Universidad Politécnica de Apodaca

“Programación Orientada a objetos”

Docente: Oscar Salvador Salas Peña

Grupo: 37 A

Carrera: Sistemas computacionales

Cuatrimestre: 6

Ximena Huerta Reta

362312887

Andrea Viridiana Avalos Quiroz

362312892

José Pablo Plata Arratia

362313093

Luis Gerardo Del Rio Rodríguez

362312850

Indice

Introducción

En el siguiente documento se presenta de manera detallada el proceso completo de realización de un proyecto académico, cuyo propósito principal es dar solución a una problemática específica planteada durante las sesiones de clase. Este trabajo no solo tiene como objetivo aplicar los conocimientos adquiridos, sino también fomentar el desarrollo de habilidades prácticas relacionadas con el análisis, diseño e implementación de soluciones tecnológicas. El proyecto se orienta a la creación de un sistema que sea plenamente compatible con las necesidades reales de clientes o usuarios, buscando optimizar procesos y facilitar la gestión de diversas áreas dentro de un negocio.

Problemática

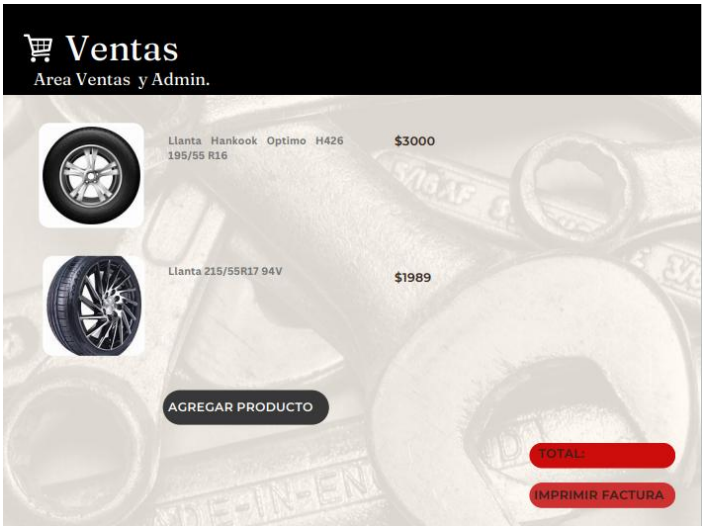
La problemática planteada para resolver fue la siguiente: actualmente, una llantera lleva el registro de sus ventas de forma manual, sin contar con un sistema que permita a los empleados conocer de manera rápida y precisa qué mercancía se encuentra disponible, cuáles fueron las ventas realizadas en el día, cuántos productos quedan en existencia, entre otras dificultades relacionadas con la gestión y el control del inventario.

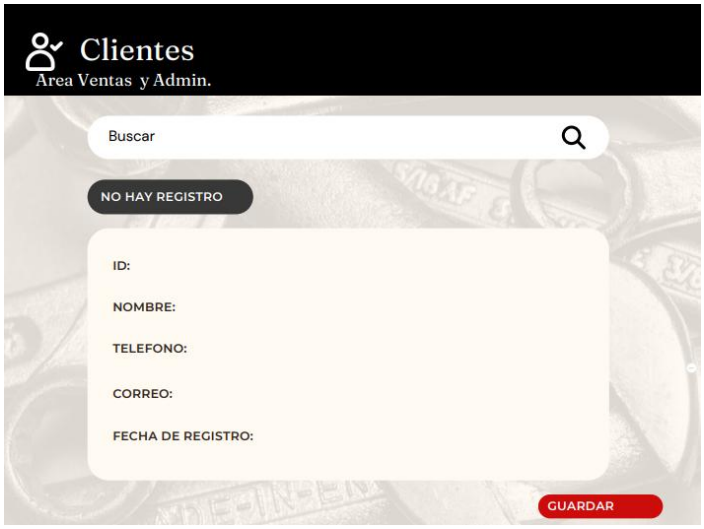
Propuesta

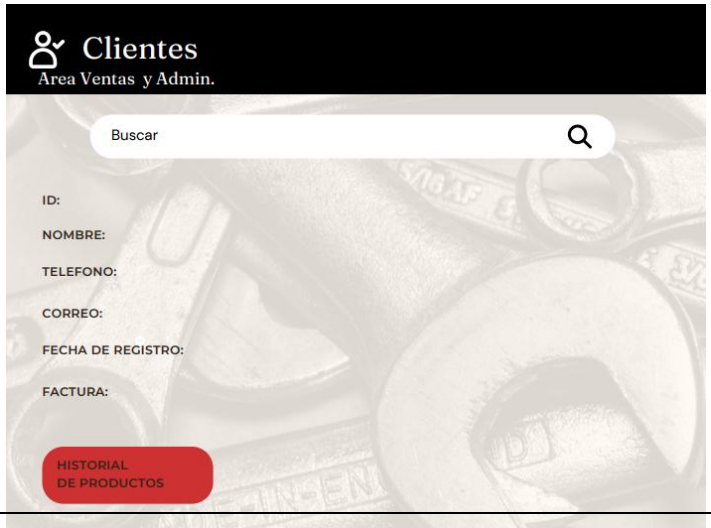
La propuesta formulada tras analizar el problema y sus consecuencias consiste en el desarrollo de un sistema dividido en tres áreas principales, a las cuales se podrá acceder de acuerdo con el puesto asignado en el inicio de sesión: Inventario, Ventas y Administración. Cada una de estas áreas contará con funciones específicas, como el registro de ventas, la incorporación de nuevos clientes, la consulta del inventario disponible, la adición o eliminación de productos, la modificación de precios, entre muchas otras herramientas orientadas a optimizar la gestión y el control de la llantera.

