Inventory Management System

OBJECTIVE:

Create a C program to manage product inventory allowing add, update, delete, search, and stock operations using file handling.

OVERVIEW:

The Product Inventory Management System is a menu-driven C application used to manage product records. It uses structures to store product ID, name, quantity, and price, along with file handling to maintain persistent storage. The system allows users to add, view, update, and delete product details, as well as generate simple inventory reports. This project helps students practice CRUD operations, validation, and modular programming concepts.

PROGRAM:

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

#define MIN_STOCK 5
#define LINE 256

struct Product {
   int productId;
   char name[100];
   char category[50];
   int quantity;
   float price;
};

/* Function declarations */
void login();
void addProduct();
void displayAll();
void searchProduct();
void updateProduct();
void deleteProduct();
void stockIn();
void stockOut();
void inventoryReport();

/* ---------------- LOGIN ---------------- */
void login() {
   char u[20], p[20];
   do {
      printf("\n--- LOGIN ---\n");
      printf("Username: ");
      scanf("%19s", u);
      printf("Password: ");
      scanf("%19s", p);
      if (strcmp(u, "vcet") || strcmp(p, "1234"))
         printf("Invalid login!\n");
```

```
    } while (strcmp(u, "vcet") || strcmp(p, "1234"));
    printf("Login successful!\n");
}

/* ---------------- MAIN ---------------- */
int main() {
    int ch;
    login();
    do {
        printf("\n1.Add\n2.Display\n3.Search\n4.Update\n5.Delete\n");
        printf("6.Stock In\n7.Stock Out\n8.Report\n0.Exit\n");
        printf("Choice: ");
        scanf("%d", &ch);

        switch (ch) {
            case 1: addProduct(); break;
            case 2: displayAll(); break;
            case 3: searchProduct(); break;
            case 4: updateProduct(); break;
            case 5: deleteProduct(); break;
            case 6: stockIn(); break;
            case 7: stockOut(); break;
            case 8: inventoryReport(); break;
        }
    } while (ch != 0);
    return 0;
}

/* ---------------- ADD ---------------- */
void addProduct() {
    FILE *fp = fopen("products.txt", "a");
    struct Product p;

    printf("ID: "); scanf("%d", &p.productId);
    printf("Name: "); scanf(" %99[^\n]", p.name);
    printf("Category: "); scanf(" %49[^\n]", p.category);
    printf("Qty: "); scanf("%d", &p.quantity);
    printf("Price: "); scanf("%f", &p.price);

    fprintf(fp, "%d|%s|%s|%d|%.2f\n",
            p.productId, p.name, p.category,
            p.quantity, p.price);
    fclose(fp);
    printf("Added successfully!\n");
}

/* ---------------- DISPLAY (FIXED) ---------------- */
void displayAll() {
    FILE *fp = fopen("products.txt", "r");
    char line[LINE];
    struct Product p;
    int printed = 0;

    if (!fp) {
        printf("No products found.\n");
```

```
    return;
  }

  printf("\nID  Name      Category    Qty  Price  Status\n");
  printf("-------------------------------------------------\n");

  while (fgets(line, sizeof(line), fp)) {
    if (sscanf(line, "%d|%99[^|]|%49[^|]|%d|%f",
          &p.productId, p.name,
          p.category, &p.quantity,
          &p.price) == 5) {

      printf("%-4d %-11s %-12s %-4d %-6.2f ",
          p.productId, p.name,
          p.category, p.quantity, p.price);

      if (p.quantity < MIN_STOCK)
        printf("LOW STOCK");
      printf("\n");
      printed = 1;
    }
  }

  if (!printed)
    printf("No valid product records.\n");

  fclose(fp);
}

/* ---------------- SEARCH ---------------- */
void searchProduct() {
  FILE *fp = fopen("products.txt", "r");
  char line[LINE];
  struct Product p;
  int id, found = 0;

  printf("Enter ID: ");
  scanf("%d", &id);

  while (fgets(line, LINE, fp)) {
    if (sscanf(line, "%d|%99[^|]|%49[^|]|%d|%f",
          &p.productId, p.name,
          p.category, &p.quantity,
          &p.price) == 5 &&
      p.productId == id) {

      printf("Found: %s (%s)\n", p.name, p.category);
      found = 1;
      break;
    }
  }

  if (!found) printf("Not found.\n");
  fclose(fp);
}
```

```
/* ---------------- UPDATE ---------------- */
void updateProduct() {
   FILE *fp = fopen("products.txt", "r");
   FILE *tp = fopen("temp.txt", "w");
   char line[LINE];
   struct Product p;
   int id, found = 0;

   printf("ID to update: ");
   scanf("%d", &id);

