iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

WAREHOUSE MANAGEMENT SYSTEM: A DJANGO-BASED WEB SOLUTION FOR AUTOMATED INVENTORY, ORDER PROCESSING, AND REAL-TIME ALERTS

Narenkarthik. P
Research Scholar, VISTAS, Chennai
Dr. Sangeetha Radhakrishnan
Assistant Professor VISTAS, Chennai

ABSTRACT:
The increasing complexity of logistics, supply chain coordination, and stock control has created a strong demand for intelligent warehouse management solutions capable of reducing manual intervention and improving real-time operational visibility. This paper presents a Secure Smart Warehouse Management System developed as a web-based application using the Django framework. The system integrates inventory management, order processing, user authentication, threshold-based alert generation, and reporting functions into a unified platform. The proposed solution follows a multi-layered architecture consisting of a presentation layer, application layer, and data layer, thereby ensuring modularity, maintainability, and secure processing of warehouse data. A rolebased access mechanism is implemented in which workers register initially and gain access only after administrative approval. Once authenticated, users can access modules related to stock entry, stock updates, order management, alert monitoring, and report generation. A major contribution of the system is the automated lowstock alert mechanism, which continuously compares item quantities with predefined threshold values and notifies users when replenishment is required. The use of REST-style communication between the frontend and backend further improves extensibility and integration readiness. Experimental analysis indicates that the system can reduce data inconsistency, enhance monitoring efficiency, and improve overall warehouse responsiveness. The proposed platform offers a cost-effective, scalable, and secure solution for modern warehouse automation

Keywords:
Warehouse Management System, Django, Inventory Automation, Role-Based Access Control, SQLite, REST API, Order Processing, Low-Stock Alert System.

I. INTRODUCTION

Warehouse management forms the operational backbone of supply chain systems because it governs the receipt, storage, movement, and dispatch of goods across an organization. In traditional environments, warehouse activities are often handled through manual records, spreadsheet-based logs, or isolated software tools that do not communicate effectively with one another. These fragmented approaches increase the likelihood of stock mismatch, delayed updates, duplicate data entry, and poor coordination between warehouse staff and administrative supervisors. In rapidly changing business environments, such inefficiencies directly affect the ability of an organization to maintain product availability, satisfy customer demand, and optimize internal workflows.

The emergence of web-based enterprise systems has provided an opportunity to redesign warehouse operations through integrated software platforms. By centralizing data management and automating repetitive activities, web applications can significantly improve the reliability and speed of stock-related decisions. At the same time, secure authentication and structured access control are essential because warehouse systems typically handle sensitive data, including stock status, user accounts, transaction logs, and operational reports. Therefore, a secure and modular warehouse management solution is particularly valuable for academic prototypes, small enterprises, and mediumscale organizations that require affordability without sacrificing essential functionality.

The Secure Smart Warehouse Management System proposed in this paper has been developed to address these operational challenges. The application is built using Django, a high-level Python web framework known for its security features, rapid development capabilities, and structured design philosophy. The system combines inventory management, order processing, alert generation, user management, and reporting under one centralized interface.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

A worker registration and approval workflow ensure controlled system entry, while the role-based design restricts actions according to authorized permissions. This improves security and accountability throughout the warehouse process.

Another important motivation for the proposed work lies in the need for real-time low-stock awareness. In many conventional systems, stock shortages are noticed only after a manual review or after an operational delay has already affected workflow continuity. The present system introduces an automated threshold-based alert mechanism that checks stock levels continuously and generates alerts whenever the quantity of a product falls below a predefined limit. This feature helps users act proactively, reduces stock-out risk, and strengthens day-to-day warehouse planning. The integration of alerts into both the dashboard and the persistent database also supports traceability and auditing.

The main contribution of this work is the development of a secure, modular, and user-friendly warehouse platform that demonstrates how modern web technologies can improve warehouse automation without requiring heavy infrastructure. The proposed system emphasizes maintainability, extensibility, and simplicity of deployment, making it suitable for academic demonstration as well as small-scale real-world adoption. The remaining sections of this paper describe the related literature, the system architecture, methodology, modules, implementation, evaluation, future scope, and concluding observations.