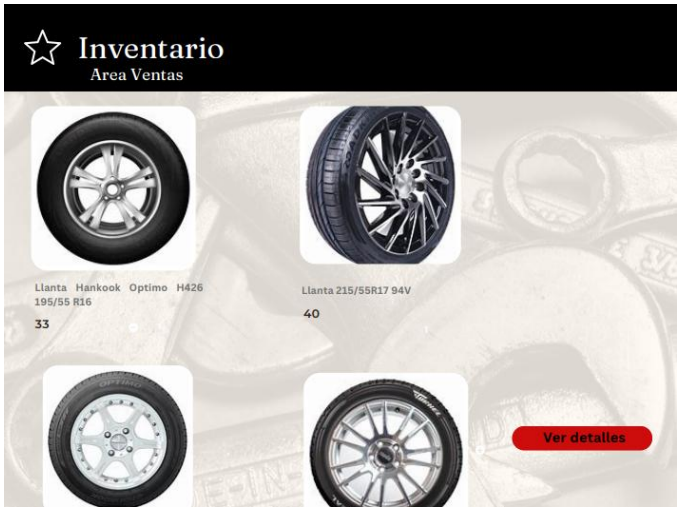
Imágenes del sistema

Imágenes del sistema, mostrando cómo se organizan las diferentes entidades con sus respectivos atributos.
Esta muestra las pantallas con las acciones y permisos que podría realizar si entra en el sistema un empleado del área de ventas:

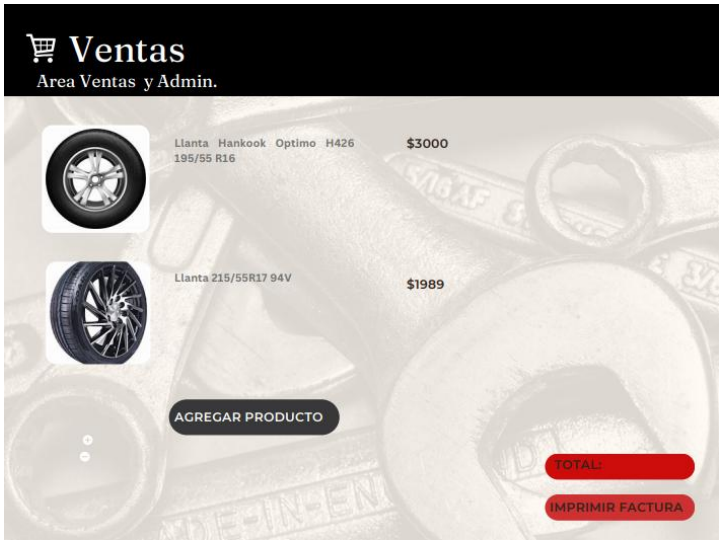


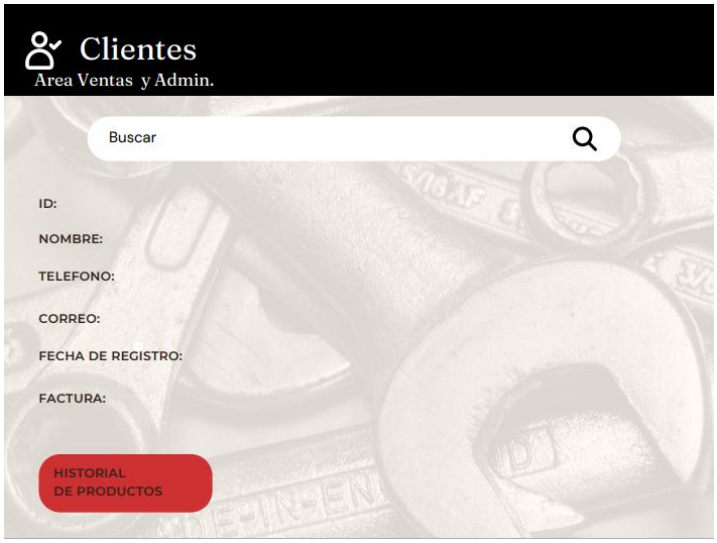


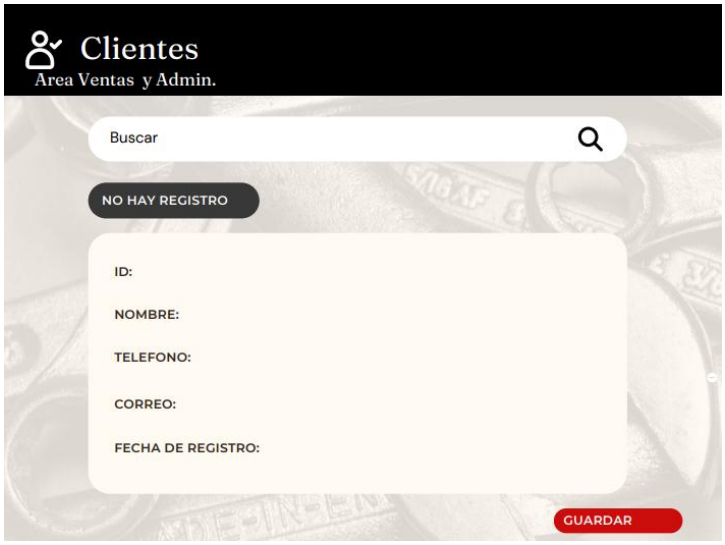




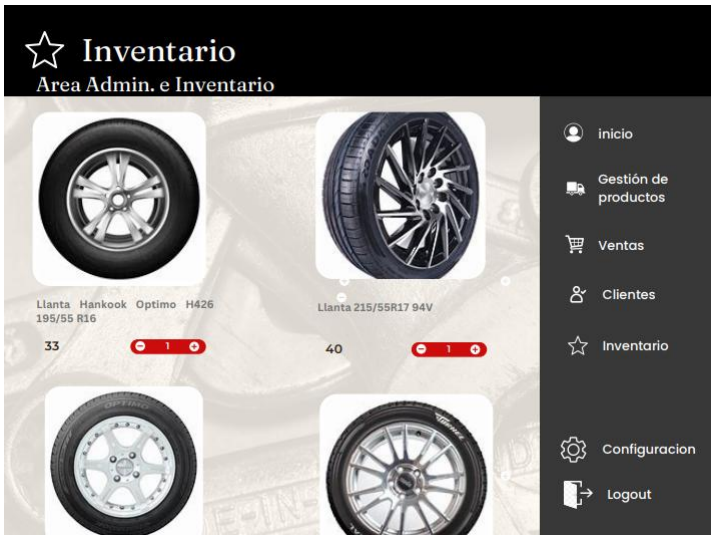
Las siguientes muestran las pantallas que le aparecen si entras al sistema como administrador, este le aparecen las todas las pantallas, además que tiene todos los permisos para modificar productos, realizar ventas, etc. :







Las siguientes son si entra al sistema un empleado del área de inventario:



Diagramas UML

Diagrama de usos de uso

En este diagrama de casos de uso, basado en las entidades y atributos que definimos para el prototipo del sistema, mostramos las operaciones que puede realizar cada persona, las acciones que tienen permitidas dentro del sistema y cómo se relacionan con su entorno. Además, permite visualizar de manera clara qué funciones realiza cada usuario.

