DEVELOPMENT OF REAL-TIME MULTI-LOCATION INVENTORY SYSTEM AT PT. AFA KARYALOKA NATA UTAMA USING V-MODEL

Arif Sa'banna Hasibuan[1)*], Tedi Setiadi[2)]
[1,2]*Informatics Study Program*
[1,2]*Faculty of Industrial Technology*
[1,2]*Ahmad Dahlan University*
E-mail :arif2100018469@webmail.uad.ac.id [1)], tedi.setiadi@tif.uad.ac.id [2)]

Abstract

Inventory management in companies with multiple locations often faces challenges when conventional tools such as spreadsheets are still used, causing inefficiency and difficulties in maintaining accurate records. Previous studies have largely focused on single-location systems without real-time integration, leaving a gap in supporting centralized and synchronized inventory control. This study aims to develop a web-based inventory information system that supports real-time and multi-location operations to improve efficiency and accuracy in stock management. The system was developed using the V-Model methodology, which includes requirements analysis, system design, architectural design, module design, implementation, and testing. Research data were obtained through interviews with company employees and literature review. System testing was conducted using the Black Box Testing method, which focuses on validating functionality based on inputs and outputs without producing quantitative data. Therefore, the evaluation was based on the conformity of the system with the specified requirements. The results show that the developed system is capable of handling core functions such as warehouse and counter stock management, shipping records, return records, and user management, all integrated into a centralized database with real-time synchronization across locations. This ensures that inventory data remains consistent and up to date. In conclusion, the research successfully addresses the identified gap by presenting a web-based, multi-location inventory information system with real-time integration, thereby supporting greater efficiency, accuracy, and control in inventory management.

Keywords- *Information System, Inventory, Multi-Location, Real-Time, V-Model*

1. INTRODUCTION

The development of information technology contributes greatly to the efficiency of company operations, especially inventory management. Information systems support fast and accurate data processing and effective decision making. Innovate Indonesia states that most manufacturing companies in Indonesia (64%) still rely on basic technologies such as manual record keeping, spreadsheets, and email, while 30% have adopted intermediate technologies such as SAP and Oracle, and only 6% have implemented advanced technologies. As operations scale up, companies need a multi-location inventory system that provides centralized, real-time data to keep inventory management in sync [1]

According to [2], inventory information systems play a role in managing stock, recording the movement of goods, and controlling inventory in a structured and integrated manner. This system allows real-time monitoring of goods and reduces the risk of shortages or excess stock.

As the scale of operations increases, companies need an inventory system that can handle multi-location. Companies with many branches require centralized and real-time inventory data to keep goods management in sync. According to [3], web-based systems speed up information distribution and allow data access anytime and anywhere, making stock control in various locations more efficient.

Good inventory management is important to maintain the smooth operation of the company.

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

[4] asserted that a computerized warehouse system records the movement of goods accurately and generates reports for stock evaluation. In contrast, manual systems are prone to errors, information delays, and data discrepancies that hinder work effectiveness. Based on preliminary observations at PT. Afa Karyaloka Nata Utama, discrepancies often occur between actual stock and manual records, which may lead to delays in distribution and impact decision-making.

The development of an inter-location inventory information system requires a systematic and structured software approach. The V-Model is considered suitable because each stage of system development is directly linked with its corresponding testing stage, allowing verification and validation to be carried out early and consistently [5]. This approach minimizes errors and ensures that integrated modules such as warehouse, store, refund, shipping, and sales can run reliably.

Several related studies have been conducted in this field. Naibaho [6] designed an inventory system using the V-Model at PT. Brickbern, but the system was limited in scope and lacked inter-unit synchronization as well as refund features. Similarly, Alfarisy and Muarie [7] developed a sales and inventory system for a single-location store, without supporting real-time integration between warehouse and counters. These studies demonstrate the need for a system that is not only web-based but also capable of handling real-time and multi-location inventory while covering broader operational functions.

Departing from these problems, this study focuses on designing and implementing a web-based inventory system at PT. Afa Karyaloka Nata Utama that integrates warehouse and counters in real time. The research addresses the challenge of ensuring stock validity by considering both stock additions and reductions, and evaluates the functional feasibility of the developed system through the V-Model approach. The objective of this research is to provide an integrated solution that improves the reliability of inventory management and supports operational decision-making within a multi-location business environment.

Black Box Testing is considered an effective approach when the source code of a system is incomplete or even unavailable, such as in the use of third-party components. This method is also suitable for testing systems or APIs with complex architectures that would be difficult to evaluate through white-box testing, especially when they consist of multiple services developed in different programming languages. One of its main advantages is that the testing process remains independent of the programming language or internal implementation used [8].

2. METHODOLOGY

The V-Model method is used in this research to build a web-based multi-location inventory information system. The V-Model is a development of the Waterfall model with similar stages, but more integrated. According to [9], the main difference lies in the relationship between the development and testing stages, where the V-Model describes the process symmetrically with testing directly connected to each development stage. This approach allows verification and validation to be done from the start, thus reducing errors and ensuring the system meets user needs.

