Bootcamp on

"Critical Thinking & Problem Solving"

TEAM NAME: SEGMENTATION FAULT

S.NO	NAME	USN
1	Nandini Hegde	4MT22CI017
2	Marushka Rachelle D Silva	4MT22CI029
3	Masudi Sai Harsha	4MT22CI030
4	Nischal K	4MT22CI036
5	Riyaz Khan	4MT22CI041
6	Sujal Revenkar	4MT22CI055
7	Vikas M	4MT22CI060
8	Yashita Vasu	4MT22CI062

CONTENTS

● INTRODUCTION :

❖ What is E-commerce?

E-commerce enables internet shopping! It resembles an online shopping mall where businesses (websites) sell goods. You peruse the merchandise, place it in your virtual cart, and proceed to pay for it.

Here is a simple product management system:

List: retains product information (price, description).Product classifications include groupings such as electronics, clothing, etc.

Shopping Cart: Stores the items you've selected to purchase. Think of it as a shopping list for you to jot down products (add to cart), check your cart to see the complete list, and then check out to make your purchase.

❖ How does it work ?

E-commerce works just like an online retailer.This is how it operates:
Online retailers: Websites that showcase their products are created by businesses, much like an electronic catalog.
Find the Need That You Have: Use search and filtering to find what you're searching for by brand, category , or even color.
Add Products to the Cart: You could add items you like to a virtual shopping cart, just like you would when you make purchases at a store.
Delivery Enchantment: The dealer delivers your stuff straight to your door. E-commerce makes shopping more convenient since it provides a wider range of options and the freedom to shop from any location.

Project Description:

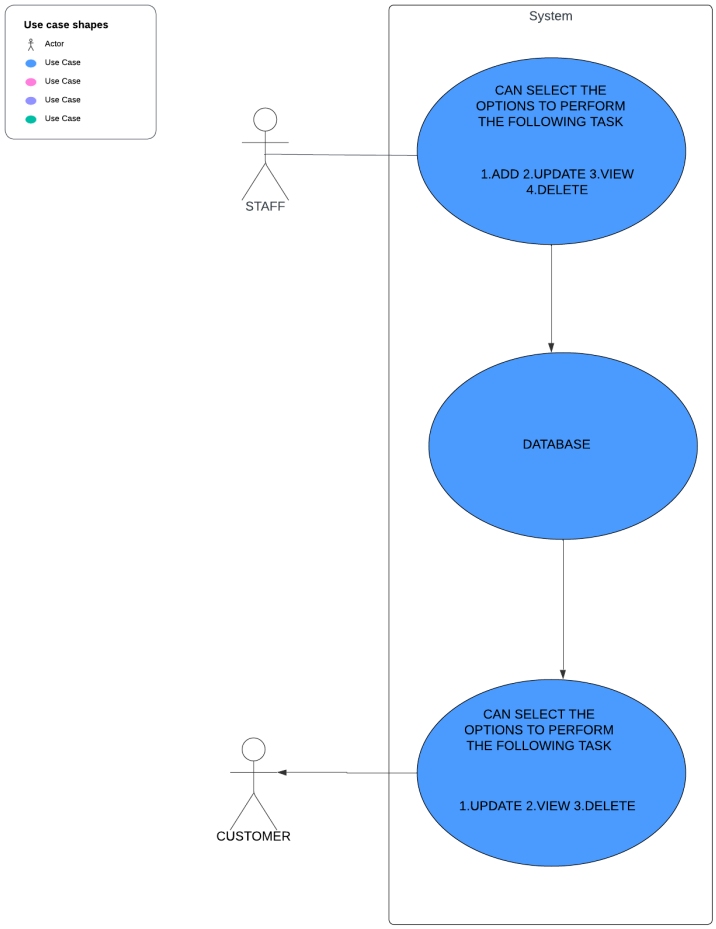
An online store's fundamental inventory management is run by this C programme. Below is a summary of its features:

1. Items:keeps track of product details such as the code, name, cost, weight, quantity, and description.lets you add new products and their details.
2. Product List:shows a list of every product that is offered along with its details.
3. Product Lookup:allows visitors to enter a product code to search for products.
4. Update on Product:allows for the product code to be used to change the name, rate, quantity, and other details of an existing product.
5. Product Removal:permits the use of the product code to remove products from the system.
6. Sales:enters the product code and quantity to process product sales. Verifies that there is enough inventory on hand before finalizing the transaction. Determines the overall cost of the bill by using the quantity and prices of the products.
7. Information Storage: stores product information in a text file .