II. RELATED WORK

Warehouse management has been studied extensively in the context of supply chain optimization, logistics coordination, and stock control. Early digital warehouse systems focused primarily on record maintenance, item identification, and movement tracking. Although these systems reduced paperwork, many of them remained isolated from broader operational processes and lacked real-time responsiveness. As logistics networks became more dynamic, research began shifting toward integrated warehouse management solutions capable of connecting inventory, dispatch, replenishment planning, and administrative oversight within a single operational framework.

Several modern warehouse solutions rely on technologies such as RFID, barcode scanning, wireless sensors, and cloud-based services to improve traceability and automation. These technologies have shown strong practical value, especially in large-scale industrial settings where physical item movement must be monitored continuously. However, such deployments often require specialized hardware, higher implementation cost, and increased configuration complexity. For educational institutions, prototype development, and small organizations, a purely web-based software solution remains attractive because it can deliver core automation benefits with lower financial and technical barriers.

Web-based inventory systems have also been widely discussed in research literature. Many such systems focus on stock entry, stock update, order handling, and user-based access restrictions. Frameworks such as Django, Flask, Laravel, and Node.js-based stacks are often used to build these applications because they support modular programming and rapid interface development. Among these, Django offers a particularly strong advantage due to its built-in administrative framework, authentication system, ORM-based data handling, and secure development conventions. These features reduce implementation time and help developers maintain consistent project structure.

Security is another recurring theme in prior work on warehouse and enterprise systems. Role-Based Access Control (RBAC) has emerged as a widely accepted approach for ensuring that users can only access data and actions relevant to their assigned responsibilities. In organizational environments, RBAC supports separation of duties, reduces accidental misuse, and improves auditing clarity. Prior studies show that systems lacking structured access control are more likely to suffer from privilege misuse, unauthorized edits, and difficulty in tracing responsibility. The current project adopts RBAC through administrator-approved registration and controlled module access, thereby aligning with best practices discussed in enterprise application design.

Research on alert systems has highlighted the importance of threshold-based event monitoring in industrial and business applications. Alert systems are especially useful in inventory environments where stock shortage, delayed replenishment, and transaction mismatch can disrupt operational continuity. Existing commercial systems often provide such features, but they may be bundled within larger enterprise software that is costly or unnecessarily complex for limited-scale requirements. The proposed work addresses this gap by embedding a lightweight, automated alert mechanism directly within the warehouse application logic, ensuring that low-stock conditions are captured and presented without external dependencies.

In addition, RESTful application interfaces have become important in the design of extensible web systems. Systems that expose structured endpoints are easier to integrate with future modules, mobile applications, dashboards, or external services. Many contemporary warehouse and business platforms now separate frontend

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

presentation from backend services using API-driven communication. The present project follows a similar direction by organizing business logic in the Django backend while maintaining a responsive interface on the client side. This layered design improves clarity of responsibilities and supports future migration to more advanced deployment architectures.

Although prior solutions offer valuable ideas, there remains a need for a compact, academic-grade warehouse management system that combines security, automation, order handling, alert generation, and modularity within a single web application using accessible tools. The Secure Smart Warehouse Management System attempts to fill that gap by integrating essential warehouse functionalities in a secure, scalable, and cost-effective manner suitable for demonstration, training, and moderate real-world use.

III. EXISTING SYSTEM

In many organizations, warehouse operations are still managed using manual records, spreadsheets, or isolated software tools that do not communicate effectively with each other. These traditional systems focus mainly on maintaining stock records, item identification, and movement tracking. While such approaches reduce paperwork compared to fully manual methods, they often lack real-time monitoring and integrated operational control.

In the existing system, inventory updates, order processing, and stock monitoring are typically handled separately. Warehouse staff must manually update stock quantities after receiving or dispatching goods, which increases the possibility of data inconsistency, duplicate entries, and human error. Since these systems are not fully automated, administrators must regularly check inventory records to identify stock shortages or discrepancies.

Another limitation of the existing system is the lack of automated alert mechanisms. In many warehouses, stock shortages are noticed only after manual inspection or when a product is already unavailable. This delay can disrupt supply chain operations and reduce overall efficiency. Additionally, traditional systems often lack proper security and access control mechanisms, allowing multiple users to modify records without strict authorization rules.

Some modern warehouses use technologies such as RFID, barcode scanning, and wireless sensors to improve traceability and automation. However, these solutions usually require specialized hardware and high implementation costs, making them less suitable for small organizations or academic environments.