The V model is so named because its structure resembles the letter V and functions as a framework in software development, as shown in Figure 1. In this model, the horizontal axis represents the sequence of time or the level of project completion moving from left to right, while the vertical axis shows the level of abstraction with the most general concepts at the top. The left side of the V model consists of top-down verification stages, while the right side consists of bottom-up validation stages. Each verification stage has a corresponding testing stage that is carried out for validation purposes[10].

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

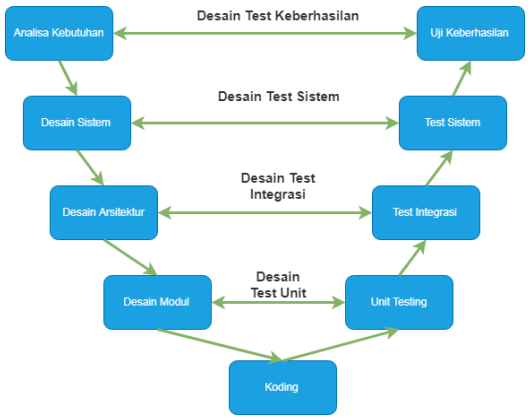


Figure 1. Methodology

The process begins with a needs analysis through interviews and observations, which are grouped into functional requirements (inventory management, shipping, refunds, sales, and access rights) and non-functional requirements (security, real-time, ease of interface, and system speed). Next, the system is designed using ERD and UML to model the database and workflow, followed by architectural design using a Three-Tier Architecture approach comprising Presentation Layer, Application Layer, and Data Layer. The module design stage covers the dashboard, users, warehouse, shipping, returns, sales, and counter, which are then implemented through coding. Recently, the system was tested using unit testing, integration testing, system testing, and success testing to ensure that all functions run as needed and support stable and real-time operations.

Data collection in this study was conducted through interviews and literature review. Primary data was obtained from interviews with one HRD staff member as a key informant who thoroughly understood the processes and requirements related to inventory management. The information provided covered important aspects that formed the basis for system development. Meanwhile, secondary data was obtained through a literature review of relevant literature and journals to strengthen the theoretical basis of the study.

The development of a web-based multi-site inventory information system begins with a needs analysis through user interviews and a review of the inventory management process to formulate system specifications. The next stage is design, including architectural design, database structure, user interface, and wireframe and flowchart creation. Implementation is done by developing a web application that includes stock management, shipping, refund, and sales features. Finally, the system is tested using the Black Box Testing method to ensure all functions run as needed before being fully implemented.

3. RESULTS AND DISCUSSION

3.1 System Requirements Analysis

System requirements analysis is a process for identifying and detailing user needs related to the information system to be created or developed [11]. Based on the results of interviews and observations that have been made, system requirements are classified into two main categories, namely functional and non-functional requirements. This classification is the basis for designing a Real-Time website-based multi-location inventory information system using the V-Model method.

Functional Requirements

Functional requirements describe the processes or services that the system must provide, including responses to inputs and behavior in certain situations [12]. To support warehouse operations, the system must have the following features or functions as functional requirements:

Table 1. Functional Requirements

No	Description
1.	The system supports login with valid credentials
2.	Easy-to-use system interface to input, edit, and delete item data
3.	The system records the name, description, and quantity of the item
4.	The system provides item search by name, category, or other attributes
5.	The system records stock changes: additions, subtractions, and adjustments
6.	The system supports setting access rights based on user roles

Non-Functional Requirements

Non-functional requirements describe the behavioral characteristics of the application, such as operations, performance, and information presentation [13]. For the system to

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

run optimally, the following components of excellence must be met as non-functional requirements:

Table 2. Non-Functional Requirements

No	Description
1.	Website accessible during working hours
2.	Fast and responsive website with low waiting time
3.	The system maintains data integrity to avoid damage or loss
4.	Data is encrypted and only accessed by authorized users
5.	Interface is easy to understand and use
6.	System runs stably with regular backup and recovery mechanisms
7.	System runs stably with regular backup and recovery mechanisms

3.2 System Design

This design aims to describe how the system will be built, starting from architecture, data structure, to user interface. At this stage, the author uses tools such as Entity Relationship Diagram (ERD) to describe the database structure and Unified Modeling Language (UML) to model the workflow and functions of the system.

Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) is used in database design to describe the relationship between entities and their attributes [14]. In this system, ERD models the relationship between entities such as goods, users, incoming and outgoing goods reports, and CRUD features in each warehouse. The ERD design is the basis for forming the table structure in the system database.

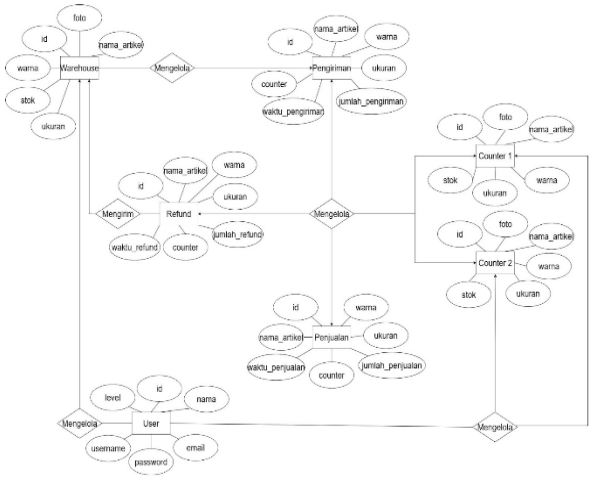


Figure 2. Entity Relationship Diagram

The Entity Relationship Diagram (ERD) shows the central role of the Warehouse entity as the main database that stores article information, connected with the Counter entity representing multiple branches. Features such as Shipping, Returns, and Sales ensure that every distribution, return, and sales activity automatically updates inventory between the warehouse and counters, supported by time attributes for real-time synchronization. The User entity manages system access and roles to maintain security and data integrity. This design emphasizes multi-location management and real-time stock synchronization as the core features of the system.

