#include #include #include #include "Stock.h"

using namespace std;

const char\* ARCHIVO\_STOCK = "stock.dat";

Stock::Stock() { \_idProducto = 0; strcpy(\_nombre, ""); strcpy(\_categoria, ""); strcpy(\_unidadMedida, ""); \_precioCosto = 0; \_precioVenta = 0; \_stockActual = 0; \_stockMinimo = 0; \_eliminado = false; }

Stock::Stock(int id, const char\* nombre, const char\* categoria, const char\* unidadMedida, float precioCosto, float precioVenta, int stockActual, int stockMinimo) { \_idProducto = id; strcpy(\_nombre, nombre); strcpy(\_categoria, categoria); strcpy(\_unidadMedida, unidadMedida); \_precioCosto = precioCosto; \_precioVenta = precioVenta; \_stockActual = stockActual; \_stockMinimo = stockMinimo; \_eliminado = false; }

// Getters int Stock::getIdProducto() { return \_idProducto; } const char\* Stock::getNombre() { return \_nombre; } const char\* Stock::getCategoria() { return \_categoria; } const char\* Stock::getUnidadMedida() { return \_unidadMedida; } float Stock::getPrecioCosto() { return \_precioCosto; } float Stock::getPrecioVenta() { return \_precioVenta; } int Stock::getStockActual() { return \_stockActual; } int Stock::getStockMinimo() { return \_stockMinimo; } bool Stock::getEliminado() { return \_eliminado; }

// Setters void Stock::setIdProducto(int id) { \_idProducto = id; } void Stock::setNombre(const char\* nombre) { strcpy(\_nombre, nombre); } void Stock::setCategoria(const char\* categoria) { strcpy(\_categoria, categoria); } void Stock::setUnidadMedida(const char\* unidadMedida) { strcpy(\_unidadMedida, unidadMedida); } void Stock::setPrecioCosto(float precio) { \_precioCosto = precio; } void Stock::setPrecioVenta(float precio) { \_precioVenta = precio; } void Stock::setStockActual(int stock) { \_stockActual = stock; } void Stock::setStockMinimo(int stockMinimo) { \_stockMinimo = stockMinimo; } void Stock::setEliminado(bool eliminado) { \_eliminado = eliminado; }

void Stock::mostrar() { cout << "ID: " << \_idProducto << endl; cout << "Nombre: " << \_nombre << endl; cout << "Categoria: " << \_categoria << endl; cout << "Unidad: " << \_unidadMedida << endl; cout << "Precio Costo: $" << \_precioCosto << endl; cout << "Precio Venta: $" << \_precioVenta << endl; cout << "Stock Actual: " << \_stockActual << endl; cout << "Stock Minimo: " << \_stockMinimo << endl; cout << "-----------------------------" << endl; }

// Funciones para manejar el archivo bool guardarStock(Stock producto) { FILE\* p = fopen(ARCHIVO\_STOCK, "ab"); if (p == NULL) return false;

bool ok = fwrite(&producto, sizeof(Stock), 1, p);  
fclose(p);  
return ok;

}

bool guardarStock(Stock producto, int posicion) { FILE\* p = fopen(ARCHIVO\_STOCK, "rb+"); if (p == NULL) return false;

fseek(p, sizeof(Stock) \* posicion, SEEK\_SET);  
bool ok = fwrite(&producto, sizeof(Stock), 1, p);  
fclose(p);  
return ok;

}

int buscarStock(int idProducto) { FILE\* p = fopen(ARCHIVO\_STOCK, "rb"); if (p == NULL) return -1;

Stock producto;  
int pos = 0;  
while (fread(&producto, sizeof(Stock), 1, p)) {  
 if (producto.getIdProducto() == idProducto && !producto.getEliminado()) {  
 fclose(p);  
 return pos;  
 }  
 pos++;  
}  
fclose(p);  
return -1;

}

Stock leerStock(int posicion) { Stock producto; FILE\* p = fopen(ARCHIVO\_STOCK, "rb"); if (p == NULL) return producto;

fseek(p, sizeof(Stock) \* posicion, SEEK\_SET);  
fread(&producto, sizeof(Stock), 1, p);  
fclose(p);  
return producto;

}

int cantidadRegistrosStock() { FILE\* p = fopen(ARCHIVO\_STOCK, "rb"); if (p == NULL) return 0;

fseek(p, 0, SEEK\_END);  
int bytes = ftell(p);  
fclose(p);  
return bytes / sizeof(Stock);

