

**CORPORACIÓN UNIVERSITARIA CORHUILA**

**ESTUDIANTES**

Juan Esteban Oliveros Duran

Daniel Stiven Poveda Cante

Angie Valentina Florez Vargas

Charith Nikool Chavarro Meneses

**PARA:**

Ing**.** Jesús Ariel Gonzales Bonilla

FACULTAD DE INGENIERÍA

NEIVA – HUILA

2024

## **Tabla de Contenido**

[Tabla de Contenido 2](#_Toc70552700)

[Introducción 2](#_Toc625859381)

[Propósito 2](#_Toc2067387490)

[Ámbito del Producto 2](#_Toc1886880681)

[Descripción del Sistema 2](#_Toc1985814764)

[1. Funcionalidades del Sistema 2](#_Toc2067602959)

[1.1 Gestión de Inventario 2](#_Toc2097853533)

[1.2 Gestión de Carrito de Compras 3](#_Toc856256874)

[Características de los usuarios: 3](#_Toc1489751277)

[• Restricciones generales: 3](#_Toc595513561)

[Requisitos de calidad: 3](#_Toc1268037705)

[Restricciones de diseño: 4](#_Toc390691210)

[Características no esenciales: 4](#_Toc471704209)

[2. Requerimientos Funcionales 4](#_Toc789690151)

[2.1 RF1 - Gestión de Productos 4](#_Toc14055852)

[2.2 RF2 - Visualización de Productos 5](#_Toc1564751402)

[2.3 RF3 - Agregar Productos al Carrito 5](#_Toc292470132)

[2.4 RF4 - Modificar Cantidad de Producto en el Carrito 5](#_Toc1358190803)

[2.5 RF5 - Eliminar Productos del Carrito 6](#_Toc1429680832)

[2.6 RF6 - Vaciar el Carrito 6](#_Toc468647133)

[2.7 RF7 - Proceder a la Compra 7](#_Toc2122272800)

[3. Requerimientos No Funcionales 7](#_Toc1687014440)

[Diagramas. 8](#_Toc1242178173)

[Diagrama Clases: 8](#_Toc984552098)

[Clases del sistema: 8](#_Toc336861582)

[Relaciones entre las clases: 10](#_Toc1378234480)

[Funcionamiento en la práctica: 11](#_Toc41913094)

[Diagrama Casos de Uso: 11](#_Toc2127570951)

[Actores: 12](#_Toc1455218600)

[Paquetes: 12](#_Toc1546277660)

[Relaciones entre casos de uso y actores: 13](#_Toc122338347)

[Diagrama Actividades: 13](#_Toc284705566)

[Inicio de sesión o Registro: 14](#_Toc533208941)

[Redirección según el rol: 14](#_Toc1960846509)

[Flujo del Administrador (bifurcación): 15](#_Toc1171480994)

[Flujo del Cliente (bifurcación): 15](#_Toc1097038606)

[Paralelismo (fork y end fork): 15](#_Toc668988135)

[Fin (stop): 15](#_Toc1921704915)

[Diagrama de Paquetes: 16](#_Toc1064609228)

[1. Paquete "Sistema" 16](#_Toc1292732037)

[2. Paquete "Usuarios" 16](#_Toc988713086)

[3. Paquete "Inventario" 17](#_Toc1548768442)

[4. Paquete "Carrito de Compras" 17](#_Toc1424511674)

[5. Paquete "Factura" 18](#_Toc365595647)

[6. Paquete "Pago" 18](#_Toc261891206)

[Resumen de las Relaciones entre los Paquetes: 19](#_Toc1016825808)

[Prototipo o Mockup. 19](#_Toc1265658168)

[2. Inicio de Sesión: 21](#_Toc456475937)

[3. Catálogo de Productos (Pestaña Productos): 21](#_Toc1380150119)

[4. Inventario (Pestaña Inventario): 22](#_Toc1717890038)

