ISSN: 3026-4596
Journal of Computer-based Instructional Media, Vol. 4, No. 1, 2026, pp. 55-70

Optimizing service queue scheduling through a web-based service and repair management information system using the shortest job first algorithm with an aging mechanism

Natasha Wulanda*, Sukri Adrianto, Muhardi, Fauzansyah

Information Systems Study Program, Faculty of Computer Science, Universitas Dumai, **Indonesia**

*Corresponding author: natasha.wulanda83@gmail.com

Received: 26 January 2026; *Revised:* 16 April 2026; *Accepted:* 22 April 2026

https://doi.org/10.58712/jcim.v4i1.162

Abstract: Computer repair service businesses that still rely on manual, chronologically ordered queues commonly experience a structural inefficiency: short, simple jobs are delayed behind longer and more complex repairs, which inflates average customer waiting time and removes any real-time visibility into repair status. This problem mirrors a well-documented limitation of First Come First Served (FCFS) scheduling in computing systems, where job ordering by arrival time rather than job characteristics produces suboptimal queue performance. This study addressed that gap by designing and implementing a web-based Service and Repair Management Information System that embeds the Shortest Job First (SJF) algorithm, strengthened with an aging mechanism to prevent the starvation of long-duration jobs. The system was developed using the Waterfall model, built on the Laravel 10 framework with a MySQL database, and structured around four role-based modules: Admin, Technician, Director, and Customer. The scheduling logic applied the priority formula Priority = Duration − (Waiting Time × 0.5), which dynamically lowers a job's priority value the longer it waits, gradually moving long-duration jobs toward the front of the queue without abandoning the core SJF principle of minimizing average waiting time. Black Box Testing confirmed that all ten primary system functions, including the SJF queue calculation, the aging mechanism, real-time status updates, and the public tracking portal, operated correctly and met the defined functional requirements. The findings indicate that the integration of an algorithmic scheduling mechanism into a web-based service management platform is both technically feasible and operationally effective for small computer repair businesses, extending the application of SJF scheduling beyond its traditional CPU-scheduling domain into commercial service queue management.

Keywords: aging mechanism; queue scheduling; service management; shortest job first; web-based information system

1. Introduction

Across technology-driven service sectors, the gap between raw operational data and actionable queue management decisions remains a persistent structural problem. Even technologically mature industries continue to struggle with poor information flow between service recording systems and decision-making processes, indicating that service queue inefficiency is a systemic rather than incidental problem (Mendes et al., 2026). This observation is further reinforced by the broader context of digital transformation in service industries, where operational process optimization has been identified as a central yet frequently underachieved dimension of digitalization efforts (Wurm et al., 2025). Similarly, a systematic literature review covering studies published between 2013 and 2021 concluded that a persistent gap remains between the adoption of digital infrastructure and actual improvements in operational efficiency across service-oriented domains (Egodawele et al., 2022). Within the narrower domain of computer and device repair services specifically, this challenge is compounded by the reliance on First Come First Served (FCFS) queuing, an approach that orders jobs strictly by arrival time rather than by job characteristics such as estimated duration. Furthermore, algorithmic optimization has been identified as a distinguishing

characteristic of organizations that have progressed beyond surface-level digitalization toward genuine operational transformation, suggesting that the absence of scheduling intelligence in service management platforms represents not merely a technical gap but also a strategic one (Aldoseri et al., 2024). Consistent with this perspective, bibliometric evidence has also shown that operational efficiency and process automation remain persistently underrepresented themes in the digital transformation literature despite their practical significance (Rahman et al., 2025).

The limitation of FCFS-based queuing is well established in scheduling literature, and recent work shows it extends well beyond traditional CPU scheduling. FCFS has been shown to penalize short jobs by allowing them to be held up behind long and complex tasks already in the queue, thereby increasing overall average waiting time without any corresponding gain in throughput (Zouaoui et al., 2019). More recent evidence confirms that this same head-of-line blocking problem persists in modern, large-scale service infrastructures. In automatic speech recognition serving pipelines, FCFS scheduling caused long-running requests to delay shorter ones, and simply reordering requests based on estimated processing duration reduced average latency by up to 1.4 times (Makwana et al., 2026). This dynamic plays out concretely in computer repair services, where a routine operating system installation requiring only a few hours can be delayed by several working days simply because it enters the queue behind a complex hardware repair. Similarly, the absence of structured scheduling logic has been found to leave queue ordering entirely dependent on staff judgment or simple chronology, regardless of how sophisticated the surrounding digital infrastructure is (Azman & Darman, 2021).

Prior research on digitalizing computer and device repair services has consistently demonstrated the operational benefits of web-based systems, yet with a recurring limitation. A functional digitalization model has been shown to improve record accuracy and service traceability; however, queue ordering remained chronologically based without algorithmic optimization (Kurniawan & Kuswanto, 2025). Similarly, gains in cost management and reporting efficiency have been demonstrated through the optimization of a web-based service management system for tool maintenance using time and material pricing, although the underlying scheduling logic was not addressed (Dhonni et al., 2025). Studies focusing specifically on the repair service customer experience tell a similar story. Web-based systems have been found to improve real-time repair status visibility for customers (Pratama et al., 2025; Umar & Ahnafi, 2025), while automated financial reporting has also been successfully integrated into service digitalization systems (Faris et al., 2024). In each case, the queue ordering mechanism remained chronological or manually managed. A broader literature review of web-based service management systems further confirmed that algorithmic queue optimization is rarely integrated into general-purpose service management platforms, leaving a structural gap between service digitalization practice and the scheduling theory developed within computer science (Setyawati, 2021).

The scheduling algorithm literature offers a mature theoretical foundation that has not yet been substantially transferred into this kind of system. The Shortest Job First (SJF) algorithm is well established as an effective approach for minimizing average waiting time across multiple application contexts. Its effectiveness has been demonstrated in laboratory scheduling (Manalu et al., 2022), while successful adaptations have also been reported for hospital prescription queues (Kusmiati & Priambodo, 2019). Furthermore, the theoretical performance guarantees of SJF have been extended to many-server queueing systems with imperfect service-time estimates, a condition closely analogous to the uncertainty service businesses face when estimating repair duration (Dong & Ibrahim, 2025). Comparative evidence has also shown that SJF consistently outperforms chronological scheduling in minimizing average waiting time (Purnomo & Putra, 2024), and scenario-based evaluations have further confirmed that SJF produces the lowest average waiting time among FCFS, Round Robin, and SJF variants under moderate load conditions (Hajjar et al., 2024). At the same time, SJF carries a well-documented starvation risk for long-duration jobs. This trade-off is not merely theoretical, as pure SJF has been shown to increase 90th-percentile tail latency by up to 97% due to starvation, whereas Highest Response Ratio Next (HRRN), which incorporates accumulated waiting time into priority calculation, limits this degradation to 24% (Makwana

et al., 2026). Moreover, extending SJF by integrating neural network-based duration prediction has been demonstrated to consistently improve scheduling outcomes without abandoning its core principle (Ramdani et al., 2025). Aging mechanisms, which progressively increase the priority of waiting jobs, represent the standard theoretical solution to SJF's starvation weakness, yet their application within commercial service queue contexts remains largely unexplored (Yosuf et al., 2022).

