Jurnal Inovasi Informatika dan Bisnis Digital (JIIBD)

Website: https://ioinformatic.org/index.php/JIIBD

Juli 2026. Vol. 1. No. 3

A Laravel-Based Web Application for Task Monitoring and Project Management at PT X

Misbahul Munir[1]*, Faisal Muttaqin[2]

[1,2]UPN "Veteran" Jatim, Indonesia
23081010251@student.upnjatim.ac.id[1], faisalmuttaqin.if@upnjatim.ac.id[2]*

Abstract

Effective project task management plays a crucial role in ensuring the operational performance of information technology companies. PT X, an Indonesian developer of Hospital Management Information Systems (SIMRS), encountered several limitations in its legacy task monitoring system developed using native PHP. These limitations included inadequate authentication mechanisms, the absence of Role-Based Access Control (RBAC), and insufficient analytical visualization to support managerial decision-making. This study aims to develop and implement a web-based task monitoring and project management system utilizing a Single Page Application (SPA) architecture based on Laravel 12, Vue.js 3, and Inertia.js. The system was developed following the Waterfall software development model, consisting of requirements analysis, system design, implementation, testing, and maintenance phases. The proposed solution comprises two integrated platforms: an Admin Panel for Product Owner teams and a Client Portal for healthcare institution clients. The system incorporates essential features such as an analytics dashboard, Kanban board, RBAC, activity logging, notification services, and Service Level Agreement (SLA) monitoring. Functional validation was conducted using the Black Box Testing method, achieving a 100% success rate across all test scenarios. The implementation of the proposed system successfully reduced manual administrative processes, enhanced project monitoring capabilities, and improved the overall efficiency of task management within the organization.

***Keyword**: Dashboard; Kanban; Laravel; Monitoring Task; SLA.*

1. Introduction

The accelerative development of information technology demands various business entities to adopt digital systems to manage operational activities, especially in supervising the progress of project execution. A systematic and transparent assignment management scheme is an absolute foundation for companies handling multiple large-scale projects with various stakeholders. The successful completion of project target times is heavily influenced by the quality of assignment distribution and the effectiveness of periodic work evaluations. Failure to implement an adequate supervision platform risks triggering delays in task completion, internal team miscommunication, and low operational transparency that ultimately leads to a drastic decline in customer satisfaction levels [5]. PT X is an information technology company that focuses on the development of Hospital Management Information System (HMIS) software and serves hundreds of healthcare facilities (faskes) spread across various regions in Indonesia. In daily operations, the Product Owner (PO) team at PT X is responsible for managing the entire lifecycle of feature development tasks, starting from receiving client requests to the product release process. Currently, PT X still uses a task monitoring system built with native PHP without a modern framework, thereby causing various significant operational problems.

The first problem relates to the aspect of system security, where the authentication mechanism used is still hardcoded without the application of hashing on user passwords, making it highly vulnerable to cyber attacks. The second problem is the absence of Role-Based Access Control (RBAC), which causes all users to have the same level of access to system data and functionality without any restrictions based on their respective positions or responsibilities. The third problem is related to the database architecture that uses string-based relations (names) instead of ID-based foreign keys, thereby potentially causing inconsistencies and damage to data integrity when entity name changes occur. The fourth problem is the lack of data visualization features and dashboard analytics that make it difficult for management to identify bottlenecks, task completion trends, and the performance of each team member [2]. The fifth problem relates to the communication process between the PO team and healthcare facility clients, which is still carried out manually through instant messaging applications like WhatsApp, resulting in high work interruptions and a lack of structured documentation regarding the task status conveyed to clients. A number of previous literature studies have described the effectiveness of digitalizing project task monitoring. The use of interactive dashboard visualizations has been proven to reduce administrative reporting time and minimize the potential for input errors compared to manual recording on spreadsheets [3]. In addition, the integration of Kanban boards in web-based management systems is significantly able to clarify the division of workflow and facilitate the early detection of bottlenecks in the production process [10]. On the other hand, the use of analytical dashboards to track service level agreements (SLAs) plays an important role in maintaining operational compliance with time commitments to service users [2].

