International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 5, Sep-Oct 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Design and Development of a Web Based Warehouse Inventory Management System – Using Python, Streamlit and MySQL

Jenil Ravindra Bahiram, Bhavna Shriram Kulkarni, Mohnish Kiran Bhangale, Dhanshree Nilkanth Kuwar
(Department of Computer Science & Engineering (Data Science), R. C. Patel Institute of Technology, and Shirpur, maharashtra
Email: jenilbahiram@gmail.com , bhavnakulkarni85@gmail.com, mohnishbhangale@gmail.com, dhanshreekuwar4@gmail.com)

----------**********----------

Abstract:

Warehouse inventory management is an important function. However, if done manually or via an offline system it can cause delays to employees. Employees may struggle to check what stock is available, find out which products need restocking and track what has been delivered by suppliers and sent to customers. This can cause problems within the warehouse and increase chances of errors. This project looks at designing and developing a web based Warehouse Inventory Management System using Python, Streamlit, and MySQL.

The system allows you to manage products, product categories, stock in/out, low stock warnings, shipments and orders. Login is provided for administrators and storekeepers. The application was developed using Python for the logic and validation, Streamlit to create the interactive web application and MySQL to store and manage data. The dashboard and reports allow users to look at their inventory information and see what items may need restocking.

The functionality of the application will be tested. Testing will be performed for logging in, creating and managing products, stock in/out, identifying low stock, tracking shipments/orders and generating reports. The expected outcome should be compared to the actual outcome. When testing the stock out functionality you should ensure that when a request is made for more than is available it does not allow it to go through. The outcomes in the final paper will be based on the tests carried out so no statements can be made about how it improves performance, efficiency or productivity.

Overall this shows how simple technologies can be used to create a basic web based inventory application. By having all important functions within one system and storing the data in a relational database it can be a useful starting point. More testing would need to be carried out within a live environment to see how usable, secure, scalable and how it affects the warehouse.

Keywords — **warehouse inventory, inventory management system, Python, Streamlit, MySQL, stock monitoring, web application.**

----------**********----------

I. INTRODUCTION

Keeping track of inventory is important in a warehouse. It ensures staff know what products they have, how much stock they have and what stock movements have been made. If records are not well organised it can be hard to know what stock is coming in and what is going out. Also, it can be difficult to know what items are running low. An information system can be used to help with this. This will use a relational database to store the information. This project uses Python, Streamlit and MySQL to create a Warehouse Inventory

 ©IJSRED: All Rights are Reserved

Management System. Python was used for the application logic, Streamlit for the forms and dashboards. MySQL was used to store user, product, category, transaction and order information. The aim of this project is to create and test a working prototype for small scale warehouse processes.

1.1 Objectives

• A web interface for managing products and categories.
• Stock IN/OUT can be recorded and ensure stock cannot go negative.
• Low stock alerts can be seen when products have a quantity less than/equal to reorder amount.
• Track shipments from suppliers and dispatches to customers.
• Dashboard for summarising information and exporting inventory and transaction reports.
• Testing the main processes using functional test cases.

1.2 Contributions

This paper details how an integrated system can be created to record inventory, track stock movements, alert for reorder amounts, track orders, restrict access based on user roles and produce CSV reports. To show how novel this is compared to current systems a literature review must be completed.

II. RELATED WORK

[Add at least 5–10 relevant sources. Critically analyse them. Explain how they work and how they relate to this project. Search Google Scholar, DOAJ and your institution's library for sources. Read the sources before adding them to your manuscript.]

III. PROBLEM STATEMENT AND REQUIREMENTS

This system helps to centralise the management of product records, stock movements, reorder amounts, orders and reports. This is a working prototype for demonstration and testing purposes. Further security, reliability and usability considerations would be required for this to be used in a production environment.

3.1 Functional Requirements

• Initial admin user created. Registered users can be authenticated.
• Authorised users can view and manage product and category information.
• Stock IN/OUT can be recorded. Operations cannot be performed to make stock negative.
• Products with quantity ≤ reorder amount will show low stock.
• Record shipment information from suppliers and dispatches to customers. Update order status.
• Dashboard for viewing information. Selected data can be exported as CSV.
• Only admins can manage users.

IV. SYSTEM ARCHITECTURE AND TECHNOLOGY. SYSTEM ARCHITECTURE AND TECHNOLOGY

This application consists of three layers. The Presentation Layer was created using Streamlit. This includes pages, forms, tables and dashboards. The Application Layer was created using Python. This checks user input and manages inventory operations. The Data Layer was created using MySQL. This contains tables for users, categories, products, stock_transactions and orders. MySQL was connected using mysql-connector-python. Configuration for the application can be set locally by using environment variables or a .env file. When deployed this can be set using Streamlit secrets. [Figure 1. Diagram showing the architecture of the application: User/browser, Streamlit UI, Python application logic and MySQL database. Note key parts and the direction of data.]

