International Journal Software Engineering and Computer Science (IJSECS)
6 (3), 2026, 1111-1120
Published Online December 2026 in IJSECS (http://www.journal.lembagakita.com/index.php/ijsecs)
P-ISSN: 2776-4869, E-ISSN: 2776-3242. DOI: https://doi.org/10.35870/ijsecs.v6i3.8308.

RESEARCH ARTICLE Open Access

Implementation of a Web-Based Inventory System to Support Operations at Hanbai Photocopy

Nandini Ladivina Sari [1*], **Sri Lestari** [2], **Mesra Betty Yel** [3]

[1*,2,3] Department of Information Systems, Faculty of Computer Technology, Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika, East Jakarta City, Special Capital Region of Jakarta, Indonesia.

Article History:

Received: August 6, 2026.
Revised: August 7, 2026.
Accepted: August 27, 2026.
Available online: September 20, 2026.
Published: December 1, 2026.

***Corresponding Author:**
nandiniladivina24@gmail.com

Abstract: Hanbai Photocopy is a micro-enterprise providing photocopying and office stationery retail services that requires structured inventory management to maintain seamless daily operations. Previously, inventory tracking relied entirely on paper-based ledger records, which frequently resulted in recording discrepancies between physical stock and written logs, slow search retrieval, and time-consuming report compilation. This study aims to design and implement a web-based inventory management system to support operations at Hanbai Photocopy through centralized and automated data administration. The research adopted a qualitative case study methodology, collecting operational data through direct field observation, semi-structured interviews with the business owner, and literature review. System development followed the classical Waterfall lifecycle, encompassing requirements analysis, system design, implementation, testing, and maintenance. The application was constructed using Google Apps Script as the execution platform and Google Sheets as the relational cloud database, enabling a cost-effective serverless architecture without independent hosting requirements. Functional validation was conducted through black box testing across key functional requirements, including master data management, transaction logging, automated stock validation, role-based access control, and activity auditing. The results demonstrate that all planned test scenarios executed successfully, effectively preventing invalid stock deductions and enabling real-time inventory adjustments. The operational deployment facilitates structured stock tracking, mitigates manual recording errors, and streamlines periodic reporting, thereby improving operational efficiency at Hanbai Photocopy.

Keywords: Inventory System; Google Apps Script; Google Sheets; Black Box Testing; Business Operations.

1. Introduction

The rapid expansion of digital technology has fundamentally altered administrative workflows across diverse business sectors, primarily by replacing conventional manual mechanisms that are inherently vulnerable to human calculation and recording discrepancies (Afifah, 2023). Within micro, small, and medium enterprises, particularly photocopy and stationery service providers, structured data administration is vital because commercial survival depends directly on uninterrupted supply availability. Ineffective stock governance introduces serious operational liabilities: inventory depletion halts daily production cycles and damages customer trust, whereas unmonitored overstocking immobilizes operational capital, increases depreciation, and elevates storage overheads (Nugroho, 2025). Consequently, implementing a robust and automated recording mechanism is indispensable to maintain equilibrium between customer responsiveness and efficient capital allocation. Hanbai Photocopy operates as a neighborhood commercial provider managing an extensive catalog of consumable items and retail products, including various paper grades, specialized printing inks, stationery goods, and bookbinding materials. Daily inventory governance at this enterprise continues to rely entirely on paper-based ledger entries, which frequently results in logging inaccuracies during inbound supplier deliveries and outbound retail transactions. Furthermore, retrieving historical transaction logs and reconciling remaining balances require considerable manual labor because physical archives lack relational indexing. This manual record-keeping routinely creates discrepancies between recorded ledger figures and physical stock counts inside the storage area. In addition, generating periodic inventory summaries becomes delayed, impairing the owner's ability to make data-driven purchasing and stock replenishment decisions. These operational bottlenecks demonstrate that traditional paper-based methods are no longer sufficient to sustain growing daily business activities. Similar administrative challenges have been widely documented across diverse organizational domains. Prior empirical research on web-based inventory systems in distribution enterprises demonstrated that digital record processing significantly accelerates transaction cycles while safeguarding historical archives against physical loss and documentation degradation (Heryanto *et al.*, 2014). An investigation within an educational chemical laboratory confirmed that integrated, real-time transaction tracking substantially enhances data accuracy and administrative efficiency compared to physical ledger cards (Denovya & Voutama, 2026). In secondary school inventory management, manual procedures were shown to involve high human error probabilities and persistent delays in data dissemination (Hidrawan *et al.*, 2026), while empirical evidence from small commercial enterprises indicated that browser-accessible inventory architectures provide practical operational benefits for non-technical users (Qisthina *et al.*, 2025). Nevertheless, most existing implementations depend on dedicated database servers such as MySQL, continuous hosting subscriptions, and specialized maintenance protocols, which impose ongoing financial commitments that micro-enterprises often cannot justify. The novelty of this study lies in the architectural implementation of a lightweight, serverless inventory framework utilizing Google Apps Script as the execution runtime and Google Sheets as the relational cloud database backend. By exploiting built-in web publishing services and Google Workspace cloud infrastructure, this technological strategy provides small businesses with an automated, browser-accessible inventory platform without requiring dedicated physical servers, complex network configurations, or recurring software licensing fees. Accordingly, this research aims to design, construct, and deploy a web-based inventory management system tailored to the operational workflows of Hanbai Photocopy, while systematically evaluating its functional validity across all transactional modules through black box testing. This approach offers a practical, low-barrier digitalization pathway to enhance inventory accuracy and operational sustainability for small-scale retail enterprises (Anwar *et al.*, 2025).