All things considered, this programme offers a fundamental basis for product inventory management in C programmes. It can be expanded to include more functionalities like user accounts, other product categories, and stronger data storage systems.

- Use Case:

For your online store, this C programme offers a powerful inventory management system. It enables employees to handle your product catalog with ease by acting as a centralized product database. Employees have the ability to add new product listings, complete with prices, descriptions, and stock levels. The program's sophisticated search features and real-time product information updating enable effective stock control. Employees can also easily remove items that are out of stock or outdated. A product catalog that is easy to use helps customers by making their purchasing experience more efficient. They can easily peruse your product offerings, get access to comprehensive product details, and decide what to buy. This programme maximizes inventory control and product accessibility, which raises employee productivity and improves consumer satisfaction.



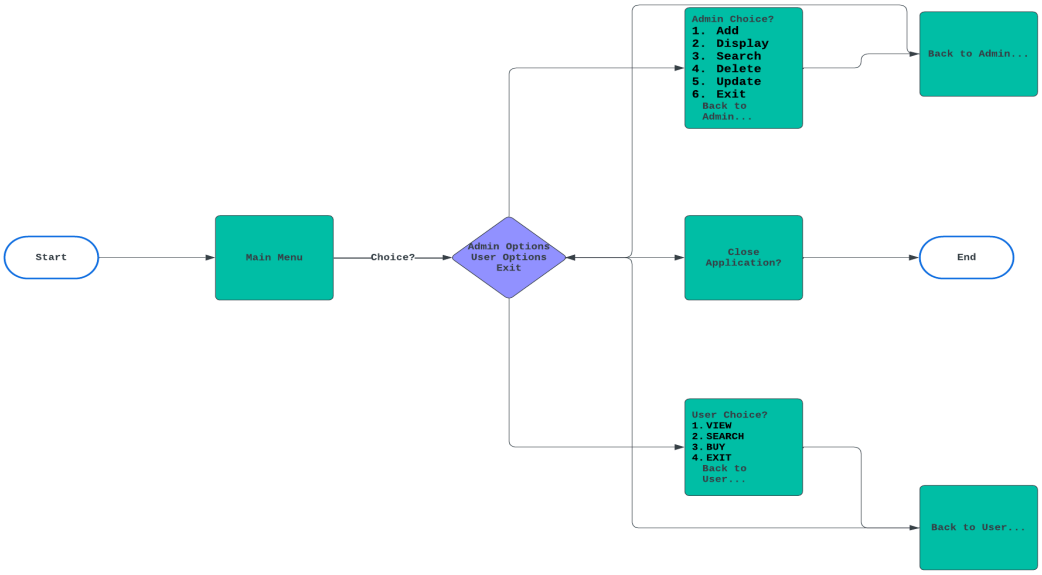
TEST CASE:

Test Case	Description	Expected Solution
Product Addition	Staff attempts to add a new product with a unique code, name, price, quantity, weight, and description.	The program successfully adds the new product to the inventory database (Record.txt) and displays a confirmation message.
Duplicate Product Code	Staff attempts to add a new product with a code that already exists in the database.	The program displays an error message indicating that the product code is already in use and prompts the staff to enter a unique code.
Invalid Input (Product Addition)	Staff enters invalid data during product addition, such as negative quantities, non-numeric characters in price, or leaving required fields blank.	The program detects the invalid input and displays an error message prompting the staff to enter valid data.
Product Display	Staff requests to view the list of available products.	The program retrieves and displays a formatted list of all products in the database, including code, name, rate, quantity, weight, and description.
Empty Inventory	Staff attempts to view the list of available products when there are no products in the database.	The program displays an appropriate message indicating that there are currently no products in the inventory.

