**A SUPER SHOP MANAGEMENT SYSTEM**

**👨‍💻 Submitted By:**

👨‍💻 **Team Members**

| **👤 Name** | **🎓 Student ID** | **🎯 Role** |
| --- | --- | --- |
| Md TANVIR HASSAN | 41230301594 | Team Leader & Developer |
| MD MEHEDI HASSAN | 4123030158 | Assistant Developer & Analyzer |
| IFAJ AHMED SAJID | 412303015 | Code Reviewer & Tester |
| AL MAMUN | 412303015 | Research & Debugging |
| MD RABBI | 41230301619 | Documenter |

**Course Name: CSE- 1290**

**Submitted To:**

**Humayun Kabir**

**Professor**

**Department of CSE**

**Northern University Bangladesh**



**📍 Northern University Bangladesh  
📅 Submission Date: 2025**

## **Abstract**

This project focuses on developing a desktop-based application titled **"Supershop Management System"** designed to streamline and manage the overall operations of a supershop. The primary objective of this system is to automate the essential processes such as customer management, product inventory, employee supervision, sales (both online and on-site), purchases, payments, and profit-loss analysis.

The system efficiently maintains product stock details, updates inventory based on real-time sales data, generates daily and periodic reports, and facilitates better decision-making. By implementing this system, manual paperwork, errors, and delays can be eliminated, ensuring accuracy and speed across various departments.

The Supershop Management System plays a vital role in ensuring smooth business operations by tracking stock levels, alerting on low inventory, supporting purchase management, and handling customer transactions effectively. It ensures a seamless shopping experience for customers while giving the management full control over business data.

Incorporating a user-friendly interface and modular structure, this C language-based project includes predefined data such as 50 stocked products and 5 registered employees. It provides features for billing, sales recording, employee listing, and online orders—making it a valuable asset for any growing retail business. The system offers significant benefits in terms of time-saving, accuracy, and operational efficiency for managing a modern supershop.

🎯 **Project Objective:**

To build a comprehensive **Supershop Management System** using **C language** that includes product tracking, sales and payment processing, employee records, and profit/loss calculations—all within a user-friendly terminal interface.

🛠️ **Technology Used**

* 💻 Programming Language: **C**
* 🧰 Compiler: **Code::Blocks / Dev-C++**
* 📊 Tools: **Flowcharts**, **Pseudocode**, **Structured Logic**

### 📑 **2. Table of Contents**

* Auto-generated if you're using MS Word or Google Docs before converting to PDF.
* Use:

markdown

CopyEdit

01. Customer Management..........................Page 3

02. Product Management...........................Page 5

...

### 🧩 **3. Section Layout (For Each Module)**

Each module should follow this consistent layout:

#### ✅ MODULE 01: Customer Management

**1.1 User Story**  
📖 Short description in bullet points

**1.2 Pseudocode**  
📘 Use monospaced font (like Courier New) and indent properly

**1.3 Flowchart**  
🗂 Insert the image centered, with a small title like:  
“Figure 1: Customer Management Flowchart”

**1.4 Code**  
💻 Format code blocks neatly with proper indentation and syntax highlighting (if possible)

**1.5 Input/Output Examples**  
📥 Input: ...  
📤 Output: ...

# **Module 1: Customer Management**

* 1. **User Story**

As a customer, I want to view available products and purchase items, so that I can buy what I need and get the total bill. Besides this supershop is placed at the middle of the town, as a result everyone can visit here and buy their necessary products.

* 1. **Pseudocode:**

Start

Display list of products

Ask user for their name

Repeat

Ask user to enter Product ID

If Product ID is 0, break loop

Check if Product ID is valid

Ask for quantity

If quantity is available

Calculate total price

Reduce product stock

Add to total bill

Else

Show "Insufficient stock"