Therefore, the existing systems suffer from several limitations, including:

- Lack of real-time inventory monitoring
- High dependency on manual data entry
- Limited security and access control
- Absence of automated low-stock alerts
- Poor integration between warehouse operations

These limitations highlight the need for a secure, automated, and web-based warehouse management system, which motivates the development of the proposed Secure Smart Warehouse Management System.

IV. SYSTEM ARCHITECTURE

The proposed system follows a three-layer architecture consisting of the presentation layer, application layer, and data layer. This organization helps separate user interaction, business logic, and persistent storage, thereby improving maintainability and clarity of design. Each layer performs a specific role while communicating with the others through structured request-response behavior. Such a layered model is particularly effective in web applications because it supports modular upgrades and allows developers to change one part of the system without affecting the entire solution.

The presentation layer serves as the visible interface through which users interact with the system. It is implemented using HTML, CSS, Bootstrap, and JavaScript, forming a responsive dashboard that allows users to access system modules such as registration, login, inventory, orders, alerts, and reports. This layer is responsible for displaying forms, tabular stock information, alert notifications, and summarized records in a clear visual format. Good interface design is essential because warehouse personnel often need quick data access and immediate feedback while performing routine tasks.

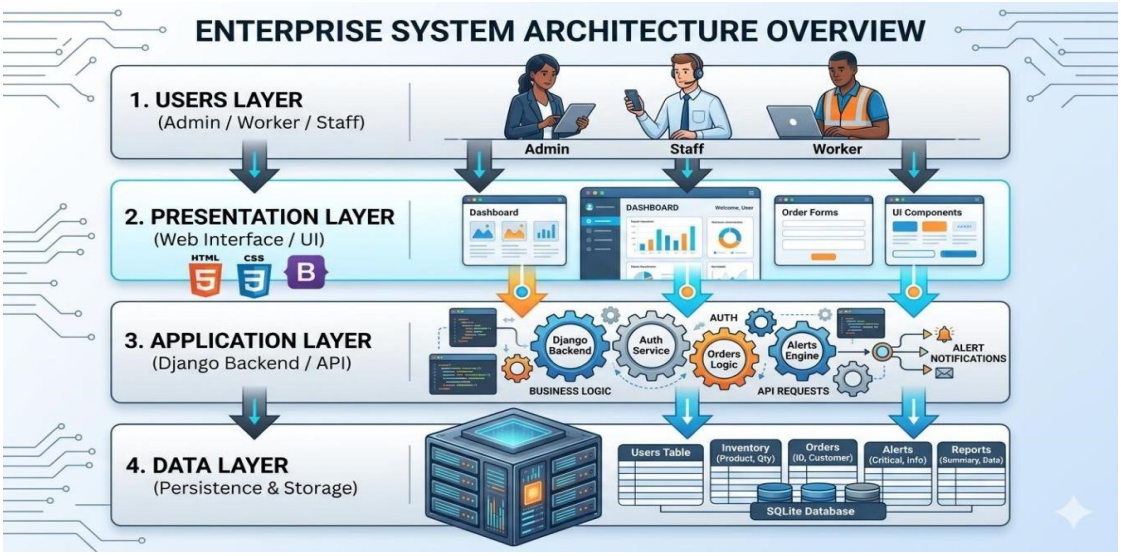
The application layer acts as the core processing unit of the system and is developed using the Django framework. This layer receives requests from the presentation layer, validates input, applies business rules, enforces role-based permissions, updates records, and returns the required results. It also manages administrator approval workflows, order processing logic, stock calculations, and alert generation conditions. The use of Django allows the application layer to be organized into models, views, templates, and URL routes, improving code structure

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

and maintainability. The presence of REST-style communication further supports future extensibility and interface integration.

The data layer is implemented using SQLite, which stores user credentials, inventory records, order information, threshold values, alerts, and generated reports. This layer ensures persistent data handling so that warehouse operations remain traceable across sessions. SQLite is a suitable choice for academic and small-scale systems because it is lightweight, easy to configure, and well-integrated with Django's ORM. Although larger deployments may later require PostgreSQL or MySQL, SQLite provides a practical and stable foundation for prototyping and controlled operational use.