   while (fgets(line, LINE, fp)) {
      if (sscanf(line, "%d|%99[^|]|%49[^|]|%d|%f",
            &p.productId, p.name,
            p.category, &p.quantity,
            &p.price) == 5) {

         if (p.productId == id) {
            printf("New Name: ");
            scanf(" %99[^\n]", p.name);
            printf("New Category: ");
            scanf(" %49[^\n]", p.category);
            printf("New Qty: ");
            scanf("%d", &p.quantity);
            printf("New Price: ");
            scanf("%f", &p.price);
            found = 1;
         }

         fprintf(tp, "%d|%s|%s|%d|%.2f\n",
               p.productId, p.name,
               p.category, p.quantity,
               p.price);
      }
   }

   fclose(fp); fclose(tp);
   remove("products.txt");
   rename("temp.txt", "products.txt");

   printf(found ? "Updated!\n" : "Not found.\n");
}

/* ---------------- DELETE ---------------- */
void deleteProduct() {
   FILE *fp = fopen("products.txt", "r");
   FILE *tp = fopen("temp.txt", "w");
   char line[LINE];
   struct Product p;
   int id, found = 0;

   printf("ID to delete: ");
   scanf("%d", &id);
```

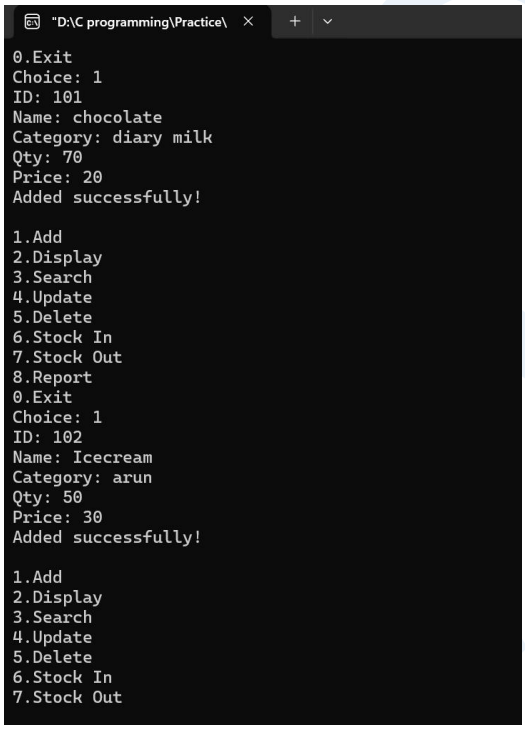
```
    while (fgets(line, LINE, fp)) {
        if (sscanf(line, "%d|%99[^|]|%49[^|]|%d|%f",
                   &p.productId, p.name,
                   p.category, &p.quantity,
                   &p.price) == 5 &&
            p.productId != id) {

            fprintf(tp, "%d|%s|%s|%d|%.2f\n",
                    p.productId, p.name,
                    p.category, p.quantity,
                    p.price);
        } else found = 1;
    }

    fclose(fp); fclose(tp);
    remove("products.txt");
    rename("temp.txt", "products.txt");

    printf(found ? "Deleted!\n" : "Not found.\n");
}

/* ---------------- STOCK IN / OUT ---------------- */
void stockIn() { updateProduct(); }
void stockOut() { updateProduct(); }

/* ---------------- REPORT ---------------- */
void inventoryReport() {
    FILE *fp = fopen("products.txt", "r");
    char line[LINE];
    struct Product p;
    float total = 0;

    while (fgets(line, LINE, fp))
        if (sscanf(line, "%d|%99[^|]|%49[^|]|%d|%f",
                   &p.productId, p.name,
                   p.category, &p.quantity,
                   &p.price) == 5)
            total += p.quantity * p.price;