[5. Imagen de Agregado con Éxito: 23](#_Toc1748457161)

[6. Desde Productos Agregado al Carrito (Pestaña Productos): 24](#_Toc1437733776)

[7. Carrito con los Productos Agregados (Pestaña Carrito): 25](#_Toc1551000862)

[8. Eliminar Producto o Vaciar Carrito (En la Pestaña Carrito): 26](#_Toc521501390)

## 

## **Introducción**

### **Propósito**

Este documento tiene como objetivo describir detalladamente el sistema de gestión de inventario y carrito de compras, su funcionamiento, los requerimientos necesarios y los detalles técnicos para su implementación y operación. Su propósito es proporcionar una visión clara del sistema para los desarrolladores, administradores y cualquier otra parte interesada en el proyecto.

### **Ámbito del Producto**

El sistema desarrollado es una aplicación web que permite a los usuarios gestionar un carrito de compras y un inventario de productos. Los usuarios pueden agregar productos al carrito, modificar la cantidad de estos, proceder con la compra y vaciar el carrito. Además, el sistema permite gestionar el inventario de productos disponibles, con la posibilidad de agregar, modificar y eliminar productos del inventario

## **Descripción del Sistema**

### **1. Funcionalidades del Sistema**

#### **1.1 Gestión de Inventario**

* **Añadir productos al inventario**: Los administradores pueden agregar nuevos productos al inventario, especificando detalles como nombre, descripción, precio, cantidad disponible e imagen del producto.
* **Modificar productos**: Los administradores pueden modificar la información de los productos existentes, como el nombre, la descripción, el precio, la cantidad en stock o la imagen.
* **Eliminar productos**: Los administradores pueden eliminar productos del inventario.
* **Consultar inventario**: Los usuarios y administradores pueden ver la lista de productos disponibles en el inventario, con sus detalles y cantidad disponible.

#### **1.2 Gestión de Carrito de Compras**

* **Añadir productos al carrito**: Los usuarios pueden agregar productos al carrito desde el apartado de productos, eligiendo la cantidad deseada.
* **Actualizar la cantidad de productos**: Los usuarios pueden modificar la cantidad de productos en su carrito antes de proceder con la compra.
* **Eliminar productos del carrito**: Los usuarios pueden eliminar productos individualmente del carrito.
* **Vaciar el carrito**: Los usuarios pueden vaciar completamente el carrito en cualquier momento.
* **Ver el total**: El sistema calcula el total de la compra en función de los productos en el carrito y sus cantidades.
* **Proceder a la compra**: Los usuarios pueden proceder al checkout para completar la compra, una vez que estén satisfechos con el carrito.

#### **Características de los usuarios:**

* **Usuarios finales**: Personas que navegarán por el sistema para ver productos, agregarlos al carrito y realizar compras. Necesitan una interfaz intuitiva y fácil de usar.
* **Administradores**: Personas encargadas de gestionar el inventario, agregar productos y mantener la base de datos actualizada. Necesitan funciones de administración más avanzadas.

#### **• Restricciones generales:**

* **Seguridad**: El sistema debe asegurar que los datos de los usuarios estén protegidos (en caso de implementar la autenticación).
* **Escalabilidad**: El sistema debe poder soportar un aumento en la cantidad de productos y usuarios.
* **Cumplimiento de CORS**: Asegurar que las peticiones entre el frontend y el backend se manejen correctamente para evitar problemas de CORS.

#### **Requisitos de calidad:**

* **Confiabilidad**: El sistema debe ser confiable y mantener la integridad de los datos.
* **Mantenibilidad**: El código debe estar bien estructurado y documentado para facilitar futuras mejoras.
* **Seguridad**: El sistema debe cumplir con las mejores prácticas de seguridad en cuanto a protección de datos y autenticación.

#### **Restricciones de diseño:**

* El sistema debe ser desarrollado utilizando Angular para el frontend y Spring Boot para el backend.
* La base de datos debe utilizar MySQL como sistema de gestión.