4.1 Database Design

Table	Purpose
users	Stores user names, password hashes, and roles.
categories	Stores product category names.
products	Stores SKU, product name, category, quantity, reorder level, location, and unit price.
stock_transactions	Stores stock IN/OUT events, quantity, user, notes, and timestamp.
orders	Stores supplier shipment/customer dispatch details, product, quantity, date, and status.

 ©IJSRED: All Rights are Reserved

4.2 Low-Stock Rule

Any products with a quantity ≤ reorder amount will be flagged as low stock. This will be visible on the website. There is no mention of email/SMS alerts in the current documentation so this cannot be claimed.

V. IMPLEMENTATION

5.1 Authentication and Roles

There are two user roles (admin & storekeeper). An initial admin user will be created. Passwords are stored as PBKDF2-HMAC-SHA256 hashes with salt.

5.2 Product and Category Management

Users can manage product information and categories. SKU, product name, quantity, reorder amount, location and unit price can be added. SKU must be unique.

5.3 Stock Transactions

Stock IN/OUT can be recorded. The system ensures that stock cannot go below zero.

5.4 Order Tracking

Shipment information from suppliers and dispatches to customers can be recorded. An order code, product, quantity, date, counterparty and status can be added.

5.5 Dashboard and Reports

Dashboard for displaying information. Data can be exported as CSV for further analysis.

VI. TESTING METHODOLOGY AND RESULTS

Functional testing should be carried out on the version of the system detailed in this paper. Test cases should be carried out using sample data. Fill in the table below with the results from actual testing. Do not claim any tests were passed until they have been performed.

Test ID	Test case	Expected result	Observed result / status
T01	Log in with valid credentials	Authorized user reaches the application	[Run test]
T02	Log in with invalid credentials	Access is denied	[Run test]
T03	Add a product with a unique SKU	Product is saved and appears in inventory	[Run test]
T04	Add a duplicate SKU	Duplicate is rejected with an understandable error	[Run test]
T05	Record stock IN	Quantity and transaction history update correctly	[Run test]
T06	Attempt stock OUT greater than available quantity	Operation is rejected; stock does not become negative	[Run test]
T07	Set quantity at or below reorder level	Product appears in low-stock warning	[Run test]
T08	Create and update an order	Order details and status are retained	[Run test]
T09	Export a report	CSV file downloads and contains expected records	[Run test]
T10	Access user management as a storekeeper	Restricted page/function is unavailable	[Run test]

6.1 Quantitative Evaluation (Optional but Recommended)

To provide more evidence a repeatable sample dataset can be used. State how many test cases were passed. How long certain actions took and any errors encountered. If measuring response times state what hardware, software versions, how large the data set was, how many times it was repeated and if this was an average or median. Do not claim the system is faster than manual or existing systems unless a fair comparison was made.

VII. DISCUSSION

This provides an integrated solution to managing product records, stock movements, low stock alerts, order status and CSV exports. The technology stack used to develop this prototype is relatively simple and is suitable for educational purposes. This system will only be useful if accurate data is entered, database connections are reliable and users have the correct permissions. This system should be analysed against similar systems before claiming it is novel or better.

VIII. LIMITATIONS AND FUTURE WORK

- This system has not had an independent security audit. It should not be assumed this is ready to handle sensitive data.
- Currently there is no email/SMS alerts for low stock. Only alerts in the website.
- This requires a database that can be accessed from a remote location. The developers local MySQL database (localhost) will not be accessible to the application hosted in the cloud.

 ©IJSRED: All Rights are Reserved

• Further work can be done to include automatic backups and restores, improved session management, audit logs and password recovery.
• Further testing can be carried out with multiple users, larger data sets, measuring response times and user feedback. • Integration with barcode scanners, purchasing planning and outside logistics services could also be considered once requirements have been defined.

IX. CONCLUSION

This paper details the design of a web based Warehouse Inventory Management System using Python, Streamlit and MySQL. The prototype provides a way to manage products and categories, record stock movements, alert for reorder amounts, track orders, restrict access based on user roles and produce CSV reports. The next essential step is to complete and document functional testing. This should be analysed against related research. This system should be considered an educational prototype until security, data protection, back up and reliability has been considered for deployment.

REFERENCES

[1] Streamlit, "Streamlit documentation." https://docs.streamlit.io/ (accessed [insert date]).
[2] Oracle, "MySQL 8.4 Reference Manual." https://dev.mysql.com/doc/ (use the version matching the database tested).
[3] Python Software Foundation, "Python Documentation." https://docs.python.org/3/ (accessed [insert date]).
[4] Pandas Development Team, "pandas documentation." https://pandas.pydata.org/docs/ (accessed [insert date]).

 ©IJSRED: All Rights are Reserved