**IVENTORY MANAGEMENT SYSTEM**

# Submitted by

**Name of the Students: ANUSMITA PAL**

**Enrolment Number: 12022002002185**

**Section: E**

**Class Roll Number: 78**

**Stream: CSE**

**Subject:** Programming for Problem Solving using C

**Subject Code: ESC103(Pr.)**

**Department:** Basic Science and Humanities

## Academic Year: 2022-26

PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE FIRST SEMESTER



**DEPARTMENT OF BASIC SCIENCE AND HUMANITITES**

**INSTITUTE OF ENGINEERING AND MANAGEMENT, KOLKATA**



# CERTIFICATE OF RECOMMENDATION

We hereby recommend that the project prepared under our supervision by RITANAKR BOSE,entitled IVENTORY MANAGEMENT SYSTEM be accepted in partial fulfillment of the requirements for the degree of partial fulfillment of the first semester.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Head of the Department Project Supervisor

Basic Sciences and Humanities IEM, Kolkata

# Introduction

**The inventory management system is a computer program designed to help businesses manage their inventory efficiently. This report presents an inventory management system implemented in the C programming language. The program allows users to add, update, delete, and display product information stored in the inventory.**

**2 Variable Description:**

* **struct Product**: A structure representing a product, containing the following fields:
* **id**: An integer representing the product ID.
* **name**: A character array to store the product name.
* **quantity**: An integer representing the quantity of the product.
* **price**: A float representing the price of the product.
* **inventory**: An array of **struct Product** to store the products in the inventory.
* **count**: An integer representing the current count of products in the inventory.
* **choice**: An integer variable to store the user's menu choice.

**3.Function Descriptions:**

* **addProduct(struct Product \*inventory, int \*count)**: This function allows users to add a new product to the inventory. It prompts the user to enter the product details (ID, name, quantity, and price) and adds the new product to the inventory if space is available.
* **updateProduct(struct Product \*inventory, int count)**: This function enables users to update the details of an existing product in the inventory. Users

provide the product ID to identify the product to be updated. If the product is

found, the function prompts the user to enter new details (name, quantity, and price) and updates the product information accordingly.

* **deleteProduct(struct Product \*inventory, int \*count)**: This function allows users to delete a product from the inventory. Users enter the product ID of the item to be deleted. If the product is found, the function removes it from the inventory and adjusts the count of products accordingly.
* **displayInventory(struct Product \*inventory, int count)**: This function displays the current inventory. It lists all the products along with their ID, name, quantity, and price. If the inventory is empty, it displays an appropriate message.

* 1. **File Description :**

The source code file contains the implementation of the inventory management system program in C. It includes necessary header files (**stdio.h**, **stdlib.h**, **string.h**) for input/output, memory allocation, and string manipulation.

The main function serves as the program's entry point. It initializes the inventory and count variables, and then presents a menu to the user using an infinite loop. The user can choose options to add a product, update a product, delete a product, display the inventory, or exit the program. Based on the user's choice, the corresponding function is called to perform the desired operation.

* 1. **Source code:**

#include <stdio.h>

#include <stdlib.h> #include <string.h> **struct** Product { **int** id; **char** name[100]; **int** quantity; **float** price;

}; **void** addProduct(**struct** Product \*inventory, **int**

\*count) { if (\*count >= 100) { printf("Inventory is full. Cannot add more products.\n"); return; } **struct** Product newProduct; printf("Enter product details:\n"); printf("Product ID: "); scanf("%d", &newProduct.id); printf("Product Name: "); scanf(" %[^\n]", newProduct.name); printf("Quantity: "); scanf("%d", &newProduct.quantity); printf("Price: "); scanf("%f", &newProduct.price); inventory[\*count] = newProduct;

(\*count)++; printf("Product added successfully.\n");

} **void** updateProduct(**struct** Product \*inventory, **int** count) { **int** productId; **int** found = 0; printf("Enter the product ID to update: "); scanf("%d", &productId); for (**int** i = 0; i < count; i++) { if (inventory[i].id == productId) { printf("Enter new details for the product:\n"); printf("Product Name: "); scanf(" %[^\n]", inventory[i].name); printf("Quantity: "); scanf("%d", &inventory[i].quantity); printf("Price: "); scanf("%f", &inventory[i].price); printf("Product details updated successfully.\n"); found = 1; break;

} } if (!found) { printf("Product not found.\n");

} } **void** deleteProduct(**struct** Product \*inventory, **int**

\*count) { **int** productId; **int** found = 0; printf("Enter the product ID to delete: "); scanf("%d", &productId); for (**int** i = 0; i < \*count; i++) { if (inventory[i].id == productId) { for (**int** j = i; j < \*count - 1; j++) { inventory[j] = inventory[j + 1];

} (\*count)--; printf("Product deleted successfully.\n");

found = 1; break;

} } if (!found) { printf("Product not found.\n");

} } **void** displayInventory(**struct** Product \*inventory, **int** count) { if (count == 0) { printf("Inventory is empty.\n"); return; } printf("Product Inventory:\n"); printf("ID\tName\tQuantity\tPrice\n"); for (**int** i = 0; i < count; i++) { printf("%d\t%s\t%d\t\t%.2f\n", inventory[i].id, inventory[i].name, inventory[i].quantity, inventory[i].price);

} } **int** main() { **struct** Product inventory[100]; **int** count = 0; **int** choice; printf("Inventory Management System\n"); while (1) { printf("\nSelect an option:\n"); printf("1. Add a product\n"); printf("2. Update a product\n"); printf("3. Delete a product\n"); printf("4. Display inventory\n"); printf("5. Exit\n"); printf("Enter your choice: "); scanf("%d", &choice); switch (choice) { case 1: addProduct(inventory, &count); break; case 2:

updateProduct(inventory, count); break; case 3: deleteProduct(inventory, &count); break; case 4: displayInventory(inventory, count); break; case 5:

printf("Exiting the program...\n"); exit(0); default:

printf("Invalid choice. Please try again.\n");

break;

} } return 0;

}

**6.Conclusion:**

The inventory management system program provides a user-friendly interface for managing inventory efficiently. It allows businesses to add, update, delete, and display product information easily. The program is implemented in the C language, making it portable and suitable for various platforms.

The program can be further enhanced by adding additional features such as search functionality, data persistence, and error handling. Overall, the inventory management system program simplifies inventory management tasks and improves productivity in businesses.

[Grab your reader’s attention with a great quote from the document or use this space to emphasize a key point. To place this text box anywhere on the page, just drag it.]