#### **Características no esenciales:**

* **Recomendaciones de productos**: Se puede agregar una funcionalidad en el futuro que sugiera productos relacionados a los usuarios.

**Requerimientos del Sistema**

### **2. Requerimientos Funcionales**

### **2.1 RF1 - Gestión de Productos**

1. **Definir el problema**: Los administradores necesitan gestionar los productos del inventario para agregar, editar y eliminar productos según sea necesario.
2. **Entrada y salida de datos**:
   1. **Entrada**: Nombre del producto, descripción, precio, cantidad, imagen (opcional).
   2. **Salida**: Confirmación de operación (producto agregado, editado o eliminado).
3. **Descripción**: El sistema debe permitir a los administradores agregar nuevos productos al inventario, modificar la información de productos existentes y eliminar productos que ya no están disponibles o deseados en el inventario.
4. **Proceso**: Los administradores ingresan la información del producto a través de un formulario en el sistema, que se valida y almacena en la base de datos correspondiente. El sistema actualizará la lista de productos en el inventario después de cada operación.
5. **Restricciones**:
   1. El nombre del producto debe ser único.
   2. El precio debe ser un número positivo.
   3. La cantidad debe ser un valor entero positivo.

### **2.2 RF2 - Visualización de Productos**

1. **Definir el problema**: Los usuarios deben poder ver los productos disponibles en el inventario para poder agregarlos a su carrito de compras.
2. **Entrada y salida de datos**:
   1. **Entrada**: Petición de los productos a visualizar.
   2. **Salida**: Lista de productos con sus detalles (nombre, precio, descripción, imagen).
3. **Descripción**: Los usuarios deben poder navegar por una lista de productos disponibles, ver detalles como nombre, descripción, precio y una imagen del producto. Esta información será presentada de manera ordenada y accesible.
4. **Proceso**: El sistema recupera los productos desde la base de datos y los muestra al usuario en una página de catálogo. El sistema debe permitir al usuario filtrar y ordenar los productos por categoría, precio y nombre.
5. **Restricciones**:
   1. Los productos deben ser visibles solo si tienen stock disponible (cantidad mayor a cero).

### **2.3 RF3 - Agregar Productos al Carrito**

1. **Definir el problema**: Los usuarios deben poder agregar productos al carrito de compras para realizar la compra posteriormente.
2. **Entrada y salida de datos**:
   1. **Entrada**: Producto seleccionado, cantidad.
   2. **Salida**: Producto agregado al carrito, confirmación de operación.
3. **Descripción**: Los usuarios pueden seleccionar un producto y añadirlo a su carrito especificando la cantidad deseada. El carrito debe actualizarse dinámicamente con cada adición.
4. **Proceso**: Cuando un usuario selecciona un producto y define la cantidad, el sistema valida que la cantidad solicitada esté disponible en el inventario. El producto y la cantidad se almacenan en la tabla del carrito, y el total se recalcula.
5. **Restricciones**:
   1. La cantidad solicitada no debe exceder la cantidad disponible en el inventario.

### **2.4 RF4 - Modificar Cantidad de Producto en el Carrito**

1. **Definir el problema**: Los usuarios deben poder modificar la cantidad de un producto en su carrito antes de realizar la compra.
2. **Entrada y salida de datos**:
   1. **Entrada**: Producto en el carrito, nueva cantidad.
   2. **Salida**: Carrito actualizado con la nueva cantidad y total recalculado.
3. **Descripción**: Los usuarios pueden cambiar la cantidad de productos dentro de su carrito de compras, y el sistema debe recalcular el total automáticamente en función de la cantidad modificada.
4. **Proceso**: Cuando un usuario cambia la cantidad de un producto en su carrito, el sistema valida que la nueva cantidad no exceda el stock disponible y luego actualiza el carrito y recalcula el total de la compra.
5. **Restricciones**:
   1. La cantidad no debe ser negativa ni exceder el stock disponible.