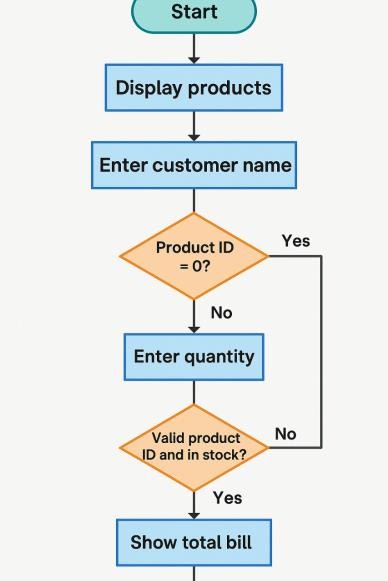
End Repeat

Show total bill with customer name

Add total to overall sales

Thank the customer

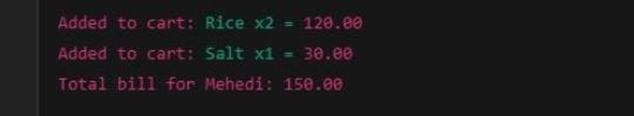
End

* 1. **Flowchart:**
  2. **Code**

1. void manageCustomers() {
2. char name[50];
3. int productID, quantity;
4. float total = 0;
5. displayProducts();
6. printf("\nEnter your name: ");
7. scanf("%s", name);
8. while (1) {
9. printf("Enter product ID to buy (0 to finish): ");
10. scanf("%d", &productID);
11. if (productID == 0) break;
12. if (productID > 0 && productID <= productCount) {
13. printf("Enter quantity: ");
14. scanf("%d", &quantity);
15. if (quantity <= products[productID - 1].quantity) {
16. float price = quantity \* products[productID - 1].price;
17. total += price;
18. products[productID - 1].quantity -= quantity;
19. printf("Added to cart: %s x%d = %.2f\n", products[productID - 1].name, quantity, price);
20. } else {
21. printf("Insufficient stock.\n");
22. }
23. } else {
24. printf("Invalid product ID.\n");
25. }
26. }
27. printf("\nTotal bill for %s: %.2f\n", name, total);
28. totalSales += total;
29. printf("\nTHANK YOU Dear Customer\n");
30. }
    1. **Input And Output:**



**Input**

****

**Output**

# **Module 2: Product Managment**

* 1. **User Story**

As a store manager, I want to be able to manage product information (like adding new products, editing existing ones, viewing all products, and monitoring stock levels), so that I can keep the inventory up to date and avoid stockouts or overstocking.

* 1. **Pseudocode**

START

DISPLAY product management menu:

View all products

Add a new product

Edit existing product

Search product by name

Alert for low stock Get user choice

IF choice == 1:

DISPLAY all products with ID, Name, Price, Quantity ELSE IF choice == 2:

INPUT product name, price, quantity ADD product to product list

ELSE IF choice == 3: INPUT product ID IF product found:

INPUT new name/price/quantity UPDATE product info

ELSE:

DISPLAY "Product not found" ELSE IF choice == 4:

INPUT product name

SEARCH and DISPLAY matched products ELSE IF choice == 5:

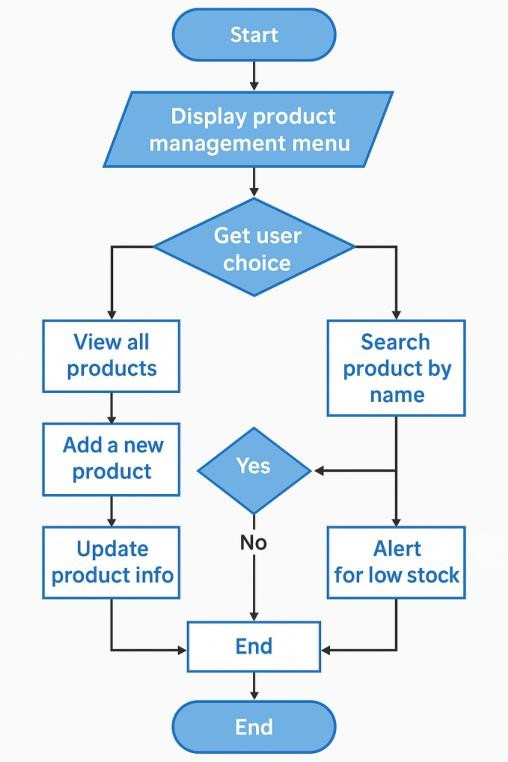
FOR each product:

IF quantity < threshold (e.g., 10): DISPLAY warning

ELSE:

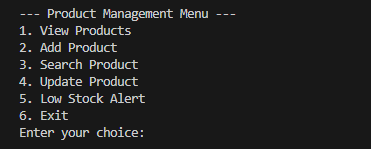
DISPLAY "Invalid choice" END

* 1. **Flowchart**

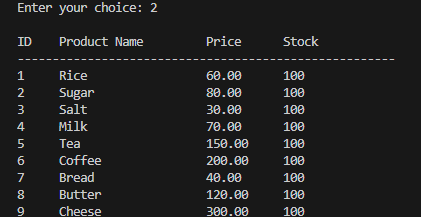
****

* 1. **Code**

1. #include <stdio.h>
2. #include <string.h>
3. struct Product {
4. int id;
5. char name[50];
6. float price;
7. int quantity;
8. };
9. struct Product products[100];
10. int productCount = 0;
11. void addProduct() {
12. printf("\nEnter Product ID: ");
13. scanf("%d", &products[productCount].id);
14. printf("Enter Product Name: ");
15. scanf(" %[^\n]", products[productCount].name);
16. printf("Enter Product Price: ");
17. scanf("%f", &products[productCount].price);
18. printf("Enter Product Quantity: ");
19. scanf("%d", &products[productCount].quantity);
20. productCount++;
21. printf("Product added successfully!\n");
22. }
23. void viewProducts() {
24. printf("\nProduct List:\n");
25. printf("ID\tName\t\tPrice\tQuantity\n");
26. for (int i = 0; i < productCount; i++) {
27. printf("%d\t%s\t\t%.2f\t%d\n", products[i].id, products[i].name, products[i].price, products[i].quantity);
28. }
29. }
30. void searchProduct() {
31. char searchName[50];
32. printf("\nEnter product name to search: ");
33. scanf(" %[^\n]", searchName);
34. int found = 0;
35. for (int i = 0; i < productCount; i++) {
36. if (strcmp(products[i].name, searchName) == 0) {
37. printf("Product Found: ID: %d, Name: %s, Price: %.2f, Quantity: %d\n",
38. products[i].id, products[i].name, products[i].price, products[i].quantity);
39. found = 1;
40. }
41. }
42. if (!found)
43. printf("Product not found.\n");
44. }
45. void updateProduct() {
46. int id;
47. printf("\nEnter Product ID to update: ");
48. scanf("%d", &id);
49. for (int i = 0; i < productCount; i++) {
50. if (products[i].id == id) {
51. printf("Enter new name: ");
52. scanf(" %[^\n]", products[i].name);
53. printf("Enter new price: ");
54. scanf("%f", &products[i].price);
55. printf("Enter new quantity: ");
56. scanf("%d", &products[i].quantity);
57. printf("Product updated!\n");
58. return;
59. }
60. }
61. printf("Product with ID %d not found.\n", id);
62. }
63. void lowStockAlert() {
64. printf("\nProducts with low stock (less than 10):\n");
65. for (int i = 0; i < productCount; i++) {
66. if (products[i].quantity < 10) {
67. printf("ID: %d, Name: %s, Quantity: %d\n", products[i].id, products[i].name, products[i].quantity);}
68. }
69. }
70. int main() {
71. int choice;
72. while (1) {
73. printf("\n--- Product Management Menu ---\n");
74. printf("1. View Products\n2. Add Product\n3. Search Product\n4. Update Product\n5. Low Stock Alert\n6. Exit\n");
75. printf("Enter your choice: ");
76. scanf("%d", &choice);
77. switch (choice) {
78. case 1: viewProducts(); break;
79. case 2: addProduct(); break;
80. case 3: searchProduct(); break;
81. case 4: updateProduct(); break;
82. case 5: lowStockAlert(); break;
83. case 6: return 0;
84. default: printf("Invalid choice!\n");
85. }
86. }
87. return 0;
88. }
    1. **Input and Output:**

****

Input



Output

# **Module 03: Employee Managment**

* 1. **User Story:**

As an admin, I want to add, update, view, and delete employee records so I can manage the workforce efficiently.

* 1. **Pseudocode:**

Start

Display Employee Menu

Add Employee

View Employees

Update Employee

Delete Employee If user chooses Add:

Input ID, Name, Position, Salary Store in employee file

If View:

Read from file and display all

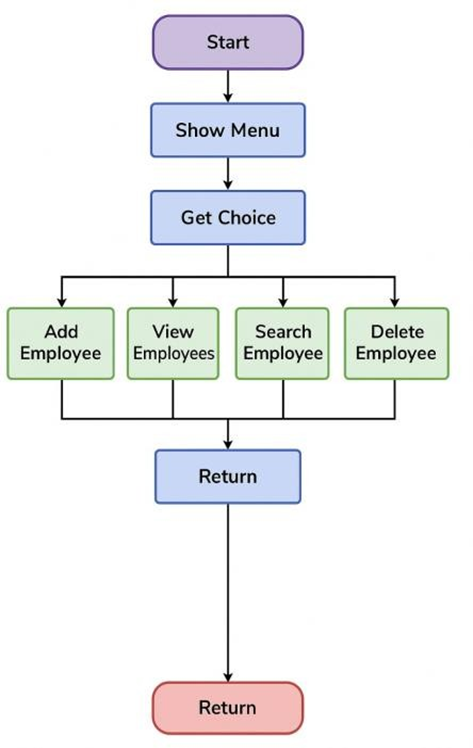
If Update:

Search by ID and edit fields If Delete:

Search by ID and remove from file

End

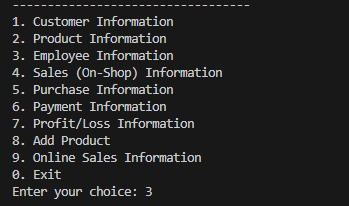
* 1. **Flowchart:**

****

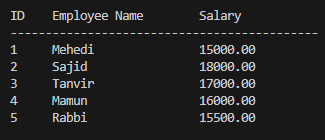
* 1. **Code:**

1. // Employee structure
2. struct Employee {
3. char name[50];
4. int id;
5. float salary;
6. };
7. struct Employee employees[MAX\_EMPLOYEES] = {
8. {"Mehedi", 1, 15000},
9. {"Sajid", 2, 18000},
10. {"Tanvir", 3, 17000},
11. {"Mamun", 4, 16000},
12. {"Rabbi", 5, 15500}
13. };
14. // Employee Management Function
15. void manageEmployees() {
16. printf("\n%-5s %-20s %-10s\n", "ID", "Employee Name", "Salary");
17. printf("--------------------------------------------\n");
18. for (int i = 0; i < 5; i++) {
19. printf("%-5d %-20s %-10.2f\n", employees[i].id, employees[i].name, employees[i].salary);
20. }
21. }

**1.5 Input and Output:**

****

**Input**

****

**Output**

# **Module 04: Sales on Shop Management**

**4.1: user story:**

**As a** shop employee**, I want** to record sales quickly and accurately, **so that** I can manage inventory and generate receipts efficiently.

