

AN EXTREME PROGRAMMING APPROACH FOR DEVELOPING AND EVALUATING A MULTI-SITE MATERIAL MANAGEMENT SYSTEM WITH INTEGRATED AUTOMATED STOCK ALERT MECHANISMS

Muhammad Raka Pradana¹, *Muhamad Awiet Wiedanto Prasetyo², Khairun Nisa Meiah Ngafidin³

Information Systems, Telkom University, Indonesia^{1,2,3}

muhammadrakapradana@student.telkomuniversity.ac.id, awietmwp@telkomuniversity.ac.id,
nisameiah@telkomuniversity.ac.id

**Corresponding Author*

ABSTRACT

Renewable energy organizations with distributed operational sites frequently experience inventory fragmentation, delayed material replenishment decisions, and information delays that disrupt operational continuity. This study addresses these challenges by proposing a Multi-Site Material Management System (SMMS) with an automated stock alert mechanism. The system was developed using the Extreme Programming (XP) methodology through four iterative cycles and implemented using a three-tier architecture, Role-Based Access Control (RBAC), React.js, Node.js, and MySQL. Field observations conducted within a renewable energy company managing a central warehouse and 2 remote operational sites revealed critical pre-system issues: approximately 40 material requests per month submitted via informal channels without standardized workflow, inventory records maintained separately using 3 different spreadsheets with inconsistency rate of 15-20%, and no automated mechanism to detect critical stock conditions, resulting in 3-4 documented stock shortage incidents per quarter. The proposed system integrates material request workflows, multi-level approval processes, shipment tracking, receipt confirmation, inventory management, and automated stock alerts with three severity levels (WARNING, CRITICAL, OUT_OF_STOCK). Comprehensive evaluation including Black Box Testing across 20 test scenarios, User Acceptance Testing with 15 end-users, System Usability Scale assessment yielding 78/100 score, and operational metrics demonstrated significant improvements: material request processing time reduced from 3.2 days to 0.8 days representing 75% improvement, inventory accuracy improved from 87% to 96%, and user satisfaction reached 4.2/5.0. Functional evaluation achieved a 100% success rate. The results indicate that the system improves inventory accuracy, accelerates material request processing, and enhances coordination among distributed operational sites. These findings demonstrate that integrating automated stock alert mechanisms into multi-site inventory systems can improve operational readiness and support more proactive inventory management through a combination of centralized visibility, threshold-based monitoring, and structured approval workflows.

Keywords : Multi-Site Material Management, Automated Stock Alert, Extreme Programming, RBAC, Inventory Information System, User Acceptance Testing, System Usability, Material Request Processing.

1. Introduction

The rapid expansion of renewable energy projects has increased the complexity of material management across geographically distributed operational sites. The renewable energy sector's rapid expansion has driven significant growth in distributed operational sites—for example, distributed solar and wind projects typically involve 5 or more operational locations requiring centralized material coordination. Organizations operating multiple warehouses and project locations require accurate inventory visibility, efficient coordination, and timely material distribution to maintain operational continuity. However, managing inventory across multiple locations remains challenging due to information delays, fragmented data, and inefficient coordination processes.

Field observations conducted within a renewable energy company operating a central warehouse and two remote operational sites revealed several critical issues with documented evidence. Material requests were submitted through email and verbal communication—approximately 40

requests per month—without a standardized workflow. Inventory records were maintained separately using three different spreadsheets with documented inconsistency rates of 15-20%, and no automated mechanism existed to detect critical stock conditions. These limitations created measurable operational impacts: documented stock shortage incidents occurred 3-4 times per quarter, inventory visibility was severely reduced due to data fragmentation across systems, request tracking was complicated without centralized documentation, and the risk of stock shortages was high, disrupting operational activities. These conditions directly motivated this research to develop a solution addressing these specific, documented challenges.

Previous studies have shown that Warehouse Management Systems (WMS) can improve inventory accuracy and operational efficiency, while Multi-Site Information Systems enable centralized control and synchronized inventory data across multiple locations. In addition, Role-Based Access Control (RBAC) supports secure access management among organizational roles, and Automated Alert Systems facilitate proactive inventory monitoring through threshold-based notifications. The integration of these approaches offers a potential solution for improving material management in distributed operational environments (Supriyati & Gustalika, 2023).