Use Case Diagram

One model that can be used to demonstrate the integrity of the information system to be built is the Use Case Diagram. This diagram shows the available system functions and the parties (actors) who have the right to utilize these functions [15].

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

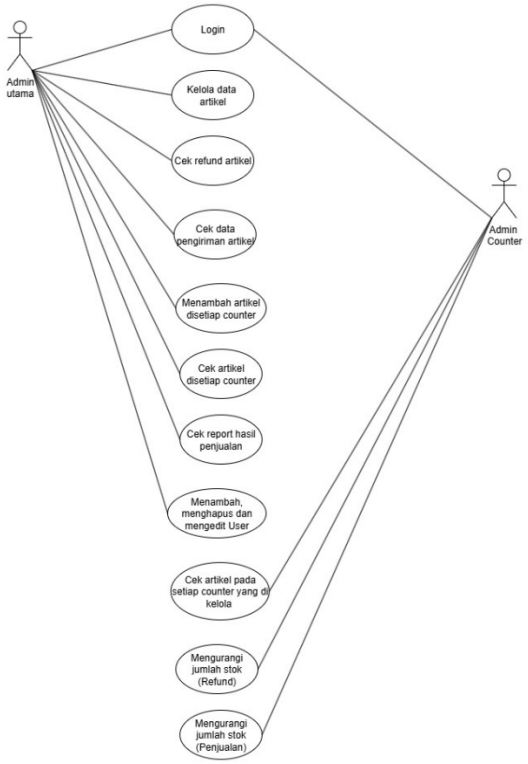


Figure 3. Use Case Diagram

In this web-based inventory information system, the diagram shows the roles of the Main Admin and Counter Admin. The Main Admin manages all the main functions of the system, such as articles, users, and reports. Meanwhile, the Counter Admin is in charge of managing the stock at each counter, including checking articles and stock settings for refunds and sales.

Sequence Diagram

A sequence diagram is a type of interaction diagram that depicts processes based on chronological order. Each diagram represents one flow from the various flows contained in a use case [16].

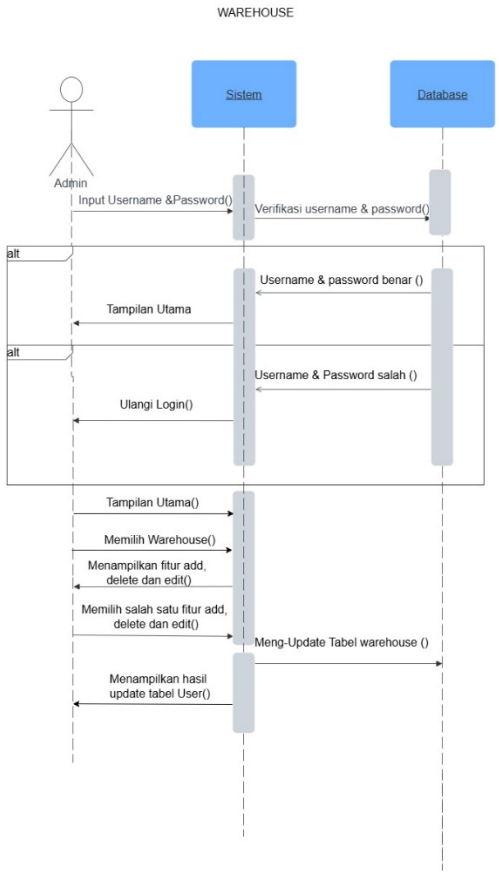


Figure 4. Sequence Diagram

The sequence diagram above illustrates the flow of Admin in managing the warehouse data. The process begins when the Admin inputs a username and password, which are then verified by the system through the database. If the credentials are correct, the system directs the Admin to the main page; otherwise, the system requests a re-login. Once successfully logged in, the Admin selects the warehouse menu where several features are available, including add, edit, delete, detail, and send. When one of these actions is chosen, the system processes the request by updating the warehouse table in the database. After the update, the system immediately displays the results on the user interface, ensuring that changes are recorded and synchronized in real time. This flow represents not only the warehouse process but also reflects the general interaction pattern applied across other modules such as user management, counter, sales, shipping, and refund, thereby supporting multi-location and real-time integration.

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

3.3 Architectural Design

At this stage, the system requirements that have been analyzed are poured into a transformation mapping as an initial representation of the system [17], then developed using a three-tier architecture consisting of Presentation Layer, Application Layer, and Data Layer, to describe the communication flow between users, applications, and databases in a web-based multi-location inventory information system.