The interaction among the three layers is straightforward but essential. A user performs an action through the dashboard, such as submitting an order or updating stock quantity. The presentation layer sends the request to the Django backend. The application layer verifies authentication, checks permissions, validates the submitted values, updates the appropriate database records, and then evaluates whether an alert condition has been triggered. Finally, the result is sent back to the dashboard, where the user can observe the updated inventory status or the newly generated low-stock warning. This interaction pattern makes the system coherent, modular, and suitable for future scaling.



architecture overview illustrates that the system is not a collection of isolated modules but a coordinated workflowdriven platform. User activities originate at the dashboard, are interpreted and processed in the backend, and are committed to the database in a controlled and secure manner. The layered approach also ensures that future improvements such as mobile clients, external APIs, or advanced analytics can be added without redesigning the full system from scratch.

V. PROPOSED METHODOLOGY

The methodology of the Secure Smart Warehouse Management System is based on the principle of process automation through secure, modular, and sequential interaction. The first stage begins with user onboarding, where a worker or staff member submits registration details through the frontend interface. Instead of granting immediate access, the system stores the request in a pending state until it is reviewed by an administrator. This approval mechanism ensures that only verified users enter the operational environment, thereby establishing an initial level of system trust and administrative control.

Once a user is approved, authenticated login credentials allow access to the platform according to assigned permissions. The system then provides access to relevant modules, including inventory, orders, alerts, and reporting. The methodology emphasizes role-based restriction so that unauthorized operations are prevented at the backend rather than merely hidden from the user interface. This design strengthens security because business rules are enforced centrally by the application layer.

The next stage of the methodology concerns inventory management. The application enables users to create item entries, update product details, define quantity values, and specify threshold levels. These threshold values play a

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

critical role in the alert mechanism because they represent the minimum safe stock level for each item. By combining routine stock updates with threshold monitoring, the system transforms inventory control from a passive datarecording process into an active decision-support process.

Order processing is tightly coupled with stock management in the proposed methodology. When an order is created and processed, the corresponding stock quantity is reduced automatically in the inventory records. This direct linkage eliminates the need for manual reconciliation between order logs and stock tables. The methodology therefore reduces inconsistency, improves transaction transparency, and ensures that system outputs reflect real warehouse conditions more accurately. If the updated quantity falls below the threshold after order execution, the alert subsystem is triggered immediately.