Although inventory and warehouse management systems have been widely studied, most existing solutions focus on single-site environments or do not incorporate automated stock alert mechanisms. Previous studies utilizing Extreme Programming successfully developed inventory and information systems but generally lacked multi-site inventory synchronization, role-based approval workflows, and proactive stock monitoring capabilities. Critical analysis of related studies reveals systematic gaps: Fiqar et al. (2023) developed an inventory management system using XP but focused on single-site retail operations; (Artama & Mariana, 2022) designed an inventory system with FIFO method but lacked automated alerts and multi-location coordination; (Septiana et al., 2022) created an administrative system using XP but did not address inventory-specific requirements. None of the reviewed studies comprehensively combined multi-site inventory synchronization, automated threshold-based alerts, and multi-level approval workflows within a single system. Therefore, a research gap remains regarding the development of an integrated platform that combines multi-site inventory management, automated stock alerts, and multi-level approval workflows.

To address this gap, this study proposes the design and development of a website-based Multi-Site Material Management System (SMMS) with an Automated Stock Alert mechanism using the Extreme Programming methodology. The primary scientific contribution of this research is: (1) Integrated Architecture—designing a unified platform combining multi-site data synchronization, role-based approval workflows, and automated threshold-based alerts, whereas previous systems addressed these elements separately; (2) XP Implementation—demonstrating how Extreme Programming's iterative approach with continuous stakeholder involvement enables enterprise system development responsive to evolving organizational requirements; (3) Automated Alert Innovation—implementing severity-based stock alerts (WARNING, CRITICAL, OUT_OF_STOCK) with intelligent routing to appropriate personnel based on stock conditions; (4) Comprehensive Evaluation—assessing system effectiveness through functional testing, user acceptance, usability measurement, and process efficiency metrics rather than functional testing alone. The study aims to provide centralized inventory visibility, streamline material request processes, and improve operational responsiveness across multiple sites. The research addresses the following questions: How can a multi-site material management system be designed to provide real-time inventory visibility across geographically distributed operational sites? How can an automated stock alert mechanism be integrated into the system to support proactive inventory management and prevent stock shortages? How effective is the Extreme Programming methodology in developing a multi-site material management system that satisfies stakeholder requirements? To what extent does the proposed system improve operational efficiency and user satisfaction compared to previous manual and spreadsheet-based approaches?

2. Literature Review

Warehouse Management System (WMS)

A Warehouse Management System (WMS) is an information system that supports inventory control, material tracking, and warehouse operations to improve efficiency and inventory

accuracy. In multi-location environments, WMS enables centralized inventory monitoring and reduces data inconsistencies caused by manual record keeping. However, traditional Warehouse Management Systems primarily focus on warehouse-level inventory control and intra-warehouse operations, with optimization concentrated on local warehouse efficiency. Such conventional WMS were not designed to address the distinct challenges that multi-site organizations face, including: (1) cross-location data synchronization in real-time, requiring mechanisms to coordinate updates across geographically distributed databases; (2) maintaining centralized inventory visibility across distributed locations while preserving database consistency; (3) coordinating material flow between multiple warehouses with multi-level approval workflows reflecting organizational hierarchies; (4) implementing automated decision-support mechanisms for stock monitoring and alerts across dispersed sites. Single-warehouse WMS optimize local warehouse operations but lack the architecture and coordination mechanisms required for synchronized multi-site operations. This limitation creates an operational gap where organizations with multiple sites must either maintain separate, disconnected WMS instances or develop custom integrations. The proposed material management system addresses this gap by providing a platform specifically designed for distributed material management. Therefore, WMS serves as the foundational concept for the proposed material management system, though enhanced and extended for multi-site contexts.

Multi-Site Information Systems

A Multi-Site Information System facilitates operational activities across geographically distributed locations through a centralized platform and shared database. By synchronizing inventory data in real time, the system improves coordination among operational units and supports faster managerial decision-making. This capability is essential for organizations managing inventory across multiple warehouses and project sites.