### **2.5 RF5 - Eliminar Productos del Carrito**

1. **Definir el problema**: Los usuarios deben poder eliminar productos del carrito antes de proceder con la compra.
2. **Entrada y salida de datos**:
   1. **Entrada**: Producto seleccionado para eliminar.
   2. **Salida**: Producto eliminado del carrito y total actualizado.
3. **Descripción**: Los usuarios pueden eliminar productos de su carrito de compras si cambian de opinión o deciden no comprarlos. El carrito se actualizará automáticamente después de la eliminación.
4. **Proceso**: Cuando un usuario selecciona un producto para eliminar, el sistema lo elimina del carrito y recalcula el total de la compra.
5. **Restricciones**:
   1. El sistema no debe permitir eliminar productos de un carrito que ya ha sido confirmado para compra.

### **2.6 RF6 - Vaciar el Carrito**

1. **Definir el problema**: Los usuarios deben poder vaciar completamente su carrito de compras antes de proceder con la compra.
2. **Entrada y salida de datos**:
   1. **Entrada**: Acción de vaciar el carrito.
   2. **Salida**: Carrito vacío, confirmación de operación.
3. **Descripción**: Los usuarios tienen la opción de vaciar todos los productos de su carrito si deciden no proceder con la compra. Esta opción debe ser clara y fácil de usar.
4. **Proceso**: El sistema eliminará todos los productos del carrito y recalculará el total, dejándolo en cero.
5. **Restricciones**:
   1. Esta acción es irreversible una vez confirmada.

