**TITLE OF THE PROJECT**

**Inventory Management System for Retail Businesses**

*A CAPSTONE PROJECT REPORT*

DSA0198 - Object Oriented Programming with C++

**Submitted to**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**In partial fulfillment for the award of the degree of**

**BACHELOR OF ENGINEERING IN COMPUTER SCIENCE & ENGINEERING**

**By**

**KRISHNA VASAVI.V (192210271)**

**SASIDHARA.V(192210289)**

**LAKSHMI NARAYANA(192210352)**

**Course Faculty**

**Dr. Selvaraju sir**

****

**SAVEETHA SCHOOL OF ENGINEERING,**

**SIMATS, CHENNAI - 602105**

**MARCH-2024**

**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUE OF MEDICAL AND TECHNICAL SCIENCES, CHENNAI - 602105**

**BONAFIDE CERTIFICATE**

Certified that this project report “**Inventory Management System for Retail Businesses**” is the Bonafide work of “**KRISHNA VASAVI.V**” who carried out the project work under my supervision.

**Submitted to**

**Dr. Selvaraju sir**

**(Course Faculty),**

**Assistant Professor (SG),**

**Saveetha School of Engineering,**

**SIMATS**

**ACKNOWLEDGEMENT**

This project work would not have been possible without the contribution of many people. It gives me immense pleasure to express my profound gratitude to our Honorable Chancellor Dr. N. M. Veeraiyan, Saveetha Institute of Medical and Technical Sciences, for his blessings and for being a source of inspiration. I sincerely thank our Director of Academics Dr.Deepak Nallaswamy, SIMATS, for his visionary thoughts and support. I am indebted to extend my gratitude to our Director Dr. Ramya Deepak, Saveetha School of Engineering, for facilitating us all the facilities and extended support to gain valuable education and learning experience.

I register my special thanks to Dr. B. Ramesh, Principal, Saveetha School of Engineering for the support given to me in the successful conduct of this project. I wish to express my sincere gratitude to my Course faculty Dr. A.Hyils Sharon Magdalene, for his inspiring guidance, personal involvement and constant encouragement during the entire course of this work.

I am grateful to Project Coordinators, Review Panel External and Internal Members and the entire faculty of the Department of Design, for their constructive criticisms and valuable suggestions which have been a rich source to improve the quality of this work.

**TABLE OF CONTENTS:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TOPIC** | **PAGE NO** |
| 1 | Abstract | **5** |
| 2 | Introduction | **5-6** |
| 3 | Problem Domain | **7** |
| 4 | Algorithm | **7-9** |
| 5 | Source code and output | **10-16** |
| 6 | Implementation and Limitations | **17-18** |
| **7** | conclusion | **19** |
| **8** | References | **20-21** |

**ABSTRACT:**

The Inventory Management System for Retail Businesses is designed to automate the process of tracking and managing inventory in retail environments. This system provides real-time stock updates, ensuring that retail businesses can maintain the right balance of products to meet customer demand. Key features include automated reordering, low-stock alerts, supplier management, and the ability to generate detailed reports on inventory levels and sales trends. By optimizing inventory processes, this system helps businesses reduce overstock, minimize stockouts, and improve overall operational efficiency.

The project aims to offer a user-friendly interface that enables store employees and managers to efficiently monitor and update inventory. This system will also integrate supplier information, allowing for automated purchase orders when stock levels fall below a certain threshold. Through this project, retail businesses can achieve improved accuracy in inventory management, leading to cost savings, enhanced productivity, and a better customer experience.

**INTRODUCTION:**

Inventory management is a critical component of retail business operations, ensuring that products are available to meet customer demand while minimizing overstock and waste. In today’s competitive market, retailers must rely on efficient inventory systems to keep track of stock levels, manage suppliers, and optimize product availability. A well-designed inventory management system helps businesses automate and streamline these processes, allowing them to make informed decisions that reduce costs and improve customer satisfaction.

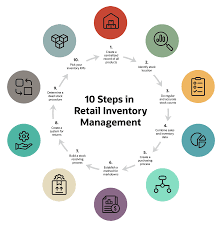
This Inventory Management System for Retail Businesses project aims to provide an effective solution for managing inventory, tracking stock in real-time, and maintaining accurate records of product availability. By implementing such a system, retailers can monitor stock levels, forecast demand, automate reordering processes, and generate insightful reports. Additionally, the system will feature user-friendly interfaces for easy navigation, enabling store managers and employees to quickly access and update inventory data.

The system will also include functionality to alert the business when stock levels are low, ensuring timely restocking. It will integrate seamlessly with supplier databases and generate purchase orders, reducing the risk of stockouts and ensuring that popular products remain in stock. Ultimately, this project seeks to improve operational efficiency, increase profitability, and provide a better shopping experience for customers by optimizing the way inventory is managed.