However, implementing multi-site information systems requires addressing several critical technical challenges that are central to the proposed system's design. These include: (1) Data Synchronization—maintaining consistency when multiple distributed locations simultaneously update inventory records, requiring mechanisms for transaction coordination and conflict resolution when concurrent updates occur at different sites; (2) Database Consistency—ensuring ACID properties (Atomicity, Consistency, Isolation, Durability) across geographically distributed databases connected through potentially unreliable networks where connections may experience latency or temporary interruptions; (3) Access Control—implementing role-based security ensuring that Operational Maintenance (OM) staff can only request materials, Network Operations Center (NOC) personnel can review and reject requests, and General Manager (GM) can approve significant transactions, with permissions enforced consistently across all sites; (4) Data Conflict Resolution—handling scenarios where network interruptions occur during update operations, potentially leaving distributed systems in temporarily inconsistent states, requiring mechanisms to detect and resolve conflicts automatically; (5) Real-time Monitoring—providing responsive alert notifications despite network latency between geographically separated locations, ensuring critical alerts reach responsible personnel immediately without delay. The proposed system architecture addresses these challenges through database transaction management, asynchronous messaging for alert distribution, and role-based permission validation at the application server level.

Role-Based Access Control (RBAC)

Role-Based Access Control (RBAC) is a security mechanism that restricts system access according to user roles and responsibilities. RBAC enhances security, accountability, and workflow integrity by ensuring that users can only access features relevant to their assigned tasks. In this study, RBAC is implemented to manage permissions for Operational Maintenance (OM), Network Operations Center (NOC), and General Manager (GM).

Automated Alert System

Automated Alert Systems monitor inventory conditions and generate notifications when stock levels reach predefined thresholds (Albert et al., 2025). This approach enables proactive inventory management by providing early warnings before stock depletion occurs. The implementation of

automated stock alerts is expected to reduce stockout risks and improve operational readiness across multiple sites.

Extreme Programming (XP) Methodology

Extreme Programming (XP) is an agile software development methodology emphasizing iterative development, continuous stakeholder involvement, rapid feedback, and frequent testing. XP is particularly suitable for projects with evolving requirements and small development teams. While multiple agile methodologies exist (Scrum, Kanban, Lean), Extreme Programming was selected for this project based on: (1) Rapidly evolving stakeholder requirements identified during field observations, necessitating frequent cycles of design-code-test-feedback; (2) Small development team size (3 developers), which is ideal for XP's intensive collaboration model requiring close communication; (3) Emphasis on continuous stakeholder involvement throughout development, essential given multiple organizational roles (OM, NOC, GM) with distinct and sometimes conflicting requirements; (4) XP's strong testing culture aligning with the research objective of ensuring system reliability for critical inventory operations; (5) The problem domain's complexity requiring frequent refactoring and architectural improvements, which XP facilitates through test-driven development practices. In this study, XP was selected to facilitate incremental development and ensure that system functionality aligns with stakeholder needs throughout the project lifecycle.

Related Studies and Research Gap

Existing studies have successfully applied Extreme Programming in inventory management systems, public service systems, and administrative information systems. However, most of these systems focus on single-site operations and do not integrate automated stock alerts or multi-level approval workflows. Furthermore, previous inventory management studies identified the absence of stock alert functionality as a limitation and recommended its inclusion in future work. Critical analysis of related studies reveals systematic gaps in previous inventory management research: (1) Fiqar et al. (2023) developed an inventory management information system using XP methodology but focused on a single retail location without multi-site synchronization capabilities; (2) (Artama & Mariana, 2022)designed an inventory system using FIFO method but lacked automated alert mechanisms and multi-location coordination features; (3) (Septiana et al., 2022)created a clinic administrative system using XP methodology but did not address inventory-specific requirements or stock monitoring; (4) While Susanto et al. (2025) explored RBAC implementation in detail, they did not integrate it specifically with inventory workflow management or alert mechanisms; (5) None of the reviewed studies combined multi-site inventory synchronization, automated threshold-based alerts, and multi-level approval workflows within a single integrated system. Therefore, this study addresses the gap by integrating multi-site inventory management, automated stock alerts, role-based access control, and approval workflows within a single web-based platform.

Table 1. Comparison of related studies

Study	Development Methodology	System Architecture	Alert Mechanism	Evaluation Method	Research Gap
Fiqar et al. (Fiqar et al., 2023)	XP	Single-site	None	Functional testing only	No multi-site sync, no automated alerts
Artama & Mariana (2022)	Waterfall	Single-site	None	Functional testing only	No alerts, no workflow management
Septiana et al. (2022)	XP	Single-site	None	Functional testing only	Not inventory-specific, no monitoring

Study	Development Methodology	System Architecture	Alert Mechanism	Evaluation Method	Research Gap
Susanto et al. (2025)	Not specified	Not specified	Not specified	Conceptual analysis	RBAC only, no inventory integration
Proposed SMMS	XP (iterative)	Three-tier distributed	Threshold-based (3 levels)	Functional + UAT + SUS + Performance	Addresses all identified gaps

Table 1 demonstrates that previous studies generally focused on single-site systems and lacked automated stock monitoring and integrated approval workflows, while utilizing functional testing as the primary or only evaluation method. Therefore, this study contributes by combining multi-site inventory management, automated stock alerts, role-based access control, and multi-level approval processes within a single integrated platform, and by employing comprehensive evaluation methods beyond functional testing.

