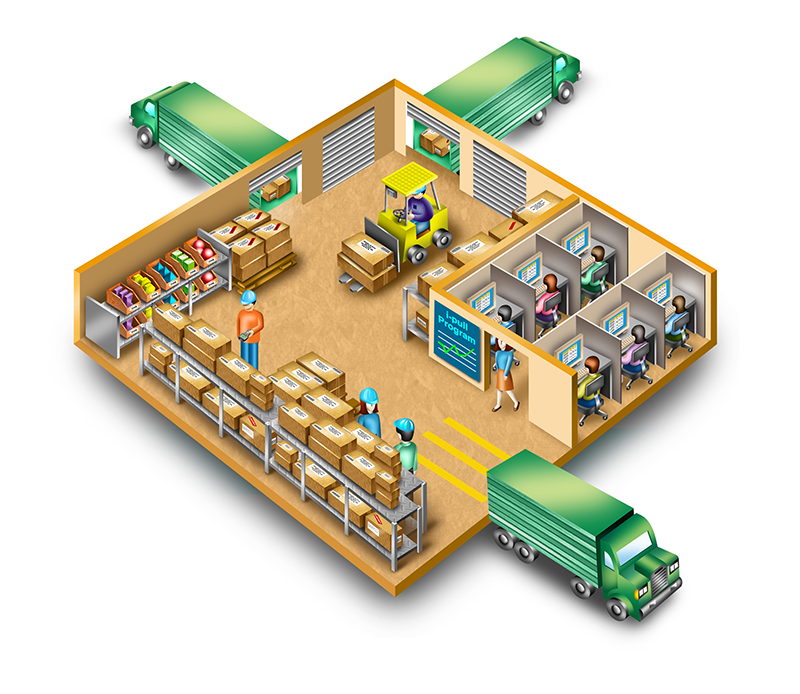


**Palestine Technical University-Kadoorie**

Computer Systems Engineering Department



**A Warehouse Management System**

**Prepared by:**

Yousef Jaber (202110266)

Ahmad Tomeh (202110060)

Yazan Hussien (202112844)

Yaseen Ashqar (202110871)

**Supervisor:**

Dr. Osama Hamed

**Abstract:**

This project introduces a sophisticated Warehouse Management System (WMS) designed to meticulously track inbound and outbound inventory movements, monitor stock levels, manage expiration dates, and advance reservation for warehouses ,easy access to the storage with financial reports.

The system offers real-time visibility into inventory flows, ensuring accurate records of goods entering and leaving the warehouse. Additionally, it includes features to monitor and prevent stock from expiring, minimizing wastage and maximizing profitability. By integrating payment methods, the system facilitates seamless transactions, enhancing operational efficiency. Through this comprehensive approach, the WMS optimizes inventory control, mitigates losses due to expiration, and streamlines financial processes, ultimately bolstering warehouse operations.

**1 . INTRODUCTION**

This project introduces a sophisticated Warehouse Management System (WMS) designed to meticulously track inbound and outbound inventory movements, monitor stock levels, manage expiration dates, and streamline payment methods with easy access to the storage

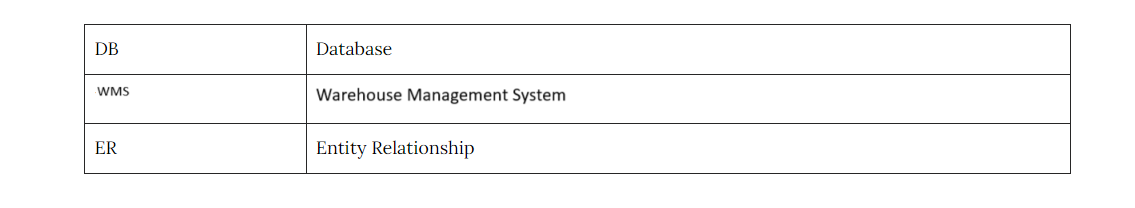
The system offers real-time visibility into inventory flows, ensuring accurate records of goods entering and leaving the warehouse. Additionally, it includes features to monitor and prevent stock from expiring, minimizing wastage and maximizing profitability. By integrating payment methods, the system facilitates seamless transactions, enhancing operational efficiency. Through this comprehensive approach, the WMS optimizes inventory control, mitigates losses due to expiration, and streamlines financial processes, ultimately bolstering warehouse operations

**1.1 Purpose**

The purpose of this document is to build an online system to manage the warehouses and track inbound and outbound inventory to ease access to the storage.