****

**PROBLEM DOMAIN:**

The problem domain for the Inventory Management System for Retail Businesses focuses on the challenges retailers face in accurately tracking and managing inventory levels. Retailers often struggle with overstocking, which ties up capital and leads to waste, or understocking, which results in lost sales and dissatisfied customers. Additionally, manual inventory management is prone to human error, inefficiency, and lack of real-time visibility, making it difficult to maintain optimal stock levels, forecast demand, and coordinate effectively with suppliers. These issues can lead to operational inefficiencies, higher costs, and reduced profitability, highlighting the need for an automated, real-time inventory management solution.

****

**Algorithm: Inventory Management System**

Step 1: Initialization

1. Load all product data from the inventory database.
2. Set reorder thresholds and supplier details for each product.
3. Initialize transaction logs for tracking inventory movement.

Step 2: Real-Time Stock Update on Sales

4.For each sale transaction:

* Retrieve the product sold.
* Deduct the sold quantity from the product's current stock.
* Update the inventory database.
* Log the transaction in the system (transaction log).

5.If (product stock < reorder threshold):

* Trigger the low-stock alert.
* Proceed to Step 6 for the reorder process.

Step 3: Restocking Inventory

6. For each restock:

* Receive new stock from the supplier.
* Add the received quantity to the product's current stock.
* Update the inventory database.
* Log the restock transaction.

Step 4: Automated Reordering Process

7. If a product's stock falls below the reorder threshold:

* Retrieve supplier information for the product.
* Generate a purchase order (PO) for the required quantity.
* Send the PO to the supplier (automatically or with manager approval).
* Log the reorder action for auditing.

Step 5: Report Generation

8. At specified intervals (daily/weekly/monthly):

* Generate inventory reports including:
* Current stock levels
* Sales trends per product
* Overstock and understock products
* Forecast of product demand
* Provide reports to management for review.

Step 6: Demand Forecasting and Monitoring

9. Analyze historical sales data to identify patterns (fast-moving or slow-moving items).

10. Adjust reorder thresholds based on sales trends and demand forecasting.

11. Monitor stock levels continuously in real-time.

Step 7: Transaction and Error Handling

12. If a transaction error occurs (e.g., incorrect quantity entry):

* Allow the user to manually correct or reverse the transaction.
* Update the inventory and transaction logs accordingly.

Step 8: System Shutdown

13. At the end of the day/session:

* Save all updates to the database.
* Perform backup operations to ensure data integrity.



**SOURCE CODE:**

#include <iostream>

#include <vector>

#include <string>

using namespace std;

// Product class to represent individual items in the inventory

class Product {

public:

string name;

int productID;

int stock;

int reorderLevel;

int price;

Product(string name, int productID, int stock, int reorderLevel, int price) {

this->name = name;

this->productID = productID;

this->stock = stock;

this->reorderLevel = reorderLevel;

this->price = price;

}

void displayProductInfo() {

cout << "Product ID: " << productID << ", Name: " << name

<< ", Stock: " << stock << ", Price: $" << price << endl;

}

// Update stock after sale or restock

void updateStock(int quantity) {

stock += quantity;

if (stock < reorderLevel) {

cout << "Warning: Stock for " << name << " is below reorder level. Please reorder soon!" << endl;

}

}

// Function to check if a reorder is needed

bool isReorderNeeded() {

return stock < reorderLevel;

}

};

// Inventory class to manage all products

class Inventory {

public:

vector<Product> products;

// Add product to the inventory

void addProduct(Product product) {

products.push\_back(product);

}

// Find product by ID

Product\* findProduct(int productID) {

for (auto& product : products) {

if (product.productID == productID) {

return &product;

}

}

return nullptr;

}

// Display all products in the inventory

void displayInventory() {

cout << "------ Inventory ------" << endl;

for (auto& product : products) {

product.displayProductInfo();

}

}

// Handle sales transactions

void sellProduct(int productID, int quantity) {

Product\* product = findProduct(productID);

if (product != nullptr) {

if (product->stock >= quantity) {

product->updateStock(-quantity); // Deduct stock

cout << "Sold " << quantity << " units of " << product->name << endl;

} else {

cout << "Not enough stock available for " << product->name << endl;

}

} else {

cout << "Product not found!" << endl;

}

}

// Restock products

void restockProduct(int productID, int quantity) {

Product\* product = findProduct(productID);

if (product != nullptr) {

product->updateStock(quantity); // Add stock

cout << "Restocked " << quantity << " units of " << product->name << endl;

} else {

cout << "Product not found!" << endl;

