
DESIGN AND IMPLEMENTATION OF AN AUTOMATED QUEUE MANAGEMENT SYSTEM

Fatima Shehu Tofa¹, Nura Muhammad Baba^{1, 2}, Muhammad Sani Rabi¹,
Salisu Yusuf Muhammad^{1,3}

¹Department of Computer Science, Faculty of Science and Computing

¹Baba-Ahmed University, Kano, Nigeria

Email: ¹fatima.tofa@babaahmeduniversity.edu.ng, ²nabnurtrn@gmail.com,

³sykabara@gmail.com

Abstract

Managing queues remains a common service-delivery problem, particularly when the number of people requesting services temporarily exceeds the available service capacity. Conventional physical lines and manual ticketing may result in prolonged waiting, congestion, uncertainty regarding queue position, inconsistent service allocation, and inadequate administrative records. This study presents the design and implementation of a web-based Automated Queue Management System (AQMS) developed to automate user registration, sequential queue-number assignment, queue-status monitoring, and administrative reporting. The system adopts a three-tier architecture consisting of presentation, application, and database layers. It was implemented using HTML5, CSS3, JavaScript, Bootstrap, jQuery, PHP 8.1.12, Apache 2.4.54 through XAMPP, and MySQL Server 8.0. User, service, queue, and reporting information is maintained in the database, while authentication, ticket generation, monitoring, and reporting are handled by the application layer. Functional testing covered registration, login, queue-number generation, queue monitoring, and report generation, and all five documented test cases were successfully passed, representing a 100% pass rate. However, the project did not include measured before-and-after waiting-time observations, server response-time benchmarks, or large-scale concurrency experiments. Consequently, this study does not claim a numerical percentage reduction in waiting time. The prototype nevertheless demonstrates the practical feasibility of replacing manual queue procedures with a structured, traceable, and data-oriented web-based workflow (Maister, 1985; Katz et al., 1991).

Keywords: Web-Based System; Queue Management; Client-Server Architecture; Real-Time Updates; Service Management.

I. INTRODUCTION

1.1 Background

Waiting lines emerge when the demand for a service is greater than the capacity immediately available to provide that service. Within universities and other service-oriented institutions, users frequently interact with units such as registries, bursary departments, admission offices, student affairs, ICT support centers, and departmental offices. In many of these settings, queues are still managed through physical lines, paper-based tickets, or verbal instructions. Such practices can make it difficult for users to determine their position in a queue and may also limit the availability of dependable records for service analysis. Queuing theory offers a formal framework for examining these situations through concepts such as arrival rates, service rates, service channels, and queue disciplines (Taha, 2017; Gross et al., 2018).

The growing adoption of digital service delivery provides an opportunity to transform physical waiting into an organized digital queue process. A web-based queue management application can register users, generate sequential queue numbers, maintain queue records in a database, and provide current queue information to both users and administrators. Earlier studies cited in the project report have linked effective queue organization and waiting conditions with service quality and customer satisfaction in Nigerian banking, healthcare, and telecommunications environments (Burodo et al., 2019; Ahmed et al., 2022). This study extends these principles to a web-based Automated Queue Management System designed for institutional service environments. Recent research similarly emphasizes the influence of queue structure and information availability on waiting experiences and service decisions (Ainbinder et al., 2024; Kang et al., 2024; Liu et al., 2026).

1.2 Problem Statement

Manual queue management presents several operational difficulties. Users may spend excessive time waiting because they are required to remain physically present without reliable information about their queue position. At the same time, both users and service administrators may have limited visibility of queue progression. Manual processes can also create opportunities for queue disputes, queue jumping, and inconsistent service allocation. In addition, large physical queues can produce congestion within service areas, while supervisors may lack dependable records for evaluating service throughput, identifying periods of high demand, and assessing operational performance. These challenges demonstrate the need for an automated solution capable of assigning queue numbers, maintaining queue states, supporting service allocation, and producing useful management information.

1.3 Project Objectives

The main aim of this study is to design and implement a web-based Automated Queue Management System that improves service allocation, minimizes dependence on physical waiting, and facilitates administrative reporting. The specific objectives are to: (1) examine existing manual queue-management procedures; (2) develop a multi-tier architecture for service selection, ticket generation, and queue monitoring; (3) implement customer, monitoring, and administrative functions; (4) provide administrative facilities for managing users and services; and (5) produce reports that support service-performance monitoring.