Product Search (Existing Product)	Staff searches for a product using a valid existing product code.	The program retrieves and displays the details of the matching product, including code, name, rate, quantity, weight, and description.
Product Search (Non-existent Product)	Staff searches for a product using a code that does not exist in the database.	The program displays a message indicating that no product was found with the provided code.
Product Update	Staff attempts to update the details (name, price, quantity, etc.) of an existing product using its code.	The program retrieves the product information, allows staff to modify the desired details, and saves the updated information to the database. A confirmation message is displayed.
Product Update (Non-existent Product)	Staff attempts to update the details of a product using a code that does not exist in the database.	The program displays an error message indicating that no product was found with the provided code.
Product Deletion	Staff attempts to delete a product from the database using its code.	The program locates the product, confirms deletion with the staff, and removes the product information from the database. A confirmation message is displayed.
Deletion Attempt (Non-existent Product)	Staff attempts to delete a product using a code that does not exist in the database.	The program displays an error message indicating that no product was found with the provided code.
Sale - Sufficient Stock	A customer places an order for a product with a quantity less than or equal to the available stock.	The program verifies the stock availability, processes the sale, and displays the total bill amount. Updates the product quantity in the database.
Sale - Insufficient Stock	A customer places an order for a product with a quantity exceeding the available stock.	The program displays an error message indicating insufficient stock and prompts the customer to enter a lower quantity or choose another product.

WORKFLOW:

1. **Start:** Begin with an oval labeled "Start."
2. **Main Menu:** Draw a rectangle labeled "Main Menu" with options: Admin, User, and Exit.
3. **Decision:** Add a diamond labeled "Choice?" to determine the user's selection.
    - If Admin is selected, go to Admin Options.
    - If User is selected, go to User Options.
    - If Exit is selected, go to Close Application.
4. **Admin Options:** Draw a rectangle labeled "Admin Options" with choices: Add Product, Display Products, Search Product, Delete Product, Update Product, Close Application, and Sale Product.
5. **Decision (Admin):** Add a diamond labeled "Admin Choice?" to handle Admin's selections.
    - For each Admin option, go to its respective process or return to Admin Options.
6. **User Options:** Draw a rectangle labeled "User Options" with options: View Products, Search Product, Buy Product, and Close Application.
7. **Decision (User):** Add a diamond labeled "User Choice?" to handle User's selections.
    - For each User option, go to its respective process or return to User Options.
8. **Close Application:** Draw a diamond labeled "Close Application?" with options Yes and No.
    - If Yes, end the program.
    - If No, return to the previous menu.
9. **End:** Conclude with an oval labeled "End."
10. **Connectivity:** Ensure each process has a way back to its respective options menu, and the Close Application decision leads to either ending the program or returning to the previous menu.



Requirements:

Feature	Description
Product Management	Add new products (code, name, price, quantity, weight, description). View list of all products with details. Search for products by code. Update product information (name, price, quantity, etc.). Delete products from inventory.
Inventory Control	Track product stock levels (quantity). Display low stock or out-of-stock alerts (optional). Allow for setting minimum stock levels (optional).
Sales Processing	Process product sales by entering product code and quantity. Verify sufficient stock before completing a sale. Calculate total bill amount based on product price and quantity.
User Interface	Menu-driven interface for staff interaction. Clear and user-friendly prompts and messages. Handle invalid user inputs gracefully.
Data Storage	Store product information in a text file (e.g., Record.txt). Consider alternative data storage methods for scalability (optional, e.g., database).
Non-Functional Requirements	Basic access control mechanisms (optional). Responsive performance and timely user feedback. Scalability to handle increasing products and transactions (optional). Well-structured, documented, and maintainable code.

CODE:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX 20

typedef struct items {
    char product_code[MAX];
    char product_name[MAX];
    int rate;
    int quantity;
    char weight[MAX];
    char description[30];
} ITEM;

ITEM item;
int isCodeAvailable(char code[]) {
    FILE *file = fopen("Record.txt", "r");
    while (fread(&item, sizeof(item), 1, file)) {
        if (strcmp(code, item.product_code) == 0) {
            fclose(file);
            return 1;
        }
    }
    fclose(file);
    return 0;
}

int isCodeAvailableclass(char code[]) {
    FILE *file = fopen("Record.txt", "r");
    while (fread(&item, sizeof(item), 1, file)) {
        if (strcmp(code, item.weight) == 0) {
            fclose(file);
            return 1;
        }
    }
    fclose(file);
    return 0;
```