From the web development technology perspective, the utilization of the Laravel framework is considered highly efficient for designing project monitoring systems due to the availability of built-in security features, route management, and structured database interactions through Eloquent ORM [4]. Furthermore, the collaboration between Laravel, Vue.js, and the Inertia.js protocol is capable of producing applications with a dynamic Single Page Application (SPA) architecture without the need to separate backend and frontend repositories in

a complex manner [7]. Through the identification of the challenges above, this research is directed toward designing and building an integrated task monitoring system using Laravel 12, Vue.js 3, and Inertia.js. The final product of this research is aimed at solving the architectural weaknesses of the old system, providing visual data analysis as a basis for managerial decision-making, and presenting an independent portal for healthcare facility partners to track the progress of assignments directly without depending on manual communication.

2. Research Method

This research uses the Waterfall software development method, which consists of five sequential stages. The choice of the Waterfall method is based on the clarity of the system requirements that have been defined from the beginning and its suitability for projects with a specifically determined scope [4]. This approach is also consistent with the methodology commonly used in the development of monitoring information systems, as has been applied in various previous studies in the field of web-based project management [5], [7]. The execution procedure of this research is divided into five structured stages in accordance with the development cycle of the Waterfall model. The first stage, requirements analysis, begins with an in-depth investigation of the weaknesses of the old task monitoring system. This investigation was carried out through direct observation of the daily work rhythm of the Product Owner as well as intensive interviews with users to formulate the functional and non-functional requirement specifications of the Admin Panel platform [8]. Next, in the system design stage, the functionality aspects and database relationships are modeled using Use Case Diagrams, Activity Diagrams, and relational schema designs (Entity Relationship Diagrams). The user interface design at this stage is compiled using a combination of Soft UI visual concepts.

The third stage is the implementation phase, which translates all designs into program code using a combination of Laravel 12 technology on the backend, Vue.js 3 on the frontend, and Inertia.js as a dynamic data bridge to realize Single Page Application (SPA) performance. After the system is integrated, the fourth stage is followed by external functional evaluation using the Black Box Testing method to test the alignment of feature inputs and outputs without involving internal modification of the program code [1], [5]. Finally, the maintenance stage includes deploying the application to the production environment, conducting operational training sessions for internal users, and preparing periodic database backup procedures to ensure long-term reliability.

The system architecture designed adopts a Monolith SPA pattern using Inertia.js as a connector protocol between server-side Laravel and client-side Vue.js. On the first visit, the server sends a complete HTML page along with the JavaScript bundle. On subsequent navigations, Inertia.js only transfers data in JSON format without doing a full page reload, thereby providing a user experience equivalent to a pure SPA application while still taking advantage of server-side routing and middleware from Laravel [7]. The system is accessed by the internal team through the 'web' authentication guard with two roles, namely Admin (PO Lead) and Member (PO Member/Engineer). The relationship of this architecture can be seen in Figure 1.