2. Related Work

2.1 Conceptual Foundations of Information Systems and Inventory Governance

An information system is an integrated assembly of interconnected elements operating collectively through inputs, processing engines, and functional outputs to achieve defined organizational objectives (Mulyati, 2017). Within logistical operations, technical implementation translates conceptual structural models into deployable digital tools designed to govern inventory assets, which represent the physical commodities held by an enterprise to maintain uninterrupted commercial workflows (Afifah, 2023). An inventory management system systematically logs stock movements, tracks fluctuating supply levels in real time, and regulates inbound and outbound goods to maintain operational balance between customer demand and capital commitment (Nugroho, 2025). A conventional regulatory framework in inventory control is the First-In, First-Out (FIFO) method, which prioritizes the distribution of earlier-received stock to prevent item degradation and stock obsolescence (Nugroho, 2025; Sekti *et al.*, 2024). In the current iteration of the system developed for Hanbai Photocopy, FIFO principles are incorporated as a theoretical baseline rather than an automated transaction-routing algorithm; this method establishes a conceptual benchmark for subsequent algorithmic

refinements, as elaborated in the conclusion. Modern administrative workflows frequently deploy web-based architectures where analytical processes are hosted remotely and accessed through standard web browsers, eliminating the need for client-side software installations across multiple business devices (Abdulloh, 2018; Suryawinata, 2019).

2.2 Google Apps Script, Google Sheets, and the Waterfall

Model Google Apps Script is a cloud-based development environment utilizing modern JavaScript syntax to automate workflows and construct lightweight web applications that integrate natively with the Google Workspace ecosystem without requiring standalone database configurations or physical hosting hardware (Google Developers, 2024). Published Apps Script routines handle client-side communication via doGet() and doPost() HTTP methods, functioning as an accessible middleware layer. For transactional data persistence, Google Sheets serves as a cloud-based relational data repository, structuring entities across distinct sheets that support collaborative, real-time synchronization (Google, 2024). This combination offers a viable, zero-infrastructure computing framework for micro-enterprises seeking centralized data coordination without recurring subscription overheads. Security protocols within this serverless environment rely on centralized Google identity authentication coupled with fine-grained access permissions governing script execution and spreadsheet read-write privileges, preventing unauthorized data modification (Google Developers, 2024). To execute system development systematically, this research employs the classic Waterfall software lifecycle model, which advances linearly through requirements analysis, system architectural design, code implementation, functional verification, and ongoing maintenance (Pressman, 2014). This linear progression ensures that operational requirements gathered via direct field observation and stakeholder interviews are comprehensively established before programming commences.

2.3 Related Work and Comparative

Synthesis A review of prior literature was conducted to benchmark technical architectures and methodological approaches across comparable web-based inventory initiatives. Table 1 summarizes relevant empirical studies alongside the specific gaps addressed by this investigation.

