**Project 1 – last version**

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
  
#define MAX\_ITEMS 100  
#define MAX\_NAME\_LEN 100  
#define MAX\_CATEGORY\_LEN 100  
#define FILENAME "inventory.csv"  
  
struct InventoryItem {  
 char name[50];  
 char category[50];  
 int quantity;  
 double price;  
};  
  
void addItem(struct InventoryItem \*inventory, int \*count);  
void updateItem(struct InventoryItem \*inventory, int count);  
void deleteItem(struct InventoryItem \*inventory, int \*count);  
void searchByName(struct InventoryItem inventory[], int count);  
void searchByCategory(struct InventoryItem inventory[], int count);  
void displayAllItems(struct InventoryItem \*inventory, int count);  
void displayLowQuantityItems(struct InventoryItem \*inventory, int count);  
void saveDataToFile(struct InventoryItem \*inventory, int count);  
void retrieveDataFromFile(struct InventoryItem \*inventory, int \*count);  
void displayItem(struct InventoryItem item);  
  
int main() {  
 struct InventoryItem inventory[MAX\_ITEMS];  
 int count = 0, choice;  
  
 retrieveDataFromFile(inventory, &count);  
  
 printf("\n====================== Inventory Mangement System ======================");  
 printf(" ");  
 printf(" ");  
 do {  
 printf("\n\n");  
 printf("1. Add item\n");  
 printf("2. Update item\n");  
 printf("3. Delete item\n");  
 printf("4. Search by name\n");  
 printf("5. Search by category\n");  
 printf("6. Display all items\n");  
 printf("7. Display low quantity items\n");  
 printf("8. Save data to file\n");  
 printf("0. Exit\n");  
 printf("Enter choice: ");  
 scanf("%d", &choice);  
  
 switch (choice) {  
 case 1:  
 addItem(inventory, &count);  
 break;  
 case 2:  
 updateItem(inventory, count);  
 break;  
 case 3:  
 deleteItem(inventory, &count);  
 break;  
 case 4:  
 searchByName(inventory, count);  
 break;  
 case 5:  
 searchByCategory(inventory, count);  
 break;  
 case 6:  
 displayAllItems(inventory, count);  
 break;  
 case 7:  
 displayLowQuantityItems(inventory, count);  
 break;  
 case 8:  
 saveDataToFile(inventory, count);  
 break;  
 case 0:  
 printf("Exiting program.\n");  
 break;  
 default:  
 printf("Invalid choice. Please try again.\n");  
 break;  
 }  
 } while (choice != 0);  
  
 return 0;  
}  
  
void addItem(struct InventoryItem \*inventory, int \*count) {  
 if (\*count == MAX\_ITEMS) {  
 printf("Inventory is full. Cannot add more items.\n");  
 return;  
 }  
  
 printf("Enter item name: ");  
 scanf("%s", inventory[\*count].name);  
 printf("Enter item category: ");  
 scanf("%s", inventory[\*count].category);  
 printf("Enter item quantity: ");  
 scanf("%d", &inventory[\*count].quantity);  
 printf("Enter item price: ");  
 scanf("%lf", &inventory[\*count].price);  
  
// saveDataToFile(inventory, \*count);  
  
 (\*count)++;  
}  
  
void updateItem(struct InventoryItem \*inventory, int count) {  
 char itemName[50];  
 int i, found = 0;  
  
 printf("Enter name of item to update: ");  
 scanf("%s", itemName);  
  
 for (i = 0; i < count; i++) {  
 if (strcmp(inventory[i].name, itemName) == 0) {  
 printf("Enter new quantity: ");  
 scanf("%d", &inventory[i].quantity);  
 printf("Enter new price: ");  
 scanf("%d", &inventory[i].price);  
 found = 1;  
 break;  
 }  
 }  
  
 if (!found) {  
 printf("Item not found.\n");  
 }  
}  
  
void deleteItem(struct InventoryItem \*inventory, int \*count) {  
 char itemName[50];  
 int i, j, found = 0;  
  
 printf("Enter name of item to delete: ");  
 scanf("%s", itemName);  
  
 for (i = 0; i < \*count; i++) {  
 if (strcmp(inventory[i].name, itemName) == 0) {  
 found = 1;  
 break;  
 }  
 }  
  