### **2.7 RF7 - Proceder a la Compra**

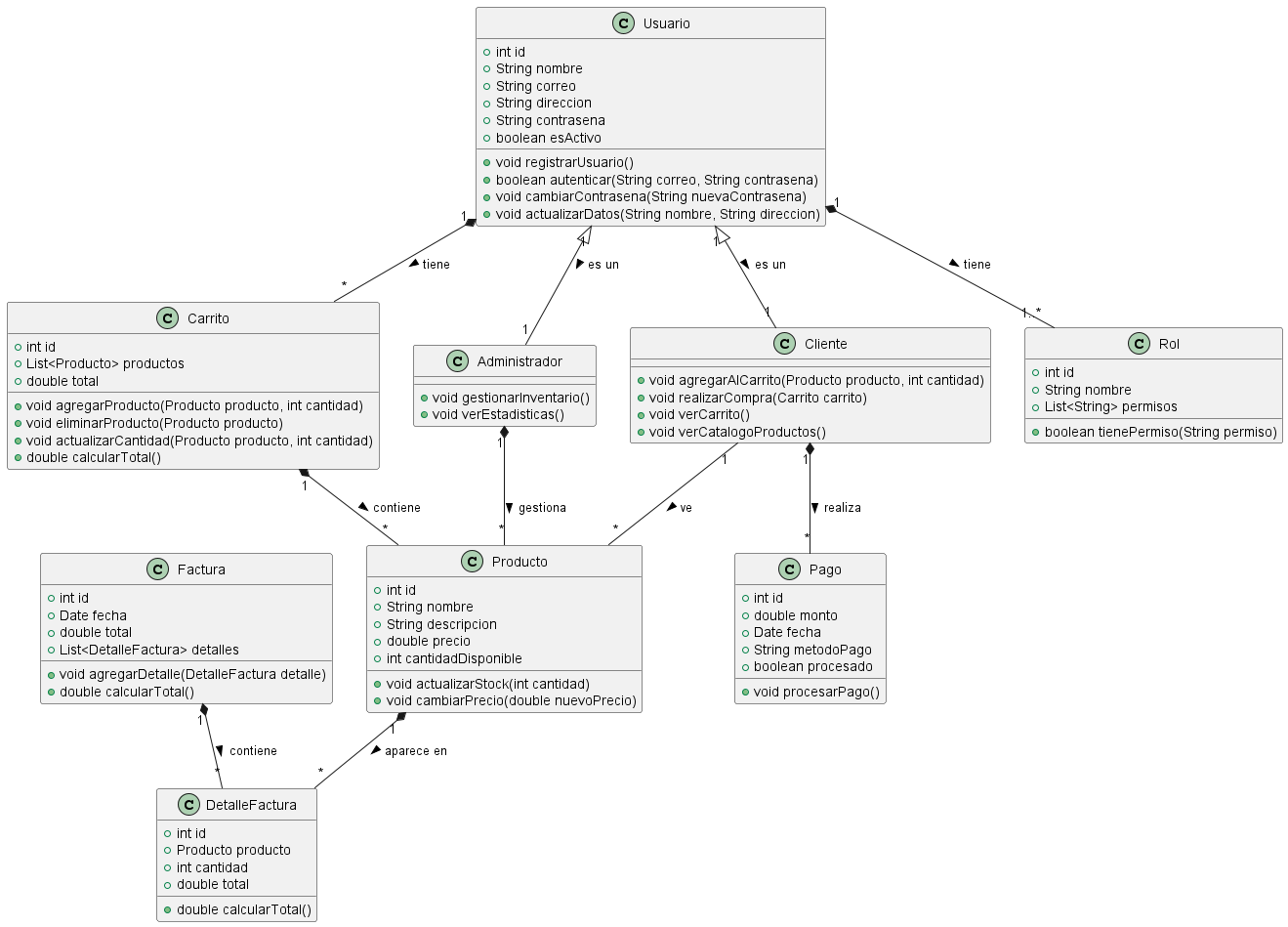
1. **Definir el problema**: Los usuarios deben poder proceder a la compra de los productos en su carrito, confirmando los detalles y completando el pago.
2. **Entrada y salida de datos**:
   1. **Entrada**: Productos del carrito, detalles de pago, dirección de envío.
   2. **Salida**: Confirmación de la compra, generación de factura.
3. **Descripción**: Los usuarios pueden revisar los productos en su carrito, realizar ajustes finales y confirmar la compra proporcionando sus datos de pago y dirección de envío. El sistema procesará la compra y generará una factura.
4. **Proceso**: El sistema valida la información de pago y dirección, realiza el pago (con integración de sistema de pago) y genera una factura de compra. El sistema también actualiza el inventario para reflejar la compra.
5. **Restricciones**:
   1. El sistema debe asegurar que los detalles de pago estén correctamente procesados y que la transacción sea segura.

### **3. Requerimientos No Funcionales**

* **Interfaz de usuario amigable**: El sistema debe ser fácil de usar, con una interfaz intuitiva tanto para los administradores como para los usuarios.
* **Acceso desde múltiples dispositivos**: El sistema debe ser accesible desde computadoras de escritorio y dispositivos móviles.
* **Rendimiento**: El sistema debe poder manejar un número adecuado de productos y usuarios sin sufrir caídas de rendimiento.
* **Seguridad**: El sistema debe contar con seguridad básica para prevenir accesos no autorizados a la gestión de productos y carritos de compras.