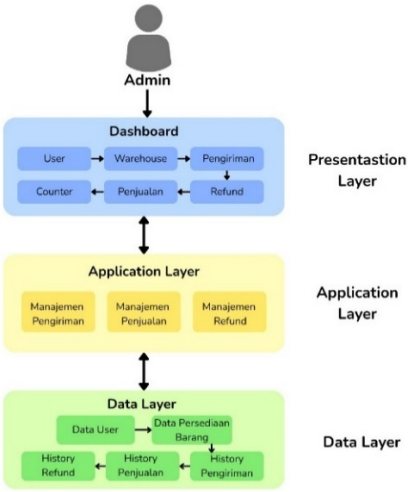
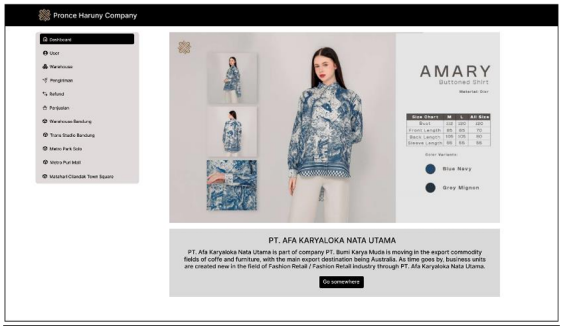


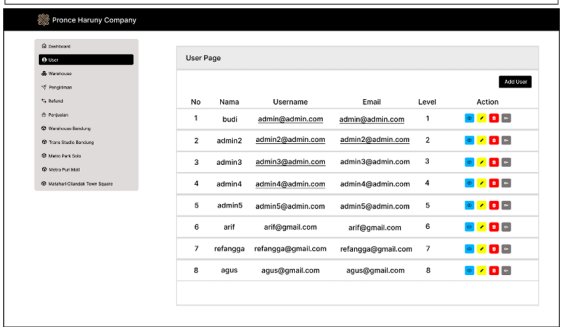
Figure 5. Architectural Design

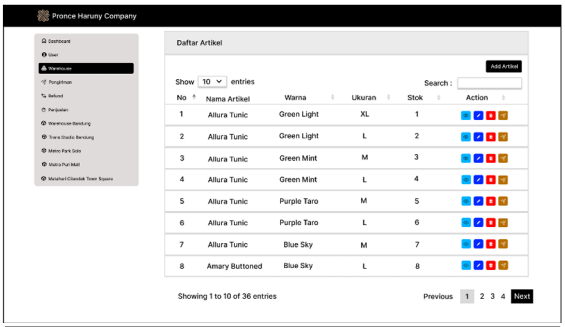
In this research, inventory information system is designed using a Three-Tier Architecture model, as illustrated in Figure 5. The architecture consists of three layers: Presentation Layer, Application Layer, and Data Layer. In the Presentation Layer, the admin interacts through a web interface to access menus such as dashboard, user management, warehouse, shipping, refunds, sales, and counters. The Application Layer is responsible for executing business logic, including shipment management, sales, and refund processing. Meanwhile, the Data Layer functions as a centralized storage that manages user data, inventory records, and transaction histories of shipments, sales, and returns. This three-tier approach not only structures the system more clearly but also ensures real-time synchronization and multi-location integration, while making the system easier to develop and maintain.

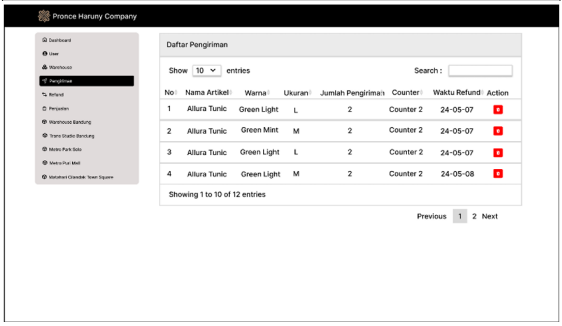
3.4 Module Design

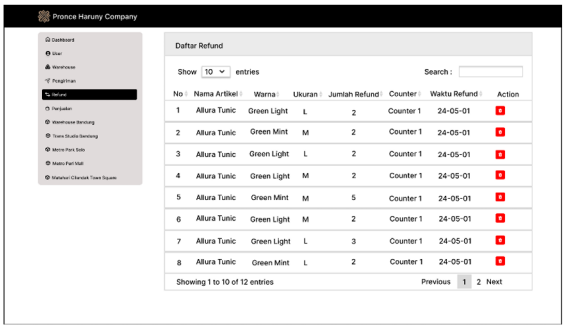
Module design is carried out to divide the system into functional parts that can be developed separately but remain integrated, with the aim of supporting real-time inventory management and centralized multi-location operational environments.











DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

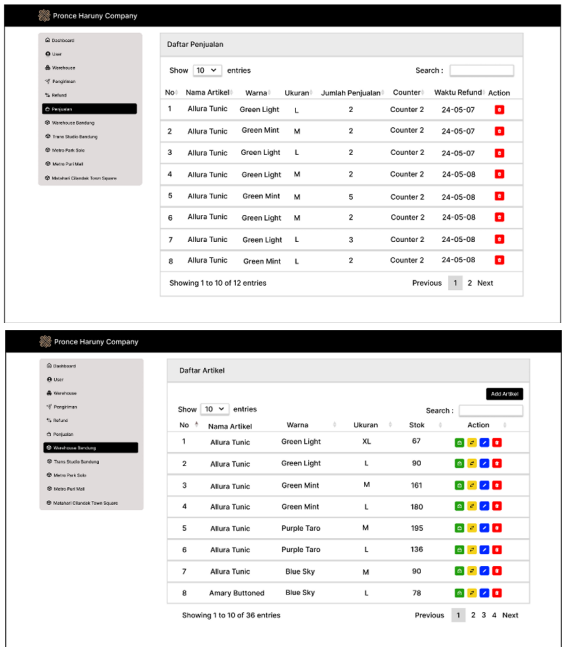


Figure 6. Module Design

The modules in this multi-site inventory information system consist of functional parts that are integrated with each other. The Dashboard module displays the catalog of goods and company profile. The User module manages user data, including add, edit, delete, and password reset. The Warehouse module records item data such as name, color, size, and stock, and supports delivery to counters. The Shipping module records the distribution of goods from the warehouse to the counters, including quantity and destination location. The Refund module records the return of goods from the counter to the warehouse, along with the quantity and time. The Sales module stores transaction data at each counter, such as item name, amount sold, and transaction time. The Counter module manages items at each location, with sales, returns, edit, delete, and stock management features. All modules are connected in one centralized system and work in real-time, with customized access control based on user roles.

3.5 Coding Implementation

The implementation of this warehouse inventory information system uses an ASUS VivoBook laptop with AMD Ryzen 7 processor, 8 GB RAM, and 512 GB SSD. The software used includes Windows 11, Google Chrome, Visual Studio Code, Figma, Draw.io, and Laragon as a local server. The programming languages used are HTML, CSS, PHP, JavaScript, and MySQL, with Bootstrap for the interface. The code structure is modular and organized, covering display, process logic, and database access. The implementation focused on managing inventory, shipping, refund, and sales data through the main functions in the system.

Login Authentication

```
$username = (isset($_POST['username'])) ? htmlentities($_POST['username']) : "" ;
$password = (isset($_POST['password'])) ? md5(htmlentities($_POST['password'])) : "" ;
$query = mysqli_query($conn, "SELECT * FROM tb_user WHERE username =
'$username' AND password = '$password'");
$hasil = mysqli_fetch_array($query);
if($hasil){
    $_SESSION['username_inventory'] = $username;
    $_SESSION['level_inventory'] = $hasil['level'];
    header('location:../home');
}
```

Figure 7. Login Authentication

The code snippet above is part of the user login authentication process. The system takes the username and password input from the login form, then matches it with the data in the tb_user table. The password is encoded using the md5() function before being compared. If they match, the system creates a login session with $_SESSION and redirects the user to the main page. Otherwise, the system displays a login error message.

Item Data Management

```
<?php
include "connect.php";
$nama_artikel = (isset($_POST['nama_artikel'])) ? htmlentities($_POST['nama_artikel']) :
"";
$warna = (isset($_POST['warna'])) ? htmlentities($_POST['warna']) : "";
$ukuran = (isset($_POST['ukuran'])) ? htmlentities($_POST['ukuran']) : "";
$stok = (isset($_POST['stok'])) ? htmlentities($_POST['stok']) : "";
if (!empty($_POST['input_warehouse_validate'])) {
    $query = mysqli_query($conn, "INSERT INTO tb_warehouse (nama_artikel, warna,
ukuran, stok) VALUES ('$nama_artikel', '$warna', '$ukuran', '$stok')");
    if ($query) {
        $message = "<script>alert('Data berhasil dimasukkan');
window.location='../warehouse'</script>";
    } else {
        $message = "<script>alert('Gagal memasukkan data')</script>";
    }
}
echo $message;
?>
```

Figure 8. Item Data Management

The code above is part of the backend process to add new article data to the tb_warehouse table. The data is taken from the input form, including article name, color, size, and stock. For basic security, the input is filtered with the htmlentities() function to prevent special character injection. The data is then saved to the database using the SQL INSERT INTO command. If successful, the system provides feedback via a JavaScript alert and redirects the user to the warehouse main page.

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

Goods Delivery Process

```
// Check stock in warehouse
$query_get_stok_warehouse = "SELECT stok FROM tb_warehouse WHERE id = $id";
$result_stok_warehouse = mysqli_query($conn, $query_get_stok_warehouse);
// Reduce stock in the warehouse
$query_update_stok_warehouse = "UPDATE tb_warehouse SET stok = stok - $jumlah_pengiriman WHERE id = $id";
// Add stock at the selected counter
$query_update_stok_counter = "UPDATE $counter_table SET stok = stok + $jumlah_pengiriman WHERE id = $id";
// Enter shipping data into the tb_pengiriman table
$query_insert_pengiriman = "INSERT INTO tb_pengiriman (nama_artikel, warna, ukuran, jumlah_pengiriman, counter, waktu_pengiriman) VALUES ('$nama_artikel', '$warna', '$ukuran', $jumlah_pengiriman, '$counter', '$waktu_pengiriman')";
```

Figure 9. Goods Delivery Process

The program code above handles the process of sending goods from the main warehouse to each counter. The process starts by checking the stock availability based on the article ID. If the stock is sufficient, the system reduces the amount of stock in the warehouse and adds the stock to the destination counter table (e.g. tb_counter1, tb_counter2, and so on). The shipment data is then recorded to the tb_shipment table, including the article name, quantity, and delivery time. This mechanism ensures that stock recording is automated and consistent between parts of the system.