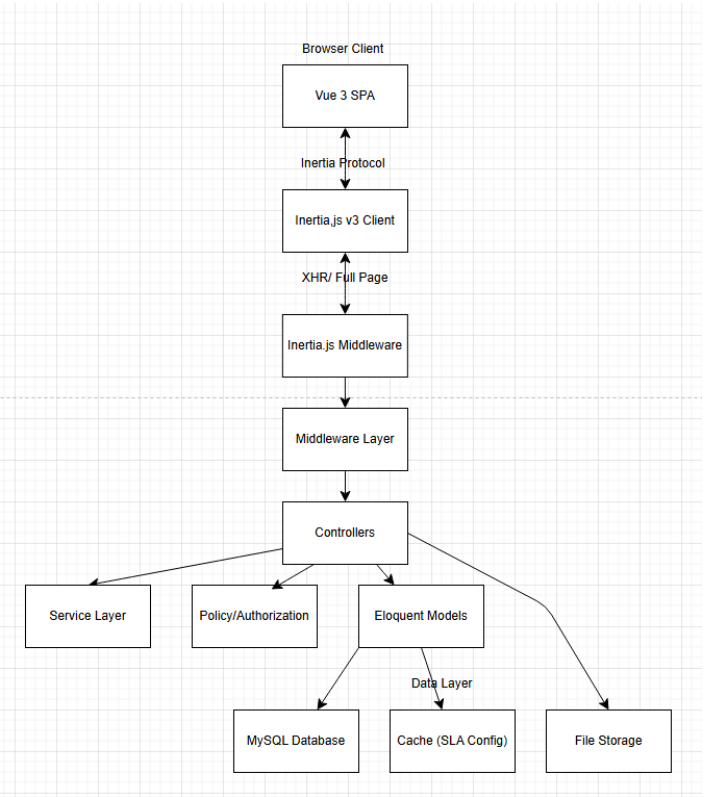

Figure 1: System Architecture

Figure 1 represents the monolith Single Page Application (SPA) system architecture scheme implemented. The integration between Laravel 12 on the backend and Vue.js 3 on the frontend is directly bridged by Inertia.js. Through this scheme, the user's browser on the Admin Panel (localhost:8000) only requests data in JSON format after the initial page load is complete. Laravel Web Engine acts as the main controller that processes all security aspects, including routing, authentication guards, role authorization (Admin and Member), and business logic management. Data storage is centralized in a MySQL relational database that accommodates 14 tables connected to each other using foreign keys to maintain data integrity relationally.

This system restricts roles only to the internal team consisting of Admin and Member. The interaction of actors with the system is modeled through a Use Case Diagram presented in Figure 2.

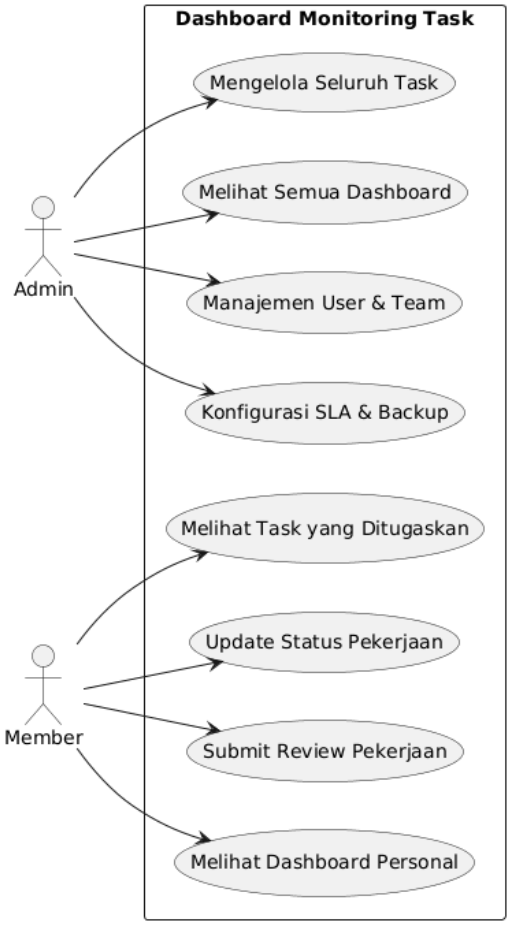

Figure 2: Use Case Diagram for Admin and Member Roles

Based on the use case diagram in Figure 2, the interaction between users and the system is divided into strict authorization boundaries to ensure operational security. The system limits roles only to the internal team consisting of Admin and Member. The Admin actor has full access rights to fundamental data management, including CRUD (Create, Read, Update, Delete) operations on team entities, clients (hospitals/healthcare facility partners), and all monitoring tasks. Meanwhile, the Member actor (PO Member/Engineer) has a limited scope that only covers monitoring the task list and updating the status of tasks specifically assigned to them.

The system database is designed with 14 tables that relate to each other using foreign keys to maintain referential data integrity. The structure of the main database tables that store system functionality data is presented in Table 1.

Table 1: Main Database Table Structure Design

Table Name	Description	Primary Key	Foreign Key
'users'	Stores account data of internal users (Admin and Member)	'id'	'team_id' -> 'teams.id'
'teams'	Stores product and engineer team data	'id'	-
'clients'	Stores profile data of partner healthcare facilities	'id'	-
'tasks'	Stores monitoring task details and time targets	'id'	'client_id', 'product_id', 'engineer_id'
'documents'	Stores client UAT/BAST documents	'id'	'client_id', 'created_by'
'sla_configs'	Stores task SLA limit configurations	'id'	-
'task_templates'	Stores recurring task template data	'id'	-
'task_comments'	Stores task discussion comments	'id'	'task_id', 'user_id'
'activity_logs'	Stores audit logs of system activities	'id'	'user_id'

The database structure listed in Table 1 details the main data storage scheme to support monitoring and audit trail functionalities. The `users` table holds login credentials with a role column to logically distinguish between Admin and Member authorizations. The `tasks` table acts as a central entity that houses the entire supervision history by referencing client, team, and engineer identities relationally. The `sla_configs` and `task_templates` tables function to automate task management, while the `activity_logs` table records system audit trails

automatically to detect user operational activities. Entity relationships in the database are modeled through an Entity Relationship Diagram (ERD) as shown in Figure 3.