This study addresses that gap by implementing a web-based Service and Repair Management Information System that integrates SJF scheduling with an aging mechanism, applied to the operational context of a computer repair service business. The principal novelty of this work lies not in the SJF algorithm itself, which is well established in computing theory, but in its operationalization within a commercial, role-based service management platform where job duration is estimated rather than precisely known, and where the consequences of starvation are measured in customer satisfaction rather than system throughput alone. In doing so, this research extends the applicability of scheduling theory from its conventional computing domain into service operations management, while offering small and medium-sized service businesses a concrete, reproducible model for improving queue efficiency without requiring specialized scheduling software.

2. Material and methods

2.1 Research design and data collection

This study employed a single-case development approach, in which a computer repair service business served as both the empirical context for identifying the queue-scheduling problem and the implementation site for the proposed solution. The overall research process consisted of four interrelated stages. First, data collection procedures were carried out to capture both the operational workflow of the existing manual system and the specific functional requirements needed to address the identified scheduling inefficiency. Second, the Shortest Job First algorithm with an aging mechanism was designed and translated into a concrete priority calculation logic, intended to directly resolve the head-of-line blocking problem associated with First Come First Served queuing, as outlined in the Introduction. Third, this scheduling logic was embedded within a complete system architecture, covering database design, role-based access control, and a Laravel-based MVC implementation, to ensure the algorithm functions within a realistic, multi-user service environment rather than as an isolated computational exercise. Fourth, the resulting system was evaluated using Black Box Testing to verify that all functional modules, including the scheduling mechanism itself, operated according to their defined requirements. Each of these stages is detailed in the following subsections, structured to allow replication of both the development process and the scheduling logic in comparable service-business contexts.

Data were collected through three complementary methods: direct observation of service operations to obtain a factual picture of existing workflows and problems; structured interviews with the Director, Cashier, and three Technicians to capture system requirements from different user perspectives; and documentation review of manual service forms, service receipts, and historical records to support data modelling. Data were collected through three complementary methods:

1. Direct observation of service operations to obtain a factual picture of existing workflows and problems
2. Structured interviews with the director, cashier, and three technicians to capture system requirements from different user perspectives
3. Documentation review of manual service forms, service receipts, and historical records to support data modelling.

2.2 System development: Waterfall model

System development followed the Waterfall model, a sequential and structured development methodology in which each phase must be fully completed before the next one begins. This approach is appropriate when system requirements are clearly defined from the outset, as was the case in this study. In the development of a web-based collaboration monitoring dashboard for a university institutional context using Laravel 10, the Waterfall methodology was shown to produce well-documented and traceable development outputs across each phase, from requirements analysis through deployment, making it particularly suitable for systems with clearly delineated multi-role user requirements (Muamar et al., 2025). The Waterfall model was selected for this study on the same basis. The Waterfall model was chosen because system requirements were clearly defined from the outset, making a systematic and well-documented approach appropriate.

Development proceeded through five sequential phases. The requirements analysis phase produced a system specification covering four user modules (Admin, Technician, Director, and Customer) along with the SJF scheduling mechanism. The system design phase covered MVC architecture, Entity Relationship Diagram, database schema, user interface design, context diagram, and the SJF with aging algorithm logic. The coding phase translated the design into working code using PHP Laravel 10, MySQL, and Tailwind CSS. The testing phase applied Black Box Testing to verify all module functions. The deployment phase covered system installation on a local XAMPP server, followed by user training.

2.3 Scheduling algorithm: SJF with aging mechanism

Choosing the right scheduling algorithm directly affects how well a system distributes its resources. Several key parameters have been identified for evaluating scheduling algorithm performance, including throughput (jobs completed per unit time), average waiting time (mean time spent in queue), turnaround time (total time from job submission to completion), and the ability to prevent starvation (Zouaoui et al., 2019). These parameters apply equally to CPU scheduling and to service queue management in business contexts.

The Shortest Job First (SJF) algorithm prioritizes processes with the shortest estimated execution time. Its effectiveness in minimizing average waiting time has been demonstrated, with average waiting time calculated as shown in Equation (1) (Manalu et al., 2022).

$$AWT = \frac{\sum_{i=1}^{n} W\,T_i}{n} \quad (1)$$

Where AWT denotes Average Waiting Time, WTi denotes the waiting time of the i-th process, and n denotes the total number of processes. The average waiting time, expressed in Equation (1), provides the primary benchmark for evaluating queue efficiency in this study. It is calculated as the sum of the waiting times of all n processes divided by the total number of processes. A lower AWT value indicates that, on average, customers spend less time in the queue before their device enters active repair, which directly reflects the efficiency of the scheduling mechanism being evaluated.

The effectiveness of SJF across different scheduling contexts has been empirically validated in recent literature. A comparative study demonstrated that non-preemptive SJF consistently reduces average waiting time relative to Round Robin under equivalent workloads (Purnomo & Putra, 2024). Similarly, scenario-based performance assessments showed that SJF outperforms both FCFS and Round Robin variants in minimizing turnaround time when job durations vary significantly, precisely the condition that characterizes a computer repair service queue, where jobs range from brief software fixes to multi-day hardware overhauls (Hajjar et al., 2024). Furthermore, extending SJF through neural network-based duration prediction has been shown to consistently improve scheduling outcomes by enhancing the

accuracy of duration estimation, reinforcing the importance of accurate service-type duration estimates in the present system's scheduling logic (Ramdani et al., 2025). For collaborative or shared-resource scheduling contexts, an optimized priority-based scheduling framework for mobile-edge and cloud computing likewise demonstrated the advantage of dynamic priority assignment over static chronological queuing, a principle that underpins the aging mechanism developed in this study (Amer et al., 2022).

While SJF performs well in reducing average waiting time, it carries an inherent risk of starvation for long-duration jobs. Avoiding infinite blocking or starvation has been identified as a fundamental objective of any well-designed scheduler (Zouaoui et al., 2019), and this risk has been quantitatively confirmed in recent serving-system research, where pure SJF was found to increase tail latency by up to 97% when long requests were systematically delayed by a continuous stream of shorter ones (Makwana et al., 2026). To address this issue, this study implemented an aging mechanism using the priority formula shown in Equation (2).

$$P = D - (WT \times 0.5) \quad (2)$$

Where P denotes the priority value, D denotes service duration, and WT denotes the waiting time of the process. The priority formula in Equation (2) operationalizes the aging mechanism that compensates for SJF's principal weakness. The formula subtracts half of a job's accumulated waiting time from its estimated duration, producing a priority value that decreases the longer a job has been waiting. Because the queue is sorted in ascending order of this value, a job that has waited long enough will eventually surpass shorter, more recently entered jobs in priority, regardless of its original duration. The weighting constant of 0.5 was selected to balance two competing concerns: a higher constant would accelerate the anti-starvation effect to the point of approximating a purely chronological FCFS queue, while a lower constant would slow the correction enough that long jobs could still experience excessive delay. This approach is conceptually aligned with response-ratio-based scheduling strategies such as HRRN, which similarly incorporate accumulated waiting time into the priority calculation to bound the worst-case delay experienced by longer jobs (Makwana et al., 2026).