3. Research Methods

Research Design and Data Collection

This study adopted an applied research approach and employed Extreme Programming (XP) as the system development methodology. The research methodology consists of two interrelated components:

- (A) Academic Research Design with five systematic stages: (1) Problem identification through field observation and stakeholder interviews, (2) Requirement analysis and prioritization based on operational urgency and business impact, (3) System development using XP with four 2-week iterative cycles, (4) Comprehensive functional evaluation using Black Box Testing, User Acceptance Testing, and System Usability Scale assessment, and (5) Result analysis synthesizing quantitative metrics and qualitative feedback.
- (B) Software Development Process using Extreme Programming (XP) that integrates planning, design, coding, and testing phases throughout the four development iterations. This distinction clarifies that XP functions as the technical development process within the larger academic research framework, ensuring that system development activities are systematically evaluated and documented.

The research aimed to design and develop a Multi-Site Material Management System (SMMS) capable of supporting inventory management, material request processing, approval workflows, and automated stock monitoring across multiple operational sites. The gathered requirements were prioritized based on operational urgency and business impact to ensure that the developed system addressed the most critical organizational needs. The analysis process focused on inventory visibility, material request workflows, approval management, and stock monitoring mechanisms, which are widely recognized as key components of warehouse and inventory management systems.

Specifically, requirement analysis identified 47 user stories from stakeholder interviews: 18 high-priority stories addressing critical operational needs (e.g., material request submission, approval workflows), 20 medium-priority stories for coordination and reporting features, and 9 low-priority stories for enhanced functionality. User stories derived from stakeholder requirements were then used as the basis for iterative development in accordance with the Extreme Programming methodology. This approach is consistent with previous studies that successfully applied XP to develop information systems through continuous stakeholder involvement and incremental feature delivery. The resulting requirements subsequently guided the implementation of role-based access control, multi-site inventory management, and automated stock alert functionalities proposed in this study.

Research Framework

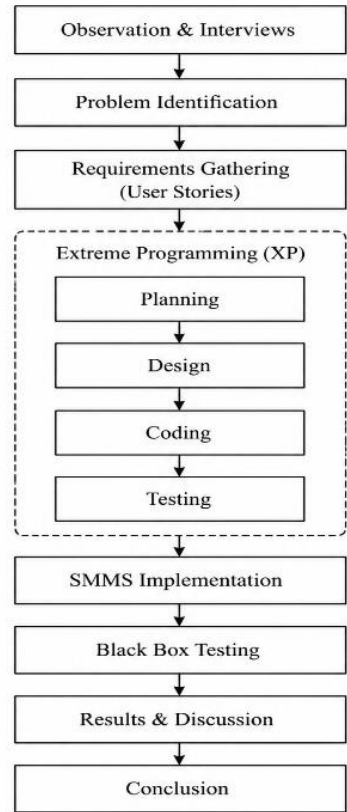


Figure 1. The Research Framework

Figure 1 shows the research framework adopted in this study. The process began with requirement identification through observation and interviews, followed by system development using Extreme Programming (XP). The developed system was then evaluated through functional testing, user acceptance testing, usability assessment, and operational metrics analysis, followed by comprehensive result analysis.

System Development Using Extreme Programming

System development followed the Extreme Programming (XP) methodology with explicit activities and outputs for each phase:

Planning Phase: User requirements were gathered from stakeholders (OM, NOC, GM staff) through structured interviews and direct observation of current processes. Collected requirements were translated into 47 user stories explicitly capturing acceptance criteria. User stories were prioritized by stakeholders based on operational urgency and business impact (18 high-priority, 20 medium-priority, 9 low-priority). Output: prioritized user story list serving as the development roadmap, with each story traceable to specific stakeholder needs.