    fclose(fp);
    printf("Total Inventory Value: %.2f\n", total);
}
```

Enhancements Implemented

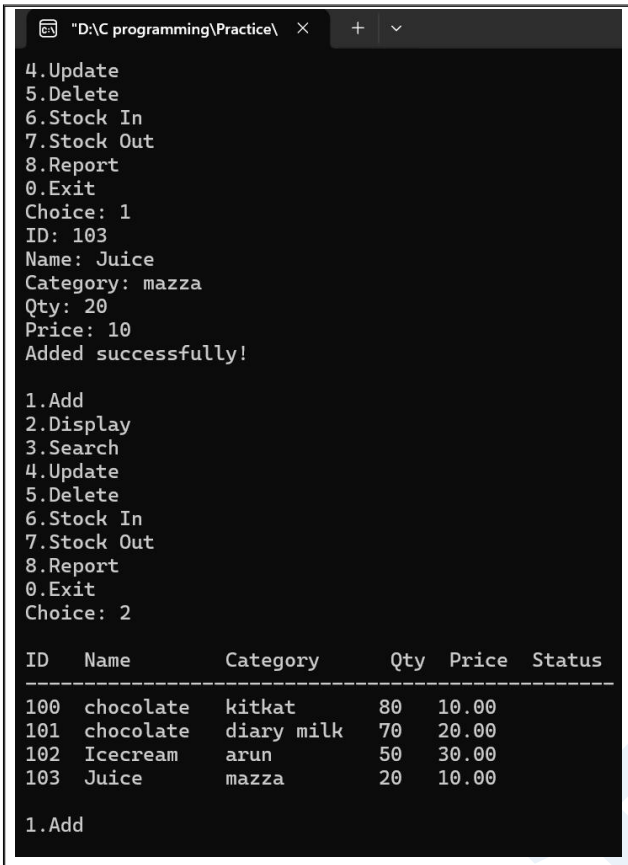
✔ Login authentication
✔ Product category classification
✔ Low stock alert (quantity < 5)
✔ Secure CRUD using temp file
✔ Modular programming

OUTPUT:

```
"D:\C programming\Practice\

--- LOGIN ---
Username: vcet
Password: 1234
Login successful!

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 1
ID: 100
Name: chocolate
Category: kitkat
Qty: 80
Price: 10
Added successfully!

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 1
```

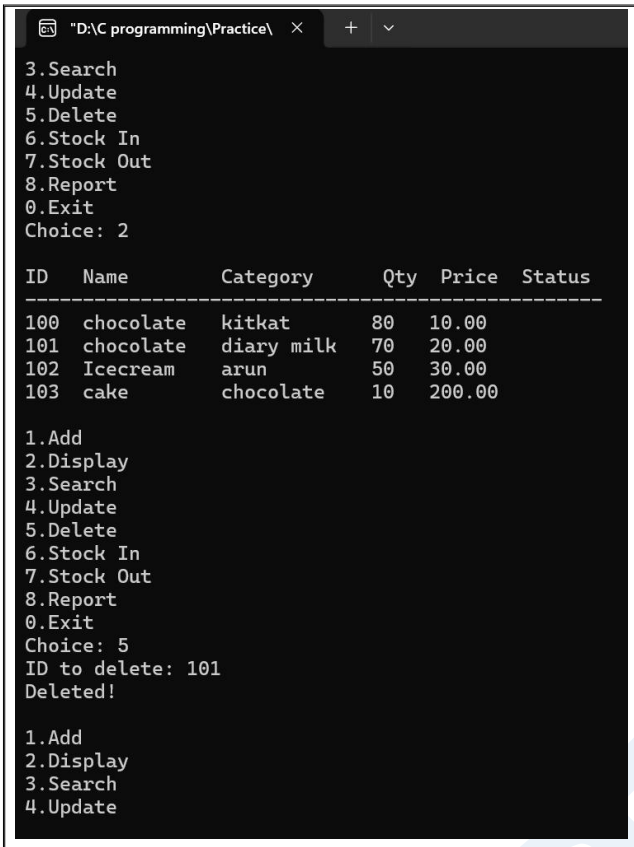


```
"D:\C programming\Practice\

0.Exit
Choice: 1
ID: 101
Name: chocolate
Category: diary milk
Qty: 70
Price: 20
Added successfully!

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 1
ID: 102
Name: Icecream
Category: arun
Qty: 50
Price: 30
Added successfully!

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
```

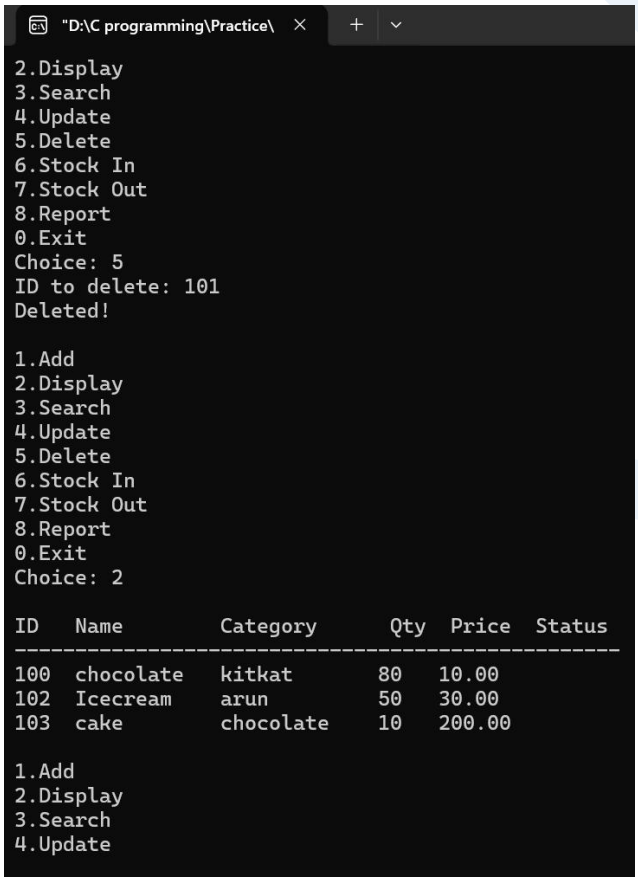