**1.2 DOCUMENT CONVENTIONS**

This document uses the following conventions.



**1.3 INTENDED AUDIENCE AND READING SUGGESTIONS**

This project is a prototype for the warehouse management system and it is restricted within the college premises. This has been implemented under the guidance of college professors(**Dr. Osama Hamed**)

This project is useful for the large companies that have very large storage spaces that and as well as to the shop owners

**1.4 PROJECT SCOPE**

The purpose of the online warehouse management system is to ease manage the warehouses and track inbound and outbound inventory to ease access to the storage and to create a convenient and easy-to-use application for shop owner to request an order and know my estimated time of arrival.

The system is based on a relational database with its warehouse management and functions to control the outbound ,inbound and storage.

We will have a database server supporting thousands of types of goods, that is available in Palestine . above all, we hope to provide a comfortable a user experience along with the best pricing available and accurate timing.

* 1. **REFERENCES**
* <https://krazytech.com/projects>
* Software Engineering 9th Edition by Ian Sommerville

## 2. OVERALL DESCRIPTION

**2.1 PRODUCT PERSPECTIVE**

* The Warehouse Management System (WMS) is designed as an integrated part of a broader supply chain management ecosystem. It interfaces with procurement, inventory management, order fulfillment, and shipping modules, providing seamless coordination and efficient management of warehouse operations. The system serves as a centralized platform that supports real-time tracking, management, and reporting of warehouse activities.

**2.2 PRODUCT FEATURES**

**Inventory Management:**

* + Real-time tracking of inventory levels, automated stock replenishment, and inventory auditing.

**Order Processing:**

* + Efficient order picking, packing, and shipping processes, including batch processing and prioritization of urgent orders.

**Receiving and Put away:**

* + Streamlined processes for receiving shipments and organizing inventory in optimal storage locations.

**Barcode/RFID Integration:**

* + Support for barcode and RFID scanning to improve accuracy and speed of operations.

**Reporting and Analytics:**

* + Comprehensive reporting tools and analytics for performance monitoring, trend analysis, and decision support.

**User Access Control:**

* + Role-based access control to ensure users only access functions relevant to their roles.

**Alerts and Notifications:**

* + Automated alerts and notifications for critical events such as low stock levels, order delays, and equipment failures.

**2.3 USER CLASS and CHARACTERISTICS**

* **Warehouse Managers:**
  + Responsibilities: Oversee overall warehouse operations, make strategic decisions, and generate performance reports.
  + Access: All system functionalities and analytics.
* **Inventory Controllers:**
  + Responsibilities: Maintain optimal inventory levels, conduct audits, and generate inventory reports.
  + Access: Inventory management and reporting features.
* **Order Fulfillment Staff:**
  + Responsibilities: Pick, pack, and ship orders.
  + Access: Order processing and barcode/RFID scanning functionalities.  
    - * **Receiving Staff:**
        + Responsibilities: Handle incoming shipments, inspect goods, and manage put-away operations.
        + Access: Receiving, put-away, and barcode/RFID scanning features.

**IT Administrators:**

Responsibilities: Maintain system infrastructure, manage user permissions, and handle integrations.

Access: Administrative access to all system settings and configurations.

**Executives:**

Responsibilities: View high-level performance metrics and reports.

Access: Dashboards and analytics features.