```
}

int isProductAvailable(int quantity) {
    FILE *file = fopen("Record.txt", "r");
    while (fread(&item, sizeof(item), 1, file)) {
        if (item.quantity >= quantity) {
            fclose(file);
            return 1;
        }
    }
    fclose(file);
    return 0;
}

int get_int() {
    int input;
    char ch;
    while (scanf("%d", &input) != 1 || input <= 0) {
        while ((ch = getchar()) != '\n' && ch != EOF);
        printf("Must be a positive integer. Enter a positive integer value: ");
    }
    return input;
}

void addProduct() {
    FILE *file = fopen("Record.txt", "ab");
    char code[MAX];
    printf("Enter \"end\" to exit\n\nEnter Product Code: ");
    scanf("%s", code);
    if (strcmp(code, "end") == 0) {
        return;
    }
    if (isCodeAvailable(code)) {
        printf("* Product is already there.\n");
        return;
    }
    strcpy(item.product_code, code);
    printf("Enter Product Name: ");
    scanf("%s", item.product_name);
    printf("Enter Product Rate: ");
    item.rate = get_int();
```

```
    printf("Enter Quantity: ");
    item.quantity = get_int();
    printf("Enter classification : ");
    scanf("%s", item.weight);
    printf("Enter Product Description: ");
    scanf(" %29[^\n]", item.description);
    fwrite(&item, sizeof(item), 1, file);
    fclose(file);
}

void display() {
    FILE *file = fopen("Record.txt", "rb");
    if (file == NULL) {
        printf("No Product is inserted.\n");
        return;
    }
    printf("-----------------------------------------------------------------------------------------------------\n");
    printf("CODE\t|\tNAME\t|\tRATE\t|\tQUANTITY\t|\tWEIGHT\t|\tDESCRIPTION\n");
    printf("-----------------------------------------------------------------------------------------------------\n");
    while (fread(&item, sizeof(item), 1, file)) {
        printf("%s\t|\t%s\t|\t%d\t|\t%d\t\t|\t%s\t|\t%s\n", item.product_code, item.product_name,
item.rate, item.quantity, item.weight, item.description);
    }
    printf("-----------------------------------------------------------------------------------------------------\n");
    fclose(file);
}

void displayc() {
    FILE *file = fopen("Record.txt", "rb");
    if (file == NULL) {
        printf("No Product is inserted.\n");
        return;
    }
    printf("-----------------------------------------------------------------------------------------------------\n");
    printf("CODE\t|\tNAME\t|\tRATE\t|\tQUANTITY\t|\tCLASS\t|\tDESCRIPTION\n");
    printf("-----------------------------------------------------------------------------------------------------\n");
    while (fread(&item, sizeof(item), 1, file)) {

        printf("%s\t|\t%s\t|\t%d\t|\t%d\t\t|\t%s\t|\t%s\n", item.product_code, item.product_name,
item.rate, item.quantity, item.weight, item.description);
    }
```

```
    printf("-----------------------------------------------------------------------------------------------------\n");
    fclose(file);
}
void search() {
    int choice;
    do {
        printf("\nSEARCH BY\n");
        printf("1. Product code\n");
        printf("2. Product classification\n");
        printf("3. Exit\n");
        printf("Enter your choice: ");
        choice = get_int();
        switch (choice) {
        case 1: searchcode(); break;
        case 2: searchclass(); break;
        case 3: closeApp(); break;
        default: printf("* Invalid choice! Please try again.\n"); break;
        }
    } while (choice != 3);
}

void searchcode() {
    FILE *file;
    char code[MAX];
    printf("Enter \"end\" to go back to menu.\n\nEnter the Product Code to search: ");
    scanf("%s", code);
    if (strcmp(code, "end") == 0) {
        return;
    }
    if (!isCodeAvailable(code)) {
        printf("* Product code not found.\n");
        return;
    }
    printf("Product Information\n");
    file = fopen("Record.txt", "rb");

    while (fread(&item, sizeof(item), 1, file)) {
        if (strcmp(item.product_code, code) == 0) {
            printf("Product Code:        %s\n", item.product_code);
            printf("Name of Product:     %s\n", item.product_name);
            printf("Rate of Product (RS): %d\n", item.rate);
```

```
            printf("Product Weight:      %s\n", item.weight);
            printf("Product Description: %s\n", item.description);
            break;
        }
    }
    fclose(file);
}
void searchclass() {
    FILE *file;
    char classification[MAX];
    printf("Enter \"end\" to go back to menu.\n\nEnter the Product Code to search: ");
    scanf("%s", classification);
    if (strcmp(classification, "end") == 0) {
        return;
    }
    while (!isCodeAvailableclass(classification)) {
        printf("* Product code not found.\n");
        return;
    }

    printf("Product Information\n");
    file = fopen("Record.txt", "rb");