```
"D:\C programming\Practice\
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 1
ID: 103
Name: Juice
Category: mazza
Qty: 20
Price: 10
Added successfully!

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 2

ID   Name        Category      Qty  Price  Status
-------------------------------------------------
100  chocolate   kitkat        80   10.00
101  chocolate   diary milk    70   20.00
102  Icecream    arun          50   30.00
103  Juice       mazza         20   10.00

1.Add
```

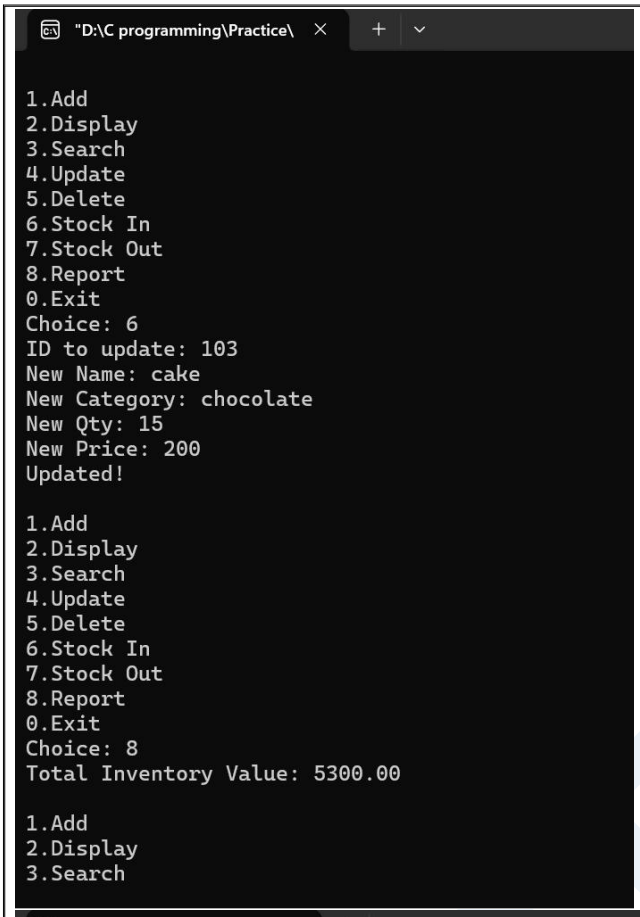
```
"D:\C programming\Practice\
1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 3
Enter ID: 102
Found: Icecream (arun)

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 4
ID to update: 103
New Name: cake
New Category: chocolate
New Qty: 10
New Price: 200
Updated!

1.Add
2.Display
3.Search
```



```
"D:\C programming\Practice\
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 2

ID   Name        Category      Qty  Price  Status
-------------------------------------------------
100  chocolate   kitkat        80   10.00
101  chocolate   diary milk    70   20.00
102  Icecream    arun          50   30.00
103  cake        chocolate     10   200.00

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 5
ID to delete: 101
Deleted!

1.Add
2.Display
3.Search
4.Update
```



```
"D:\C programming\Practice\
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 5
ID to delete: 101
Deleted!

1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 2

ID   Name        Category      Qty  Price  Status
-------------------------------------------------
100  chocolate   kitkat        80   10.00
102  Icecream    arun          50   30.00
103  cake        chocolate     10   200.00

1.Add
2.Display
3.Search
4.Update
```


"D:\C programming\Practice\
1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 6
ID to update: 103
New Name: cake
New Category: chocolate
New Qty: 15
New Price: 200
Updated!
1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 8
Total Inventory Value: 5300.00
1.Add
2.Display
3.Search


"D:\C programming\Practice\
ID to update: 103
New Name: cake
New Category: chocolate
New Qty: 15
New Price: 200
Updated!
1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 8
Total Inventory Value: 5300.00
1.Add
2.Display
3.Search
4.Update
5.Delete
6.Stock In
7.Stock Out
8.Report
0.Exit
Choice: 0
Process returned 0 (0x0) execution time : 654.865 s
Press any key to continue.