1.4 Scope and Boundaries

The developed system provides user registration and authentication, automated queue-number generation, queue-status monitoring, service management, administrative functions, and reporting. Its database model comprises users, services, queues, and reports. The study is

restricted to software development and testing and does not cover specialized physical display equipment, biometric devices, SMS gateways, or a dedicated mobile application. The project also recognizes dependence on internet or network access and limited availability of large-scale organizational data as important constraints (Lester & Mallari, 2020).

II. LITERATURE REVIEW

2.1 Queueing Theory Basics

Queueing theory concerns the mathematical analysis of waiting lines and service systems. A typical queueing model considers customers, the arrival process, the service mechanism, the queue discipline, and departures from the system (Taha, 2017; Gross et al., 2018). Kendall's notation provides a compact representation of queueing systems using a form such as $A/S/c$, where A denotes the arrival distribution, S represents the service-time distribution, and c indicates the number of service channels. Accordingly, an $M/M/c$ model describes a system with Poisson arrivals, exponentially distributed service times, and c parallel servers. Although the present prototype does not estimate an empirical $M/M/c$ model, queueing theory provides the conceptual basis for understanding how arrival intensity, service capacity, and queue discipline influence waiting.

The implemented system principally applies the First-Come, First-Served (FCFS) discipline. The next queue number is obtained from the most recent number, increased by one, stored, and displayed to the user. Although priority service may be incorporated in a future version, the available project documentation does not describe a separate priority-routing algorithm. Therefore, this study does not claim that VIP or priority routing was implemented. Recent research indicates that prioritization policies can significantly influence waiting and throughput, demonstrating why priority routing should be treated as a distinct system feature (Frazelle & Katok, 2024; Kang et al., 2024).

2.2 Review of Existing Web and Automated Solutions

Queue-management solutions currently range from traditional manual lines to electronic ticketing systems and web-based reservation platforms. Banking applications, for example, can improve queue organization through numbered tickets and service categories, although they may require dedicated display equipment and installation expenses. Healthcare queue systems can assist with scheduling and patient flow but may introduce integration challenges. Government-oriented solutions can increase transparency and accountability, but their effectiveness may depend on infrastructure and user adaptation. Studies cited in the project report have reported positive associations between queue organization, waiting conditions, and customer satisfaction in Nigerian service settings (Burodo et al., 2019; Ahmed et al., 2022).

Web-based approaches provide a practical alternative because their fundamental functions can be implemented using widely available web technologies and relational databases. Lester and Mallari (2020) and Mohammed and Yusuf (2023), as cited in the project report, describe online queue approaches intended to reduce congestion and improve service delivery. The gap addressed by the present project is the need for a relatively affordable and adaptable web application that can be used by different service-oriented organizations without requiring specialized queue hardware. More recent implementations also demonstrate the continuing relevance of web-based queue systems, quantitative waiting-time assessment, and digital service-quality evaluation (Zhao & Gilbert, 2025; Iqbal & Sari, 2025; Fauzi & Suhaimi, 2026).

2.3 Technology Synthesis

HTML, CSS, JavaScript, PHP, and MySQL were selected because they provide the necessary components for developing a conventional browser-based client-server application and can be conveniently configured in a local XAMPP environment. HTML and CSS support the presentation interface, JavaScript enables interactive client-side behaviour, PHP performs server-side processing, and MySQL provides persistent data storage. Together, these technologies support the separation of presentation, application logic, and data management associated with a three-tier architecture (Pressman & Maxim, 2020; Sommerville, 2016).

The project requirements refer to real-time queue updates and identify WebSockets, Server-Sent Events, or optimized polling as possible approaches. However, the final project documentation does not confirm implementation of WebSockets or Server-Sent Events and does not provide a formal API endpoint specification. For this reason, the system is conservatively presented as a browser-based client-server application rather than being described as a WebSocket- or RESTful-API-based prototype. WebSockets, Server-Sent Events, and a formally specified RESTful API remain appropriate architectural enhancements for a future version if they are implemented and experimentally evaluated.