Table 1. Comparative Synthesis of Related Inventory Systems and Research Positioning

No	Study Focus	Author (Year)	Key Findings	Identified Research Gap	Current Study Position
1	Web-based inventory system at PT Infinetworks	Heryanto *et al.* (2014)	Digitalization accelerates processing cycles and prevents transactional records from physical loss	Lacks cloud-native, serverless deployment architectures	Implements Google Apps Script and Google Sheets without dedicated servers
2	Prototype web-based chemical laboratory inventory	Denovya & Voutama (2026)	Real-time tracking enhances recording accuracy and operational efficiency	Confined to academic laboratory environments rather than retail services	Applied directly to retail photocopy and office stationery operations
3	Inventory management application at SMA Negeri 5 Binjai	Hidrawan *et al.* (2026)	Web system successfully replaces manual record books and reduces human error	Context limited to educational institution asset control	Tailored to commercial merchandise and daily consumable replenishment cycles
4	Web-based inventory design for MSMEs	Qisthina *et al.* (2025)	Web interfaces demonstrate high usability for non-technical small business managers	Relies on conventional commercial web-hosting subscriptions	Employs Google Apps Script to completely eliminate external hosting expenses
5	Web-based goods inventory implementation	Anwar *et al.* (2025)	Automated transaction logging substantially accelerates inventory balance reconciliation	Omits multi-level user administration and comprehensive activity logging	Incorporates role-based user management and granular activity audit trails

6	Sample product inventory management system	Islamiah & Roslina (2025)	Computerized workflows expedite inventory reporting for commercial marketing divisions	Relies on standalone relational database instances (MySQL)	Eliminates dedicated MySQL instances by utilizing cloud-based Google Sheets

As shown in Table 1, existing web inventory applications frequently depend on dedicated database servers such as MySQL and third-party hosting packages, which impose recurrent maintenance commitments that micro-enterprises cannot reliably support. The technical novelty of this study lies in utilizing Google Apps Script and Google Sheets as an integrated, serverless computational framework. This architecture delivers an automated web inventory platform at zero server maintenance cost while incorporating role-based administrative controls and activity audit logs to safeguard operational accountability for small retail enterprises.

3. Methodology

3.1 Research Approach and Empirical Object

This study applies a software engineering methodology within a qualitative single-case study framework to examine the operational mechanics of inventory governance at Hanbai Photocopy and resolve systemic ledger recording discrepancies (Sugiyono, 2019). The primary research artifact is a lightweight, serverless web-based inventory management system developed using Google Apps Script as the core execution engine and Google Sheets as the relational cloud database backend. The empirical scope encompasses all operational workflows concerning inbound merchandise logging, outbound inventory tracking, supplier directory management, and automated stock report generation at Hanbai Photocopy.

3.2 Data Collection Techniques

Empirical data were gathered systematically through three complementary techniques: direct field observation, semi-structured interviews, and literature reviews. Direct observation was conducted within Hanbai Photocopy's storage and retail facilities to document existing paper-based recording practices and identify procedural bottlenecks. A semi-structured interview was conducted with the business owner as the key informant, structured around three operational domains: real-time stock balance tracking, inbound/outbound inventory routing, and administrative reporting requirements. In addition, literature reviews of recent academic publications and official technical documentation from Google Developers were conducted to formulate the technical architecture. Table 2 summarizes the data collection protocols.

Table 2. Summary of Empirical Data Collection Protocols

No	Technique	Data Source	Investigative Objective	Analytical Output
1	Direct observation	Physical ledger entries and storage workflows at Hanbai Photocopy	Map existing business processes and operational bottlenecks	Baseline workflow diagram of current operations
2	Semi-structured interview	Business owner of Hanbai Photocopy (single key informant)	Elicit functional operational needs and reporting criteria	Formulated inventory functional and non-functional requirements
3	Literature review	Peer-reviewed journal articles and Google Developers documentation	Establish conceptual framework and architectural benchmarks	System architecture blueprint and script execution models

3.3 System Development Life Cycle

System construction was executed using the classic Waterfall software lifecycle model (Pressman, 2014), selected due to the stable and fully elicitable operational requirements established during initial fieldwork. The engineering process proceeded through five sequential phases:

1) Requirements Analysis: Synthesizing observation logs and interview findings to define core software specifications.
2) System Design: Translating functional specifications into Unified Modeling Language (UML) architectural diagrams—including use case, activity, and class diagrams—and formulating the relational structure across Google Sheets.