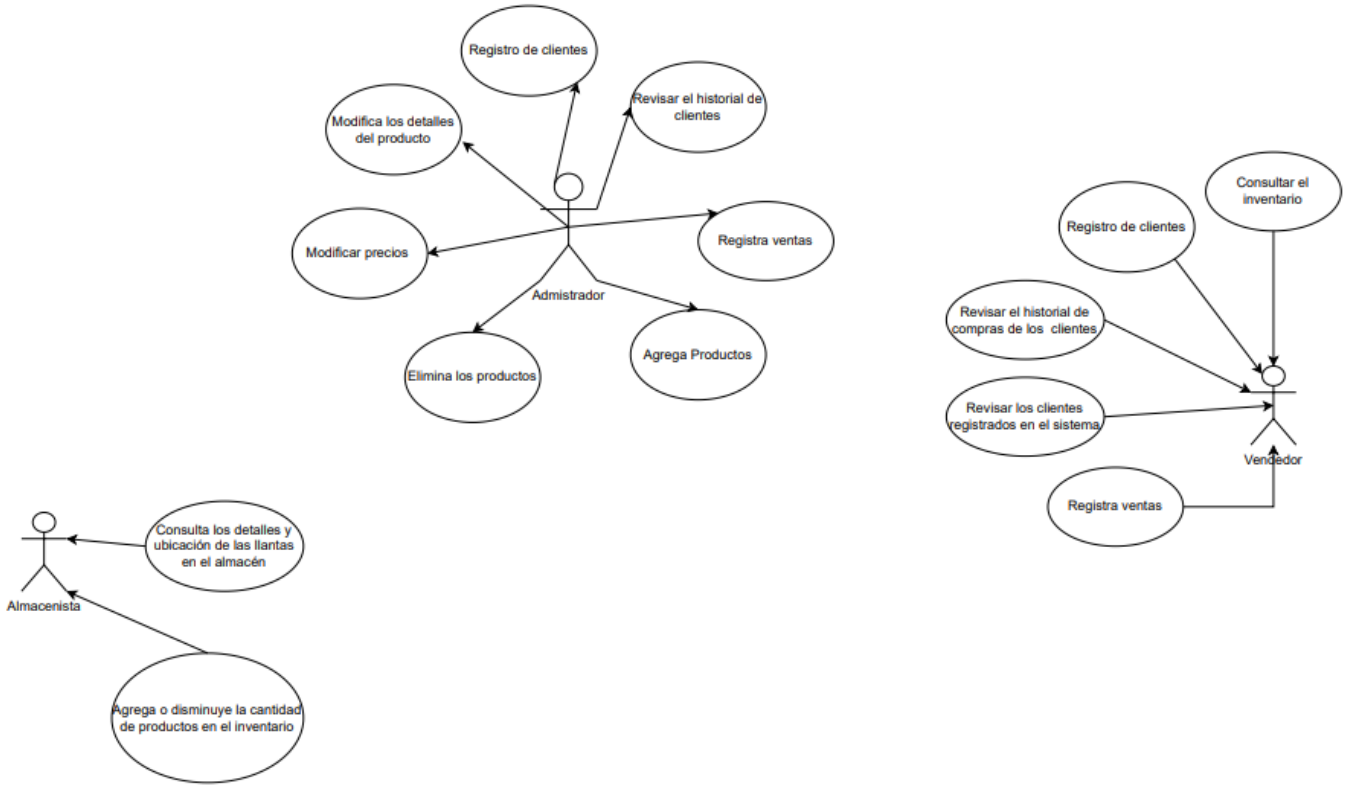
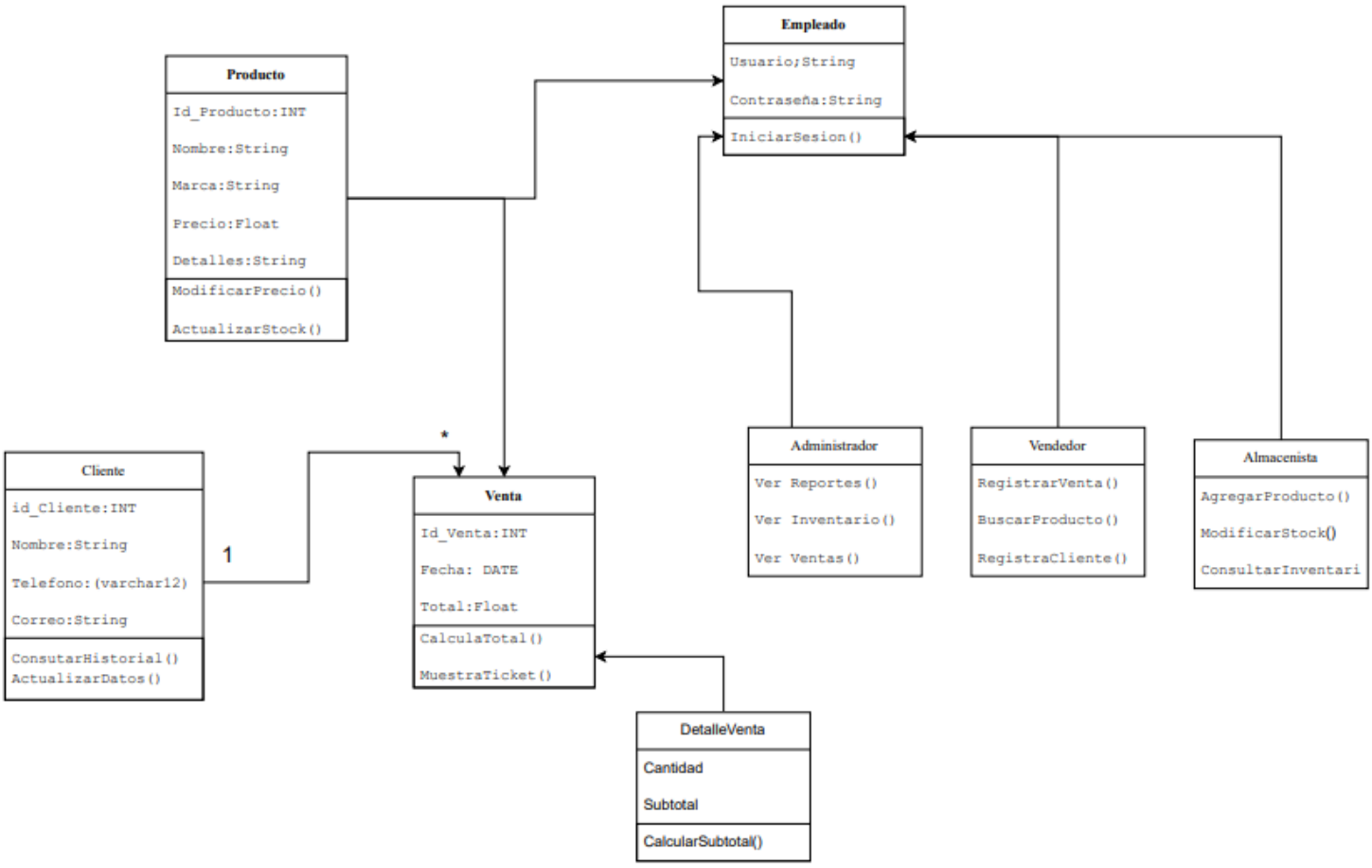


Diagrama de clases

Este diagrama de clases lo realizamos basándonos tanto en la estructura que ya teníamos del prototipo del sistema como en el diagrama de casos de uso. Aquí se describe de manera visual la estructura general del sistema, mostrando las clases, los atributos que contiene cada una y las relaciones que existen entre ellas. Gracias a este diagrama, podemos entender mejor cómo se organiza la información y cómo interactúan las diferentes partes del sistema. Además, nos servirá como guía para saber exactamente qué datos se van a necesitar y cómo se van a manejar cuando pasemos a la etapa de programación y codificación.