}

bool eliminarStock(int posicion) { Stock producto = leerStock(posicion); if (producto.getIdProducto() == 0) return false;

producto.setEliminado(true);  
return guardarStock(producto, posicion);

}

void listarStocks() { int total = cantidadRegistrosStock(); cout << "LISTADO DE PRODUCTOS" << endl; cout << "==================================================" << endl;

for (int i = 0; i < total; i++) {  
 Stock producto = leerStock(i);  
 if (!producto.getEliminado()) {  
 producto.mostrar();  
 }  
}

}

void consultarStock(int idProducto) { int pos = buscarStock(idProducto); if (pos == -1) { cout << "Producto no encontrado!" << endl; return; }

Stock producto = leerStock(pos);////////////////////////  
cout << "STOCK DEL PRODUCTO:" << endl;  
cout << "Nombre: " << producto.getNombre();  
cout << "Stock Actual: " << producto.getStockActual() << endl;  
cout << "Stock Minimo: " << producto.getStockMinimo() << endl;  
  
if (producto.getStockActual() < producto.getStockMinimo()) {  
 cout << "ALERTA: Stock por debajo del minimo!" << endl;  
}

}

void alertaStockMinimo() { int total = cantidadRegistrosStock(); bool hayAlertas = false;

cout << "ALERTAS DE STOCK MINIMO" << endl;  
cout << "==================================================" << endl;  
  
for (int i = 0; i < total; i++) {  
 Stock producto = leerStock(i);  
 if (!producto.getEliminado() && producto.getStockActual() < producto.getStockMinimo()) {  
 cout << "ALERTA: " << producto.getNombre() << " - Stock: " << producto.getStockActual()  
 << " (Minimo: " << producto.getStockMinimo() << ")" << endl;  
 hayAlertas = true;  
 }  
}  
  
if (!hayAlertas) {  
 cout << "No hay productos con stock por debajo del minimo." << endl;  
}

}

int generarIdStock() { int total = cantidadRegistrosStock(); if (total == 0)

return 1;  
  
Stock ultimo = leerStock(total - 1);  
return ultimo.getIdProducto() + 1;

}

// Funciones del menú void agregarProducto() { system("cls"); cout << "AGREGAR PRODUCTO" << endl; cout << "===============" << endl;

int id = generarIdStock();  
char nombre[40], categoria[20], unidadMedida[10];  
float precioCosto, precioVenta;  
int stockActual, stockMinimo;  
  
cout << "ID generado: " << id << endl;  
cout << "Nombre: ";  
cin.ignore();  
cin.getline(nombre, 40);  
cout << "Categoria: ";  
cin.getline(categoria, 20);  
cout << "Unidad de medida: ";  
cin.getline(unidadMedida, 10);  
cout << "Precio costo: ";  
cin >> precioCosto;  
cout << "Precio venta: ";  
cin >> precioVenta;  
cout << "Stock actual: ";  
cin >> stockActual;  
cout << "Stock minimo: ";  
cin >> stockMinimo;  
  
Stock producto(id, nombre, categoria, unidadMedida, precioCosto, precioVenta, stockActual, stockMinimo);  
  
if (guardarStock(producto)) {  
 cout << "Producto agregado correctamente!" << endl;  
} else {  
 cout << "Error al agregar producto!" << endl;  
}  
  
system("pause");

}

void modificarProducto() { system("cls"); cout << "MODIFICAR PRODUCTO" << endl; cout << "=================" << endl;

int id;  
cout << "ID del producto a modificar: ";  
cin >> id;  
  
int pos = buscarStock(id);  
if (pos == -1) {  
 cout << "Producto no encontrado!" << endl;  
 system("pause");  
 return;  
}  
  
Stock producto = leerStock(pos);  
producto.mostrar();  
  
char opcion;  
cout << "Desea modificar este producto? (s/n): ";  
cin >> opcion;  
  
if (opcion == 's' || opcion == 'S') {  
 char nombre[40], categoria[20], unidadMedida[10];  
 float precioCosto, precioVenta;  
 int stockActual, stockMinimo;  
  
 cout << "Nuevo nombre (" << producto.getNombre() << "): ";  
 cin.ignore();  
 cin.getline(nombre, 40);  
 if (strlen(nombre) > 0) producto.setNombre(nombre);  
  
 cout << "Nueva categoria (" << producto.getCategoria() << "): ";  
 cin.getline(categoria, 20);  
 if (strlen(categoria) > 0) producto.setCategoria(categoria);  
  