3) Implementation: Programmatically constructing the application modules in Google Apps Script (JavaScript syntax) and HTML/CSS, encompassing user authentication, item master data, inbound transactions, outbound transactions, supplier management, dynamic reporting, and role administration.
4) Functional Testing: Validating functional integrity through rigorous black box testing protocols across all transactional interfaces.
5) Deployment and Maintenance: Hosting the web application within the Google Workspace cloud runtime, evaluating operational reliability during live business hours, and gathering user feedback for incremental optimization.

3.4 System Requirements Specification

Analysis of business and informational workflows produced seven core functional requirements, as detailed in Table 3. These specifications address stock balance integrity, access control, auditability, and vendor records.

Table 3. Functional Requirements Specification

Code	Functional Requirement	Operational Description
FR-01	Stock transaction control	Record and validate inbound and outbound stock transactions in a structured relational format
FR-02	User authentication	Enforce credential verification via a dedicated login portal to prevent unauthorized system access
FR-03	Real-time stock monitoring	Display dynamic stock balances and generate visual threshold alerts when supplies reach safety minimums
FR-04	User activity logging	Automatically log timestamps, user identities, and action details for every database modification to maintain audit trails
FR-05	Periodic report generation	Compile structured transaction summaries filtered by transaction category and designated date ranges
FR-06	Multi-criteria search and filtering	Enable responsive inventory record querying by item identifier, item name, or product classification
FR-07	Supplier records management	Maintain an integrated vendor directory to streamline restocking procedures and procurement coordination

Non-functional requirements were established to ensure operational dependability, accessibility, and data integrity. Table 4 summarizes these criteria.

Table 4. Non-Functional Requirements Specification

Code	Quality Dimension	Performance Specification
NFR-01	Security	Enforce encrypted session handling and restrict backend spreadsheet write permissions to authorized scripts
NFR-02	Usability	Deliver an intuitive, user-friendly graphical interface suited for non-technical retail operators
NFR-03	Availability	Ensure 24/7 cloud accessibility via public internet connections without local server maintenance dependencies
NFR-04	Platform compatibility	Guarantee responsive operational rendering across modern desktop browsers and mobile computing interfaces
NFR-05	Scalability	Leverage Google Sheets capacity to sustain longitudinal growth in daily transactional entries
NFR-06	Data integrity	Execute automated server-side validations to prevent outbound stock processing that exceeds available physical inventory

3.5 Architectural System

Design System design formalizes the behavioral and structural logic of the application prior to implementation. The use case diagram models single-actor role interactions where the system administrator (business owner) possesses complete operational authority across item master catalogs, inbound/outbound logging, vendor directories, user management, audit trails, and reporting outputs. Activity diagrams define step-by-step UI interaction flows for master inventory configuration, stock deduction validations, and transactional queries. Furthermore, the class diagram defines the programmatic object structure and relational associations linking the User, Item, InboundStock, OutboundStock, Supplier, and ActivityLog entities.

3.6 Verification Technique and Implementation Setup

System verification was conducted using black box testing, an empirical evaluation technique that analyzes functional adherence against designated software specifications without inspecting internal source code execution paths. Test vectors were systematically fed into user interfaces to evaluate whether generated outputs aligned with expected system behaviors across user login, master data modification, transaction logging, vendor cataloging, report compilation, and user role administration. The technical specifications of the development and deployment setup are outlined in Table 5.

Table 5. System Development and Implementation Specifications

No	Component	Technical Environment Specification
1	Development platform	Google Apps Script runtime (modern JavaScript, utilizing doGet() and doPost() web endpoints)
2	Relational database	Google Sheets, partitioned into six structurally linked functional sheets
3	User interface layer	Native HTML Service embedded within Google Apps Script (HTML5, modern CSS styling, and client-side JavaScript)
4	Authentication mechanism	Role-based credential verification authenticated against secured records within the User sheet
5	Execution server	Serverless Google Workspace cloud infrastructure (eliminating dedicated hosting subscriptions)
6	Verification methodology	Systematic black box functional testing covering authentication, inventory transactions, and administrative modules
7	Client accessibility	Any contemporary web browser running on desktop workstations, laptops, or mobile devices

