Warehouse Management System (WMS)

Professor: Paul Laird

university name: Dublin Business School

Module Title: Programming for Information Systems

Module Code: B9IS123

Submitted By:

Group Name: - Logic Legends

Group members: - 1. ArbazKhan Mokashi (20024926)

2.Dharma Sai Mogadampuram (20025435)

3. Sahil Sinha (20027911)

Table of Contents

Introduction

Warehouse Management System (WMS) can streamline the warehouse processes by thoroughly tracking the amount of inventory, location management and supervising the product movements. It grabs attention by way of processing data in a blink of an eye without the hassle of input-output processing, updating's, and producing reports on the inventory status. We have adopted Flask in our system as it is easy to code and is also flexible. SQLite is a secure disaster recovery technique that is also robust. We have used this as for a database. It brings together an online and real-time transaction process that guarantees smooth and uncompromised back-end integration and promotes efficient and correct management of the warehouse as tasks are executed in a real time.

System Overview

The Warehouse Management System (WMS) functions as an essential element of its inventory system, with the ability to automatically organize and keep track of all the products that are in the process of moving through different storage facilities. It encapsulates four pivotal entities that serve as the building blocks of the system:

1. Products: The core trait presents the various kinds of products that these warehouses supply. Each of these products is eyetracking per se, which means that each item is being monitored and can be tracked down to its storage point within network. The WMS furnishes all the details by cataloging the mentioned items involving numerous parameters in which the identification and management of them get easier.
2. Locations: As the main representative of the warehouse network nodes, the entity 'Locations' is the physical area occupied by products—wether this be an entire warehouse or zone therein. To be successful in this, we need to give close attention to such spatial separation of the stores and their product inventory management/strategies.
3. Product Movements: Critical in transfers between locations, this department/section is the source of all the logistical details of inventory despatch and storing. It audits products' movement including emplacement of every goods' pathway from one point to another. It maps out a continuous chain of custody and also facilitates the reallocation of stocks efficiently.
4. Balance: Proving as a system's register, 'Balance' supplies with a real-time, or quantitative, highlighted view for products at certain locations. This snapshot is the main instrument to help appraisal of stock level and anticipating both the inventory needs and any demand fluctuations. Such step keeps stock level within the limit defined by these two factors – overstocking or stockouts.

Implementation Details

In under Warehouse Management System, Python is chosen for its adaptability and this is through the Flask framework which is understood for its minimal structure and ease of use. Flask's modular design is a great fit for this system, allowing one to extend the stack of backend services with this foundation. The other tool that used is SQLite, as it omits the requirement for a third-party server and its simplicity in integrating with Python and Flask. SQLite provides for a serverless architecture, saving in setting up and it is a nice fit for a simple application. Python, Flask, and SQLite integration ensures a seamless, fast development abstraction and a robust, self-supporting WMS platform.

Database Initialization

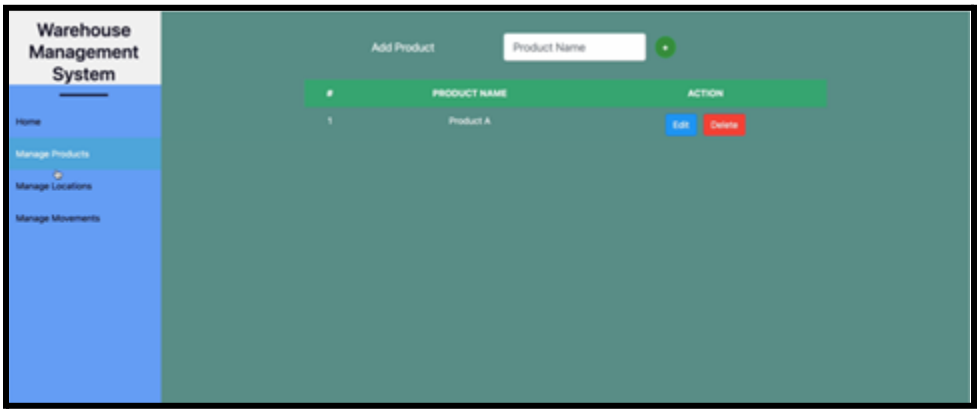
The **database.db** file is created with four tables corresponding to the entities mentioned above. The database schema includes:

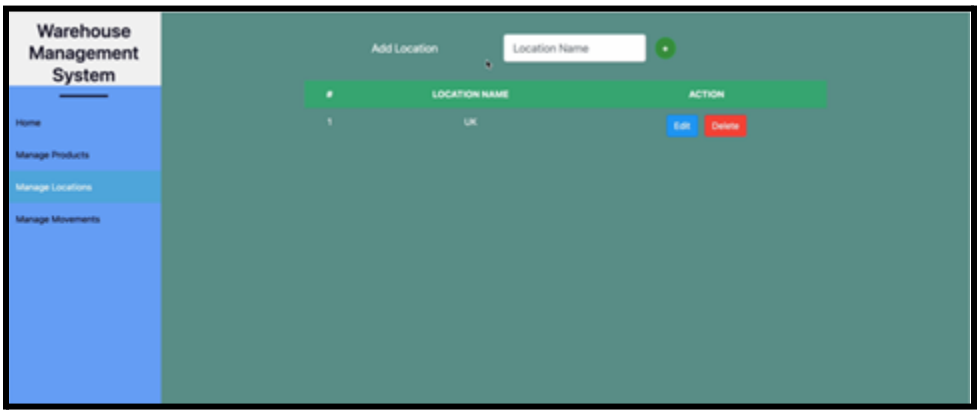
- A **products** table with **productID** and **productName**.
- A **location** table with **locationID** and **locationName**.
- A **productmovement** table with **movementID**, **atTime**, **from_location**, **to_location**, **productName**, and **qty**.
- A **balance** table that tracks the quantity of each product at each location.

Functionalities and Features

1. Data Management: WMS is developed to be a unified hub where new entries, edit operations and deletions related to products, locations and transports are accessed.
2. Movement Management: Input employees scans the product movement and it updates the balance table instantly so that each store knows the current inventory at their store.
3. User Interface: The front-end is acted out using HTML but served by Flask which will provide Flask with dynamic content. This comprises input and output forms for data entry and output tables. The models, algorithms, and programs are also the work of data scientists.
4. Data Validation: Back-end validation of the data makes the result of the final transaction foolproof, like preventing overdrawn accounts or moves from undefined positions.
5. Reporting: With a dashboard, you can get informed in a glance about the theage-size distribution in the current stock levels that exist in different locations


Warehouse Management System
Home
Manage Products
Manage Locations
Manage Movements
DASHBOARD
UK
Product
Quantity Left
Product A
1000


Warehouse Management System
Home
Manage Products
Manage Locations
Manage Movements
Add Product
Product Name
#
PRODUCT NAME
ACTION
1
Product A
Edit
Delete


Warehouse Management System
Home
Manage Products
Manage Locations
Manage Movements
Add Location
Location Name
#
LOCATION NAME
ACTION
1
UK
Edit
Delete


Warehouse Management System
Home
Manage Products
Manage Locations
Manage Movements
Add Movement
Time
03:11 AM
From Location
[Move In]
To Location
[Move Out]
Product
Product A
Quantity
SUBMIT
#
TIME
FROM LOCATION
TO LOCATION
PRODUCT
QUANTITY
ACTION
1
02:51
UK
UK
Product A
1
Edit
Delete

Testing

In order to be sure the whole Warehouse Control System is in good work and is available, serious testings were performed. It also encompasses the concept of unitary testing that is focusing on testing the isolated functions and methods in order to uncover the errors and confirm the methods' performance. The code was gradually modularized until the point it was sized into blocks that were successfully tested in stringent conditions ensuring their functionality independent of their location in the control system. Integration testing was also one of the points at which the entire system was tested and the smoothness with which it functions was confirmed checking the entire system as one component. This encompassed standalone testing of the key workflows, say adding new products with subsequent updating of inventory balances, to take care of easy information flow and system poise.

Challenges and Solutions

The process of creating the Warehouse Management System met us with a number of obstacles which we had to overcome in the course of time. For example, we had to think of the best way of ensuring smooth and uninterrupted concurrent database access and refine a user interface that is responsive. We solved the database concurrency issue by using Flask’s multi-threading check_same_thread=False argument because it allows multiple threads to safely continue access the SQLite database from different threads simultaneously and without post data conflicts. In the User interface part, we employed Bootstrap, a robust front-end framework, to achieve a tastefully designed that is interative, and responsive for any device or screen sizes. Bootstrap delivered the grid system and pre-built components which sped up the UI development process to make the final product have a consistent uniformity and user-friendliness experience. The upgrade gave the system the capability to be error-free and easy to use.