}

}

// Generate report of low-stock products

void generateLowStockReport() {

cout << "------ Low Stock Report ------" << endl;

for (auto& product : products) {

if (product.isReorderNeeded()) {

cout << "Product: " << product.name << ", Stock: " << product.stock

<< ", Reorder Level: " << product.reorderLevel << endl;

}

}

}

};

int main() {

// Creating an inventory system

Inventory inventory;

// Adding some products to the inventory

Product p1("Laptop", 101, 50, 10, 1000);

Product p2("Smartphone", 102, 30, 5, 700);

Product p3("Headphones", 103, 100, 20, 100);

inventory.addProduct(p1);

inventory.addProduct(p2);

inventory.addProduct(p3);

// Display current inventory

inventory.displayInventory();

// Simulate selling some products

inventory.sellProduct(101, 45); // Sell 45 Laptops

inventory.sellProduct(102, 15); // Sell 15 Smartphones

// Display inventory after sales

inventory.displayInventory();

// Generate low stock report

inventory.generateLowStockReport();

// Restock products

inventory.restockProduct(101, 50); // Restock Laptops

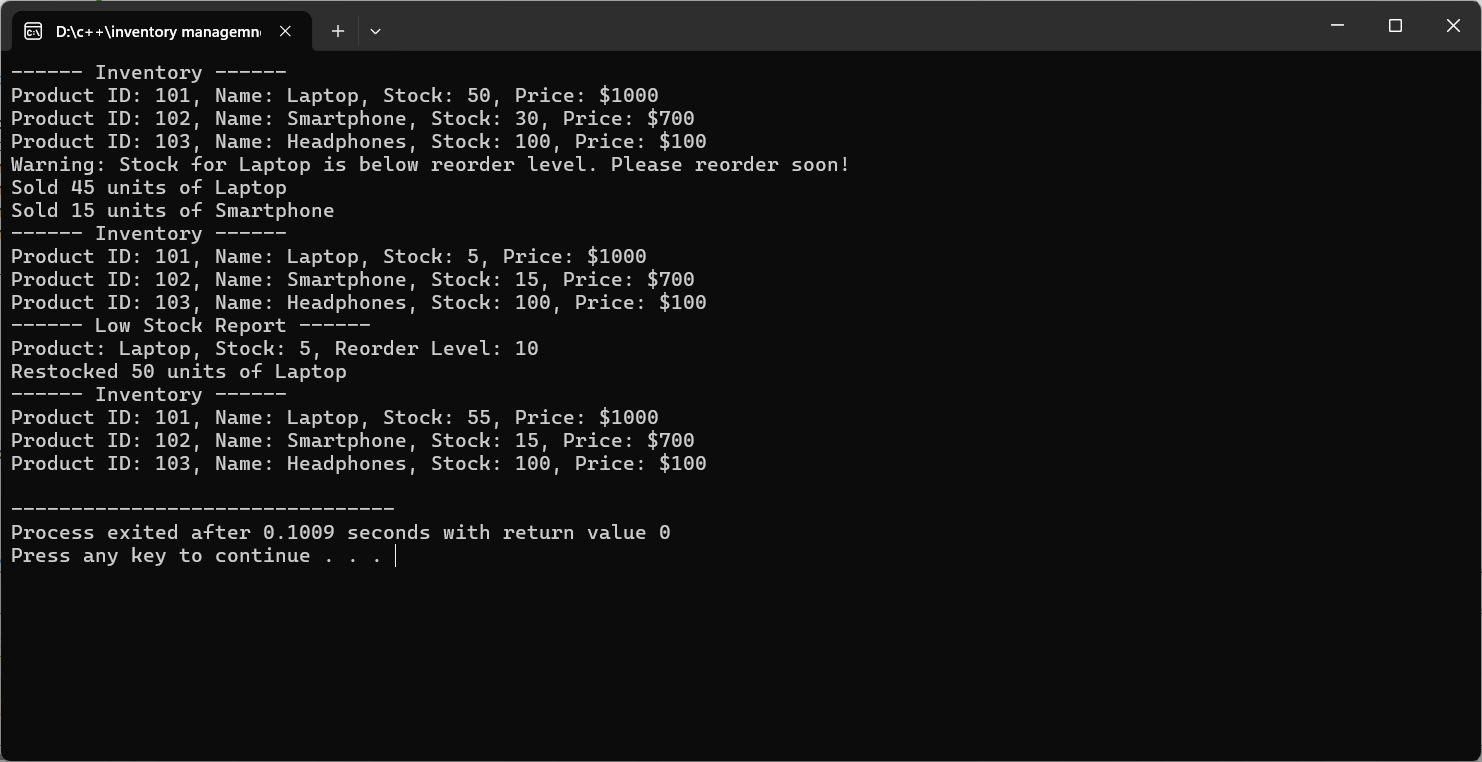
// Display final inventory

inventory.displayInventory();

return 0;

}

**OUTPUT:**

****

**IMPLEMENTATION:**

1.Product Management:

* The Product class represents individual items in the inventory, storing information such as product name, ID, current stock, reorder level, and price.
* Methods in this class include updating stock levels (for both sales and restocking) and checking if a reorder is necessary.

