system with integrated hospital information systems.

References

Alfarisi, I. A., Priandika, A. T., & Puspaningrum, A. S. (2023). Penerapan Framework Laravel Pada Sistem Pelayanan Kesehatan (Studi Kasus: Klinik Berkah Medical Center). *Jurnal Ilmiah Computer Science*, *2*(1), 1–9. https://doi.org/10.58602/jics.v2i1.11

Ananda, S. P., Fakhrurrifqi, M., Putri, D. G. P., & Wijayanti, R. (2023). Pengembangan Sistem Informasi DataRawat Berbasis Web. *Journal of Internet and Software Engineering*, *4*(2), 20–27. https://doi.org/10.22146/jise.v4i2.8540

Braa, J., Sahay, S., & Monteiro, E. (2023). Design Theory for Societal Digital Transformation: The Case of Digital Global Health. *Journal of the Association for Information Systems*, *24*(6), 1645–1669. https://doi.org/10.17705/1jais.00816

Butcher, C. J., & Hussain, W. (2022). Digital healthcare: the future. *Future Healthcare Journal*, *9*(2), 113–117. https://doi.org/10.7861/fhj.2022-0046

Chen, J., Amaize, A., & Barath, D. (2021). Evaluating Telehealth Adoption and Related Barriers Among Hospitals Located in Rural and Urban Areas. *The Journal of Rural Health*, *37*(4), 801–811. https://doi.org/10.1111/jrh.12534

Dodaro, C., Galata, G., Kamran Khan, M., Maratea, M., & Porro, I. (2022). Operating Room (Re)Scheduling with Bed Management via ASP. *Theory and Practice of Logic Programming*, *22*(2), 229–253. https://doi.org/10.1017/S1471068421000090

Fahrudin, H., Abdussalaam, F., & Sari, I. (2024). Perancangan Sistem Informasi Pengolahan Data Morbiditas Rawat Inap Guna Menunjang Tata Kelola Pelaporan Rawat Inap. *Jurnal Indonesia : Manajemen Informatika Dan Komunikasi*, *5*(3), 2145–2157. https://doi.org/10.35870/jimik.v5i3.846

Haleem, A., Javaid, M., Singh, R. P., & Suman, R. (2021). Telemedicine for healthcare: Capabilities, features, barriers, and applications. *Sensors International*, *2*. https://doi.org/10.1016/j.sintl.2021.100117

Kadim, A. A., Hadjaratie, L., Mokoginta, R., & Zakaria, A. (2025). Rancang Bangun Sistem Informasi Rekam Medis Berbasis Web dengan Framework Laravel. *Diffusion: Journal of Systems and Information Technology*, *5*(1). https://doi.org/https://doi.org/10.37031/diffusion.v5i1.30066

Kamal, M., Nasrullah, M., Rosadi, R., Anggito, Y., & Roy, S. C. (2024). Evaluation of Information Security at the XYZ Foundation Using OWASP Top 10 2021 Framework. *Journal of Advances in Information and Industrial Technology*, *6*(2), 143–152. https://doi.org/10.52435/jaiit.v6i2.397

Kim, H. S., Kwon, I. H., & Cha, W. C. (2021). Future and development direction of digital healthcare. *Healthcare Informatics Research*, *27*(2), 95–101. https://doi.org/10.4258/HIR.2021.27.2.95

Kraus, S., Schiavone, F., Pluzhnikova, A., & Invernizzi, A. C. (2021). Digital transformation in healthcare: Analyzing the current state of research. *Journal of Business Research*, *123*, 557–567. https://doi.org/10.1016/j.jbusres.2020.10.030

Makumbani, H., & Tsibolane, P. (2025). Design-Reality Gap Analysis of Health Information Systems Failure. *Procedia Computer Science*, *256*, 949–956. https://doi.org/10.1016/j.procs.2025.02.199