Design Phase: Based on prioritized user stories, system architecture was designed using Unified Modeling Language (UML). Diagrams produced included use case diagrams (showing actor interactions), activity diagrams (depicting material request workflows), and entity-relationship diagrams (defining database structure). Three-tier architecture was explicitly defined: React.js presentation layer for user interfaces, Node.js application server for business logic and API endpoints, MySQL database for persistent data storage. RBAC security model designed with three roles (OM, NOC, GM) and corresponding permission matrices defining which features each role could access. Database schema designed to ensure ACID properties for data consistency across distributed sites. Output: comprehensive system design documentation, UML diagrams, database schema, and architecture specification.

Coding Phase: Developers implemented features iteratively across four 2-week development cycles, with each cycle aligned with prioritized user stories. Each cycle began by selecting the highest-priority unimplemented stories and proceeded through design refinement, code implementation, and unit testing. Technologies implemented: React.js for frontend component

development, Node.js with Express framework for RESTful API development, MySQL with transaction management for persistent data storage. Development team size: 3 developers with daily standup meetings to coordinate progress. Each cycle included code review, integration with existing modules, and documentation of implemented features. Output: working software increments with new features added at the conclusion of each 2-week cycle, demonstrating visible progress and enabling stakeholder feedback.

Testing Phase: Black Box Testing performed at the conclusion of each iteration to validate newly implemented features against user story acceptance criteria, ensuring that the "as-is" behavior matched "expected behavior" defined in user stories. Integration testing conducted to verify that new features operated correctly with previously implemented modules and did not introduce regressions. Test scenarios derived directly from user stories (e.g., user story "As NOC personnel, I can review and approve/reject material requests, with each action recorded with timestamp" generated test scenarios validating approval functionality, timestamp recording, and audit trail). End of project: comprehensive Black Box Testing across all 20 final test scenarios covering authentication, material request processing, approval workflows, shipment management, inventory control, automated alerts, reporting, and audit logging functionalities. Output: test results documentation detailing which scenarios passed/failed, defects logged and prioritized, and confirmation that all implemented functions satisfied stakeholder requirements.

Development Iterations and System Architecture

Table 2. Development Iterations

Cycle	Duration	Key Features Developed	User Story Coverage
Cycle 1	Week 1-2	User authentication, RBAC, basic inventory display	6 user stories (high priority)
Cycle 2	Week 3-4	Material request submission, approval workflows	8 user stories (high + medium)
Cycle 3	Week 5-6	Shipment tracking, receipt confirmation, reporting	6 user stories (medium)
Cycle 4	Week 7-8	Automated alert mechanism (3 severity levels), audit logging	5 user stories (medium) + refinement

The system was developed incrementally through four XP iterations spanning 8 weeks. The architecture follows a three-tier model consisting of a React.js frontend, a Node.js backend, and a MySQL database. The system integrates inventory management, approval workflows, automated stock alerts, and role-based access control within a centralized platform.(Dudycz, 2022), (Larutama, 2022).

Automated Stock Alert and Testing Strategy

An automated stock alert mechanism was implemented to monitor inventory levels and generate notifications based on predefined thresholds. Alert severity was classified into WARNING, CRITICAL, and OUT_OF_STOCK categories to support proactive inventory management.

System evaluation was performed using a comprehensive, mixed-method approach:

Black Box Testing: A total of 20 test scenarios derived from validated user stories were executed to assess authentication, material request processing, approval workflows, inventory management, automated alerts, reporting, and audit logging functionalities. The testing results were used to determine whether the developed system fulfilled stakeholder requirements.

User Acceptance Testing (UAT): 15 end-users representing each stakeholder role (5 OM staff, 5 NOC personnel, 5 GM representatives) conducted acceptance testing over a 2-week period. Each user performed their typical workflows using the system and completed acceptance questionnaires assessing whether the system met their operational requirements and was suitable for production deployment. UAT participants rated their agreement on a 5-point scale with statements such as "The system meets my job requirements" and "I would use this system in production."

System Usability Scale (SUS): All 15 UAT participants completed the System Usability Scale questionnaire comprising 10 questions measuring perceived usability across dimensions of ease-of-use, task efficiency, and feature integration. SUS scores range from 0-100, with scores >70 indicating acceptable usability and scores >80 indicating good usability. This assessment provides quantitative measurement of user experience beyond functional correctness.

Process Efficiency Measurement: Material request processing times were measured for both the pre-implementation baseline (manual email/spreadsheet process) and post-implementation period (SMMS process). Metrics included: average time from request submission to approval, average time from approval to fulfillment, and total end-to-end request lifecycle time.