 cout << "Nueva unidad (" << producto.getUnidadMedida() << "): ";  
 cin.getline(unidadMedida, 10);  
 if (strlen(unidadMedida) > 0) producto.setUnidadMedida(unidadMedida);  
  
 cout << "Nuevo precio costo (" << producto.getPrecioCosto() << "): ";  
 cin >> precioCosto;  
 producto.setPrecioCosto(precioCosto);  
  
 cout << "Nuevo precio venta (" << producto.getPrecioVenta() << "): ";  
 cin >> precioVenta;  
 producto.setPrecioVenta(precioVenta);  
  
 cout << "Nuevo stock actual (" << producto.getStockActual() << "): ";  
 cin >> stockActual;  
 producto.setStockActual(stockActual);  
  
 cout << "Nuevo stock minimo (" << producto.getStockMinimo() << "): ";  
 cin >> stockMinimo;  
 producto.setStockMinimo(stockMinimo);  
  
 if (guardarStock(producto, pos)) {  
 cout << "Producto modificado correctamente!" << endl;  
 } else {  
 cout << "Error al modificar producto!" << endl;  
 }  
}  
  
system("pause");

}

void eliminarProducto() { system("cls"); cout << "ELIMINAR PRODUCTO" << endl; cout << "================" << endl;

int id;  
cout << "ID del producto a eliminar: ";  
cin >> id;  
  
int pos = buscarStock(id);  
if (pos == -1) {  
 cout << "Producto no encontrado!" << endl;  
 system("pause");  
 return;  
}  
  
Stock producto = leerStock(pos);  
producto.mostrar();  
  
char opcion;  
cout << "Esta seguro de eliminar este producto? (s/n): ";  
cin >> opcion;  
  
if (opcion == 's' || opcion == 'S') {  
 if (eliminarStock(pos)) {  
 cout << "Producto eliminado correctamente!" << endl;  
 } else {  
 cout << "Error al eliminar producto!" << endl;  
 }  
}  
  
system("pause");

}

// NUEVO: Submenú de consultas void menuConsultas() { int opcion; do { system("cls"); cout << "CONSULTAS DE STOCK" << endl; cout << "==================" << endl; cout << "1 - CONSULTAR POR PRODUCTO (ID)" << endl; cout << "2 - CONSULTAR POR CATEGORIA" << endl; cout << "3 - CONSULTAR POR PRECIO" << endl; cout << "4 - CONSULTAR POR NIVEL DE STOCK" << endl; cout << "0 - VOLVER AL MENU STOCK" << endl; cout << "==================" << endl; cout << "OPCION: "; cin >> opcion;

switch (opcion) {  
 case 1:  
 system("cls");  
 int id;  
 cout << "ID del producto: ";  
 cin >> id;  
 consultarStock(id);  
 system("pause");  
 break;  
 case 2:  
 system("cls");  
 consultarPorCategoria();  
 system("pause");  
 break;  
 case 3:  
 system("cls");  
 consultarPorPrecio();  
 system("pause");  
 break;  
 case 4:  
 system("cls");  
 consultarPorNivelStock();  
 system("pause");  
 break;  
 case 0:  
 cout << "Volviendo al menu stock..." << endl;  
 break;  
 default:  
 cout << "Opcion invalida!" << endl;  
 system("pause");  
 break;  
 }  
} while (opcion != 0);

}

// Función mostrarMenuStock void mostrarMenuStock() { system("cls"); cout << "STOCK" << endl; cout << "---" << endl; cout << "1 - AGREGAR PRODUCTO" << endl; cout << "2 - MODIFICAR PRODUCTO" << endl; cout << "3 - ELIMINAR PRODUCTO" << endl; cout << "4 - LISTAR PRODUCTOS" << endl; cout << "5 - CONSULTAS DE STOCK" << endl; // Cambiado cout << "6 - ALERTA DE STOCK MINIMO" << endl; cout << "---" << endl; cout << "0 - VOLVER" << endl; cout << "---" << endl; cout << "OPCION: "; }