Maspupah, A. (2024). Literature Review: Advantages And Disadvantages Of Black Box And White Box Testing Methods. *Jurnal Techno Nusa Mandiri*, *21*(2), 151–162. https://doi.org/10.33480/techno.v21i2.5776

Perdana, C., Maharani, & Angga Wijaya, M. (2024). Implementasi Framework Bootstrap 5 Pada Perancangan Front-End Website MC BRO di PT X. *Jurnal Sistem Informasi Galuh*, *2*(1), 30–43. https://doi.org/10.25157/jsig.v2i1.3634

Rahimi, I., & Gandomi, A. H. (2021). A Comprehensive Review and Analysis of Operating Room and Surgery Scheduling. *Archives of Computational Methods in Engineering*, *28*(3), 1667–1688. https://doi.org/10.1007/s11831-020-09432-2

Salsabila, D. S., Ardilla, Y., & Wahyudi, N. (2025). Penerapan Iso 29119 Sebagai Upaya Peningkatan Kualitas Layanan Website E-Procurement. *Jurnal Inovasi Pendidikan Dan Teknologi Informasi (JIPTI)*, *6*(2), 475–484. https://doi.org/10.52060/jipti.v6i2.3740

Salunke, S. V., & Ouda, A. (2024). A Performance Benchmark for the PostgreSQL and MySQL Databases. *Future Internet*, *16*(10). https://doi.org/10.3390/fi16100382

Santoso, M. F. (2019). Teknik Responsive Web Design Bootstrap 4 Serta Penerapannya Dalam Rancang Bangun Layout WEB. *Jurnal Pilar Nusa Mandiri*, *15*(1), 61–68. https://doi.org/10.33480/pilar.v15i1.101

Sarwosri, S., Rochimah, S., Laili Yuhana, U., & Balqis Hidayat, S. (2023). Software Quality Measurement for Functional Suitability, Performance Efficiency, and Reliability Characteristics Using Analytical Hierarchy Process. *JOIV : International Journal on Informatics Visualization*, *7*(4), 2421. https://doi.org/10.62527/joiv.7.4.2441

Sekarini, I. G. A. A., & Widhiyanti, A. A. S. (2023). Aplikasi Informasi Kunjungan Harian Pasien pada Rumah Sakit Swasta Kota Denpasar Menggunakan Framework Laravel. *Jurnal Saintikom (Jurnal Sains Manajemen Informatika Dan Komputer)*, *22*(2), 546. https://doi.org/10.53513/jis.v22i2.8627

Sinlae, F., Irwanda, E., Maulana, Z., & Syahputra, V. E. (2024). Penggunaan Framework Laravel dalam Membangun Aplikasi Website Berbasis PHP. *Jurnal Siber Multi Disiplin (JSMD)*, *2*(2). https://doi.org/10.38035/jsmd.v2i2

Siregar, L. K., & Yahfizham, Y. (2024). Manajemen Proyek Sistem Informasi Surat Perintah Perjalanan Kegiatan Setdaprovsu Berbasis Website. *JUKTISI: Jurnal Komputer Teknologi Informasi Sistem Informasi*, *3*(1), 628–638.

Subecz, Z. (2021). Web development with Laravel framework. *Gradus*, *8*(1), 211–218. https://doi.org/10.47833/2021.1.csc.006

Taaffe, K., Pearce, B., & Ritchie, G. (2021). Using kernel density estimation to model surgical procedure duration. *International Transactions in Operational Research*, *28*(1), 401–418. https://doi.org/10.1111/itor.12561

Tristiyanto, T., & Adelliani, A. (2021). Sistem Informasi Pelayanan Kesehatan Rawat Inap Menggunakan Framework Laravel. *Jurnal Pepadun*, *2*(1), 90–100. https://doi.org/10.23960/pepadun.v2i1.27

Wienert, J., Jahnel, T., & Maaß, L. (2022). What are Digital Public Health Interventions? First Steps Toward a Definition and an Intervention Classification Framework. *Journal of Medical Internet Research*, *24*(6). https://doi.org/10.2196/31921