     printf("-----------------------------------------------------------------------------------------------------\n");
    printf("CODE\t|\tNAME\t|\tRATE\t|\tQUANTITY\t|\tCLASS\t|\tDESCRIPTION\n");
    printf("-----------------------------------------------------------------------------------------------------\n");

    while (fread(&item, sizeof(item), 1, file)) {
        if (strcmp(item.weight, classification) == 0) {
        printf("%s\t|\t%s\t|\t%d\t|\t%d\t\t|\t%s\t|\t%s\n", item.product_code, item.product_name,
item.rate, item.quantity, item.weight, item.description);
    }}
    printf("-----------------------------------------------------------------------------------------------------\n");
    fclose(file);
}



void deleteRecord() {
    FILE *file1, *file2;
    char code[MAX];
```

```
    display();
    printf("Enter the Product Code to delete: ");
    scanf("%s", code);
    if (!isCodeAvailable(code)) {
        printf("* Product not available.\n");
        return;
    }
    file1 = fopen("Record.txt", "rb");
    file2 = fopen("tempfile.txt", "wb");
    while (fread(&item, sizeof(item), 1, file1)) {
        if (strcmp(item.product_code, code) != 0) {
            fwrite(&item, sizeof(item), 1, file2);
        }
    }
    fclose(file1);
    fclose(file2);
    file1 = fopen("Record.txt", "wb");
    file2 = fopen("tempfile.txt", "rb");
    while (fread(&item, sizeof(item), 1, file2)) {
        fwrite(&item, sizeof(item), 1, file1);
    }
    fclose(file1);
    fclose(file2);
    printf("* Product deleted successfully!\n");
}

void updateProduct() {
    FILE *file1, *file2;
    char code[MAX];
    printf("Enter the Product Code to update the record: ");
    scanf("%s", code);
    if (!isCodeAvailable(code)) {
        printf("* No product found for update.\n");
        return;
    }
    file1 = fopen("Record.txt", "rb");
    file2 = fopen("tempfile.txt", "wb");
    while (fread(&item, sizeof(item), 1, file1)) {
        if (strcmp(item.product_code, code) == 0) {
            printf("Updating data for the previous product %s\n", code);
            printf("Enter Product Name: ");
```

```
            scanf("%s", item.product_name);
            printf("Enter Product Rate: ");
            item.rate = get_int();
            printf("Enter Quantity: ");
            item.quantity = get_int();
            printf("Enter classification(in caps) ");
            scanf("%s", item.weight);
            printf("Enter Product Description: ");
            scanf(" %29[^\n]", item.description);
            fwrite(&item, sizeof(item), 1, file2);
        } else {
            fwrite(&item, sizeof(item), 1, file2);
        }
    }
    fclose(file1);
    fclose(file2);
    file1 = fopen("Record.txt", "wb");
    file2 = fopen("tempfile.txt", "rb");
    while (fread(&item, sizeof(item), 1, file2)) {
        fwrite(&item, sizeof(item), 1, file1);
    }
    fclose(file1);
    fclose(file2);
    printf("* Product updated successfully!\n");
}

void saleProduct() {
    FILE *file1, *file2;
    char code[MAX];
    int quantity;
    printf("Enter the Product Code: ");
    scanf("%s", code);
    if (!isCodeAvailable(code)) {
        printf("* No product found for sale.\n");
        return;
    }
    file1 = fopen("Record.txt", "rb");
    file2 = fopen("tempfile.txt", "wb");
    while (fread(&item, sizeof(item), 1, file1)) {
        if (strcmp(item.product_code, code) == 0) {
            printf("Enter the Quantity: ");
```

```
            quantity = get_int();
            if (isProductAvailable(quantity)) {
                item.quantity -= quantity;
                float total_amount = item.rate * quantity;
                fwrite(&item, sizeof(item), 1, file2);
                fclose(file1);
                fclose(file2);
                printf("* Total Amount: %.2f\n", total_amount);
                return;
            } else {
                printf("* Out of stock or insufficient quantity.\n");
                return;
            }
        } else {
            fwrite(&item, sizeof(item), 1, file2);
        }
    }
    fclose(file1);
    fclose(file2);
    printf("* Product sold successfully!\n");
}

void closeApp() {
    char choice;
    printf("Are you sure you want to exit? (Y/N): ");
    scanf(" %c", &choice);
    if (choice == 'Y' || choice == 'y') exit(0);
}

void adminOptions() {
    int choice;
    do {
        printf("\nAdmin Panel\n");
        printf("1. Add Product\n");
        printf("2. Display Products\n");
        printf("3. Search Product\n");
        printf("4. Delete Product\n");
        printf("5. Update Product\n");
        printf("6. Close Application\n");
        printf("7. Sale Product\n");
        printf("Enter your choice: ");
```