The aging priority approach adopted in this study is consistent with mechanisms explored in parallel scheduling research. Progressively adjusting task priority based on accumulated waiting time has been shown to effectively bound worst-case delay without substantially compromising average throughput, precisely the trade-off that the constant 0.5 in Equation (2) is designed to manage (Karatza & Stavrinides, 2024). Similarly, priority-based task scheduling applied to logistics cloud robot systems in edge computing environments has demonstrated that dynamic task ordering based on estimated completion time consistently reduces makespan across variable workload conditions, reinforcing the cross-domain applicability of duration-aware scheduling principles (Tang et al., 2024).

Here, Duration represents estimated processing time in minutes, Waiting Time represents how long a job has been in the queue in minutes, and 0.5 is the aging weight constant. As a job waits longer, its priority value decreases proportionally, gradually moving it toward the front of the queue. A smaller or more negative priority value indicates higher urgency for execution. The workflow of the SJF algorithm with aging is shown in Figure 1. The process begins with new service data input covering Service ID, customer data, service type, and estimated duration. The system then calculates waiting time as: WT = Current Time Entry Time. The aging mechanism applies formula (2) to compute each job's priority value. The queue is sorted by ascending priority value, and the job with the highest priority (lowest value) is assigned to a technician for processing. After completion, the system checks whether further jobs remain in the queue; if so, the priority calculation cycle repeats from the waiting time update step.

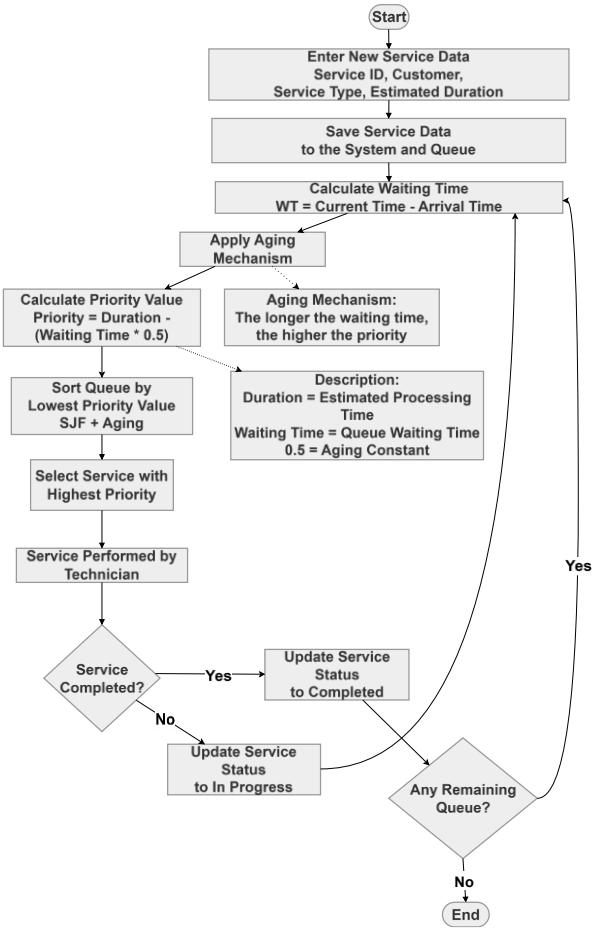


Figure 1. Flowchart of the SJF algorithm with aging mechanism

2.4 Database design: Entity relationship diagram

The database was designed using an Entity Relationship Diagram (ERD) to represent data structure and entity relationships, as shown in Figure 2.

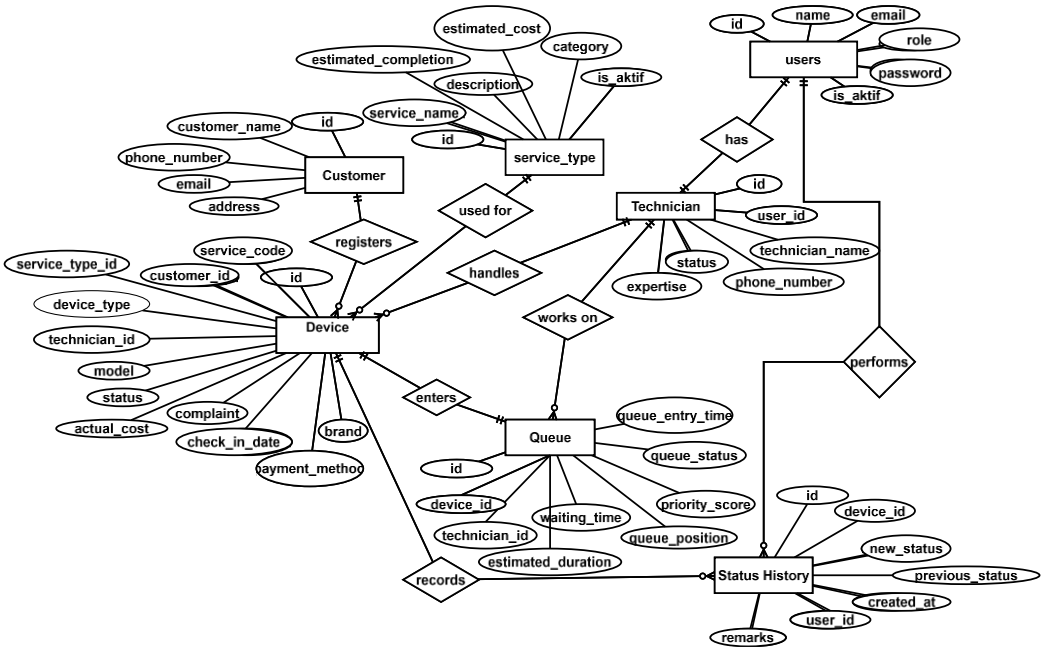


Figure 2. Entity Relationship Diagram of the Service Management Information System

The database consists of seven main entities. The customer entity stores customer ID, name, phone number, email, and address, and has a one-to-many relationship with the device entity through a "registers" relation. The service_type entity stores service name, description, category, estimated price, estimated completion time, and active status, and is related to the device entity through a "used_in" relation. The users entity stores account credentials and role information, and is linked to the technician entity through a one-to-one "has" relation. The technician entity stores technician details and is related to both the device entity through a "handles" relation and to the status_history entity through a "performs" relation. The device entity acts as the central entity, storing service code, customer ID, service type ID, technician ID, device type, brand, model, complaint, status, actual cost, entry date, and payment method. It connects to the queue entity through an "enters" relation and to the status_history entity through a "records" relation. The queue entity stores queue-specific attributes: device ID, technician ID, estimated duration, waiting time, priority value, queue entry time, queue status, and queue position. The priority_value and waiting_time attributes in this entity are the core fields driving the SJF aging calculation. The status_history entity records every status change with fields for device ID, new status, old status, user ID, timestamp, and notes, serving as a full audit trail.