Código

El código muestra los métodos que se necesitan para llevar a cabo cada acción que necesitan las diferentes áreas, algunas áreas comparten métodos y son aplicadas de la misma manera.

```
#include <cstring>
#include <cstdlib>
class Llantera{
    private:
        bool sesion;
        int puesto;
        char usuario[20];
        char contra[20];


public:
    Llantera(){
        sesion=false;  //sesion cerrada
        puesto=0;
        usuario[20] = '\0';
        contra[20] = '\0';
    }

void CrearUsuario(){
    int puestoNuevo;
    printf("\nCrea un nombre de usuario: ");
    scanf(" %s", usuario);
    printf("\n Crea una contrasena: ");
    scanf(" %s", contra);
    printf("Ingresa el puesto 1.Ventas 2.Almacen 3.Administracion: ");
        scanf("%d", &puestoNuevo);
        //GUARDAR USUARIOS EN ARCHIVO
        FILE* archivo = fopen("usuarios.txt", "a");
        if (archivo == NULL) {
            printf("Error al abrir el archivo.\n");
            return;
        }
    fprintf(archivo, "%s %s %d\n", usuario, contra, puestoNuevo);
        fclose(archivo);

        printf("Usuario registrado correctamente.\n");
    }
```

```c
void IniciarSesion(){
    char usuario_ing[20], contra_ing[20];
    int puesto_ing;
    sesion = false;

    printf("\nBienvenido a Llantera Pablus");


do{
    printf("\nIngresa tu usuario: ");
    scanf(" %s", usuario_ing);
    printf("\nIngresa tu contrasena: ");
    scanf(" %s", contra_ing);
    printf("\nSelecciona tu area: 1.Ventas  2.Almacen  3.Administracion: "); //seleccionar area de trabajo
    scanf(" %d", &puesto_ing);

    FILE* archivo = fopen("usuarios.txt", "r");
        if (archivo == NULL) {
            printf("Error al abrir el archivo.\n");
            return;
        }

     char usuario_arch[20], contra_arch[20];
     int puesto_arch;

    while (fscanf(archivo, " %s %s %d", usuario_arch, contra_arch, &puesto_arch) != EOF) {
            if (strcmp(usuario_ing, usuario_arch) == 0 &&
                strcmp(contra_ing, contra_arch) == 0 &&
                puesto_ing == puesto_arch) {
                sesion = true;
                puesto=puesto_ing;
                break;
            }
        }
    fclose(archivo);

        if (!sesion) {
            printf("\nUsuario, contrasena o puesto incorrectos. Intenta de nuevo.\n");
        }

    } while (!sesion);

    sesion = true;
    switch (puesto_ing) {
        case 1: printf("Bienvenido al area de Ventas\n"); break;
        case 2: printf("Bienvenido al area de Almacen\n"); break;
        case 3: printf("Bienvenido al area de Administracion\n"); break;
        default: printf("Área desconocida\n"); break;
    }
```

```
        }
    }
 int getPuesto() {
        return puesto;
    }
public:
    int getSesion(){
        return sesion;
    }
};



class Productos: public Llantera{
    public:
    int Id_Producto;
    char Nombre[20];
    char Marca[20];
    float Precio;

public:
 Productos(){
    Id_Producto=0;
    Nombre[20] = '\0';
    Marca[20] = '\0';
    Precio = 0;

}
//METODO
void EliminarProductos() {
    printf("\n=== ELIMINAR PRODUCTOS ===\n");
    int IDMod;
    printf("Ingresa el ID del producto a eliminar: ");
    scanf("%d", &IDMod);

    FILE* archivo = fopen("productos.txt", "r");
    if (archivo == NULL) {
        printf("No se pudo abrir productos.txt.\n");
        return;
    }

    FILE* temp = fopen("temp.txt", "w");
    if (temp == NULL) {
        printf("No se pudo crear archivo temporal.\n");
        fclose(archivo);
        return;
    }
```

```
    int id, stock;
    char nombre[50], marca[50];
    float precio;
    int encontrado = 0;

    while (fscanf(archivo, "%d %49s %49s %f %d", &id, nombre, marca, &precio, &stock) == 5) {

        if (id == IDMod) {
            encontrado = 1;
            printf("Producto a eliminar encontrado: %d %s %s %.2f %d\n", id, nombre, marca, precio, stock);
            // No escribimos este producto al archivo temporal (lo eliminamos)
        } else {
            fprintf(temp, "%d %s %s %.2f %d\n", id, nombre, marca, precio, stock);
        }
    }

    fclose(archivo);
    fclose(temp);

    if (encontrado) {
        remove("productos.txt");
        rename("temp.txt", "productos.txt");
        printf("Producto eliminado correctamente.\n");
    } else {
        remove("temp.txt");
        printf("No se encontró un producto con ID %d.\n", IDMod);
    }
}



void ModificarStock() {
    printf("\n ===MODIFICAR STOCK===\n");
    int StockNew, idModificar;
    printf("\nIngresa el Id del producto a modificar: ");
    scanf("%d", &idModificar);
    printf("\nIngresa el stock nuevo: ");
    scanf("%d", &StockNew);

    FILE* archivo = fopen("productos.txt", "r");
    if (archivo == NULL) {
        printf("Error al abrir el archivo.\n");
        return;
    }

    // Guardar todas las líneas en memoria
    char lineas[1000][200]; // máximo 1000 líneas de 200 caracteres