```
        choice = get_int();
        switch (choice) {
        case 1: addProduct(); break;
        case 2: display(); break;
        case 3: search(); break;
        case 4: deleteRecord(); break;
        case 5: updateProduct(); break;
        case 6: closeApp(); break;
        case 7: saleProduct(); break;
        default: printf("* Invalid choice! Please try again.\n"); break;
        }
    } while (choice != 6);
}

void userOptions() {
    int choice;
    do {
        printf("\nUser Panel\n");
        printf("1. View Products\n");
        printf("2. Search Product\n");
        printf("3. Buy Product\n");
        printf("4. Close Application\n");
        printf("Enter your choice: ");
        choice = get_int();
        switch (choice) {
        case 1: display(); break;
        case 2: search(); break;
        case 3: saleProduct(); break;
        case 4: closeApp(); break;
        default: printf("* Invalid choice! Please try again.\n"); break;
        }
    } while (choice != 4);
}

void mainMenu() {
    int choice;
    do {
        printf("\nInventory Management System\n");
        printf("1. Admin\n");
        printf("2. User\n");
        printf("3. Exit\n");
```

```
        printf("Enter your choice: ");
        choice = get_int();
        switch (choice) {
        case 1: adminOptions(); break;
        case 2: userOptions(); break;
        case 3: closeApp(); break;
        default: printf("* Invalid choice! Please try again.\n"); break;
        }
    } while (choice != 3);
}

int main() {
    mainMenu();
    return 0;
}
```

TEST RESULT:

```
E-COMERCE STORE
1. Admin
2. User
3. Exit
Enter your choice: 1
```

```
Admin Panel
1. Add Product
2. Display Products
3. Search Product
4. Delete Product
5. Update Product
6. Close Application
7. Sale Product
Enter your choice: 1
Enter "end" to exit

Enter Product Code: 1
Enter Product Name: pen
Enter Product Rate: 25
Enter Quantity: 5
Enter classification : tech
Enter Product Description: good

Admin Panel
1. Add Product
2. Display Products
3. Search Product
4. Delete Product
5. Update Product
6. Close Application
7. Sale Product
Enter your choice: 1
Enter "end" to exit

Enter Product Code: 2
Enter Product Name: apple
Enter Product Rate: 15
Enter Quantity: 10
Enter classification : food
Enter Product Description: better
```

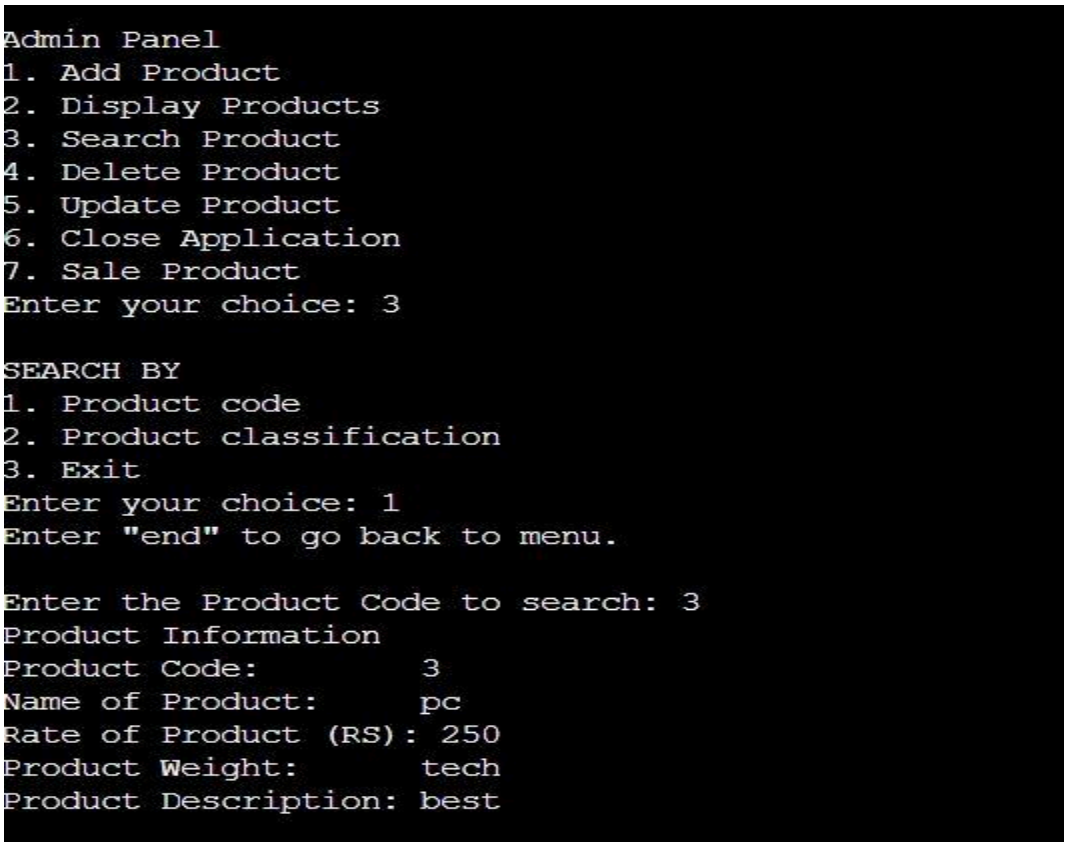
```
Admin Panel
1. Add Product
2. Display Products
3. Search Product
4. Delete Product
5. Update Product
6. Close Application
7. Sale Product
Enter your choice: 2
-----------------------------------------------------------------------------------------------------
CODE    ||      NAME    ||      RATE    ||      QUANTITY        ||      WEIGHT  ||      DESCRIPTION
-----------------------------------------------------------------------------------------------------
1       ||      pen     ||      25      ||      5               ||      tech    ||      good
2       ||      apple   ||      15      ||      10              ||      food    ||      better
3       ||      pc      ||      250     ||      20              ||      tech    ||      best
4       ||      banana  ||      10      ||      6               ||      food    ||      good
-----------------------------------------------------------------------------------------------------
```