## **Diagramas.**

### **Diagrama Clases:**



### **Clases del sistema:**

1. **Producto:**
   1. Esta clase contiene toda la información relevante sobre un producto, como el nombre, la descripción, el precio, y la cantidad disponible. Además, tiene métodos para actualizar el stock y cambiar el precio del producto.
   2. **Métodos:**
      1. actualizarStock(int cantidad): Actualiza la cantidad disponible del producto.
      2. cambiarPrecio(double nuevoPrecio): Cambia el precio del producto.
2. **Carrito:**
   1. El **Carrito** es la representación de lo que un **Cliente** quiere comprar. Contiene una lista de productos y un método para calcular el total de la compra.
   2. **Métodos:**
      1. agregarProducto(Producto producto, int cantidad): Agrega un producto al carrito con la cantidad especificada.
      2. eliminarProducto(Producto producto): Elimina un producto del carrito.
      3. actualizarCantidad(Producto producto, int cantidad): Actualiza la cantidad de un producto en el carrito.
      4. calcularTotal(): Calcula el total de la compra en el carrito.
3. **Factura:**
   1. La **Factura** representa la compra finalizada de un **Cliente**. Contiene los detalles de los productos comprados y su costo total.
   2. **Métodos:**
      1. agregarDetalle(DetalleFactura detalle): Agrega un detalle de factura, que contiene un producto y su cantidad.
      2. calcularTotal(): Calcula el total de la factura.
4. **DetalleFactura:**
   1. El **DetalleFactura** es un registro individual de cada producto comprado, incluyendo la cantidad y el total de esa compra específica.
   2. **Métodos:**
      1. calcularTotal(): Calcula el total de un detalle específico, multiplicando el precio del producto por la cantidad comprada.
5. **Usuario:**
   1. La clase **Usuario** contiene la información básica de un usuario (nombre, correo, dirección, contraseña, etc.). Los usuarios pueden ser de diferentes tipos (como **Administrador** o **Cliente**), por lo que esta clase es una clase base.
   2. **Métodos:**
      1. registrarUsuario(): Registra un nuevo usuario en el sistema.
      2. autenticar(String correo, String contrasena): Autentica a un usuario con su correo y contraseña.
      3. cambiarContrasena(String nuevaContrasena): Cambia la contraseña del usuario.
      4. actualizarDatos(String nombre, String direccion): Permite que el usuario actualice su información personal.
6. **Administrador:**
   1. Los **Administradores** gestionan el inventario (productos) y pueden ver estadísticas relacionadas con el sistema.
   2. **Métodos:**
      1. gestionarInventario(): Permite al administrador gestionar los productos disponibles (agregar, eliminar, editar).
      2. verEstadisticas(): Muestra estadísticas relacionadas con las compras, ventas, productos, etc.
7. **Cliente:**
   1. Los **Clientes** son usuarios que compran productos. Pueden agregar productos al carrito, realizar la compra y ver el carrito de compras.
   2. **Métodos:**
      1. agregarAlCarrito(Producto producto, int cantidad): Agrega un producto al carrito del cliente.
      2. realizarCompra(Carrito carrito): Finaliza la compra del carrito.
      3. verCarrito(): Muestra los productos que están actualmente en el carrito del cliente.
      4. verCatalogoProductos(): Método que permite al **Cliente** ver todos los productos disponibles en el sistema, es decir, acceder al catálogo de productos.
8. **Pago:**
   1. La clase **Pago** representa la transacción financiera realizada por el cliente al realizar la compra.
   2. **Métodos:**
      1. procesarPago(): Procesa el pago, estableciendo que el pago ha sido completado.
9. **Rol:**
   1. El **Rol** define los permisos y responsabilidades de cada usuario dentro del sistema.
   2. **Métodos:**
      1. tienePermiso(String permiso): Verifica si un usuario tiene un permiso específico, como el de gestionar inventario o realizar compras.