III. SYSTEM METHODOLOGY AND SOFTWARE ARCHITECTURE

3.1 Requirements Specification

The project followed the System Development Life Cycle (SDLC) using the Waterfall model. The documented development stages comprised requirements gathering, system analysis, system design, development, testing, deployment, and maintenance (Pressman & Maxim, 2020; Sommerville, 2016). This sequential approach provided a structured link between the identified requirements, system design, implementation, and subsequent testing.

Functional and non-functional requirements:

Requirement Type	Requirement
Functional	User registration and authentication
Functional	Service selection and unique sequential token generation
Functional	Queue storage and status monitoring
Functional	Administrative queue and service management
Functional	Report generation
Non-functional	Security, reliability, availability and access control
Non-functional	Usability and cross-browser web accessibility
Non-functional	Scalability and performance efficiency

3.2 Web Architecture and Data Flow

The system is organized around a three-tier client-server architecture. The presentation tier contains browser-based interfaces for activities such as registration, login, dashboard access, and queue-status viewing. The application tier is responsible for authentication, queue-number generation, monitoring, and reporting processes. The database tier maintains information relating to users, services, queues, and reports. The overall data flow can be

represented as: user request → presentation layer → PHP application logic → MySQL database → application response → browser interface.

[Presentation Layer: Customer / Administrator Browser] → [Application Layer: PHP Queue, Authentication, Reporting Logic] → [Database Layer: MySQL Users, Services, Queues, Reports]

Figure 1. Logical three-tier architecture of the Automated Queue Management System.

The final project report identifies the same three architectural layers. Login, registration, dashboard, and queue-status pages belong to the presentation layer; queue generation, authentication, report processing, and monitoring are handled within the application layer; while Users, Queue Records, Services, and Reports form the principal database components.

3.3 Communication Protocol and Queue Updates

The documented implementation relies on PHP for server-side processing and JavaScript for client-side interaction. No dedicated WebSocket or Server-Sent Events service is documented in the final report. The prototype is therefore best understood as a conventional web client-server application in which the browser requests information from the application server and the server retrieves or modifies the relevant MySQL records. For a production deployment that requires consistently low-latency push notifications, WebSockets or Server-Sent Events could be introduced and evaluated separately.

3.4 Database Design

The database structure separates identity information, service definitions, queue transactions, and reporting information. The principal entities are Users, Services, Queues, and Reports. A user can be associated with multiple queue records, while a service can be linked to numerous queue transactions. Reports summarize information derived from queue activities. The queue entity records the queue number, user and service relationships, status, and relevant timestamps.

Entity	Primary Key	Representative Fields / Purpose
Users	user_id	full_name, email, password_hash, role — identity and access control
Services	service_id	service_name, description, status — service catalogue
Queues	queue_id	user_id, service_id, queue_number, status, created_at, completed_at — queue transactions
Reports	report_id	report_date, served_count, waiting_count — management summaries

[Users] 1 —< [Queues] >— 1 [Services] [Queues] —> [Reports]

Figure 2. Logical entity relationship structure.

3.5 Algorithmic Logic

Queue-number assignment follows a sequential FCFS procedure. After a user has authenticated and selected a service, the application retrieves the most recently issued queue number, increments it by one, stores the new number, and presents the resulting ticket to the user. The procedure is represented by the following pseudo code:

BEGIN

LOGIN USER

SELECT SERVICE

GET LAST_QUEUE_NUMBER

NEW_QUEUE_NUMBER = LAST_QUEUE_NUMBER + 1

SAVE NEW_QUEUE_NUMBER

DISPLAY QUEUE_TICKET

END

Within the documented workflow, sequential numbering is used to establish queue-number uniqueness. Nevertheless, the project does not report a concurrency benchmark; therefore, no claim is made concerning collision-free operation under hundreds of simultaneous requests. In a production environment, atomic database transactions or appropriate locking should be applied during number allocation to minimize race conditions.

IV. IMPLEMENTATION AND SYSTEM WALKTHROUGH

4.1 Technology Stack and Development Toolchain

Component	Implementation
Operating system	Microsoft Windows 11 Pro, 64-bit
Web server	Apache 2.4.54 through XAMPP
Database	MySQL Server 8.0
Server-side engine	PHP 8.1.12
Frontend	HTML5, CSS3 and JavaScript ES6
UI libraries	Bootstrap 5.2 and jQuery
Development environment	Visual Studio Code and XAMPP
Access	Web browser and internet/local network environment

The final project report identifies this development environment and indicates that the application was developed and tested locally. XAMPP supplied the Apache server environment, PHP handled server-side requests, and MySQL stored customer, queue, and administrative data. The implementation also incorporated authentication, password protection, input validation, session management, and access-control measures.