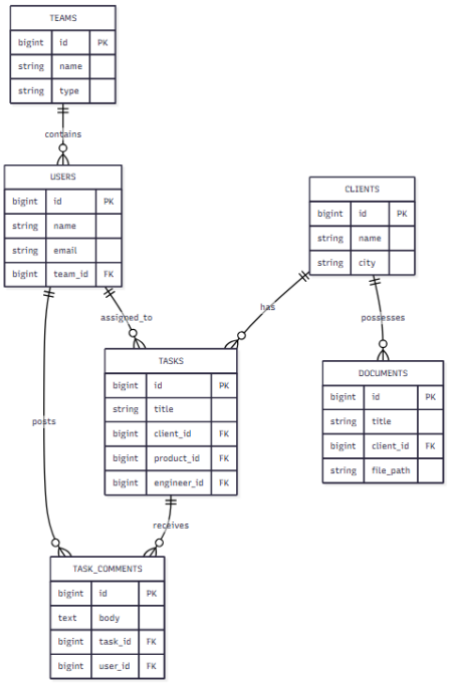

Figure 3: Database Entity Relationship Diagram (ERD)

The entity relationship schema in Figure 3 shows the relational database structure optimized to replace the old system that still used string relations. A one-to-many relationship is applied between the `TEAMS` and `USERS` tables to group users based on their work teams. The `CLIENTS` table has a one-to-many relationship with `TASKS` and `DOCUMENTS` to associate tasks and handover documents with each partner hospital. All assignments in the `TASKS` table strictly refer to `CLIENTS` and `USERS` through ID-based foreign keys (integers), thereby ensuring referential integrity of the database and preventing orphan data when deleting or updating master data.

3. Results and Discussion

The implementation of system functionalities on the Admin Panel begins with the authentication and authorization modules. The login security mechanism is built using Laravel Fortify with the application of bcrypt hashing on user passwords, replacing the previous hardcoded login system. Role authorization is implemented through middleware-based Role-Based Access Control, restricting administrative access rights only to the Admin role, while the Member role is only allowed to manage personal tasks specifically assigned to them [1]. Furthermore, the dashboard analytics module is designed to present a real-time summary of project execution data. The dashboard view is developed with a modern layout (Figure 4) displaying four task status metric cards (Open, In Progress, Revision, Completed), weekly progress charts based on Chart.js, and an overdue task detection panel.

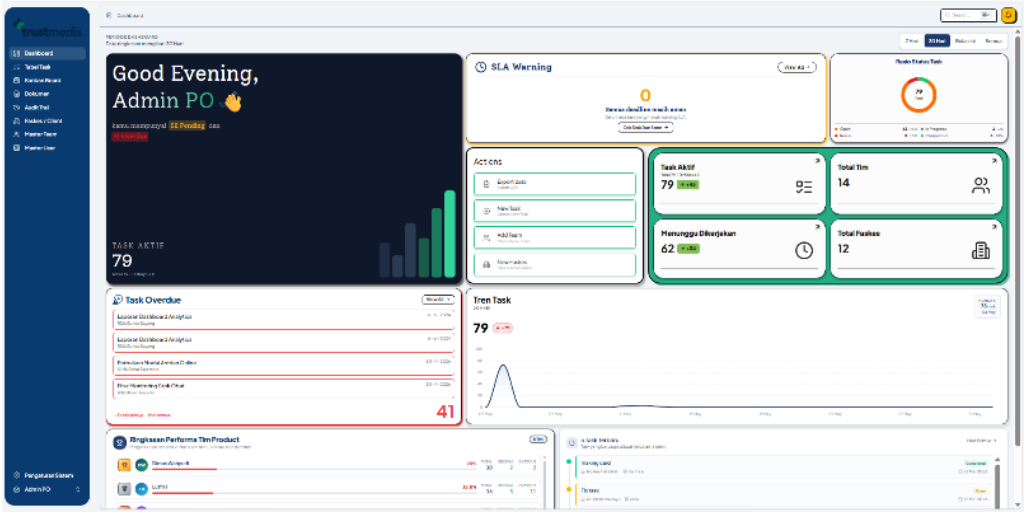

Figure 4: Admin Dashboard View

The dashboard interface visualization in Figure 4 shows the soft-UI layout designed to facilitate the Product Owner team in analyzing operational performance. The metric panel at the top presents the total assignments based on four work status categories, equipped with real-time Service Level Agreement (SLA) compliance percentages. The graph section combines step-line chart visualizations to track