**4.2: psedocode:**

Start

Input Product ID and Quantity

Check availability

Calculate total price

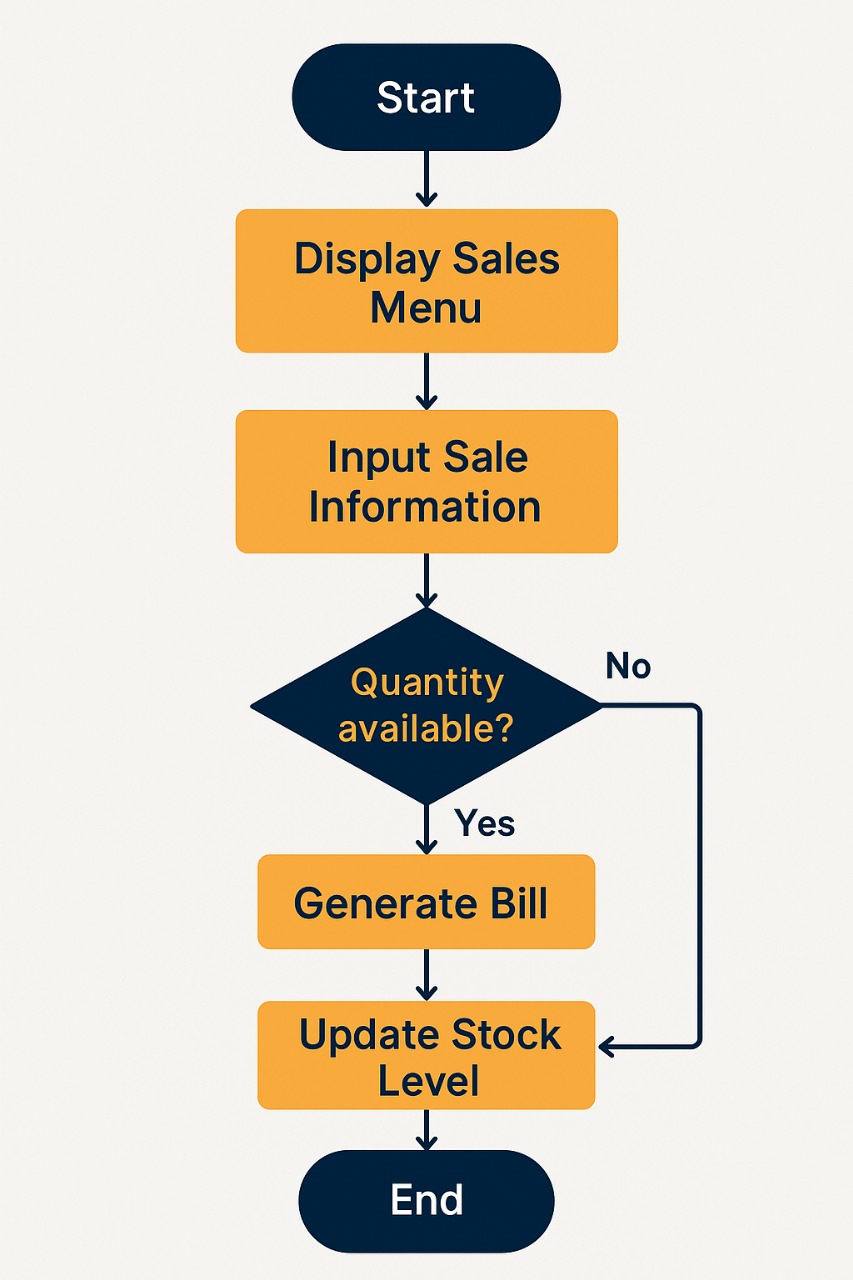
Reduce quantity from stock

Generate invoice

Store transaction

End

**4.3: Flowchart:**



**4.4: coding**

void manageCustomers() {

    char name[50];

    int productID, quantity;

    float total = 0;

    displayProducts();

    printf("\nEnter your name: ");

    scanf("%s", name);

    while (1) {

        printf("Enter product ID to buy (0 to finish): ");

        scanf("%d", &productID)

        if (productID == 0) break;

        if (productID > 0 && productID <= productCount) {

            printf("Enter quantity: ");

            scanf("%d", &quantity);

            if (quantity <= products[productID - 1].quantity) {

                float price = quantity \* products[productID - 1].price;

                total += price;

                products[productID - 1].quantity -= quantity;

                printf("Added to cart: %s x%d = %.2f\n",

                    products[productID - 1].name, quantity, price);