**2.4 OPERATING ENVIRONMENT**

**2.5 DESIGN and IMPLEMENTATION CONSTRAINTS**

**The system must adopt a modular architecture to ensure flexibility and scalability. Each module, such as inventory management and order processing, should function independently while integrating seamlessly with others. It must support both on-premise and cloud-based deployment options to cater to different organizational preferences and requirements.**

* **User Interface:**
  + The interface should be intuitive and user-friendly to minimize the learning curve for users. Consistency in design and navigation is critical.
  + A responsive design is necessary to ensure usability across various devices, including desktops, tablets, and smartphones.

**Data Management:**

The system should support high-volume transactions and large datasets typical in warehouse operations. Efficient database design and indexing are crucial for performance.

Data redundancy and replication strategies must be employed to ensure data integrity and availability.

**3. Functional Requirements**

* **Inventory Management**

1. The system must provide real-time tracking of inventory levels, showing current stock for all items.
2. The system should support automated stock replenishment based on predefined thresholds.
3. Users should be able to conduct regular inventory audits and cycle counts.
4. The system needs to manage inventory across multiple locations and warehouses.
5. It must track lot and serial numbers for items requiring detailed traceability.

* **Order Processing**

1. Users should be able to input and manage customer orders.
2. The system must facilitate picking, packing, and shipping processes.
3. It should support batch and wave picking strategies for optimized order fulfillment.
4. Priority should be given to urgent orders based on user-defined criteria.
5. The system must handle returns and reverse logistics, including restocking or disposal of returned items.

* **Receiving and Put away**

1. The system must record the receipt of incoming shipments, including quantity and condition of goods.
2. It should support inspection and quality control processes for incoming goods.
3. Efficient put away processes should be facilitated, recommending optimal storage locations.
4. Cross-docking operations should be supported where items are directly transferred from receiving to shipping.

* **Barcode/RFID Integration**

1. The system must support barcode and RFID scanning for inventory management and order processing tasks.
2. Inventory data must be updated in real-time upon scanning items.
3. The system should generate and print barcode labels and RFID tags as needed.

* **Reporting and Analytics**

1. The system should generate standard reports on inventory levels, order status, and warehouse performance.
2. Users should be able to create customizable reports tailored to specific needs.
3. Dashboards for real-time monitoring of key metrics should be provided.
4. Trend analysis and forecasting tools should be available to support decision-making.

* **User Access Control**

1. Role-based access control should be implemented to restrict user access based on their roles.
2. User authentication should be required for access, supporting secure login mechanisms.
3. Audit trails and activity logs should be maintained for user actions within the system.

* **Alerts and Notifications**

1. Automated alerts should be generated for critical events such as low stock levels and order delays.
2. Notifications should be sent to relevant users regarding equipment failures and maintenance needs.

**4. EXTERNAL INTERFACE REQUIREMENTS**

**4.1 User Interfaces:**

* Front-end software: Developed using HTML, CSS, Bootstrap, JavaScript, and React.js.
* Features:
* Responsive design that supports various devices (desktops, tablets, and smartphones).
* User-friendly interface for warehouse management tasks (e.g. inventory tracking, reporting).
* Back-end software: Powered by Node.js with MySQL Database integration.
* Features:
* Handles server-side logic and API requests.
* Provides Authentication.

**4.2 Hardware Interfaces:**

* Windows.
* Any device with a web browser that supports HTML, CSS, JavaScript, and React.js

**4.3 Software Interfaces:**

|  |  |
| --- | --- |
| **Software used** | **Description** |
| **Operating system** | We have chosen Windows operating system for its best support and user-friendliness. |
| **Database** | MySQL Database is chosen to store data related to the project, such as user information, content, and other application data. |
| **JavaScript (JS)** | To implement the project we have chosen **JS** language for its more interactive support, and this language give us a good library like React. |
| **Front-end Technologies** | HTML, CSS, Bootstrap, JavaScript, and React.js |
| **Back-end Technology** | Node.js is used to develop the back-end logic of the application, handling server-side operations, routing, and integrating with the MySQL Database to fetch and manipulate data. |

**4.4 COMMUNICATION INTERFACES:**

* **This project supports all types of web browsers,**
* **JWT (JSON Web Token):**
* JWT will be used for user authentication and secure information exchange between parties
* Each JWT will contain user information and an expiration time, used to validate incoming requests.
* **API (Application Programming Interface):**
* RESTful APIs will be provided to integrate the system with external applications and software.
* The API will include endpoints for login, inventory management, report generation, and payment processing.