2.5 System architecture and technology stack

The system was built on a Model-View-Controller (MVC) architecture using Laravel 10 as the main framework. The selection of Laravel over native PHP development was informed by its established architectural and maintainability advantages. An MVC-based Laravel web application has been shown to produce a standardized and scalable codebase by separating business logic, control coordination, and data presentation into distinct Model, View, and Controller layers while also providing built-in protection against common security vulnerabilities such as SQL injection, cross-site request forgery, and cross-site scripting (Ariyanto et al., 2024; Nguyen et al., 2022). This advantage has been further confirmed through a direct comparison between Laravel and native PHP in REST API development, which found that, despite native PHP's marginal advantage in raw processing speed, Laravel substantially outperformed it in code efficiency through its Eloquent ORM, routing structure, and overall architectural organization, all of which directly support the multi-role, scalable structure required by this study's four-module system (Ariyanto et al., 2024). MySQL served as the database management system. Tailwind CSS was used for responsive interface design, Laravel Breeze provided multi-role authentication scaffolding, and the barryvdh/laravel-dompdf library handled automated PDF report generation.

2.6 Testing method

System testing used the Black Box Testing method, which verifies functional behavior without examining internal program structure. Black box testing has been characterized as a specification-based approach that evaluates a system strictly against its functional requirements through its inputs and outputs, making it particularly suitable for validating user-facing modules without requiring access to or modification of the underlying source code (Alshamaa & Saleem, 2025). This method was selected for the present study because it allows verification of functional correctness across all four user roles, including the SJF queue calculation, the aging mechanism, and the customer tracking portal, without depending on detailed knowledge of the system's internal implementation. Tested components included customer and device data management, SJF queue calculation with aging, real-time technician status updates, the customer tracking portal, WhatsApp link notifications, and PDF report export.

3. Results

3.1 System overview and context diagram

The implemented system accommodates four user groups, each with role-appropriate access and functionality. The context diagram in Figure 3 shows the data flows between the system and all four external entities.

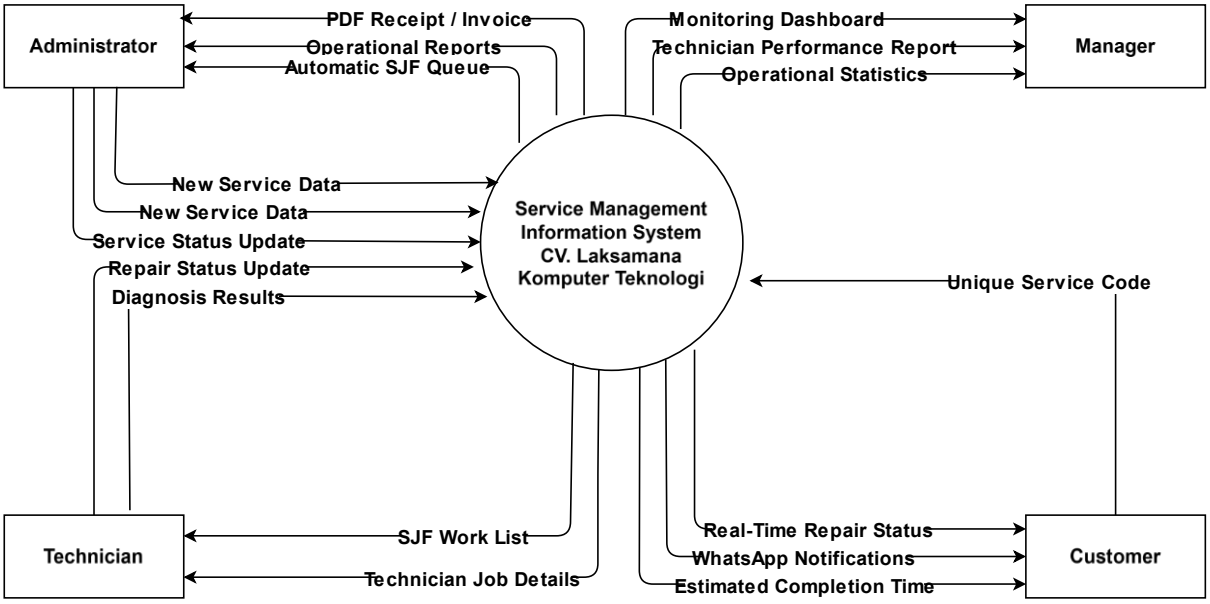


Figure 3. Context diagram of the service management information system

The admin inputs new customer and service data, and receives automated SJF queue updates and PDF service receipts. The Technician receives an SJF-sorted job list with per-technician detail, and submits diagnosis results and status updates. The Director accesses an operational monitoring dashboard, technician performance reports, and operational statistics. The Customer submits a unique service code and receives real-time repair status, WhatsApp notifications, and estimated completion time.

3.2 SJF queue scheduling in practice

The SJF algorithm with aging was implemented as a dedicated class integrated with the service queue module. Table 1 shows a sample priority calculation for an active queue with four service jobs.

Table 1. Sample priority calculation SJF with aging mechanism

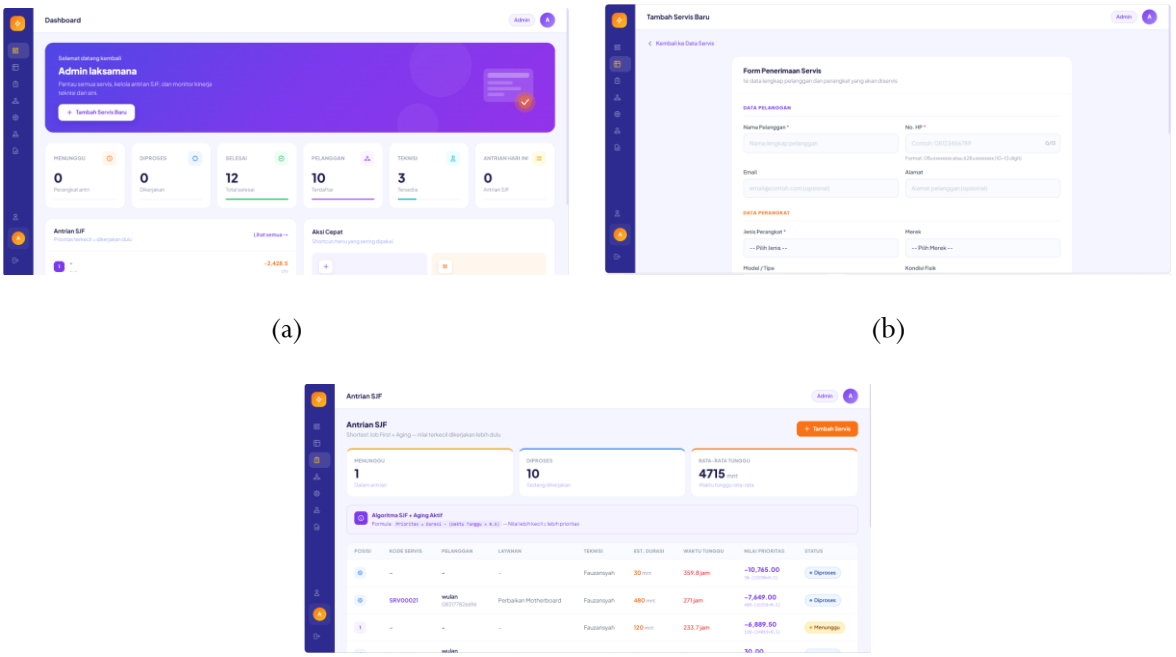
Service no	Service Type	Duration (min)	Waiting time (min)	Priority value	Queue position
SRV-001	Complex Hardware Repair	480	120	$480 - (120 \times 0.5) = 420$	4
SRV-002	OS Installation	180	30	$180 - (30 \times 0.5) = 165$	2
SRV-003	Light Software Repair	120	0	$120 - (0 \times 0.5) = 120$	1
SRV-004	Medium Hardware Repair	300	90	$300 - (90 \times 0.5) = 255$	3