weekly task completion trends and a donut chart to monitor the current task status distribution proportion. The utilization of this integrated visualization is empirically able to speed up the tactical decision-making process for the managerial team in identifying workflow bottlenecks accurately and timely [3], [2]. On the task monitoring page, the interface is designed as a table supported by eight server-side filter parameters such as product team, client, engineer, category, link status, check status, and date range. The implementation of these server-side filters is proven to reduce the client-side computational load when processing large datasets, while the inline status toggle feature allows instant task status updates without page reload [5].

To present a more interactive workflow visualization, the kanban board module is implemented using the vuedraggable library based on SortableJS. Moving task cards between status columns triggers a PATCH request to the server automatically to update the database, supported by optimistic updates to provide instant visual feedback to users [10]. In maintaining the system's accountability aspect, the activity log module automatically records every user data modification in JSON format through the ActivityLogger service class. This audit trail documents user information, action type, and detailed data value changes [4], [9].

In addition, the SLA tracking feature monitors the completion duration of each task category based on deadline rules configured by the administrator. The system dynamically displays visual indicators of SLA compliance in the form of colored badges on the dashboard and task table to maintain developer operational discipline [2], [6]. The Admin Panel is also equipped with other supporting modules such as an integrated Global Search feature (Ctrl+K), in-app notifications, data import from CSV/Excel files, bulk actions, report export to Excel, document versioning of UAT/BAST, collaborative comment threads, recurring task templates, dark mode, and database backup-restore functions.

Testing was conducted using the Black Box Testing method on all main system functionalities. The testing results on critical modules are presented in Table 2.

Table 2: Functional Testing Results

No	Tested Module	Test Scenario	Result
1	Login	Input valid and invalid credentials	Successful
2	CRUD Team	Create, read, update, delete team data	Successful
3	CRUD Client	Create, read, update, delete client data	Successful
4	CRUD Task	Create, read, update, delete, filter tasks	Successful
5	Kanban Board	Drag and drop between status columns	Successful
6	Dashboard	Metric cards and charts display	Successful
7	Import Data	CSV/Excel file upload and validation	Successful
8	SLA Tracking	SLA calculation and indicators	Successful
9	Notifications	Trigger and notification display	Successful
10	Role Access	Access restriction Admin vs Member	Successful
11	Export Excel	Download filtered Excel file	Successful

The system functionality testing documented in Table 2 confirms the operational feasibility of the platform with a 100% pass rate on 11 critical modules. Each test scenario was executed by entering valid and invalid input parameters to observe the system's reaction. The test results show that access authorization (role access) is able to restrict Admin and Member privileges consistently, the automatic calculation process in the SLA tracking module runs accurately, and the data export and import modules successfully synchronize feature execution data directly without any data latency. This external functional evaluation method is effectively able to prove the alignment of system behavior with user requirements expectations [5], [7].

The task monitoring system built in this study successfully resolves the five main problems identified in the old system. First, system security is significantly improved through the implementation of bcrypt hashing, CSRF protection, and prepared statements via Eloquent ORM that prevent SQL injection. Second, role-based access control ensures that each user only accesses data and functionality in accordance with their responsibilities. Third, data integrity is maintained through ID-based foreign key relations across all 14 database tables. Fourth, dashboard analytics provides real-time data visualization that supports evidence-based decision-making. Fifth, digitalization of the monitoring system eliminates the need for manual communication and provides high internal transparency for the development team.