4.2 Module Implementation

4.2.1 Customer/User Interface

The user-facing interface supports account registration, authentication, service selection, queue-number generation, and queue-status monitoring. A generated queue ticket contains the queue number, date, and selected service type. The dashboard provides information such as the user's queue number, the current service number, queue status, and estimated waiting time. In this way, users can obtain queue-position information digitally rather than relying solely on their physical presence in a waiting line (Nielsen, 1993; Norman, 2004).

4.2.2 Public Queue Display

The system includes a queue-display interface intended to communicate the current ticket being served and the progression of the queue. The design documentation indicates that the display may present the current number, assigned counter, and next number. However, the implementation report does not document installation of a dedicated physical display board. Accordingly, this study treats the feature as a software-based display interface rather than a specialized hardware integration.

4.2.3 Staff/Administrative Interface

The administrative interface supports user management, queue administration, service management, monitoring, and report generation. The design documentation describes operator functions including viewing waiting users, calling the next queue number, skipping or recalling a number, and marking a service as completed. Administrative users can also manage service points and generate operational reports.

4.2.4 Reporting Module

The reporting component generates daily queue reports, customer statistics, and service-performance information. Reports are maintained separately from individual queue transactions within the database, allowing management summaries to be produced from stored queue activity without interfering with the ticket-generation process.

4.3 System Testing

Testing was undertaken to determine whether the implemented application performed the functions defined in its requirements. The project documentation identifies unit testing, integration testing, system testing, and user acceptance testing. The five detailed functional cases presented in the main results section addressed registration, login, queue-number generation, queue monitoring, and report generation. All five cases were recorded as successfully passed (Dennis et al., 2020).

Table 2. Functional test results reported in the final project report. Five of five documented cases passed (100%).

Test Case	Expected Result	Observed Result	Status
User registration	Account created successfully	Successful	Pass
User login	User gains access	Successful	Pass
Queue number generation	Unique queue number assigned	Successful	Pass
Queue monitoring	Correct queue displayed	Successful	Pass
Report generation	Reports generated correctly	Successful	Pass

V. EVALUATION AND PERFORMANCE ANALYSIS

5.1 Verification of System Correctness

The documented functional tests indicate that the prototype successfully executed its principal workflows, including user registration, authentication, queue-number generation, queue monitoring, and report generation. All five detailed test cases were marked Pass, resulting in a 100% pass rate for the main documented functional tests. The project report also includes a broader thirteen-item release checklist in an appendix, where the listed prototype checks are marked Pass. Collectively, these results provide evidence of functional correctness at the prototype-testing level.

5.2 Computational Performance and Benchmarking

The original project requirements proposed quantitative assessment of server response time, database-query performance, and concurrency involving 50, 100, and 500 virtual users. However, the final project report does not present measured response times, page-load benchmarks, database-query timings, or results from a documented 50/100/500-user load test. Similarly, no measured dataset comparing waiting times before and after implementation is provided. Consequently, this manuscript does not introduce or fabricate numerical performance results. The principal quantitative finding available from the documented testing is the 100% pass rate obtained across the five detailed functional test cases.

A more comprehensive journal-level performance evaluation should deploy the system in a controlled server environment and repeatedly measure request latency, database transaction time, page-load time, queue-generation throughput, and performance under different levels of concurrent access. Waiting-time improvement can be calculated using: $\text{Reduction (\%)} = ((T_{\text{manual}} - T_{\text{AQMS}}) / T_{\text{manual}}) \times 100$, where T_{manual} represents the baseline mean waiting time and T_{AQMS} represents the mean waiting time after deployment of the automated system. Such measurements should be obtained before making any numerical claim regarding waiting-time reduction. Recent studies provide useful methodological references for this type of evaluation, including Zhao and Gilbert (2025) and Iqbal and Sari (2025).

5.3 Analytical Value

The reporting functionality is intended to support daily queue statistics, counts of customers served, and service-efficiency information. Since queue transactions are stored in the database, the system establishes a foundation for data-driven administrative decision-making by allowing activity to be summarized according to date, service, and status. Nevertheless, the final project report does not provide the numerical dataset required to calculate measures such as average service time per teller or peak-hour throughput. These measures are therefore treated as potential analytical outputs rather than experimentally measured findings.