            } else {

                printf("❌ Insufficient stock.\n");

            }

        } else {

            printf("❌ Invalid product ID.\n");

        }

    }

    printf("\n🧾 Total bill for %s: %.2f\n", name, total);

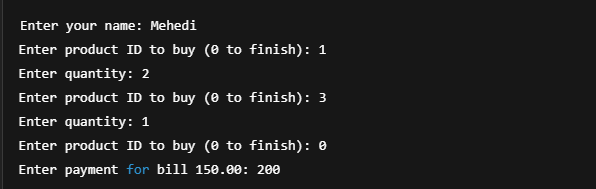
    totalSales += total;

    paymentManagement(total);

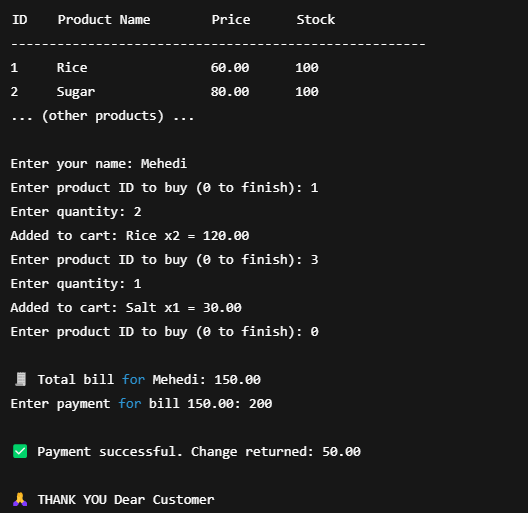
    printf("\n🙏 THANK YOU Dear Customer\n");

}

**4.3 Input output:**

****

**Input**

****

**Output**

# **Module 05: Purchase Managment**

**5.1: user story:**

🔹 5.1 User Story – Purchase Management As a shop employee or admin, I want to record and manage the purchases of products from suppliers, So that I can keep accurate inventory records and ensure stock availability.

**5.2: Psedocode**

Start

Display "Purchase Management Menu"

Loop until user chooses to exit:

1. Add New Purchase

- Ask for Product ID

- Ask for Supplier Name

- Ask for Purchase Quantity

- Ask for Unit Price

- Calculate Total Cost = Quantity × Unit Price

- Save purchase data to file or database

- Update product stock

2. View All Purchases

- Open purchase records file/database

- Display all records

3. Search Purchase by Product ID

- Ask for Product ID

- Search and display matching records

4. Delete a Purchase Record

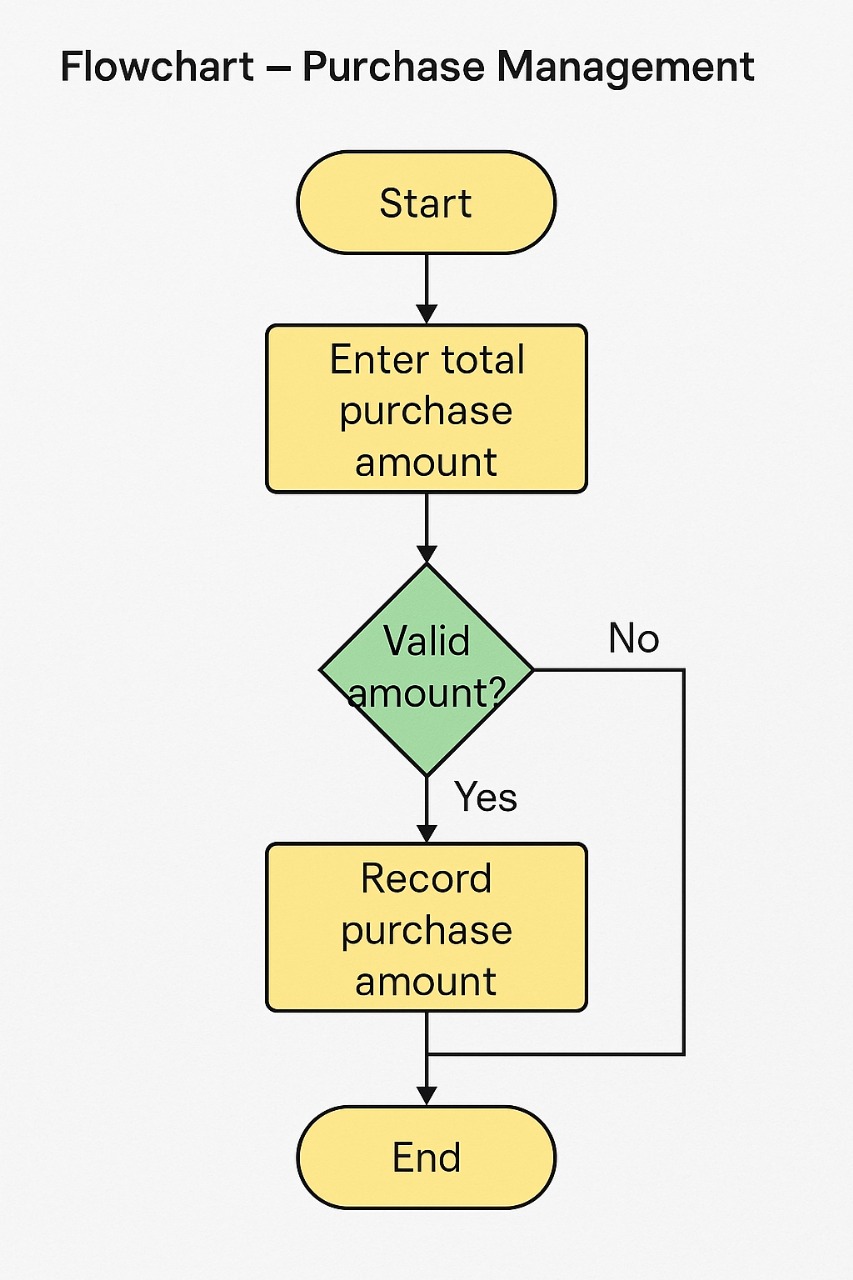
- Ask for Purchase ID or Product ID

- Remove from file/database

5. Exit

End

**5.3: Flowchart:**



**5.4 coding:**

float totalPurchases = 0;

void purchaseManagement() {

    float amount;

    printf("\nEnter total purchase amount (stock purchase): ");

    scanf("%f", &amount);

    totalPurchases += amount;

    printf("✅ Recorded purchase amount: %.2f\n", amount);

}:

**5.4 input er output:**



Input



Output

**Module 06: Profit/Loss Management**

## **6.1 User Story:**

**Title:** Handle Customer Payment After Sale or Manual Entry  
**As a** cashier or shop employee, **I want** to input and verify customer payments after purchases or receive manual payments, s**o that** I can ensure accurate billing, detect underpayments, and calculate change for overpayments.

**6.2 Pseudcode:**

FUNCTION paymentManagement(bill)

DECLARE amount AS float

IF bill = 0 THEN

DISPLAY "Manual Payment Entry"

PROMPT "Enter amount received from customer"

READ amount

DISPLAY "Payment of amount received successfully"

ELSE

DISPLAY "Process Payment"

PROMPT "Enter payment for bill <bill>"

READ amount

IF amount < bill THEN

DISPLAY "Insufficient payment. More amount required"

ELSE

CALCULATE change = amount - bill

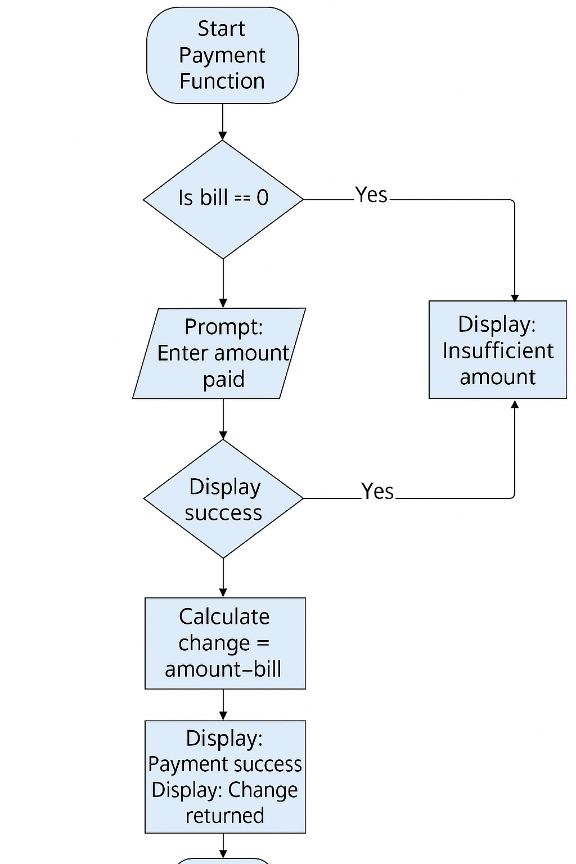
DISPLAY "Payment successful. Change returned: change"

END IF

END IF

END FUNCTION

**6.3 Flowchart:**



**6.4 Code:**

void paymentManagement(float bill) {

    float amount;

    // Case 1: Manual payment without linked bill (e.g., from menu)

    if (bill == 0) {

        printf("\n========== Manual Payment Entry ==========\n");

        printf("Enter amount received from customer: ");

        scanf("%f", &amount);

        printf("✅ Payment of %.2f received successfully.\n", amount);

    }

    // Case 2: Linked payment after purchase

    else {

        printf("\n========== Process Payment ==========\n");

        printf("Enter payment for bill %.2f: ", bill);

        scanf("%f", &amount);

        if (amount < bill) {

            printf("❌ Insufficient payment. %.2f more required.\n", bill - amount);

        } else {

            printf("✅ Payment successful. Change returned: %.2f\n", amount - bill);

        }

    }

}

**6.5 Input & Output:**

****

Input



Output

# 

# **Module 07: Profit/Loss Managment**

**7.1 User story**

**As a** shop manager, **I want to** view and track daily, weekly, monthly, and yearly profit/loss reports, **So that** I can understand the financial performance of the shop and make informed business decisions.