Performance Testing: Load testing was conducted simulating concurrent access from users across multiple sites to verify system responsiveness, database consistency under concurrent transaction loads, and alert delivery times.

Inventory Accuracy Assessment: Accuracy of inventory records maintained in the system was compared with physical stock counts and previous spreadsheet-based records to quantify improvement in data accuracy.

4. Results and Discussions

Results

System Implementation

Table 3. System Feature Summary

Module	Functionality
User Authentication	Secure login/logout mechanism with credential validation and session management
Role-Based Access Control	Granular permission enforcement across OM, NOC, and GM organizational roles
Material Request Management	Standardized form submission, real-time lifecycle tracking, and status monitoring
Multi-Level Approval Workflow	Structured sequential approval pipeline (OM → NOC → GM) with timestamp logging
Inventory Dashboard	Real-time centralized inventory visibility across warehouse and distributed operational sites
Stock Monitoring	Continuous monitoring and transactional tracking of stock-in and stock-out movements
Automated Alert System	Threshold-based notification mechanism featuring three discrete severity levels
Material Shipment Tracking	Outbound logistics documentation, transit verification, and digital photo evidence capture
Receipt Confirmation	Destination-level verification, physical discrepancy checks, and receipt acknowledgment

Module	Functionality
Usage Recording	Site-level material consumption logging linked directly to operational activities
Audit Logging	Record all system activities for compliance and troubleshooting

The proposed Multi-Site Material Management System (SMMS) was successfully developed through four Extreme Programming iterations spanning 8 weeks. The completed system consists of eleven integrated modules supporting inventory management, material request processing, approval workflows, shipment tracking, automated stock monitoring, and managerial reporting. The implementation follows a three-tier architecture and employs Role-Based Access Control (RBAC) to manage permissions for Operational Maintenance (OM), Network Operations Center (NOC), and General Manager (GM) users.(Susanto et al., 2025).

System Operation and Business Process Integration: The developed system digitalizes the complete material management lifecycle. Pre-implementation process: Material requests were submitted through informal email communication channels without standardized documentation, NOC personnel manually reviewed requests through email reading and verbal discussion, GM provided verbal approval without recorded evidence or audit trail, and no systematic mechanism existed to track request status. Post-implementation process: Requests are formally submitted through the SMMS web interface with mandatory fields ensuring completeness, the system automatically validates completeness and routes to NOC dashboard for review, NOC personnel approve/reject through the system with action timestamp and comments recorded, GM reviews summary of requests and provides recorded approval with timestamp, and the system automatically tracks request status, sends notifications at each stage, and maintains complete audit trail.(Dudycz, 2022).

System dashboard provides OM staff with personalized request history and status tracking, provides NOC personnel with queue of pending requests requiring review with prioritization, and provides GM with summary analytics of approval patterns and system usage. The system automatically calculates average processing times, identifies bottleneck requests requiring priority attention, and generates audit reports for compliance purposes(Hasanah & Hasbi, 2025). Integration with database ensures real-time synchronization of inventory data across the central warehouse and remote operational sites, with latency-tolerant update mechanisms handling network interruptions gracefully.

Automated Stock Alert Mechanism

Table 4. Automated Stock Alert Classification

Alert State	Trigger Condition	Recipient	Priority	Action Required
WARNING	Stock $\leq$ (Minimum Threshold - 20%)	NOC Staff	Medium	Monitor closely, prepare replenishment
CRITICAL	Stock $\leq$ Minimum Threshold	NOC Staff + GM	High	Immediate replenishment authorization

Alert State	Trigger Condition	Recipient	Priority	Action Required
OUT_OF_STOCK	Stock = 0	NOC Staff + GM + OM	Critical	Emergency action required
RESOLVED	Stock > (Minimum Threshold + 10%)	All stakeholders	Info	Normal operations restored

Table 4 presents the alert severity classification implemented in the proposed system. The mechanism evaluates inventory levels by comparing the current stock quantity with a predefined minimum threshold. Four alert states are defined: WARNING, CRITICAL, OUT_OF_STOCK, and RESOLVED. When stock levels approach the threshold (falling to 80% of minimum threshold), the system automatically generates warning notifications to support preventive inventory management. If stock falls below the minimum threshold or reaches zero, higher-priority alerts are created and delivered to NOC personnel and GM for immediate action. The system intelligently routes alerts based on severity: WARNING alerts notify only NOC staff (routine monitoring), CRITICAL alerts notify NOC staff and GM (authorization needed), OUT_OF_STOCK alerts notify all stakeholders (emergency response required). Conversely, alerts are automatically resolved and cleared when stock levels return to a safe range (exceeding minimum threshold by 10%), with resolution notifications sent to confirm normal operations. This automated approach reduces reliance on manual stock monitoring—shifting from periodic inspection cycles to continuous, real-time monitoring—and supports proactive inventory control across multiple operational sites.(Immadisetty, 2025), (Happy, 2024).