4. Results and Discussion

4.1 Results

The web-based inventory management system was constructed using Google Apps Script as the execution runtime and Google Sheets as the relational cloud database backend. The application integrates seven functional modules to resolve the administrative bottlenecks identified at Hanbai Photocopy. Access to the environment is guarded by an authentication gateway that restricts system operations strictly to verified user accounts. Upon successful login, users interact with an executive dashboard that synthesizes current stock positions, logs incoming and outgoing transaction totals, and triggers visual warning alerts for inventory items falling below predetermined safety thresholds. Routine item governance is managed through a master inventory catalog where operators record and update essential attributes, including item identifiers, nomenclature, classification categories, measurement units, and baseline quantities. Transactional activities are divided into inbound and outbound pathways. The inbound transaction module logs restocking shipments and immediately increments on-hand inventory levels upon submission. Conversely, the outbound transaction module incorporates automated server-side verification to process merchandise dispatches while systematically blocking any transaction where the requested quantity exceeds physical availability. Procurement management is supported by an integrated supplier directory cataloging contact details and operational addresses to facilitate vendor communication. Administrative oversight is reinforced through dynamic reporting tools capable of compiling transactional statements across custom calendar intervals, complemented by role-based user administration and an automated audit trail that records user actions across the platform.

Data persistence is structured across six interconnected sheets within Google Sheets, which operates as a serverless relational data repository. Credential records, user privileges, and authentication parameters are held in the User sheet, while foundational product specifications and current balances are preserved in the Barang sheet. Transactional histories are segregated into the Barang Masuk sheet for incoming vendor supplies and the Barang Keluar sheet for distribution records, with both sheets triggering automatic balance recalculations upon entry. Vendor communication profiles are maintained in the Supplier sheet, and system-wide modifications are chronologically documented in the Activity Log sheet, capturing operator identities, action timestamps, and descriptive event logs to ensure administrative transparency. To verify functional integrity, black box testing was systematically conducted across all implemented application modules. Testing scenarios were structured based on predefined test vectors and expected software responses, as outlined in Table 6.

Table 6. Black Box Test Scenarios for System Functional Verification

No	Functional Module	Test Scenario	Expected System Output
1	Authentication	Input valid username and valid password	System authenticates credentials and grants access to main dashboard
		Input invalid username or invalid password	System rejects access and displays credential error alert
2	Master Inventory	Update existing item attributes	System updates modified records in database and updates catalog table
		Delete existing item record	System removes selected item record from database
3	Inbound Stock	Submit valid inbound stock receipt	System records inbound transaction and automatically increments inventory balance
4	Outbound Stock	Submit outbound quantity within current stock balance	System logs outbound transaction and automatically deducts inventory balance
		Submit outbound quantity exceeding current stock balance	System blocks transaction and presents warning notification indicating insufficient stock
5	Supplier Directory	Add new vendor profile	System saves supplier record and renders it in vendor directory
		Edit existing vendor profile	System updates existing supplier attributes in database
		Delete vendor profile	System removes targeted vendor record from database
6	Transaction Reports	Query transaction logs by specific date range	System filters and displays matching transactional summaries corresponding to selected period
7	User Management	Register new operator account	System saves account credentials and displays user in active user directory
		Delete existing operator account	System revokes account access and removes record from user database
		Retrieve operational activity log	System generates chronological audit list showing user identities, timestamps, and action descriptions

The execution results of the black box verification protocol across all defined scenarios are detailed in Table 7.

Table 7. Black Box Testing Execution Results

No	Functional Module	Test Scenario	Observed Output	Functional Status
1	Authentication	Input valid username and valid password	User successfully redirected to dashboard	Passed
		Input invalid username or invalid password	Credential failure warning displayed	Passed
2	Master Inventory	Update existing item attributes	Database attributes updated successfully	Passed
		Delete existing item record	Record removed from catalog successfully	Passed
3	Inbound Stock	Submit valid inbound stock receipt	Stock balance incremented automatically	Passed
4	Outbound Stock	Submit outbound quantity within current stock balance	Stock balance deducted automatically	Passed
		Submit outbound quantity exceeding current stock balance	Transaction rejected with alert dialog	Passed
5	Supplier Directory	Add new vendor profile	Vendor profile stored and displayed	Passed
		Edit existing vendor profile	Vendor information updated accurately	Passed

		Delete vendor profile	Vendor record purged successfully	Passed
6	Transaction Reports	Query transaction logs by specific date range	Filtered report compiled accurately	Passed
7	User Management	Register new operator account	New user account populated in registry	Passed
		Delete existing operator account	User record removed successfully	Passed
		Retrieve operational activity log	Historical event log rendered chronologically	Passed

As established in Table 7, all 14 execution test cases across the seven functional modules achieved a 100% success rate, conforming precisely to system requirements specifications. Automated calculation scripts updated inventory balances immediately upon transaction submission, while input validation routines prevented negative stock values.