## 5. NONFUNCTIONAL REQUIREMENTS 5.1 PERFORMANCE REQUIREMENTS

## A) E-R DIAGRAM

## 

## 5.2 SAFETY REQUIREMENTS

**5.2.1 Data Integrity**

## Financial Reports Table

## Implement data validation checks to ensure that revenue, expenses, and profit values are within acceptable ranges and formats.

## Enforce referential integrity constraints to maintain consistency between financial reports and associated data.

## Users Table

## Implement mechanisms to prevent duplicate usernames and ensure uniqueness of user accounts.

## Regularly validate and sanitize user input to prevent SQL injection attacks and other forms of data manipulation.

## Reservations Table

## Validate reservation dates to prevent overlaps and conflicts with existing reservations.

## Customers Table

## Enforce uniqueness of customer IDs and email addresses to prevent duplicate customer accounts.

## Implement validation checks to ensure the correctness of customer data entered into the system.

## Warehouse Table

## Validate section capacity values to ensure they are realistic and within acceptable limits.

## Employees Table

## Implement validation checks to ensure the correctness of employee data entered into the system.

## Mechanisms Table

## Validate mechanism prices to prevent negative or unrealistic values.

## Access Controls

## Enforce least privilege principles to ensure that users only have access to the data and functionalities necessary for their roles.

* 1. **SECURITY REQUIREMENTS**

**These security requirements aim to protect the confidentiality, integrity, and availability of data stored in the database and ensure compliance with regulatory requirements. Implementing these measures will help mitigate security risks and safeguard sensitive information against unauthorized access and malicious activities.**

* + 1. **Access Control**

1. **Financial Reports Table**

* Only authorized users with appropriate roles (e.g., finance manager) shall have access to financial reports data.

1. **Users Table**

* Enforce access control measures to ensure that only authenticated users can access the Users table.

1. **Reservations Table**

* Access to reservation data shall be restricted to authorized users such as warehouse managers and customer service representatives.

1. **Customers Table**

* Protect customer data by enforcing access control measures and encrypting sensitive information such as passwords and email addresses.

1. **Warehouse Table**

* Access to warehouse section data shall be restricted to authorized personnel such as warehouse managers and supervisors.

1. **Employees Table**

* Secure employee data by enforcing access control measures and encrypting sensitive information such as employee names and contact details.

1. **Mechanisms Table**

* Access to mechanism data shall be restricted to authorized users such as inventory managers and procurement officers.
  + 1. **Authentication Mechanisms**
* Implement secure authentication mechanisms, such as username/password authentication .
  + 1. **Data Encryption**
* Implement encryption mechanisms to protect sensitive data stored in the database, such as passwords, email addresses, and financial information.

**5.4 Software Quality Attributes**

## Availability

## Data in the database should be available when needed, especially for critical operations such as reservations.

## Application to the Database:

## Implement measures such as redundancy, load balancing, and failover to ensure continuous availability of data.

## Correctness

## Enforce data integrity constraints to maintain data consistency and accuracy.

## Validate input data to ensure that it meets specified criteria and prevent data errors or inconsistencies.

## Maintainability

## Implement version control for database schema changes and document database updates to ensure traceability and ease of maintenance.

## Usability

## The database system should be user-friendly and accessible on multiple devices to cater to a diverse user base.

## Ensure compatibility with various devices and screen sizes by implementing responsive design principles for web-based interfaces.

## 6. Software Diagram

## A) Use Case Diagram :

## 

## B) Class Diagram

## 

## GitHup resp: [Warehouse-management-system](https://github.com/ahmadtomeh03/Warehouse-management-system)