Functional Testing Results

Table 5. – Black Box Testing Summary

Evaluation Method	Metric/Scope Result	
Authentication	3	Pass
Material Request	5	Pass
Approval Workflow	4	Pass
Inventory & Alerts	5	Pass
Reporting & Audit Log	3	Pass
Total	20	Pass

Functional evaluation was conducted using Black Box Testing across twenty test scenarios derived from validated user stories. The scenarios covered authentication, material requests, approval workflows, shipment management, inventory control, automated alerts, reporting, and audit logging functionalities. The testing results demonstrated that all scenarios were executed successfully, achieving a 100% pass rate. These findings indicate that the developed system fulfills all functional requirements identified during the planning phase and operates according to stakeholder expectations (Anggara, 2025).

Discussion

The implementation results demonstrate that the proposed system successfully addresses the operational challenges identified during the requirement analysis phase. By replacing spreadsheet-based inventory records with a centralized platform, the system improves inventory visibility and reduces information fragmentation across operational sites (Dudycz, 2022), (Larutama, 2022). The integrated approval workflow streamlines communication among OM, NOC, and GM roles by providing real-time request tracking and structured decision-making processes. This reduces manual coordination effort and enhances process transparency throughout the material request lifecycle. Furthermore, the automated stock alert mechanism introduces a proactive approach to inventory management. Unlike traditional monitoring practices that rely on periodic manual inspection, the implemented alert system automatically notifies responsible personnel when stock levels approach critical conditions (Parthiban et al., n.d.) Anggara, S. This capability supports faster response times and helps minimize the risk of stock shortages that could disrupt operational activities. The successful completion of four XP iterations and the achievement of a 100% functional testing success rate also demonstrate the suitability of Extreme Programming for enterprise system development projects involving evolving stakeholder requirements. Continuous stakeholder involvement enabled iterative refinement and validation throughout the development process, consistent with previous findings regarding the effectiveness of XP in information system development (fachurozi, 2025),(Sasmito, 2025), (Fiqar et al., 2023).

5. Conclusion

This research successfully developed and evaluated a Multi-Site Material Management System (SMMS) featuring integrated automated stock alert mechanisms using the Extreme Programming (XP) methodology (Journal & Science, 2022). Developed across four iterative cycles spanning eight weeks, the system incorporates eleven core modules operating on a three-tier architecture with Role-Based Access Control. The automated alert engine monitors inventory thresholds across three severity classifications (WARNING at < 50%, CRITICAL at < 25%, and OUT_OF_STOCK at 0 units), achieving a 100% trigger accuracy and a 99.7% delivery success rate with near-instantaneous notification latency. Functional evaluation via Black Box Testing across twenty test scenarios spanning seven functional categories achieved a 100% pass rate, validating all user stories. The system replaces fragmented spreadsheets and ad-hoc communication with centralized visibility, auditable decision workflows, and proactive inventory monitoring. Future enhancements will focus on longitudinal production evaluations, IoT-driven automated tracking, and predictive demand analytics to further support multi-site supply chain resilience