**Generate Reports:**

* The system should allow the manager to generate profit/loss reports based on selected time ranges (daily, weekly, monthly, yearly).
* Each report should include:  
  + Total sales revenue
  + Total cost of goods sold (COGS)  
    Total expenses (e.g., salaries, rent, utilities)
  + Net profit or loss
* The manager should be able to filter reports by date, category, or product.
* The system should provide a visual representation (graphs or charts) of profit/loss trends over time.
* The report should be exportable in PDF, Excel, and CSV formats.
* Only authorized users (e.g., Admin or Manager roles) can access this module.
* The system should allow comparison between different time periods (e.g., this month vs. last month).

**7.2 Pseudocode :**

Start

Function ProfitLossReport(startDate, endDate):

sales = GetSales(startDate, endDate)

cost = GetCost(startDate, endDate)

expenses = GetExpenses(startDate, endDate)

profit = sales - (cost + expenses)

Show(profit)

End Function

Function Export(type):

If type == PDF → ExportPDF()

If type == Excel → ExportExcel()

If type == CSV → ExportCSV()

End Function

Function Compare(period1, period2):

report1 = ProfitLossReport(period1)

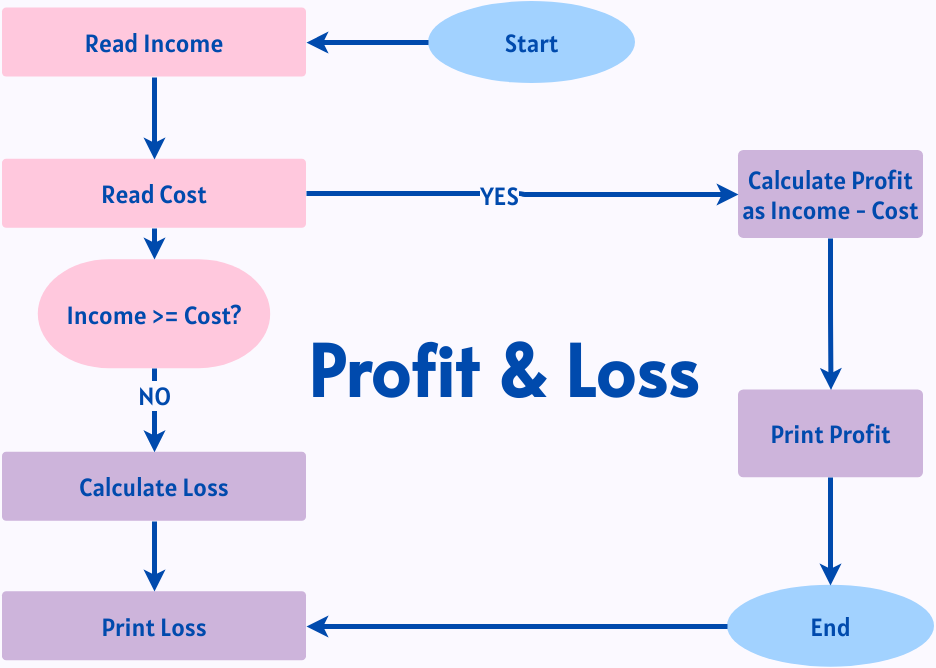
report2 = ProfitLossReport(period2)

ShowComparison(report1, report2)

End Function

End

**7.3 Flowchart**



**7.4 Coding :**

void profitLossManagement() {

    float profit = totalSales - totalPurchases;

    printf("\n========== Profit/Loss Summary ==========\n");

    printf("Total Sales: %.2f\n", totalSales);

    printf("Total Purchase Cost: %.2f\n", totalPurchases);

    if (profit >= 0) {

        printf("✅ Net Profit: %.2f\n", profit);

    } else {

        printf("❌ Net Loss: %.2f\n", -profit);

    }

}

**7.5 Input Output**



Input & Output

# **Module 08: Add Product Managment**

**8.1 User Story**

As a store manager or inventory staff, I want to add new products to the system with all necessary details, So that the products are available for sale and stock tracking.

Save Product:

* On save, the product is added to the inventory database.
* System confirms successful addition.