Reflection and Learning Outcomes

Working into this Warehouse Management System project profoundly thought me the full stack development. We took on the nature of relations data models design working with users interface craftsmanship. It was a fun way to concretely learn Python and Flask, while got a taste of the real world by primarily working on the frontend with Bootstrap. The interactive attribute proved beneficial which each person bringing different strengths, and in turn, this led to all of us having a deeper learning experience. No, we did not just design the WMS, but we also created the basics of working together and technical competence that will support our future projects in the software development field.

Conclusion

The developing of the (WMS) represents actual instantiation of top of the line high-tech technologies which involve incredible speed of data transfer and excellent user interface. The main objective of the project is to illustrate how an excellent site is created for the web by taking into account that the site must be intuitive and convenient for use. By its nature, it is designed to be scalable and set the stage for more sophisticated analytics and mobile responsiveness among other improvements in the future. When WMS currently, it is sufficient for performing its function well, where it is a reliable device for supporting stock management. The scalability and feature integration make it possible for continuous change, therefore, guaranteeing that the system continues to exist and grow stronger by taking about the new business requirement.

Reference

Ghimire, D. (2020). Comparative study on Python web frameworks: Flask and Django.

Kaur, A., Singh, G., Kukreja, V., Sharma, S., Singh, S., & Yoon, B. (2022). Adaptation of IoT with blockchain in Food Supply Chain Management: An analysis-based review in development, benefits and potential applications. *Sensors*, *22*(21), 8174.

Abu Zwaida, T., Pham, C., & Beauregard, Y. (2021). Optimization of inventory management to prevent drug shortages in the hospital supply chain. *Applied Sciences*, *11*(6), 2726.

Ricardianto, P., Kholdun, A., Fachrey, K., Nofrisel, N., Agusinta, L., Setiawan, E., ... & Endri, E. (2022). Building green supply chain management in pharmaceutical companies in Indonesia. *Uncertain Supply Chain Management*, *10*(2), 453-462.

Kler, R., Gangurde, R., Elmirzaev, S., Hossain, M. S., Vo, N. V., Nguyen, T. V., & Kumar, P. N. (2022). Optimization of meat and poultry farm inventory stock using data analytics for green supply chain network. *Discrete Dynamics in Nature and Society*, *2022*, 1-8.

Monczka, R. M., Handfield, R. B., Giunipero, L. C., & Patterson, J. L. (2015). *Purchasing and Supply Chain Management.* Cengage Learning. [This textbook provides an overview of warehouse management systems and their role in supply chain management.]

Kumar, S., & Kumar, P. (2016). *Warehouse Management: A Complete Guide to Improving Efficiency and Minimizing Costs in the Modern Warehouse.* Kogan Page Publishers. [This book offers practical insights into warehouse management systems, including product tracking, inventory management, and logistics.]

Chopra, S., & Meindl, P. (2015). *Supply Chain Management: Strategy, Planning, and Operation.* Pearson. [This textbook covers various aspects of supply chain management, including warehouse management systems and their importance in optimizing inventory flow.]

Waters, D. (2011). *Supply Chain Management: An Introduction to Logistics.* Palgrave Macmillan. [This book provides a comprehensive introduction to logistics and warehouse management systems, including discussions on product tracking and inventory control.]

Roussos, G., Georgopoulos, N., & Assimakopoulos, V. (2017). *Inventory Management: Principles, Concepts and Techniques.* Springer. [This book explores inventory management principles, including the role of warehouse management systems in maintaining optimal inventory levels.]

Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python.* O'Reilly Media. [This book is a comprehensive guide to Flask, covering everything from basic setup to advanced topics like database integration and authentication.]

Baumann, E. (2019). *Flask By Example.* Packt Publishing. [This book offers practical examples and projects to help you learn Flask by building real-world applications.]

Ronacher, A., Grinberg, M., & Hill, D. (2020). *Flask Documentation.* [The official Flask documentation is an invaluable resource for learning Flask. It covers everything from installation and basic usage to more advanced topics like testing and deployment.]

Miguel Grinberg's Flask Mega-Tutorial. [This online tutorial series by Miguel Grinberg is a comprehensive guide to building web applications with Flask. It covers a wide range of topics and provides step-by-step instructions and code examples.]

Udemy Flask Courses. [There are several Flask courses available on Udemy, covering different aspects and skill levels. Look for courses with high ratings and reviews to ensure quality.]