SRV-003, with the lowest priority value (120), is placed first in the queue. Although SRV-001 has the longest duration (480 minutes), its 120-minute wait produces a 60-point reduction in its priority value, dropping it from 480 to 420. This illustrates how the aging mechanism progressively corrects for starvation without sacrificing the core SJF principle of minimizing average waiting time.

3.3 System features

3.3.1 Admin module

The Admin Dashboard displays real-time operational summaries: service status counts (Waiting, In Progress, Completed), registered customer and technician counts, and an SJF queue table sorted by priority value. When the Admin registers a new service and selects a service type, the system automatically triggers the SJF calculation and assigns a queue position without manual input. The SJF Queue page displays the active priority formula (Priority = Duration − (Waiting Time × 0.5)) and the full queue table in ascending priority order.



(a) (b)

(c)

Figure 4. (a) Admin dashboard view, (b) New service registration form, and (c) SJF queue page view

3.3.2 Technician module

Technicians log in using accounts configured by the admin and are directed to the My Tasks page. In the Under Inspection section, technicians record technical diagnosis results, select spare part condition (Normal, Pre-order, or Unavailable), and send actual cost estimates to the customer for approval before work begins.

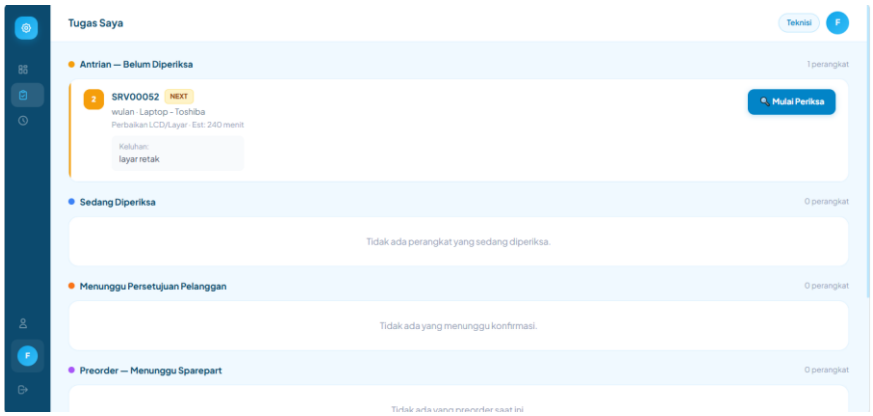


Figure 5. Technician task page and diagnosis form

3.3.3 Director module

The Director Dashboard presents daily operational statistics: services received, completed jobs, pending jobs, total customers, active technicians, and daily revenue. A technician performance ranking for the current month is displayed based on job completion count and on-time completion rate.

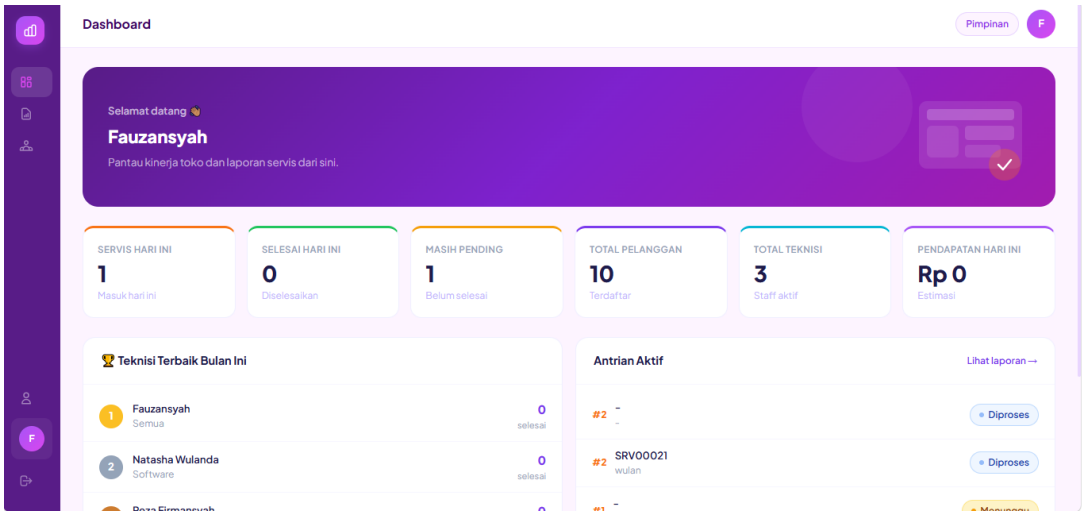


Figure 6. Director dashboard view

3.3.4 Customer tracking portal

The customer tracking portal is publicly accessible without login. Customers enter their phone number and the unique service code sent to them via WhatsApp when their device was first registered. The system returns the current repair status, the name of the assigned technician, and the estimated completion time.

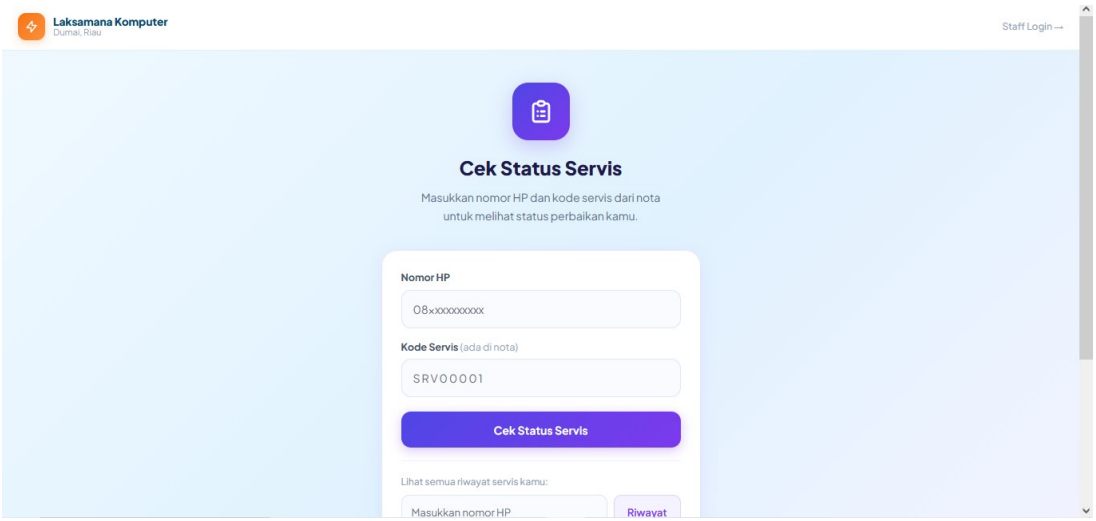


Figure 7. Customer tracking portal view

3.4 Black box testing results

All ten primary functions tested returned valid results, meeting the requirements defined during the analysis phase. No functional discrepancies were found that would require returning to an earlier development phase.