Refund Process

```
// Save refund data into tb_refund table
$query_refund = "INSERT INTO tb_refund (nama_artikel, warna, ukuran, jumlah_refund, counter, waktu_refund)
        VALUES ('$nama_artikel', '$warna', '$ukuran', '$jumlah_refund', '$counter', '$waktu_refund')";
// Reduce the stock of articles at the counter
$query_update_stok_counter = "UPDATE tb_counter1 SET stok = stok - $jumlah_refund WHERE id = $id";
// Add stock articles back to warehouse
$query_update_stok_warehouse = "UPDATE tb_warehouse SET stok = stok + $jumlah_refund WHERE id = $id";
```

Figure 10. Refund Process

The refund feature of the system allows the return of items from the counter to the main warehouse. When a user makes a refund, the system saves the data to the tb_refund table, recording the article name, quantity, counter origin, and return time. The system then automatically subtracts the stock at the counter (e.g. tb_counter1) and adds the same amount to tb_warehouse. This process maintains consistency of stock data between the counter and the warehouse and makes it easy to track returned items.

Sales Process

```
// Save sales data into the tb_penjualan table
$query_penjualan = "INSERT INTO tb_penjualan (nama_artikel, warna, ukuran, jumlah_penjualan, counter, waktu_penjualan)
                    VALUES ('$nama_artikel', '$warna', '$ukuran', '$jumlah_penjualan', '$counter', '$waktu_penjualan')";
// Reduce the stock of articles at the counter
$query_update_stok_counter = "UPDATE tb_counter1 SET stok = stok - $jumlah_penjualan WHERE id = $id";
```

Figure 11. Sales Process

The sales feature in the system is used to record the sales activities of each counter. When a sale is made, the system will store the sales information in the tb_sales table, including the item name, sales quantity, originating counter, and time of sale. After recording, the system automatically reduces the stock of the item at the relevant counter (e.g., tb_counter1) by the quantity sold. This process ensures that stock data remains accurate and supports real-time monitoring of item sales from each outlet or counter.

3.6 Unit Testing

Unit testing is the process of testing the smallest parts of a program (units), such as functions or modules, separately to ensure that they work as expected [18]. This testing is usually done during the development process to detect errors early on. At this stage, unit testing is performed on all major functions of the system, including user authentication, inventory management, shipping, returns, and sales recording, to ensure that each module runs according to specifications. As an example, testing is shown on the login function, because the verification procedure in unit testing is similar at each stage, so it is sufficient to represent it through one function.

```
proses > login_helper.php > ...
1   <?php
    3 references
2   function proses_login($conn, $username, $password): array {
3       $username = htmlentities(string: $username);
4       $password = md5(string: htmlentities(string: $password)); // sesuai database
5
6       $query = mysqli_query(mysqli: $conn, query: "SELECT * FROM tb_user WHERE username = '$username' AND password = '$password'");
7       $hasil = mysqli_fetch_assoc(result: $query);
8
9       if ($hasil) {
10          return [
11              'status' => true,
12              'username' => $hasil['username'],
13              'level' => $hasil['level']
14          ];
15      } else {
16          return ['status' => false];
17      }
18  }
19
```

Figure 12. Login Unit Testing

Unit testing was conducted to test the login function in the inventory information system using PHPUnit with two scenarios: successful login and failed login. The successful scenario uses data that matches the database (username: admin@admin.com, password: password), while the failed scenario uses an incorrect password. Testing is done through the login_helper.php file which verifies the username and password against the database. The unit test results show that the system can distinguish valid and invalid logins correctly.

```
PS C:\laragon\www\Inventory> vendor/bin/phpunit tests/LoginTest.php
PS C:\laragon\www\Inventory> vendor/bin/phpunit tests/LoginTest.php
PHPUnit 8.5.42 by Sebastian Bergmann and contributors.

..                                                                  2 / 2 (100%)

Time: 121 ms, Memory: 4.00 MB

OK (2 tests, 3 assertions)
PS C:\laragon\www\Inventory>
```

Figure 13. Login Unit Testing Results

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

The figure shows that unit testing of the login module was successfully run using PHPUnit version 8.5.42. Two scenarios were tested: login with correct data and incorrect data. Both were successful, with a total of 2 tests and 3 assertions passing 100% without error or failure. These results show that the login function is able to verify user credentials according to the tested scenarios.

3.7 Integration Testing

Integration testing is conducted after unit testing to ensure that all modules in the web-based multi-location inventory information system function in an integrated manner according to the designed flow. With the Black Box Testing method, the author compiles test cases for each integration process, such as shipping goods from the warehouse to the counter, refunding from the counter to the warehouse, and recording sales. The goal is to ensure that data changes in one module are reflected automatically and consistently in other modules [19].