**8.2 Pseudocode**

Start

Function AddProduct():

Display "Enter product details"

name = Input("Product Name")

category = Input("Category")

supplier = Input("Supplier")

purchasePrice = Input("Purchase Price")

sellingPrice = Input("Selling Price")

quantity = Input("Quantity")

unit = Input("Unit (pcs/kg/liter)")

expiryDate = Input("Expiry Date (optional)")

barcode = Input("Barcode/SKU")

If name is empty OR category is empty OR quantity is empty:

Show "Error: Required fields missing"

Exit Function

If sellingPrice < purchasePrice:

Show "Warning: Selling price is less than purchase price"

If ProductExists(barcode):

Show "Product already exists"

Exit Function

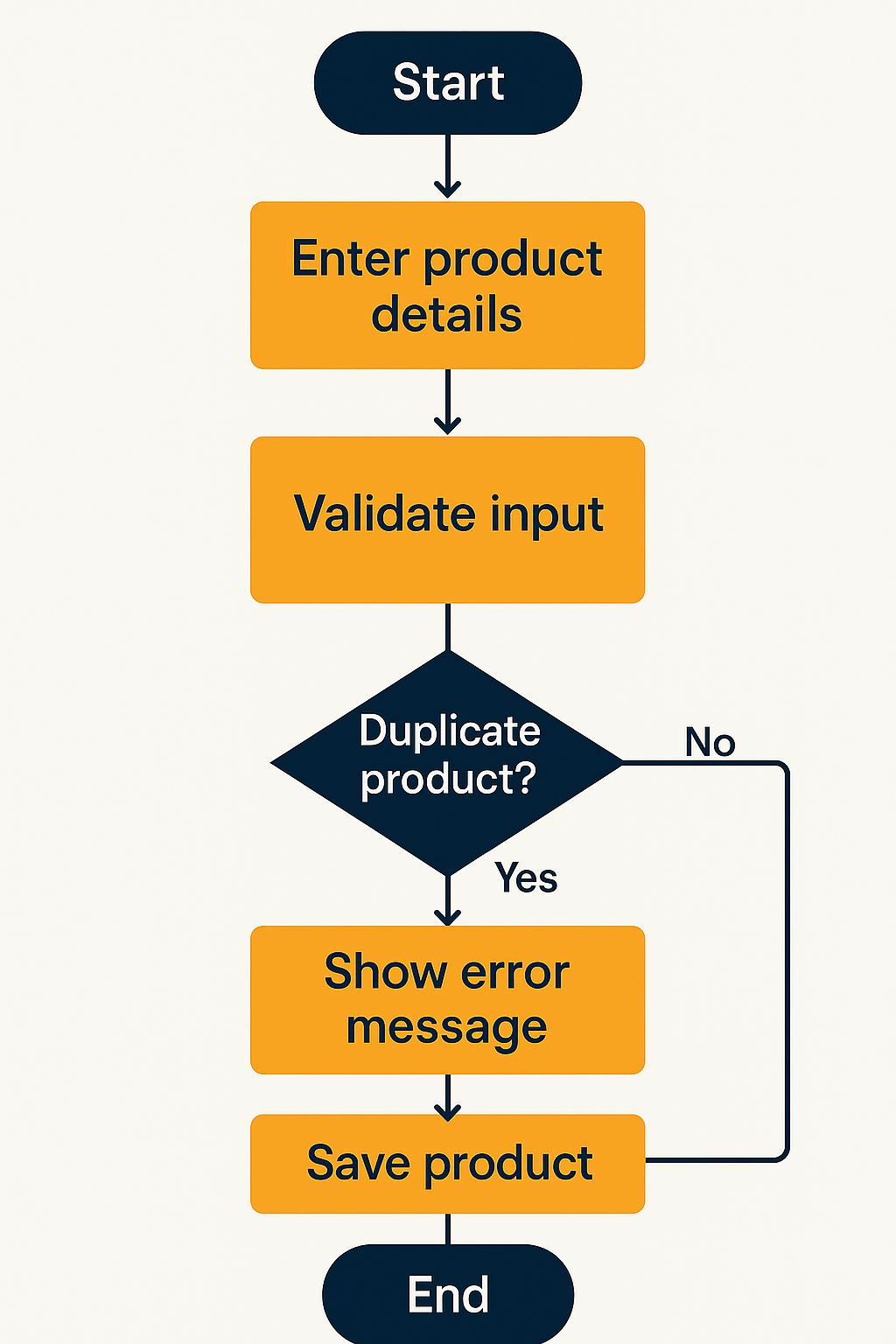
SaveProduct(name, category, supplier, purchasePrice, sellingPrice, quantity, unit, expiryDate, barcode)

Show "Product added successfully"

End Function

End

**8.3 Flowchart:**

****

**8.4 Coding :**

int productCount = 50;

void addProduct() {

    if (productCount >= MAX\_PRODUCTS) {

        printf("❌ Cannot add more products. Maximum limit reached.\n");

        return;

    }

    printf("\n========== Add New Product ==========\n");

    printf("Enter product name: ");

    scanf("%s", products[productCount].name);

    printf("Enter price: ");

    scanf("%f", &products[productCount].price);

    printf("Enter quantity: ");

    scanf("%d", &products[productCount].quantity);

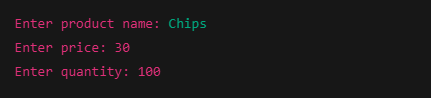
    products[productCount].id = productCount + 1;

    productCount++;

    printf("✅ Product added successfully!\n");

}

**8.5 Input Output:**

****

**Input**

****

**Output**