Compared to previous studies that built monitoring systems using native PHP [5] or the Laravel framework with conventional Bootstrap interfaces [4], [7], the system in this study has an advantage in terms of user experience through the SPA architecture that eliminates full page reloads, as well as advanced features like kanban boards with drag and drop, SLA tracking, and UAT/BAST document versioning. The use of Inertia.js as a bridge allows developers to leverage Laravel routing and middleware on the server side while still delivering a responsive SPA experience on the client side, without the complexity of managing a separate API. Digitalizing assignment management is proven to optimize internal coordination efficiency and minimize communication friction between managerial parties and executing staff [8]. The application of data visualization through dynamic dashboards is proven to reduce the complexity of operational reporting while sharpening the accuracy of direct team performance analysis [3], [2]. On the other hand, workflow visualization with Kanban boards is empirically proven to reduce the waiting time for bottleneck detection while strengthening collaboration among team members [10].

4. Conclusion

This research successfully designed and built a web-based task monitoring and project management system using Laravel 12, Vue.js 3, and Inertia.js frameworks at PT X. The developed system is realized in the form of an Admin Panel platform for the internal Product Owner team with two access roles, namely Admin and Member. The main features built include dashboard analytics, a kanban board, role-based

access control, SLA tracking, activity logs, notifications, global search, and data import/export features. Testing using the Black Box Testing method validates that all 11 system functionalities run in accordance with requirement specifications. Future research can develop the system by adding a Client Portal as independent access for partner healthcare facilities to monitor tasks directly, implementing real-time notifications using WebSockets, and applying machine learning for task completion time estimation predictions.

References

[1] M. I. Alifudin, "Analisis dan Perancangan Sistem Informasi Manajemen Tugas Multi-Peran Berbasis Web," *Jurnal Publikasi Ilmu Komputer dan Multimedia*, vol. 4, no. 3, pp. 74–85, 2025, doi: 10.55606/jupikom.v4i3.5228.

[2] D. Andriansyah, "Visualisasi Data Perhitungan SLA Pengiriman Unit Periode Januari-Desember 2021," *Jurnal Teknik Informatika (JTI) STMIK Antar Bangsa*, vol. 8, no. 1, pp. 7–11, 2022, doi: 10.51998/jti.v8i1.471.

[3] C. E. Budiawan and S. Halim, "Perancangan Dashboard Monitoring Contract Lifecycle Management pada PT X," *Jurnal Titra*, vol. 10, no. 2, pp. 57–64, 2022.

[4] D. F. A. Rizki and U. Chotijah, "Perancangan Sistem Monitoring dan Manajemen Proyek Pegawai Berbasis Website dengan Framework Laravel," *Jurnal Informatika dan Teknik Elektro Terapan (JITET)*, vol. 13, no. 3S1, pp. 692–699, 2025, doi: 10.23960/jitet.v13i3S1.7770.

[5] Y. Febriani and A. Hadi, "Perancangan Sistem Informasi Tracking Task Management Berbasis PHP dan MySQL pada PT. Cakrawala Telekomunikasi Indonesia," *ANTIVIRUS: Jurnal Ilmiah Teknik Informatika*, vol. 19, no. 1, pp. 92–103, 2025.

[6] S. Nicolazzo, A. Nocera, and W. Pedrycz, "Service Level Agreements and Security SLA: A Comprehensive Survey," *arXiv preprint arXiv:2405.00009*, 2023.

[7] R. A. Prasetio and Cuhenda, "Perancangan Sistem Monitoring dan Evaluasi Proyek Berbasis Laravel Untuk Optimalisasi Manajemen Proyek PT. Inkaba (Perseroda)," *Jurnal Informatika dan Teknologi Pendidikan*, vol. 5, no. 1, pp. 44–53, 2025, doi: 10.59395/jitp.v5i1.126.

[8] A. B. Pratomo, "Pengembangan Website Manajemen Tugas Berbasis Web Untuk Pengukuran Kinerja Sumber Daya Manusia," *Jurnal TEKNIMEDIA*, vol. 4, no. 2, pp. 139–143, 2023.

[9] Muthmainnah, W. I. Syahputra, and A. Bintoro, "Sistem Informasi Monitoring Kegiatan Berbasis WEB pada Badan Pusat Statistik Kota Lhokseumawe," *SISFO: Jurnal Ilmiah Sistem Informasi*, vol. 9, no. 2, pp. 144–159, 2025.

[10] F. A. Widjaja and B. Hakim, "Sistem Manajemen Proyek Berbasis Website Dengan Metode Kanban," *Computer Science (CO-SCIENCE)*, vol. 5, no. 2, pp. 77–86, 2025, doi: 10.31294/coscience.v5i2.8864.