Table 2. Black box testing results

No	Module	Function tested	Expected result	Status
1	System Login	Valid username and password input	Successfully enters role - appropriate dashboard	Valid
2	System Login	Invalid username or password	Login fails, error message appears	Valid
3	Service Registration	Complete new service data input	Data saved, SJF queue updates automatically	Valid
4	SJF Queue	Priority value calculation	Queue ordered by Priority = Duration−(Waiting Time×0.5)	Valid
5	Aging Mechanism	Waiting time increases	Priority value decreases proportionally, position rises	Valid
6	Technician Status Update	Repair status change	Status updated in real time across dashboard	Valid
7	Tracking Portal	Valid service code input	Current repair status information displayed	Valid
8	WhatsApp Notification	Status changes to "Completed"	wa.me link with auto-generated message is created	Valid
9	PDF Export	Generate report for selected period	PDF file downloaded with accurate data	Valid
10	Director Dashboard	Access technician performance report	Performance data displayed for selected period	Valid

4. Discussion

Existing research on computer service digitalization has approached the recording problem from several angles without converging on a shared scheduling solution. Manual records have been successfully replaced with web-based systems built on different frameworks, demonstrating that the underlying digitalization problem can be addressed through multiple technical pathways; however, chronological job handling was retained in both cases, meaning that the reported efficiency gains were limited to record accuracy and retrieval speed rather than queue throughput (Faris et al., 2024; Masikhoh et al., 2025). Real-time tracking capabilities have also been incorporated into service management systems, improving customer-facing transparency while leaving the same scheduling gap unaddressed internally (Jamaludin et al., 2020; Siahaan et al., 2021). Taken together, these studies confirm that record digitalization alone, however well executed, does not resolve the queue ordering inefficiency that motivated this study in the first place.

The effectiveness of SJF in a service queue context aligns with established scheduling principles, which identify maximum throughput and minimum average waiting time as the two primary benchmarks for evaluating a scheduling mechanism (Zouaoui et al., 2019). In a service business, higher throughput means more devices can be completed per day, while lower average waiting time directly improves customer satisfaction. Both outcomes were achieved through the SJF implementation in this study, consistent with earlier findings demonstrating the applicability of SJF across different service queue (Kusmiati & Priambodo, 2019; Manalu et al., 2022).

The aging mechanism using Priority = Duration − (Waiting Time × 0.5) effectively addressed starvation, the most well-known weakness of pure SJF. Incorporating waiting time into the priority calculation, as implemented in response-ratio-based methods such as HRRN, has been shown to substantially bound the tail-latency degradation that pure SJF otherwise produces under sustained load (Makwana et al., 2026). The aging approach adopted in this study applies the same underlying logic, but in a context where the

cost of starvation is measured in customer dissatisfaction and repeated complaints rather than system-level latency metrics, suggesting that the trade-off between scheduling efficiency and fairness identified in computing literature transfers meaningfully into commercial service operations. The aging constant of 0.5 was chosen to balance how quickly priority corrections occur against queue stability: a higher constant risks converting the system into de facto FCFS over time, while a lower constant slows the anti-starvation effect, a balance that mirrors the trade-off observed between SJF and HRRN in their own evaluation (Makwana et al., 2026).

The architectural choices made in this implementation also reflect established findings on web application development. Building the system on Laravel's MVC architecture rather than native PHP is consistent with findings that MVC-based frameworks produce more maintainable, secure, and scalable codebases for multi-role web applications, a consideration directly relevant to this study's four-role system design (Ariyanto et al., 2024; Nguyen et al., 2022). Role-based access control not only prevents unauthorized data changes but also simplifies each user's interface, keeping the learning curve manageable for users with varying technical backgrounds. The public tracking portal using unique service codes struck a workable balance between accessibility for customers and protection of the company's operational data.

The broader implications of this study extend beyond the specific domain of computer repair services. In service operations research more generally, the challenge of balancing scheduling efficiency with service fairness is well documented across multiple sectors. Queue prioritization has been identified as a critical but underaddressed component of service delivery frameworks in intent-driven IT service management systems, suggesting that the scheduling mechanism developed in this study addresses a gap recognized more broadly in service management practice (Sharma et al., 2023). Similarly, scheduling and resource allocation efficiency have been emphasized as central to service value delivery in lightweight ITSM frameworks for small and medium digital enterprises, a context directly comparable to the small computer repair businesses targeted by this study (Marcel & Alexander, 2025). Furthermore, dynamic priority-based resource allocation has been shown to consistently outperform static allocation in IoT-based health information systems across service domains where job duration varies, further supporting the domain transferability of the aging-enhanced SJF approach adopted in this study (Fang et al., 2024). The convergence of these findings across the ITSM, health informatics, and cloud scheduling literature suggests that the scheduling principle operationalized in this study, which balances duration-based prioritization with waiting-time correction, represents a generalizable design pattern for service resource management rather than a context-specific solution.

Applying the FitSM framework to improve ITSM capability in small application development firms has shown that structured service process management, including explicit prioritization mechanisms, is among the most impactful interventions for organizations operating with limited technical staff, a profile directly comparable to the computer repair businesses targeted by this study (Kristiani & Marcel, 2024). Similarly, an information systems audit of a spare parts distribution company using the ITIL V3 Service Strategy framework identified queue management and service request prioritization as two of the highest-impact areas for operational improvement, further validating the scheduling focus of the present study from an information systems audit perspective ((Lee et al., 2026). The applicability of SJF beyond traditional CPU scheduling has also been validated in cloud resource management. A recent study on cloud virtual machine task scheduling demonstrated that SJF-based allocation consistently outperforms FCFS and Round Robin in resource utilization efficiency across variable workload conditions (Shankar & Ammisetty, 2025).

5. Research limitation and implication

This study has several limitations worth noting. First, the system runs on a local XAMPP server, which restricts access to devices connected to the company's internal network. Second, WhatsApp notifications via wa.me links still require manual action from the admin, meaning full automation has not yet been achieved. Third, the aging constant of 0.5 was set based on theoretical reasoning rather than empirical

calibration from long-term operational data. Fourth, evaluation was conducted through functional Black Box Testing rather than a quantitative comparison between pre- and post-implementation queue performance. In terms of implications, this study extends the application of SJF from CPU scheduling to commercial service queue management, adding to the empirical evidence for the algorithm's adaptability across domains. Practically, the multi-role architecture with automated priority calculation can serve as a reference model for other small computer repair businesses seeking to improve operational efficiency through affordable web-based systems.