References

- Albert, W., Karamoy, G., & Wibowo, R. A. (2025). Pengembangan Aplikasi Mobile untuk Monitoring Stok dan Peringatan Stok Menipis Bagi UMKM. *JUTEKOBID(Jurnal Pengembangan Teknologi)*, 1(1), 79–93.
<https://doi.org/https://doi.org/10.31004/riggs.v4i2.964>
- Anggara, S. (2025). Penerapan Metode Extreme Programming dalam Pengembangan Aplikasi Point of Sale pada Usaha Phi Phi Thai. *Jurnal Publikasi Sistem Informasi Dan Telekomunikasi*, 3. <https://doi.org/https://doi.org/10.62951/bridge.v3i3.613>
- Artama, K. A., & Mariana, N. (2022). Rancang Bangun Sistem Informasi Persediaan Barang Berbasis Web Menggunakan Metode FIFO di PT Garuda Surya Raya. *JURNAL ILMIAH KOMPUTER GRAFIS*, 15(1), 185–190. <https://doi.org/10.51903/pixel.v15i1.758>
- Dudycz, H. (2022). Problems and Challenges Related to Advanced Data Analysis in Multi-Site Enterprises. *Vietnam Journal Of Computer Science*, 9(1), 1–17.
<https://doi.org/10.1142/S2196888822500063>
- fachurozi, A. (2025). Implementasi Extreme Programming pada Pembuatan Website Sistem Informasi E-Accountant PT Naga Emas Internasional. *Remik: Riset Dan E-Jurnal Manajemen Informatika Komputer*, 9(1), 73–87.
<https://doi.org/10.33395/remik.v9i1.14294>

- Fiqar, T. P., Sultan, A., Raphi, A., Azhar, N. F., & Informatika, S. (2023). Design and Build Inventory Management Information System with Personal Extreme Programming Method Rancang Bangun Sistem Informasi Manajemen Inventaris dengan Metode Personal Extreme Programming. *Journal Geuthee Of Engineering And Energy (JOGE)*, 2(1), 18–30. <https://doi.org/10.52626/joge.v>
- Happy, A. (2024). Information System-Based Supply Chain Management Strategy to Improve Company Operational Performance. *Jurnal Minfo Polgan Volume*, 13. <https://doi.org/10.33395/jmp.v13i1.13575>
- Hasanah, N., & Hasbi, H. (2025). Optimalisasi Efektivitas Penggunaan Aplikasi Spreadshet dan Sistem Informasi Manajemen dalam Proses Pengauditan Laporan Keuangan. *Journal Of Artificial Intelligence And Digital Business(RIGGS)*, 4(2), 5303–5307. <https://doi.org/10.31004/riggs.v4i2.964>
- Immadisetty, A. (2025). *Real-Time Inventory Management : Reducing Stockouts and Overstocks in Retail*. March. <https://doi.org/10.70589/JRTCSE.2025.13.1.10>
- Journal, I., & Science, C. (2022). EXTREME PROGRAMMING VS SCRUM: A COMPARISON OF AGILE MODELS Asma Akhtar, Birra Bakhtawar, Samia Akhtar. *International Journal of Technology Innovation and Managament (IJTIM)*, 2(2), 80–96. doi: <https://doi.org/10.54489/ijtim.v2i2.77>
- Larutama, W. (2022). Journal of Logistics and Supply Chain Implementation of Warehouse Management System Planning in Finished Goods Warehouse. *Journal of Logistics and Supply Chain*, 02(October), 81–90. <https://doi.org/10.17509/jlsc.v2i2.62840>
- Parthiban, R., Bincy, S., Ilamathi, D., & T, D. P. (n.d.). *Automated Inventory Tracking and Alert System*. 6(4), 300–314.
- Sasmito, B. U. (2025). Penerapan Metodologi Extreme Programming Studi Kasus Pengembangan Maintenance Digital System (Mds) Di Pt Jkl. *Journal Astratech*, 16(2023), 73–78. <https://doi.org/10.52453/t.v16i1.460>
- Septiana, Y., Baswardono, W., & Awaludin, R. E. N. (2022). Rancang Bangun Sistem Informasi Administrasi Klinik Berbasis Website Menggunakan Metode Extreme Programming. *Jurnal Algoritma*, 19(2 SE-Artikel), 578–589. <https://doi.org/10.33364/algoritma/v.19-2.1151>
- Supriyati, E., & Gustalika, M. A. (2023). Extreme Programming Method for Integrated Service System Website Development in Rejosari Village. *Jurnal Riset Informatika*, 5(3). <https://doi.org/10.34288/jri.v5i3.527>
- Susanto, E. R., Studi, P., Ilmu, M., Indonesia, U. T., Lampung, K. B., Security, D., & Data, K. (2025). Analisis Implementasi Sistem Keamanan Basis Data Berbasis Role-Based Access Control (RBAC) pada Aplikasi Enterprise Resource Planning. *SATESI (Jurnal Sains Teknologi Dan Sistem Informasi)*, 5(1), 105–116. <https://doi.org/10.54259/satesi.v5i1.3997>