// Función menuStock void menuStock() { int opcion; do { mostrarMenuStock(); cin >> opcion;

switch (opcion) {  
 case 1: agregarProducto(); break;  
 case 2: modificarProducto(); break;  
 case 3: eliminarProducto(); break;  
 case 4: system("cls"); listarStocks(); system("pause"); break;  
 case 5: menuConsultas(); break; // Ahora llama al submenú de consultas  
 case 6: system("cls"); alertaStockMinimo(); system("pause"); break;  
 case 0: cout << "Volviendo al menu principal..." << endl; break;  
 default: cout << "Opcion invalida!" << endl; system("pause"); break;  
 }  
} while (opcion != 0);

}

void consultarPorCategoria() { system("cls"); cout << "CONSULTA POR CATEGORIA" << endl; cout << "=====================" << endl;

char categoria[20];  
cout << "Ingrese la categoria a buscar: ";  
cin.ignore();  
cin.getline(categoria, 20);  
  
int total = cantidadRegistrosStock();  
bool encontrado = false;  
  
cout << "PRODUCTOS DE LA CATEGORIA: " << categoria << endl;  
cout << "==================================================" << endl;  
  
for (int i = 0; i < total; i++) {  
 Stock producto = leerStock(i);  
 if (!producto.getEliminado() && strcmp(producto.getCategoria(), categoria) == 0) {  
 producto.mostrar();  
 encontrado = true;  
 }  
}  
  
if (!encontrado) {  
 cout << "No se encontraron productos en esta categoria." << endl;  
}

}

void consultarPorPrecio() { system("cls"); cout << "CONSULTA POR RANGO DE PRECIO" << endl; cout << "============================" << endl;

float precioMin, precioMax;  
cout << "Precio minimo: $";  
cin >> precioMin;  
cout << "Precio maximo: $";  
cin >> precioMax;  
  
if (precioMin > precioMax) {  
 cout << "Error: El precio minimo no puede ser mayor al maximo." << endl;  
 system("pause");  
 return;  
}  
  
int total = cantidadRegistrosStock();  
bool encontrado = false;  
  
cout << "PRODUCTOS ENTRE $" << precioMin << " Y $" << precioMax << endl;  
cout << "==================================================" << endl;  
  
for (int i = 0; i < total; i++) {  
 Stock producto = leerStock(i);  
 if (!producto.getEliminado() &&  
 producto.getPrecioVenta() >= precioMin &&  
 producto.getPrecioVenta() <= precioMax) {  
 producto.mostrar();  
 encontrado = true;  
 }  
}  
  
if (!encontrado) {  
 cout << "No se encontraron productos en este rango de precio." << endl;  
}

}

void consultarPorNivelStock() { system("cls"); cout << "CONSULTA POR NIVEL DE STOCK" << endl; cout << "===========================" << endl; cout << "1 - STOCK MINIMO (igual al stock minimo)" << endl; cout << "2 - STOCK BAJO (por debajo del stock minimo)" << endl; cout << "3 - STOCK NORMAL (por encima del stock minimo)" << endl; cout << "===========================" << endl;

int opcion;  
cout << "Seleccione nivel: ";  
cin >> opcion;  
  
int total = cantidadRegistrosStock();  
bool encontrado = false;  
  
switch (opcion) {  
 case 1: // Stock mínimo  
 cout << "PRODUCTOS CON STOCK MINIMO" << endl;  
 cout << "==================================================" << endl;  
 for (int i = 0; i < total; i++) {  
 Stock producto = leerStock(i);  
 if (!producto.getEliminado() && producto.getStockActual() == producto.getStockMinimo()) {  
 producto.mostrar();  
 encontrado = true;  
 }  
 }  
 break;  
  
 case 2: // Stock bajo  
 cout << "PRODUCTOS CON STOCK BAJO" << endl;  
 cout << "==================================================" << endl;  
 for (int i = 0; i < total; i++) {  
 Stock producto = leerStock(i);  
 if (!producto.getEliminado() && producto.getStockActual() < producto.getStockMinimo()) {  
 producto.mostrar();  
 encontrado = true;  
 }  
 }  
 break;  
  
 case 3: // Stock normal  
 cout << "PRODUCTOS CON STOCK NORMAL" << endl;  
 cout << "==================================================" << endl;  
 for (int i = 0; i < total; i++) {  
 Stock producto = leerStock(i);  
 if (!producto.getEliminado() && producto.getStockActual() > producto.getStockMinimo()) {  
 producto.mostrar();  
 encontrado = true;  
 }  
 }  
 break;  
  
 default:  
 cout << "Opcion invalida!" << endl;  
 system("pause");  
 return;  
}  
  
if (!encontrado) {  
 cout << "No se encontraron productos con este nivel de stock." << endl;  
}

}