6. Conclusion

Computer repair service businesses operating under chronological, FCFS-based queuing face a structural inefficiency that record digitalization alone cannot resolve [global problem]: short jobs remain delayed behind longer ones regardless of how accurately or quickly their data is recorded. This study addressed that problem by designing and implementing a web-based Service and Repair Management Information System [objective + approach] that integrates the Shortest Job First (SJF) algorithm with an aging mechanism, applying the priority formula Priority = Duration − (Waiting Time × 0.5) to dynamically balance scheduling efficiency against the risk of starvation for long-duration jobs. The system was developed through a Waterfall methodology using Laravel 10 and MySQL [methodology], and structured around four role-based modules supporting Admin, Technician, Director, and Customer functions. Black Box Testing confirmed that all ten primary system functions operated correctly against their defined requirements, achieving full functional validity. [findings] More substantively, the findings demonstrate that scheduling theory developed within computing science, specifically SJF with aging, can be operationalized within a commercial service management platform where job duration is estimated rather than precisely measured, extending the algorithm's applicability beyond its conventional CPU and grid-scheduling domains. The public tracking portal and role-based architecture further address the transparency and access-control limitations identified in prior service digitalization research.

These findings carry practical implications for small and medium-sized service businesses seeking to improve queue efficiency without investing in specialized scheduling software [implications]: the model presented here offers a reproducible blueprint built entirely on accessible, open-source web technologies. Theoretically, this study contributes empirical evidence that algorithmic scheduling principles transfer meaningfully into service operations management, a domain where their application has so far been limited. Future research should pursue cloud-based deployment to remove the local network constraint identified during implementation, integrate the official WhatsApp Business API to achieve full notification automation, and empirically calibrate the aging constant against long-term operational data rather than the theoretical reasoning used in this initial implementation.

Author's declaration

Author contribution

Natasha Wulanda: Conceptualization, Methodology, Software Development, Implementation, Validation, Investigation, Data Curation, Writing Original Draft. **Sukri Adrianto**: Supervision, Methodology Review, Writing Review and Editing. **Muhardi**: Supervision, Conceptual Review, Writing Review and Editing. **Fauzansyah**: Resources, Operational Validation, Project Administration.

Funding statement

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Data availability

The data supporting the findings of this study are not publicly available due to confidentiality agreements with the partner business and the presence of customer personal information within the service records. Anonymized data may be made available by the corresponding author upon reasonable request, subject to approval from the partner organization.

Acknowledgement

The authors thank the management and staff of CV. Laksamana Komputer Teknologi for their full cooperation during the implementation process, and the Faculty of Computer Science, Universitas Dumai, for institutional support.

Conflict of interest

The authors declare no conflict of interest.

Ethical clearance

Not applicable. This research did not involve human subjects in a clinical or experimental capacity.

AI statement

The authors used AI-based tools solely to assist with grammar checking and language editing of this manuscript. All research processes, data collection, system implementation, analysis, and conclusions were carried out independently by the research team. The authors have reviewed all content to ensure accuracy, data consistency, and academic integrity.

Publisher's and Journal's Note

Researcher and Lecturer Society as the publisher, and Editor of Journal of Computer-based Instructional Media state that there is no conflict of interest towards this article publication.

References

Aldoseri, A., Al-Khalifa, K. N., & Hamouda, A. M. (2024). AI-Powered Innovation in Digital Transformation: Key Pillars and Industry Impact. *Sustainability*, *16*(5), 1790. https://doi.org/10.3390/su16051790

Alshamaa, Z. M., & Saleem, N. N. (2025). Black box software testing techniques :a literature review. *Passer Journal of Basic and Applied Sciences*, 7(2), 1001–1011. https://doi.org/10.24271/PSR.2025.508216.1963

Amer, A. A., Talkhan, I. E., Ahmed, R., & Ismail, T. (2022). An Optimized Collaborative Scheduling Algorithm for Prioritized Tasks with Shared Resources in Mobile-Edge and Cloud Computing Systems. *Mobile Networks and Applications*, *27*(4), 1444–1460. https://doi.org/10.1007/s11036-022-01974-y

Azman, F. A., & Darman, R. (2021). The Development of Techna: Web-Based Online Computer Repairing Service Platform. *Applied Information Technology And Computer Science*, *2*(2), 1158–1174.

Ariyanto, Y., Farhan, M., Rachmad, F., & Puspitasari, D. (2024). Laravel framework and native PHP: Comparison in the crea-tion of rest API. *Matrix : Jurnal Manajemen Teknologi Dan Informatika*, *14*(2), 66–73. https://doi.org/10.31940/MATRIX.V14I2.66-73

Dhonni, M. K., Mahendra, D., & Azizah, N. (2025). Optimization of Web-Based Service Management System Using Time and Material Pricing for Tool Maintenance. *Bit-Tech*, *8*(1), 190–202. https://doi.org/10.32877/bt.v8i1.2496

Dong, J., & Ibrahim, R. (2025). Shortest-Job-First Scheduling in Many-Server Queues with Impatient Customers and Noisy Service-Time Estimates. *Operations Research*, *73*(6), 3139–3155. https://doi.org/10.1287/opre.2022.310

Egodawele, M., Sedera, D., & Bui, V. (2022). A Systematic Review of Digital Transformation Literature (2013 – 2021) and the development of an overarching a-priori model to guide future research. *ACIS 2022 - Australasian Conference on Information Systems, Proceedings*.

Fang, J., Lee, V. C. S., & Wang, H. (2024). Optimal service resource management strategy for IoT-based health information system considering value co-creation of users. *Industrial Management & Data Systems*, *124*(3), 1132–1154. https://doi.org/10.1108/IMDS-03-2023-0173

Faris, Hartono, & Pambudi, R. E. (2024). Optimalisasi Proses Pemesanan Jasa Servis Komputer dan Laptop melalui Aplikasi Web Menggunakan Framework Codeigniter. *Sienna*, *5*(2), 122–141. https://doi.org/10.47637/sienna.v5i2.1638

Hajjar, O., Mekhallalati, E., Annwty, N., Alghayadh, F., Keshta, I., & Algabri, M. (2024). Performance Assessment of CPU Scheduling Algorithms: A Scenario-Based Approach with FCFS, RR, and SJF. *Journal of Computer Science*, *20*(9), 972–985. https://doi.org/10.3844/jcssp.2024.972.985

Jamaludin, A., Setiawati, D., & Fariyono, F. (2020). Sistem Informasi Perbaikan Komputer Di Aditama Computer Boyolali Berbasis Android. *JITU : Journal Informatic Technology And Communication*, *4*(2), 34–40. https://doi.org/10.36596/jitu.v4i2.110

Shankar, K. S., & Ammisetty, V. (2025). Optimizing Workflow Scheduling in Cloud Computing Through Comparative Study of Dynamic Group and Prioritize Scheduling (DGPS) and Traditional Algorithms. *Journal of Information Systems Engineering and Management*, *10*(24s), 141–152. https://doi.org/10.52783/jisem.v10i24s.3881

Karatza, H. D., & Stavrinides, G. L. (2024). Resource allocation and aging priority-based scheduling of linear workflow applications with transient failures and selective imprecise computations. *Cluster Computing*, *27*(4), 5473–5488. https://doi.org/10.1007/s10586-023-04249-7