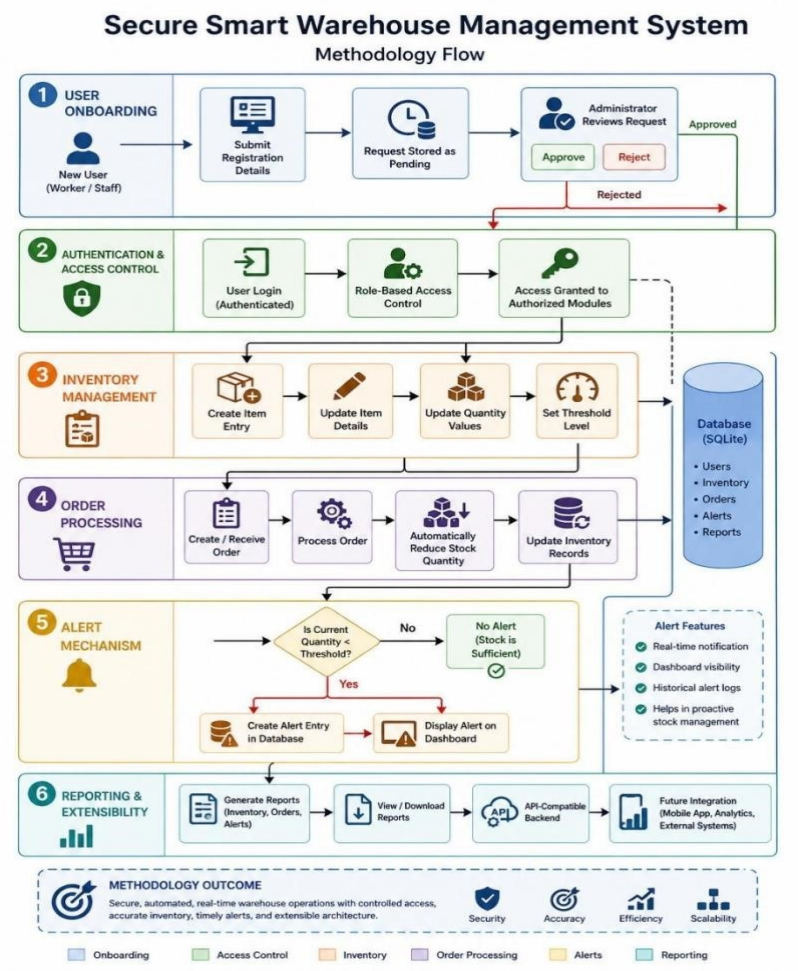


Figure 1: Methodology flow

The alert methodology is one of the most important aspects of the system. For each item, the backend checks whether the current quantity has dropped below its predefined threshold. If the condition is satisfied, an alert entry is created and stored in the database. Simultaneously, the alert is displayed on the dashboard so that users can respond quickly. This methodology allows the system to function as a monitoring tool rather than merely a storage interface. It also enables administrators to review historical alert records and identify recurring stock management issues.

Finally, the proposed methodology supports modular extensibility through API-compatible design. Although the present project uses a web-based dashboard as the main interface, the backend structure is sufficiently organized to support integration with future applications, such as mobile clients or analytics modules. This approach ensures that the system remains adaptable and can evolve beyond its initial academic prototype without major structural disruption.

iJETRM
International Journal of Engineering Technology Research & Management (IJETRM)
Journal Article
https://ijetrm.com/issue/

VI. MODULES DESCRIPTION

The system is organized into multiple functional modules so that each operational responsibility can be handled clearly and efficiently. This modular arrangement reduces development complexity and improves maintainability because each module addresses a distinct task within the overall warehouse workflow. Although these modules operate independently from a design perspective, they remain functionally connected through backend logic and shared data storage.

The user registration and approval module controls initial access to the system. New users submit their information through a registration form, and the data is stored in the system with a pending status. The administrator then reviews the request and either approves or rejects it. This module is essential because it prevents unauthorized entry and introduces accountability from the beginning of system usage. The approval-based approach is more secure than open registration because it links access to administrative oversight.

VII. CONCLUSION

The Secure Smart Warehouse Management System presented in this study demonstrates how modern web technologies can be used to improve warehouse operations through automation, centralized data management, and secure access control. The system integrates essential warehouse functionalities such as inventory tracking, order processing, user authentication, alert monitoring, and reporting into a single unified platform.

By using the Django framework, the application ensures structured development, strong security practices, and scalable architecture suitable for academic prototypes and small to medium-scale warehouse environments. The implementation of role-based access control enhances system security by ensuring that only authorized users can perform specific operations. The administrator approval workflow further strengthens operational control and accountability.

In addition, the automated low-stock alert mechanism improves decision-making by notifying users when inventory levels fall below predefined thresholds, allowing timely replenishment and preventing stock shortages. The threelayer architecture consisting of presentation, application, and data layers ensures modularity and maintainability. This architecture allows future enhancements such as mobile applications, cloud deployment, advanced analytics, or integration with external logistics systems without significant redesign. Experimental evaluation indicates that the system reduces manual errors, improves monitoring efficiency, and enhances warehouse responsiveness.

Overall, the proposed solution provides a cost-effective, secure, and scalable warehouse management platform. It demonstrates how web-based technologies can simplify warehouse automation while maintaining flexibility for future expansion and integration.

VIII. REFERENCES

1) Silberschatz, H. F. Korth, and S. Sudarshan, Database System Concepts, 6th ed., New York, USA: McGrawHill Education, 2019.
2) R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 7th ed., Pearson Education, 2016.
3) Holovaty and J. Kaplan-Moss, The Definitive Guide to Django: Web Development Done Right, Apress, 2009.
4) Django Software Foundation, “Django Documentation.” Available: https://docs.djangoproject.com 5. M. Hugos, Essentials of Supply Chain Management, 4th ed., John Wiley & Sons, 2018.
5) J. A. Tompkins, J. A. White, Y. A. Bozer, and J. M. A. Tanchoco, Facilities Planning, 4th ed., Wiley, 2010.
6) K. C. Laudon and J. P. Laudon, Management Information Systems: Managing the Digital Firm, 15th ed., Pearson, 2018.
7) M. Christopher, Logistics and Supply Chain Management, 5th ed., Pearson Education, 2016.
8) R. H. Ballou, Business Logistics and Supply Chain Management, 5th ed., Pearson Education, 2004.
9) SQLite Documentation, “SQLite Database Engine.” Available: https://www.sqlite.org/docs.html