    int totalLineas = 0;
```

```
    bool encontrado = false;

    while (fgets(lineas[totalLineas], sizeof(lineas[totalLineas]), archivo)) {
        int id, stock;
        char nombre[50], marca[50];
        float precio;

        if (sscanf(lineas[totalLineas], "%d %49s %49s %f %d", &id, nombre, marca, &precio, &stock) == 5) {
            if (id == idModificar) {
                stock = StockNew; // actualizar
                snprintf(lineas[totalLineas], sizeof(lineas[totalLineas]),
                         "%d %s %s %.2f %d\n", id, nombre, marca, precio, stock);
                encontrado = true;
            }
        }
        totalLineas++;
    }
    fclose(archivo);

    // Reescribir el archivo
    archivo = fopen("productos.txt", "w");
    if (archivo == NULL) {
        printf("Error al escribir el archivo.\n");
        return;
    }
    for (int i = 0; i < totalLineas; i++) {
        fputs(lineas[i], archivo);
    }
    fclose(archivo);

    if (encontrado)
        printf("Producto actualizado correctamente.\n");
    else
        printf("Producto con ese ID no encontrado.\n");
}
//METODO
void AgregarProductos(){
        printf("\n ===AGREGAR PRODUCTOS===\n");
    int id_prod,stockk;
    float prec;
    char nom[50], mar[50];
    printf("Agrega un Id_Producto: \n");
    scanf("%d", &id_prod);
    printf("Nombre: \n");
    scanf(" %49s", nom);
    printf("Marca: \n");
    scanf(" %49s", mar);
    printf("Precio: \n");
```

```
    printf("Precio: \n");
    scanf("%f", &prec);
    printf("Stock: \n");
    scanf("%d", &stockk);

    FILE* archivo = fopen("productos.txt", "a");
        if (archivo == NULL) {
            printf("Error al abrir el archivo.\n");
            return;
        }
        fprintf(archivo, "%d %s %s %.2f %d \n", id_prod, nom, mar, prec, stockk);
            fclose(archivo);
}

//METODO
void ModificarPrecio() {
    printf("\n ===MODIFICAR PRECIOS===\n");
    float PrecioNew;
    int idModificar;

    printf("\nIngresa el Id del producto a modificar: ");
    scanf("%d", &idModificar);
    printf("\nIngresa el precio nuevo: ");
    scanf("%f", &PrecioNew);

    FILE* archivo = fopen("productos.txt", "r");
    if (archivo == NULL) {
        printf("Error al abrir el archivo.\n");
        return;
    }


    char lineas[100][200];
    int totalLineas = 0;
    bool encontrado = false;

    while (fgets(lineas[totalLineas], sizeof(lineas[totalLineas]), archivo)) {
        int id, stock;
        char nombre[50], marca[50];
        float precio;

        if (sscanf(lineas[totalLineas], "%d %49s %49s %f %d", &id, nombre, marca, &precio, &stock) == 5) {
            if (id == idModificar) {
                precio = PrecioNew; // actualizar el precio
                snprintf(lineas[totalLineas], sizeof(lineas[totalLineas]),
                         "%d %s %s %.2f %d\n", id, nombre, marca, precio, stock);
                encontrado = true;
            }
        }
```

```
        totalLineas++;
    }
    fclose(archivo);

    // Rponer el precio nuevo en el archivo
    archivo = fopen("productos.txt", "w");
    if (archivo == NULL) {
        printf("Error al escribir el archivo.\n");
        return;
    }
    for (int i = 0; i < totalLineas; i++) {
        fputs(lineas[i], archivo);
    }
    fclose(archivo);

    if (encontrado)
        printf("Producto actualizado correctamente.\n");
    else
        printf("Producto con ese ID no encontrado.\n");
}


void ConsultarInventario() {
    printf("\n ===INVENTARIO===\n");

    FILE* archivo = fopen("productos.txt", "r");
    if (archivo == NULL) {
        printf("No se pudo abrir el archivo de productos.\n");
        return;
    }

    int id, stock;
    char nombre[50], marca[50];
    float precio;

    printf("\n--- Lista de Productos ---\n");
    printf("\nId_Producto\tNombre\t\tMarca\t\tPrecio\t\tStock\n");
    printf("-------------------------------------------------------------\n");

    while (fscanf(archivo, "%d %49s %49s %f %d", &id, nombre, marca, &precio, &stock) == 5) {
        printf("%d\t%-15s %-15s $%.2f\t%d\n", id, nombre, marca, precio, stock);
    }

    fclose(archivo);
}
//METODO
void RegistrarVentas() {
```

```c
printf("\n ===REGISTAR VENTAS===\n");

int numArticulos;
int idventa = 0;

printf("Asigna un Id_Venta: \n");
scanf("%d", &idventa);
printf("Cuantos articulos diferentes en la venta\n ");
scanf("%d", &numArticulos);

float total = 0.0;

// Estructura para almacenar los productos temporalmente
struct Producto {
    int id;
    char nombre[50];
    char marca[50];
    float precio;
    int stock;
};

struct Producto productos[100]; // ajusta el tamaño si tienes muchos
int numProductos = 0;

// Leer todos los productos del archivo
FILE* arch = fopen("productos.txt", "r");
if (arch == NULL) {
    printf("Error al abrir productos.txt\n");
    return;
}

while (fscanf(arch, "%d %49s %49s %f %d",