For example, when goods are shipped from the warehouse to the counter, the stock in the warehouse automatically decreases and the stock in the counter increases by the amount shipped. If a refund occurs, the counter stock decreases and the warehouse stock increases again. Recording sales at the counter also reduces stock and records transaction history. All these processes are automatically recorded in their respective history tables, such as shipments, refunds, and sales.

No.	Feature Tested	Main Scenario	Expected Result	Test Result
1.	Login	Valid input / incomplete input / invalid email format	System only accepts valid input	Success
2.	User	Add, edit, delete, reset password	User data can be managed according to procedure	Success
3.	Warehouse	Add, edit, delete, shipment	Item data stored & distribution recorded	Success
4.	Shipment	Search, delete shipment data	Shipment report matches search & updates	Success
5.	Refund	Search, delete refund data	Refund report matches search & updates	Success
6.	Sales	Search, delete sales data	Sales report matches search & updates	Success
7.	Counter	Add, edit, delete, sales, refund	Counter data managed and synchronized in real-time	Success

Figure 14. Table Black-Box Testing

Test results show that there are no data inconsistencies between modules, and all processes run according to the designed workflow. This finding proves that the system meets data integrity and feature integration aspects and is ready to support real-time and centralized inventory management in a multi-location operational environment.

3.8 System Testing

System testing was conducted to ensure that all features of the website-based multi-location inventory system functioned as required. Tests included login validation, user management, article data handling, inter-location shipping, refunds, and sales recording. Each scenario was tested to verify that the outputs matched expectations. The results showed that all features operated correctly and consistently, indicating the system is stable, meets user needs, and is ready for real-time, integrated stock management implementation.

3.9 Success Test

Success tests were conducted to assess whether the web-based multi-location inventory information system has met the functional needs of users. The test results show that the system is able to run all the main processes such as managing stock in the warehouse and counter, shipping, refunding, and recording sales properly and Real-Time. Each module functions according to the designed flow and can be accessed according to user access rights. Thus, the system was declared successful in improving the efficiency of integrated inventory management and supporting effective, centralized, and structured operations in a multi-location environment.

This research shows excellence in the development of a website-based inventory information system that supports real-time stock management and is integrated between locations, with key features such as inventory management, shipping, refunds, sales, and user access control. The system was built using the V-Model method and a three-tier architecture that supports structured testing at each stage. The test results, ranging from unit to integration, prove that all modules run stably, data between features is consistent, and the system is ready to be deployed in a multi-site environment to improve operational efficiency and data-driven decision making.

System testing was carried out using the Black Box Testing method, which focuses on

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76

validating system functions based on the inputs provided and the outputs produced. Due to its nature, which emphasizes functional suitability rather than numerical measurement, this method does not generate quantitative data such as success percentages or system response times. The results showed that all core features, including warehouse management, counter management, shipping, refunds, sales, and user management, operated in accordance with the specified requirements. Nevertheless, the system still has several limitations, particularly in terms of security, which currently relies on basic encryption, and the absence of non-functional testing such as performance, load, and security evaluation. These limitations can serve as recommendations for future research to adopt more comprehensive testing approaches, including quantitative evaluation, while also improving security and scalability so that the multi-location inventory information system becomes more robust and reliable in supporting organizational needs.

4. CONCLUSIONS

This research aims to develop a website-based inventory information system that can support real-time inventory management across multiple locations. The system is designed using the V-Model method with stages of requirements analysis, system design, implementation, and testing, and is implemented in the form of a web application with key features of warehouse and counter stock management, shipping, returns, sales, and user settings. Testing was conducted using the Black Box Testing method, which included unit testing, integration testing, system testing, and success testing on all main functions. The test results showed that all features worked according to the specified requirements, so the purpose of this research to build a real-time, integrated, multi-location website-based inventory information system was successfully achieved.

Future research is recommended to add mobile features, automatic notifications, as well as strengthening security aspects and testing on a larger user scale.

REFERENCES

[1] M. of Finance Republic of Indonesia and A. Development Bank, "Innovate Indonesia Unlocking Growth Through Technological Transformation March," Philippines, 2020. doi: http://dx.doi.org/10.22617/SGP200085-2.

[2] H. H. Muflihin, H. Dhika, and S. Handayani, "Perancangan Sistem Informasi Inventory Pada Toko Rosadah," *Jurnal Bianglala Informatika*, vol. 8, no. 2, pp. 91–99, 2020, doi: https://doi.org/10.31294/bi.v8i2.8712.

[3] H. Alfarizi and A. H. Anshor, "KLIK: Kajian Ilmiah Informatika dan Komputer Sistem Informasi WIP (Work In Process) Berbasis Web Menggunakan Metode Waterfall," *Media Online*, vol. 4, no. 3, pp. 1483–1492, 2023, doi: 10.30865/klik.v4i3.1503.

[4] F. Baihaqqi, N. Suarna, and O. Nurdiawan, "Sistem Informasi Gudang Berbasis Web Untuk Penyimpanan Barang Di Pt Mitra Sukses Bangun Bersama," 2023.