2.Inventory Management:

* The Inventory class maintains a collection of Product objects in a vector.
* Functions include:
* Adding Products: Adding new products to the inventory.
* Finding Products: Searching for products by their ID.
* Selling Products: Reducing stock based on sales and generating warnings when stock is low.
* Restocking Products: Updating stock after new stock arrives.
* Low Stock Reporting: Identifying products below the reorder threshold and generating a report.

3.Transaction Management:

* Each time a product is sold or restocked, the system updates the stock levels and logs the transaction.
* Transactions are handled in real-time, ensuring the current stock levels reflect the latest activity.

4.User Interface:

* The system uses a command-line interface (CLI) to interact with the user.
* Users can display inventory details, make sales, restock products, and generate low-stock reports.

**LIMITATIONS:**

The current Inventory Management System has several limitations. It lacks scalability, as it operates entirely in memory, making it unsuitable for large-scale retail operations with vast product inventories. The system also lacks persistence, meaning data is lost between sessions since it doesn’t save to a file or database. Additionally, the command-line interface (CLI) is not user-friendly for non-technical users, and there is no real-time supplier integration for automated reordering. Product information is basic, with no advanced attributes like batch tracking or expiration dates. Error handling is minimal, and there's no security or access control in place, allowing any user to modify inventory. Lastly, it doesn't include demand forecasting or historical sales analysis, which could help in better inventory planning. These limitations restrict its use in more complex retail environments and leave room for enhancements like database integration, GUI development, and security improvements.

**CONCLUSION:**

In conclusion, the Inventory Management System for Retail Businesses serves as a foundational tool for automating and optimizing inventory management processes. By enabling real-time tracking of stock levels, facilitating timely reordering, and generating insightful reports, the system enhances operational efficiency and helps retailers better meet customer demand. The implementation demonstrates the potential for increased accuracy and reduced human error in managing inventory, ultimately leading to improved customer satisfaction and profitability.

However, to fully realize its potential in a competitive retail landscape, the system must address its current limitations, such as scalability, data persistence, and user interface design. Enhancements like database integration, a graphical user interface, and advanced features such as demand forecasting would significantly improve its functionality and usability. By evolving the system to meet these needs, retailers can gain a comprehensive inventory management solution that adapts to their growth and operational challenges.

**REFERENECES:**

* Bowersox, D. J., Closs, D. J., & Cooper, M. B. (2013). Supply Chain Logistics Management. McGraw-Hill Education.
* This book provides insights into logistics and supply chain management, including inventory management strategies.
* Heizer, J., & Render, B. (2017). Operations Management. Pearson.
* A comprehensive resource that covers various operations management concepts, including inventory control and management.
* Chopra, S., & Meindl, P. (2016). Supply Chain Management: Strategy, Planning, and Operation. Pearson.
* This textbook explores supply chain strategies and includes discussions on inventory management practices.
* Stevenson, W. J. (2018). Operations Management. McGraw-Hill Education.
* This book addresses various aspects of operations management, with a focus on inventory systems and control techniques.
* Guan, L., & Zhao, S. (2015). "An Integrated Inventory Management System." International Journal of Production Research, 53(2), 516-530.
* A research article discussing the integration of various components in inventory management systems.
* Wild, T. (2017). Best Practice in Inventory Management. Routledge.
* This book outlines best practices and methodologies for effective inventory management in various industries.
* Axsater, S. (2006). Inventory Control. Springer.
* A detailed exploration of inventory control methods and their applications in different business scenarios.
* Kumar, S., & Saini, S. (2018). "A Review of Inventory Management Techniques." International Journal of Research in Engineering and Technology, 7(6), 2321-7308.
* This article reviews various inventory management techniques and their effectiveness in different contexts.
* Coyle, J. J., Langley, C. J., Gibson, B., Novack, R. A., & Bardi, E. J. (2016). Supply Chain Management: A Logistics Perspective. Cengage Learning.
* This book provides insights into logistics and inventory management in the context of supply chains.
* Simchi-Levi, D., Kaminsky, P., & Simchi-Levi, E. (2018). Designing and Managing the Supply Chain: Concepts, Strategies, and Case Studies. McGraw-Hill Education.
* A comprehensive resource that covers various supply chain topics, including inventory management strategies.