5.4 Discussion

The implemented system addresses several weaknesses associated with manual queue management by automating ticket assignment, maintaining queue information electronically, making queue status available to users, and producing administrative reports. The project identifies improved organization, more efficient allocation, reduced dependence on physical waiting, increased transparency, and improved record management as the principal benefits. These observations are broadly consistent with the related studies reviewed in the project, which associate automated queue practices with improved service organization and customer experience (Burodo et al., 2019; Mohammed & Yusuf, 2023; Ainbinder et al., 2024; Zhao & Gilbert, 2025; Iqbal & Sari, 2025; Fauzi & Suhaimi, 2026). However, the strongest evidence

generated by the present prototype concerns functional feasibility. Claims regarding numerical reductions in waiting time or scalability beyond the documented test environment would require additional empirical bench marking.

VI. CONCLUSION AND FUTURE DIRECTIONS

6.1 Conclusion

This study described the design and implementation of a web-based Automated Queue Management System developed to address challenges associated with physical and manually managed queues. The system employs a three-tier architecture comprising a browser-based presentation layer, PHP-based application logic, and MySQL data storage. Its major functions include user registration, authentication, sequential queue-number generation, queue monitoring, administrative management, and reporting. The five principal documented functional test cases were all successfully passed, demonstrating that the major workflows were implemented successfully. The system therefore offers a practical basis for reducing dependence on physical queues, improving transparency, and establishing structured records of service activities. Since the project did not collect baseline and post-implementation waiting-time measurements or server-load benchmarks, no numerical percentage reduction in waiting time is claimed. Future evaluation should incorporate these measurements to determine the system's quantitative operational impact.

6.2 Future Directions

Future work should emphasize empirical performance evaluation and deployment beyond the prototype environment. A subsequent version could incorporate WebSockets or Server-Sent Events to provide push-based queue updates, a formally specified RESTful API to improve interoperability, SMS or email notifications, mobile access, cloud deployment, digital display hardware, and stronger authentication mechanisms. Machine-learning techniques could also be explored to predict waiting times using historical arrival and service data. Further studies should compare manual and automated queue processes through controlled measurements of waiting time, service time, throughput, congestion, and user satisfaction. The project report additionally identifies mobile applications, cloud deployment, advanced analytics, biometric integration, and voice-based notifications as potential enhancements. Recent web-based queue research further supports real-time queue-status information, online reservation, and usability evaluation as relevant directions for future development (Fauzi & Suhaimi, 2026).

REFERENCES

- Adebayo, A., Ibrahim, M., & Yusuf, S. (2021). Electronic queue management systems and customer satisfaction in Nigerian banking institutions. *Journal of Information Systems*, 12(3), 45–58.
- Ahmed, A. B., Burodo, M. S., & Suleiman, A. (2022). Assessment of queue management and customer satisfaction in telecommunication service centres: A case study of MTN Sokoto branch. *Sokoto Journal of Management Studies*, 20(1), 112–128.
- Ainbinder, I., Temnikov, E., & Allalouf, M. (2024). A study comparing waiting times in global and local queuing systems with heterogeneous workers. *Applied Sciences*, 14(9), 3799. <https://doi.org/10.3390/app14093799>
- Burodo, M. S., Suleiman, A., & Shaba, Y. (2019). Application of queueing theory to customer service management in Nigerian commercial banks: A case study of First