```
Admin Panel
1. Add Product
2. Display Products
3. Search Product
4. Delete Product
5. Update Product
6. Close Application
7. Sale Product
Enter your choice: 1
Enter "end" to exit

Enter Product Code: 3
Enter Product Name: pc
Enter Product Rate: 250
Enter Quantity: 20
Enter classification : tech
Enter Product Description: best

Admin Panel
1. Add Product
2. Display Products
3. Search Product
4. Delete Product
5. Update Product
6. Close Application
7. Sale Product
Enter your choice: 1
Enter "end" to exit

Enter Product Code: 4
Enter Product Name: banana
Enter Product Rate: 10
Enter Quantity: 6
Enter classification : food
Enter Product Description: good
```


Admin Panel
1. Add Product
2. Display Products
3. Search Product
4. Delete Product
5. Update Product
6. Close Application
7. Sale Product
Enter your choice: 3
SEARCH BY
1. Product code
2. Product classification
3. Exit
Enter your choice: 1
Enter "end" to go back to menu.
Enter the Product Code to search: 3
Product Information
Product Code: 3
Name of Product: pc
Rate of Product (RS): 250
Product Weight: tech
Product Description: best

```
SEARCH BY
1. Product code
2. Product classification
3. Exit
Enter your choice: 2
Enter "end" to go back to menu.

Enter the Product classification to search: food
Product Information
-------------------------------------------------------------------------------------------------------
CODE    ||      NAME    ||      RATE    ||      QUANTITY        ||      CLASS   ||      DESCRIPTION
-------------------------------------------------------------------------------------------------------
2       ||      apple   ||      15      ||      10              ||      food    ||      better
4       ||      banana  ||      10      ||      6               ||      food    ||      good
-------------------------------------------------------------------------------------------------------

SEARCH BY
1. Product code
2. Product classification
3. Exit
Enter your choice: 2
Enter "end" to go back to menu.

Enter the Product classification to search: tech
Product Information
-------------------------------------------------------------------------------------------------------
CODE    ||      NAME    ||      RATE    ||      QUANTITY        ||      CLASS   ||      DESCRIPTION
-------------------------------------------------------------------------------------------------------
1       ||      pen     ||      25      ||      5               ||      tech    ||      good
3       ||      pc      ||      250     ||      20              ||      tech    ||      best
-------------------------------------------------------------------------------------------------------

SEARCH BY
1. Product code
2. Product classification
3. Exit
Enter your choice:
```