4.2 Discussion

The operational deployment of this web-based inventory framework resolves the structural limitations of conventional paper logbooks previously utilized at Hanbai Photocopy. Operating through a unified cloud repository eliminates fragmented physical ledgers and centralizes inventory movements, supplier relations, and user actions into a coherent data architecture. From an administrative efficiency perspective, automated mathematical recalculation eliminates the labor-intensive task of manual ledger reconciliation. Stock balances adjust dynamically following inbound receipts and outbound distributions, eliminating administrative latency and preventing record-keeping backlogs. This operational enhancement aligns with prior findings in commercial distribution environments, where automated inventory handling reduced transaction durations and minimized document attrition (Heryanto *et al.*, 2014). From an operational data integrity perspective, the system implements automated validation logic that disallows outbound requests when requested amounts exceed physical inventory on hand. This safeguard prevents bookkeeping disparities and mitigates the incidence of phantom inventory that previously disrupted Hanbai Photocopy's daily customer service. This outcome corroborates findings from chemical laboratory asset tracking, which demonstrated that real-time transactional validation substantially elevates data accuracy compared to manual paper tracking (Denovya & Voutama, 2026). Furthermore, the automated audit trail in the Activity Log establishes accountability by recording operational timestamps and user identities for every catalog alteration, addressing functional gaps identified in earlier small-business inventory architectures (Anwar *et al.*, 2025). From an economic and technical deployment standpoint, utilizing Google Apps Script and Google Sheets provides a low-barrier digitalization pathway for small commercial entities. Unlike conventional database implementations relying on standalone MySQL instances and commercial hosting subscriptions (Islamiah & Roslina, 2025; Qisthina *et al.*, 2025), this serverless cloud approach eliminates hosting expenditures, complex local network configurations, and ongoing database administrative overheads. Nevertheless, this study acknowledges clear operational boundaries. Functional verification was conducted primarily through developer-side black box testing within controlled operational workflows. Comprehensive quantitative User Acceptance Testing (UAT) measuring usability dimensions, operator cognitive workload, and user satisfaction indices was not executed during this implementation phase due to the limited duration of active live deployment. Longitudinal monitoring is recommended to evaluate system responsiveness as transactional volume expands within the Google Sheets repository over multi-year business cycles.

5. Conclusion and Recommendations

The implementation and functional evaluation of the web-based inventory management system at Hanbai Photocopy successfully address the operational vulnerabilities associated with conventional paper-based record-keeping. Prior to system deployment, inventory governance at the enterprise experienced recurrent transaction logging errors, labor-intensive record retrieval, and prolonged reporting intervals. The developed application integrates core operational workflows—including item cataloging, inbound and outbound transaction tracking, supplier directory management, multi-level user administration, and automated audit logging—into a unified, serverless architecture utilizing Google Apps Script and Google Sheets. Functional verification via black box testing confirmed that all defined test scenarios executed successfully, demonstrating robust input validation routines that prevent stock deductions beyond physical availability and maintain dynamic balance synchronization. The transition from physical ledgers to a centralized cloud repository streamlines transactional recording, accelerates item queries, and enables automated summary compilation.

Nevertheless, observed enhancements in administrative efficiency and operational accuracy remain qualitative, as baseline quantitative metrics—such as comparative processing durations or longitudinal error-rate reductions—were not systematically compiled prior to implementation. Furthermore, comprehensive User Acceptance Testing measuring standardized usability and satisfaction indices remains to be completed. To build upon these findings, future research should implement automated messaging alerts for safety stock thresholds, conduct quantitative user acceptance evaluations, and programmatically incorporate First-In, First-Out (FIFO) routing logic directly into outbound distribution routines. Future studies should also gather empirical comparative datasets measuring transaction duration and error frequency before and after deployment, while extending system integration toward point-of-sale terminals and financial accounting modules to deliver comprehensive digital governance for retail operations.

References

Abdulloh, R. (2018). *7 in 1 pemrograman web untuk pemula*. PT Elex Media Komputindo.