- Bank Kaura Namoda branch. *International Journal of Operational Research and Management*, 11(2), 34–48.
- Burodo, M. S., Suleiman, A., & Yusuf, R. (2021). Queue management and patient satisfaction in selected tertiary healthcare institutions in North-Western Nigeria. *African Journal of Health Operations and Management*, 14(1), 88–102.
- Dennis, A., Wixom, B. H., & Roth, R. M. (2020). *Systems analysis and design* (8th ed.). Wiley.
- Fauzi, N. A. binti M., & Suhaimi, N. H. S. (2026). Design and evaluation of QKlinik: A web-based queue management system for outpatient clinics. *International Journal of Software Engineering and Computer Systems*, 11(2), 147–159. <https://doi.org/10.15282/ijsecs.11.2.2025.12.0144>
- Fitzsimmons, J. A., & Fitzsimmons, M. J. (2020). *Service management: Operations, strategy and information technology* (9th ed.). McGraw-Hill Education.
- Frazelle, A. E., & Katok, E. (2024). Paid priority in service systems: Theory and experiments. *Manufacturing & Service Operations Management*. <https://doi.org/10.1287/msom.2021.0387>
- Gross, D., Shortle, J. F., Thompson, J. M., & Harris, C. M. (2018). *Fundamentals of queueing theory* (5th ed.). Wiley.
- Iqbal, A. M., & Sari, D. P. (2025). Optimization of services through the development of a website-based online queueing system. *INOVTEK Polbeng - Seri Informatika*, 10(3), 1846–1856. <https://doi.org/10.35314/pdx5hg42>
- ISO/IEC. (2009). *ISO/IEC 25010:2009: Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—System and software quality models*. International Organization for Standardization.
- Kang, K., Doroudi, S., & Delasay, M. (2024). Prioritization in the presence of self-ordering opportunities in omnichannel services. *Production and Operations Management*, 33(3), 737–756. <https://doi.org/10.1177/10591478241231874>
- Kendall, K. E., & Kendall, J. E. (2019). *Systems analysis and design* (10th ed.). Pearson.
- Larson, R. C. (2021). Perspectives on queuing: Psychology of waiting lines and applied operations research. *Operations Research*, 69(3), 701–714.
- Katz, K. L., Larson, B. M., & Larson, R. C. (1991). Prescription for the waiting-in-line blues: Entertain, enlighten, and engage. *Sloan Management Review*, 32(2), 44–53.
- Maister, D. H. (1985). The psychology of waiting lines. In J. A. Czepiel, M. R. Solomon, & J. C. Surprenant (Eds.), *The service encounter: Managing employee/customer interaction in service businesses* (pp. 113–123). Lexington Books.
- Nielsen, J. (1993). *Usability engineering*. Academic Press.
- Norman, D. A. (2004). *Emotional design: Why we love (or hate) everyday things*. Basic Books.

- Parasuraman, A., Zeithaml, V. A., & Malhotra, A. (2005). E-S-QUAL: A multiple-item scale for assessing electronic service quality. *Journal of Service Research*, 7(3), 213–233. <https://doi.org/10.1177/1094670504271156>
- Pressman, R. S., & Maxim, B. R. (2006). *Software engineering: A practitioner's approach* (6th ed.). McGraw-Hill.
- Sommerville, I. (2008). *Software engineering* (8th ed.). Addison-Wesley.
- Laudon, K. C., & Laudon, J. P. (2022). *Management information systems: Managing the digital firm* (17th ed.). Pearson.
- Lester, R., & Mallari, P. (2020). A web-based queue management system for improving service delivery. *International Journal of Computer Applications*, 176(12), 20–28.
- Liu, Y., Meng, R., Tang, X., et al. (2026). Optimizing waiting experience: How passenger progress information and entertainment fillers jointly influence online ride-hailing drivers' time perception and emotion. *Scientific Reports*, 16, 4636. <https://doi.org/10.1038/s41598-025-34811-9>
- Mohammed, U., & Yusuf, A. (2023). Design and development of an online queue reservation system. *International Journal of Computer Science and Information Technology*, 15(2), 34–49.
- Ogunleye, T., & Ibrahim, A. (2022). Queue management practices in healthcare facilities: A case study approach. *African Journal of Health Informatics*, 8(1), 61–75.
- Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill.
- Suleiman, A., Burodo, M. S., & Maruf, A. (2022). Evaluation of patient wait times and satisfaction at Federal Medical Centre Gusau. *Nigerian Journal of Health Services Management*, 9(2), 143–157.
- Taha, H. A. (2017). *Operations research: An introduction* (10th ed.). Pearson.
- Zhao, X., & Gilbert, K. (2025). A queuing theory-based approach to evaluate service delivery quality. *International Journal of Quality and Service Sciences*, 17(2), 155–167. <https://doi.org/10.1108/IJQSS-10-2024-0149>

Author Note

This manuscript was rewritten from the submitted final project report. The recent references already included in the source manuscript were retained, as were the documented findings and stated limitations. No unsupported waiting-time, response-time, or concurrency results have been introduced.