[5] R. M. Lhokseumawe, Yuyun Yunengsih, and Yuda Syahidin, "Perancangan Sistem Informasi Rekam Medis Dalam Menunjang Pelaporan Morbidittas UGD dengan Metode V-Model," *Decode: Jurnal Pendidikan Teknologi Informasi*, vol. 4, no. 2, pp. 660–674, Jul. 2024, doi: 10.51454/decode.v4i2.592.

[6] R. Naibaho, "Perancangan Sistem Informasi Inventory Dengan V-Model Berbasis Web Pada PT Brickbern," *Bridge : Jurnal publikasi Sistem Informasi dan Telekomunikasi*, vol. 2, no. 3, pp. 114–129, Jul. 2024, doi: 10.62951/bridge.v2i3.126.

[7] A. Alfarisy and M. Son Muarie, "Sistem Informasi Penjualan dan Persediaan Barang Berbasis Web Menggunakan Metode V-Model pada Toko Arif Gorden," 2021. [Online]. Available: https://journal-computing.org/index.php/journal-ita/index

[8] M. D. Gustinov *et al.*, "Analysis of Web-Based E-Commerce Testing Using Black Box and White Box Methods," 2023.

[Online]. Available: www.toko.ghaniib.my.id.

[9] A. A. Permana, B. Fadillah, and R. Taufiq, "Penggunaan Metode V-Model Untuk merancang Sistem Informasi E-Logbook Berbasis Website," *Jurnal Minfo Polgan*, vol. 12, no. 2, pp. 30–2023, 2023, doi: 10.33395/jmp.v12i2.12347.

[10] P. Sen Liu, J. F. Chin, H. Ab-Samat, and Z. P. Xie, "Digital variant design V-model for rapid product development," *International Journal of Advanced Manufacturing Technology*, vol. 139, no. 3, pp. 2103–2121, Jul. 2025, doi: 10.1007/s00170-025-15982-1.

[11] L. Setiyani, Y. Rostiani, and T. Ratnasari, "Analisis Kebutuhan Fungsional Sistem Informasi Persediaan Barang Perusahaan General Trading (Studi Kasus : PT. Amco Multitech)," *Owner*, vol. 4, no. 1, p. 288, Feb. 2020, doi: 10.33395/owner.v4i1.205.

[12] J. Sains *et al.*, "Yayasan Insan Cipta Medan Aplikasi Buku Tamu Menggunakan Fitur Kamera Dan Ajax Berbasis Website Pada Kantor Dispora Kota Medan," *Jurnal SITek (Jurnal Sains, Informasi dan Teknologi)*, vol. 1, no. 3, pp. 94–99, 2022.

[13] M. A. Sumarto, "Analisis dan Perancangan Aplikasi Point of Sale (POS) untuk Usaha Mikro, Kecil, dan Menengah (UMKM) dengan Metode Rapid Application Development (RAD)," *Jurnal Studi Komunikasi dan Media*, vol. 27, no. 1, pp. 17–34, Jun. 2023, doi: 10.17933/jskm.2023.5115.

[14] B. Simare Mare, A. A. Yana, and U. N. Mandiri, "Perancangan Sistem Informasi Berbasis Web Pada Koperasi Simpan Pinjam Sejahtera Bersama," Online, Jakarta, 2022. doi: http://dx.doi.org/10.55181/ijns.v11i2.1776.

[15] T. Wijayanti, F. Nugraha, and A. P. Utomo, "Rancang Bangun Sistem Manajemen Pengelolaan Pengaduan Masyarakat Di Kabupaten Kudus," *Journal of Computer and Information Systems Ampera* , vol. 3, no. 1, pp. 56–65, 2022, doi: https://doi.org/10.51519/journalcisa.v3i1.141.

[16] F. Arif Novianto and dan Hari Purwanto, "Perancangan Sistem Informasi Land Transportation Assistance Taxi Puskopau Pada Bandara XYZ," Jakarta, 2022.

[17] A. Dwi Herlambang, A. Rachmadi, A. Putri Rahmatika, D. Indah Dwi Utami, and S. Widya Hapsari, "V-Model Untuk Pengembangan Sistem Informasi Manajemen Ruang Rapat," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)* , vol. 7, no. 2, pp. 313–322, 2020, doi: 10.25126/jtiik.202071893.

[18] A. A. Yunanto, W. Yuwono, J. Ghaniyyah, and P. Arrochim, "Unit Testing Dan User Review Pada Sistem Informasi Kegiatan Pengabdian Masyarakat Berbasis Website dan Android," *Print) Jurnal Poros Teknik*, vol. 15, no. 1, pp. 30–35, 2023.

[19] M. Ruswiansari, A. Fayi Farozi, S. Rosetya Wardhana, N. Surabaya, and T. Surabaya, "Pengembangan Sistem Pegawai (Simpeg) Berbasis Mobile Menggunakan Metode V-Model," Surabaya, 2024. doi: https://doi.org/10.31284/j.integer.0.v9i1.5791.

DOI : https://doi.org/10.21063/jtif.2025.V13.2.66-76