 if (!found) {  
 printf("Item not found.\n");  
 return;  
 }  
  
 for (j = i; j < \*count - 1; j++) {  
 inventory[j] = inventory[j+1];  
 }  
  
 (\*count)--;  
 printf("Item deleted.\n");  
}  
  
void searchByName(struct InventoryItem inventory[], int count) {  
 char name[MAX\_NAME\_LEN];  
 int found = 0;  
  
 printf("Enter item name: ");  
 scanf(" %[^\n]", name);  
  
 printf("Search results:\n");  
 for (int i = 0; i < count; i++) {  
 if (strcmp(inventory[i].name, name) == 0) {  
 displayItem(inventory[i]);  
 found = 1;  
 }  
 }  
  
 if (!found) {  
 printf("No matching items found.\n");  
 }  
}  
  
void searchByCategory(struct InventoryItem inventory[], int count) {  
 char category[MAX\_CATEGORY\_LEN];  
 int found = 0;  
  
 printf("Enter category: ");  
 scanf(" %[^\n]", category);  
  
 printf("Search results:\n");  
 for (int i = 0; i < count; i++) {  
 if (strcmp(inventory[i].category, category) == 0) {  
 displayItem(inventory[i]);  
 found = 1;  
 }  
 }  
  
 if (!found) {  
 printf("No matching items found.\n");  
 }  
}  
  
void displayAllItems(struct InventoryItem \*inventory, int count) {  
 int i;  
  
 printf("%-20s %-20s %-20s %s\n", "Name", "Category", "Quantity", "Price");  
 printf("---------------------------------------------------------------------\n");  
 for (i = 0; i < count; i++) {  
 printf("%-20s %-20s %-20d %.2lf\n", inventory[i].name, inventory[i].category, inventory[i].quantity, inventory[i].price);  
 }  
}  
  
void displayLowQuantityItems(struct InventoryItem \*inventory, int count) {  
 int i, threshold;  
  
 printf("Enter quantity threshold: ");  
 scanf("%d", &threshold);  
  
 printf("%-20s %-20s %-20s %s\n", "Name", "Category", "Quantity", "Price");  
 printf("------------------------------------------------------------------\n");  
 for (i = 0; i < count; i++) {  
 if (inventory[i].quantity < threshold) {  
 printf("%-20s %-20s %-20d %.2.lf\n", inventory[i].name, inventory[i].category, inventory[i].quantity, inventory[i].price);  
 }  
 }  
}  
  
void saveDataToFile(struct InventoryItem \*inventory, int count) {  
 FILE \*fp;  
 int i;  
  
 fp = fopen(FILENAME, "w");  
 if (fp == NULL) {  
 printf("Error opening file for writing.\n");  
 return;  
 }  
  
 for (i = 0; i < count; i++) {  
 fprintf(fp, "%s,%s,%d,%.2lf\n", inventory[i].name, inventory[i].category, inventory[i].quantity, inventory[i].price);  
 }  
  
 fclose(fp);  
 printf("Data saved to file.\n");  
}  
  
void retrieveDataFromFile(struct InventoryItem \*inventory, int \*count) {  
 FILE \*fp;  
 char buffer[100], \*token;  
  
 fp = fopen(FILENAME, "r");  
 if (fp == NULL) {  
 printf("File does not exist. Starting with empty inventory.\n");  
 return;  
 }  
  
 \*count = 0;  
 while (fgets(buffer, 100, fp) != NULL) {  
 token = strtok(buffer, ",");  
 strcpy(inventory[\*count].name, token);  
  
 token = strtok(NULL, ",");  
 strcpy(inventory[\*count].category, token);  
  
 token = strtok(NULL, ",");  
 inventory[\*count].quantity = atoi(token);  
  
 token = strtok(NULL, ",");  
 inventory[\*count].price = atoi(token);  
  
 (\*count)++;  
 }  
  
 fclose(fp);  
 printf("Data retrieved from file.\n");  
}  
  
void displayItem(struct InventoryItem item) {  
 printf("Name: %s\n", item.name);  
 printf("Category: %s\n", item.category);  
 printf("Quantity: %d\n", item.quantity);  
 printf("Price: %.2lf\n", item.price);  
 printf("-----------------------------\n");  
}