Afifah, F. N. (2023). *Konsep sistem informasi (konsep dan penerapan)*. Yayasan Literasi Sains Indonesia.

Aghniya, A., Rivai, A. M., Febisatria, A., Harun, N., & Azza, N. (2025). Penerapan metode First In First Out (FIFO) dalam peningkatan kinerja operasional sektor ritel di Indonesia. *Jurnal Akademik Ekonomi dan Manajemen*, *2*(4), 181–192. https://doi.org/10.61722/jaem.v2i4.7265

Anwar, M. S., Nindyasari, R., & Murti, A. C. (2025). Implementasi aplikasi persediaan barang berbasis web. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *9*(4), 7166–7171. https://doi.org/10.36040/jati.v9i4.14418

Denovya, A. R., & Voutama, A. (2026). Development of a web-based chemical laboratory inventory system using the prototype model. *Jurnal Ilmiah Sistem Informasi*, *5*(2), 363–381. https://doi.org/10.51903/juisi.v5i2.1577

Google Developers. (2024). *Apps Script overview* [Technical documentation]. https://developers.google.com/apps-script

Google. (2024). *Google Sheets: Online spreadsheet editor* [Software documentation]. https://www.google.com/sheets/about/

Heryanto, A., Fuad, H., & Dananggi, D. (2014). Rancang bangun sistem informasi inventory barang berbasis web studi kasus di PT. Infinetworks Global Jakarta. *Sisfotek Global*, *4*(2), 2–5.

Hidayat, T., & Muttaqin, M. (2018). Pengujian sistem informasi pendaftaran dan pembayaran wisuda online menggunakan black box testing dengan metode equivalence partitioning dan boundary value analysis. *Jurnal Teknik Informatika UNIS*, *6*(1), 25–29. https://doi.org/10.33592/jutis.v6i1.278

Hidrawan, M. A., Badawi, A., & Fachri, B. (2026). Perancangan sistem informasi pengelolaan inventaris barang berbasis web di SMA Negeri 5 Binjai. *JUKTISI (Jurnal Kemahasiswaan Teknologi Informasi dan Sains)*, *5*(1), 257–266. https://doi.org/10.55338/juktisi.v5i1.745

Islamiah, R., & Roslina, R. (2025). Rancang bangun sistem manajemen inventory barang sample berbasis web pada divisi pemasaran (Studi kasus: PT. Berjaya Abadi). *Indonesian Journal of Informatic Research and Software Engineering (IJIRSE)*, *5*(1), 60–67. https://doi.org/10.57152/ijirse.v5i1.1396

Mulyati, Y. S. (2017). *Konsep sistem informasi*. CV. Andi Offset.

Nugroho, A. Y. (2025). *Sistem dan teknik manajemen inventory*. Gemilang Press Indonesia.

Pressman, R. S. (2014). *Software engineering: A practitioner's approach* (8th ed.). McGraw-Hill.

Qisthina, N. A. C., Andarwati, M., & Putri, D. M. (2025). Desain sistem informasi inventaris barang untuk usaha mikro kecil dan menengah (UMKM) berbasis website. *Journal of Information System and Application Development*, *3*(1), 47–57. https://doi.org/10.26905/jisad.v3i1.15396

Saputra, D., Ratnasari, M., Azhari, A., & Maisaroh, M. (2026). Pengembangan website ujian online berbasis Google Apps Script dan Google Spreadsheet. *Indonesian Journal of Multidisciplinary on Social and Technology*, *4*(3), 2278–2288. https://doi.org/10.69693/ijmst.v4i3.12603

Sekti, B. A., Gusti, A. P., & Erzed, N. (2024). Perancangan sistem informasi stok barang berbasis web dengan metode FIFO. *Jurnal Teknologi Informatika dan Komputer*, *10*(2), 506–518. https://doi.org/10.37012/jtik.v10i2.2253

Sugiyono. (2019). *Metode penelitian kuantitatif, kualitatif, dan R&D*. Alfabeta.

Suryawinata, M. (2019). *Pengembangan aplikasi berbasis web*. Umsida Press.

Wau, K. (2022). Pengembangan sistem informasi persediaan gudang berbasis website dengan metode waterfall. *Jurnal Teknik, Komputer, Agroteknologi dan Sains*, *1*(1), 10–23. https://doi.org/10.56248/marostek.v1i1.8