              &productos[numProductos].id,
              productos[numProductos].nombre,
              productos[numProductos].marca,
              &productos[numProductos].precio,
              &productos[numProductos].stock) == 5) {
    numProductos++;
}

fclose(arch);

// Proceso de venta
for (int i = 0; i < numArticulos; i++) {
    int idProducto, cantidad;
    float precioUnitario;

    printf("\nArticulo %d\n", i + 1);
```

```
    printf("Id_Producto: ");
    scanf("%d", &idProducto);

    printf("Cantidad: ");
    scanf("%d", &cantidad);

    printf("Precio unitario: ");
    scanf("%f", &precioUnitario);

    // AQUI BUSCA EL PRODUCTO PARA VER COINCIDENCIA MEDIANTE EL ID
    int encontrado = 0;
    for (int j = 0; j < numProductos; j++) {
        if (productos[j].id == idProducto) {
            if (productos[j].stock >= cantidad) {
                productos[j].stock -= cantidad;
                total += cantidad * precioUnitario;
                encontrado = 1;
            } else {

                printf("Stock insuficiente para el producto ID %d\n", idProducto);
                return; // termina todo si no hay stock suficiente
            }
            break;
        }
    }

    if (!encontrado) {
        printf("Producto con ID %d no encontrado.\n", idProducto);
        return; // termina si el producto no existe
    }
}

// Guardar cambios en productos.txt
FILE* arch2 = fopen("productos.txt", "w");
if (arch2 == NULL) {
    printf("Error al guardar productos.txt\n");
    return;
}

for (int i = 0; i < numProductos; i++) {
    fprintf(arch2, "%d %s %s %.2f %d\n",
            productos[i].id,
            productos[i].nombre,
            productos[i].marca,
            productos[i].precio,
            productos[i].stock);
}
```

```
    fclose(arch2);

    // Registrar venta
    FILE* archivoVenta = fopen("ventas.txt", "a");
    if (archivoVenta == NULL) {
        printf("Error al abrir ventas.txt\n");
        return;
    }

    fprintf(archivoVenta, "Id_Venta: %d | Articulos: %d | Total: %.2f\n", idventa, numArticulos, total);
    fclose(archivoVenta);

    printf("Venta registrada y stock actualizado correctamente.\n");
}



};

class Clientes: public Productos{
    int Id_Cliente;
    char Nombre[25];
    char Numero[20];
    char Correo[20];

public:
 Clientes(){
    Id_Cliente=0;
    Nombre[25]='\0';
    Numero[20]='\0';
    Correo[20]='\0';
 }
//METODO
void RegistrarCliente(){
    int id;
    char nom[25], num[20], cor[20];
    printf("\n===REGISTRAR CLIENTES===\n");
    printf("\nAgrega un Id_Cliente: ");
    scanf("%d", &id);
    printf("\nNombre:" );
    scanf(" %s", nom);
    printf("\nNumero:" );
    scanf(" %s", num);
    printf("\nCorreo:" );
    scanf(" %s", cor);
    FILE* archivo = fopen("clientes.txt", "a");
 if (archivo == NULL) {
        printf("Error al abrir el archivo.\n");
```

```
            return;
        }
        fprintf(archivo, "Id_Cliente: %d| Nombre: %s | Numero: %s | Correo: %s \n", id, nom, num, cor);
            fclose(archivo);
            printf("Cliente registrado correctamente\n");

}
//METODO
void BuscarCliente(){
        int id_buscar;
        printf("\n ===BUSCAR CLIENTE===\n");
        printf("Id_Cliente: ");
        scanf("%d", &id_buscar);

        FILE* archivo = fopen("clientes.txt", "r");
        if (archivo == NULL) {
            printf("Error al abrir el archivo.\n");
            return;
        }
        int id;
        char nom[50], num[20], cor[50];
        bool encontrado = false;

    while (fscanf(archivo, "Id_Cliente: %d| Nombre: %s | Numero: %s | Correo: %s \n", &id, nom, num, cor)==4){

        if (id==id_buscar){
            printf("\n Cliente encontrado: \n");
             printf("Id_Cliente: %d\nNombre: %s\nNumero: %s\nCorreo: %s\n", id, nom, num, cor);
            encontrado=true;
            break;
        }
    }
        if(!encontrado){
            printf("\nCliente no registrado\n");
        }
 fclose(archivo);
}
};

//METODO DE LIMPIAR PANTALLA Q FALTABA
void limpiarPantalla(){
    system("cls");
}

int main() {
    Llantera l;
    Productos p;
    Clientes c;
```

```
limpiarPantalla();
printf("\n===BIENVENIDO A LLANTAS PABLUS===\n");


l.IniciarSesion();

int pue = l.getPuesto();  // Obtener el puesto
int op;
bool sesi = true;

do {
    if (pue == 1) {
        do {
            limpiarPantalla();
            printf("\n===Area de Ventas===\n");
            printf("\nQue deseas hacer: 1) Consultar Inventario\n2) Registrar Venta\n3) Registrar Cliente\n4) Buscar Cliente\n0) Salir\nOpcion: ");
            scanf("%d", &op);

            switch(op) {
                case 1: limpiarPantalla();
                p.ConsultarInventario();
                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                break;
                case 2: limpiarPantalla();
                p.ConsultarInventario();