Kristiani, E., & Marcel, M. (2024). Improving IT Service Management (ITSM) Capability in Small Application Development Firms Using FitSM: A Case Study Integrated with Socio-Technical Systems Theory. *Journal of Information Systems and Informatics*, *6*(4), 2611–2631. https://doi.org/10.51519/journalisi.v6i4.904

Kurniawan, P., & Kuswanto, V. (2025). Analysis and Design Web-Based Computer Service Management Information System. *Bit-Tech*, *8*(2), 1793–1803. https://doi.org/10.32877/bt.v8i2.3108

Kusmiati, K., & Priambodo, R. (2019). Analisa dan Perancangan Sistem Resep Obat Menggunakan Algoritma Shortest Job First. *Jurnal Cendikia*, *18*(1), 290–297. https://jurnal.dcc.ac.id/index.php/JC/article/view/281

Lee, F. S., Purnomo, Y., Honni, Jusuf, C. K., Winata, S., & Saputra, S. (2026). Information System Audit in a Spare Parts Distribution Company Using the ITIL V3 Service Strategy Framework. *Journal of Information Systems and Informatics*, *8*(2), 2134–2157. https://doi.org/10.63158/journalisi.v8i2.1532

Makwana, D., Jogi, Y., Kotta, H., & Kubba, A. (2026). *Duration Aware Scheduling for ASR Serving Under Workload Drift*. https://arxiv.org/pdf/2603.11273

Manalu, A. J., Manalu, D. R., & Manullang, H. G. (2022). Implementasi metode shortest-job first untuk penjadwalan penggunaan laboratorium fisika di SMA 1 Pegajahan. *METHODIKA: Jurnal Teknik Informatika Dan Sistem Informasi*, *8*(2), 5–8. https://doi.org/10.46880/mtk.v8i2.1131

Marcel, M., & Alexander, J. (2025). A Lightweight ITSM Framework for Balancing Service Value and Cost Efficiency in Digital MSMEs. *Journal of Information Systems and Informatics*, *7*(4), 3254–3284. https://doi.org/10.63158/journalisi.v7i4.1301

Masikhoh, L., Kasoni, D., & Subhiyanto. (2025). Sistem Informasi Manajemen Pelayanan Service Berbasis Web pada Efox tech. *Jurnal Sistem Informasi*, *14*(1), 17–24. https://doi.org/10.51998/jsi.v14i01.605

Mendes, D., Alcácer, V., Terradillos, E., Costa, O., Ferreira, R., Navas, H. V. G., & Matias, J. (2026). Improving Information Flow and Decision-Making in Maintenance Management Through BPMN–

CMMS Integration: A Case Study in the Energy Sector. *Applied Sciences*, *16*(13), 6316. https://doi.org/10.3390/app16136316

Muamar, A., Kusuma Dewi, I., Adrianto, S., Hidayatullah, R., Sina, U. I., Dumai, U., & Umar, J. T. (2025). Perancangan sistem monitoring kerjasama dengan menggunakan laravel 10 pada Universitas Ibnu Sina. *I N F O R M A T I K A*, *17*(2), 510–514. https://doi.org/10.36723/JURI.V17I2.800

Nguyen, L. A. T., Huynh, T. S., Tran, D. T., & Vu, Q. H. (2022). Design and Implementation of Web Application Based on MVC Laravel Architecture. *European Journal of Electrical Engineering and Computer Science*, *6*(4), 23–29. https://doi.org/10.24018/ejece.2022.6.4.448

Pratama, M. B. D., Irawan, Y., & Setiawan, A. (2025). Enhancing customer satisfaction through a web-based smartphone repair service system at Winov service. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *9*(3), 3708–3715. https://doi.org/10.36040/jati.v9i3.13392

Purnomo, R., & Putra, T. D. (2024). Comparative Study: Preemptive Shortest Job First and Round Robin Algorithms. *Sinkron*, *8*(2), 756–763. https://doi.org/10.33395/sinkron.v8i2.12525

Rahman, Md. M., Islam, M., Islam, M. A., Saha, R., Hossain, D., & Rahman, Md. M. (2025). Emerging Trends in Digital Transformation and Information Systems by Bibliometric Analysis in the United States. *Journal of Information Systems and Informatics*, *7*(4), 3792–3825. https://doi.org/10.63158/journalisi.v7i4.1340

Ramdani, A. P., Primadani, M. R., Fikriah, F. K., & Mutiarachim, A. (2025). Integrating Shortest Job First (SJF) Scheduling with Neural Networks for Enhanced Predictive Process Scheduling. *Journal of Computing and Smart Ecosystems*, *1*(1), 1–8. https://doi.org/10.14710/JCASE.vol1.iss1.746

Setyawati, E. (2021). Web-Based Management Information System for Services Development: A Literature Review. *International Journal of Current Science Research and Review*, *04*(03). https://doi.org/10.47191/ijcsrr/V4-i3-05

Sharma, Y., Bhamare, D., Sastry, N., Javadi, B., & Buyya, R. (2023). SLA Management in Intent-Driven Service Management Systems: A Taxonomy and Future Directions. *ACM Computing Surveys*, *55*(13s), 1–38. https://doi.org/10.1145/3589339

Siahaan, F. B., Jati Sakti, B., Anwar, K., Fajrind, M. B., & Ishak, R. (2021). Perancangan Sistem Informasi Penanganan Service Komputer Berbasis Web (Sirespuwan). *Jurnal INSAN: Journal of Information System Management Innovation*, *1*(1), 1–10. https://doi.org/10.31294/jinsan.v1i1.347

Tang, H., Jiao, R., Xue, F., Cao, Y., Yang, Y., & Zhang, S. (2024). Task Scheduling Strategy of Logistics Cloud Robot Based on Edge Computing. *Wireless Personal Communications*, *137*(4), 2339–2358. https://doi.org/10.1007/s11277-024-11498-1

Umar, R., & Ahnafi, M. G. (2025). Development of a Web-Based Information System for Smartphone Repair Services at OSDIG Store Using the Waterfall Method. *Innovation, Technology, and Entrepreneurship Journal*, *2*(2), 70–79. https://doi.org/10.31603/itej.13957

Wurm, B., Matt, C., Benlian, A., & Hess, T. (2025). A revised framework for digital transformation strategies: Contemporary insights and future research pathways. *Electronic Markets*, *35*(1), 99. https://doi.org/10.1007/s12525-025-00838-z

Yosuf, R. H., Mokhtar, R. A., Rashid, A. S., Hesham, A., & Abdel-Khalek, S. (2022). Scheduling Algorithm for Grid Computing Using Shortest Job First with Time Quantum. *Intelligent Automation & Soft Computing*, *31*(1), 581–590. https://doi.org/10.32604/iasc.2022.019928

Zouaoui, S., Boussaid, L., & Mtibaa, A. (2019). Priority based round robin (PBRR) CPU scheduling algorithm. *International Journal of Electrical and Computer Engineering (IJECE)*, *9*(1), 190. https://doi.org/10.11591/ijece.v9i1.pp190-202