                p.RegistrarVentas();

                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                break;
                case 3: limpiarPantalla();
                c.RegistrarCliente();
                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                break;
                case 4: limpiarPantalla();
                c.BuscarCliente();
                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                break;
                case 0: limpiarPantalla();
                    printf("Saliendo del area de ventas...\n");
                    sesi = false;
                    printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                    break;
                default: printf("Opción no reconocida\n");
            }
        } while (op != 0);
    }

    else if (pue == 2) {
        do { limpiarPantalla();
            printf("\n===Area de Almacen===\n");
            printf("\nQue deseas hacer: 1) Consultar Inventario\n2) Modificar Stock\n3) Agregar Productos\n \n4)Eliminar Productos\n n0) Salir\nOpcion: ");

            scanf("%d", &op);

            switch(op) {
                case 1: limpiarPantalla();
                p.ConsultarInventario();
                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                break;
                case 2: limpiarPantalla();
                p.ModificarStock();
                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                break;
                case 3: limpiarPantalla();
                p.AgregarProductos();
                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                break;
                case 4: limpiarPantalla();
                p.EliminarProductos();
                printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
                    break;

                case 0: limpiarPantalla();
                    printf("Saliendo del area de almacen...\n");
                    sesi = false;
                    printf("\nPresione Enter para continuar...");
                    getchar(); // Limpia el buffer
                    getchar();
```

```
                break;
            default: printf("Opcion no reconocida\n");
        }
    } while (op != 0);
}

else if (pue == 3) {
    do { limpiarPantalla();
        printf("\n===Area Administrativa===\n");
        printf("\nQue deseas hacer: 1) Crear Usuario\n2) Agregar Productos\n3) Modificar Stock\n4) Consultar Inventario\n5) Registrar Venta\n6) Buscar Cliente\n \n8)Eliminar Productos \n7) Registrar Cliente\n0) Salir\nOpcion: ");
        scanf("%d", &op);

        switch(op) {
            case 1: limpiarPantalla();
            l.CrearUsuario();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
                break;
            case 2: limpiarPantalla();
            p.AgregarProductos();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
                break;
            case 3: limpiarPantalla();
            p.ModificarStock();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
            break;
            case 4: limpiarPantalla();
            p.ConsultarInventario();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
            break;
            case 5: limpiarPantalla();
            p.ConsultarInventario();
            p.RegistrarVentas();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
            break;
            case 6: limpiarPantalla();
            c.BuscarCliente();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
```

```
                getchar();
                break;
            case 7: limpiarPantalla();
            c.RegistrarCliente();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
            break;
            case 8: limpiarPantalla();
            p.EliminarProductos();
            printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
                break;
            case 0: limpiarPantalla();
                printf("Saliendo del area administrativa...\n");
                sesi = false;
                printf("\nPresione Enter para continuar...");
                getchar(); // Limpia el buffer
                getchar();
                break;
            default: printf("Opcion no reconocida\n");
            }
        } while (op != 0);
    }

    else {
        printf("\nPuesto no válido o no reconocido.\n");
        sesi = false;
    }

} while (sesi);

return 0;
}
```

Conclusión:

Podemos concluir que el sistema implementado para dar solución al problema planteado resulta altamente satisfactorio, ya que cumple con los requisitos especificados y demuestra ser eficaz en tareas como el registro de ventas y clientes, así como en la alta y baja de productos. En general, el sistema ofrece un funcionamiento estable, confiable y eficiente, contribuyendo de manera significativa a la optimización de los procesos en la llantera.