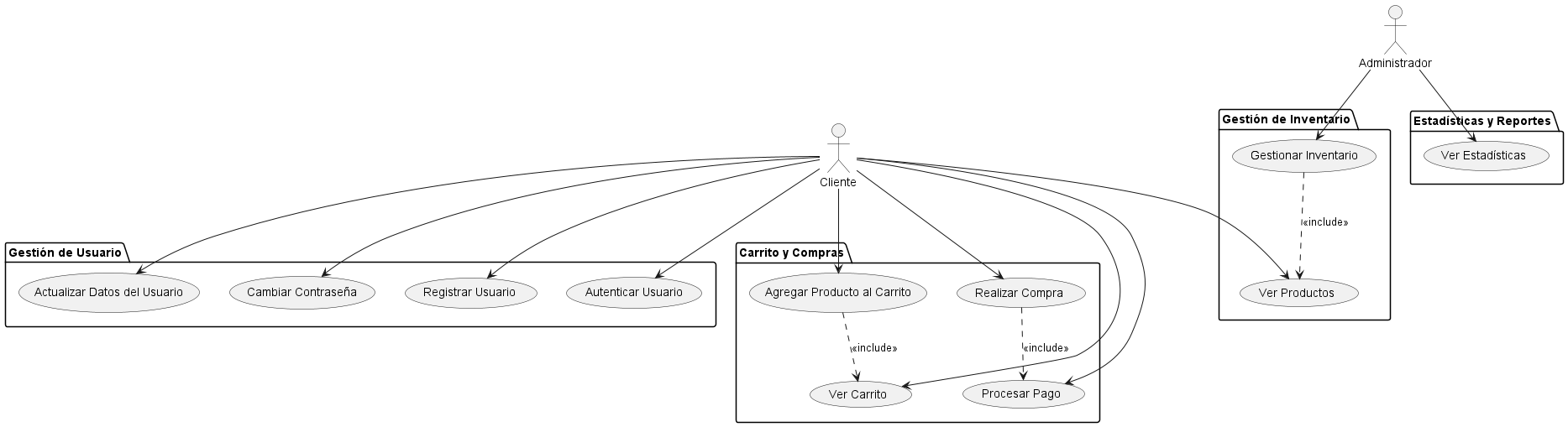
### **Relaciones entre las clases:**

* **Carrito contiene productos:** El **Carrito** puede contener múltiples **Productos**, representados en una lista de productos. Un carrito puede tener muchos productos agregados por el cliente.
* **Factura contiene detalles de factura:** Una **Factura** puede contener múltiples **DetallesFactura**, que a su vez corresponden a productos específicos en una compra.
* **Producto aparece en los detalles de la factura:** Cada **Producto** puede aparecer en múltiples **DetallesFactura**, ya que un producto puede ser comprado más de una vez, en diferentes facturas.
* **Usuario tiene un Carrito:** Un **Usuario** (tanto administrador como cliente) tiene un **Carrito** asociado, donde guarda los productos que desea comprar.
* **Administrador gestiona productos:** El **Administrador** tiene acceso al sistema de gestión de productos, lo que le permite agregar, eliminar y modificar los productos disponibles en el catálogo.
* **Cliente realiza pagos:** Un **Cliente** puede realizar **Pagos** relacionados con las compras realizadas en el sistema.
* **Cliente puede ver productos:** El **Cliente** tiene la capacidad de **ver** múltiples productos en el catálogo, lo que le permite explorar las opciones antes de agregar productos al carrito.
* **Relaciones de herencia entre Usuario, Administrador y Cliente:** Un **Administrador** y un **Cliente** son tipos específicos de **Usuario**. Ambos heredan la funcionalidad básica del **Usuario**, pero el **Administrador** tiene permisos adicionales, como gestionar el inventario.

### **Funcionamiento en la práctica:**

1. El **Cliente** inicia sesión y puede ver el catálogo de productos mediante el método verCatalogoProductos().
2. Una vez que el **Cliente** ve los productos, puede agregarlos al **Carrito** utilizando el método agregarAlCarrito(Producto producto, int cantidad).
3. El **Cliente** revisa su carrito mediante verCarrito(), y luego puede realizar la compra finalizada con el método realizarCompra(Carrito carrito).
4. Finalmente, el **Cliente** puede hacer un **Pago** por la compra.

### **Diagrama Casos de Uso:**



### **Actores:**

1. **Cliente**: Es el usuario final que interactúa con el sistema para realizar diversas acciones como visualizar productos, agregar productos al carrito, realizar compras, etc.
2. **Administrador**: Es el usuario con permisos avanzados que puede gestionar el inventario y ver estadísticas relacionadas con el sistema de compras.

### **Paquetes:**

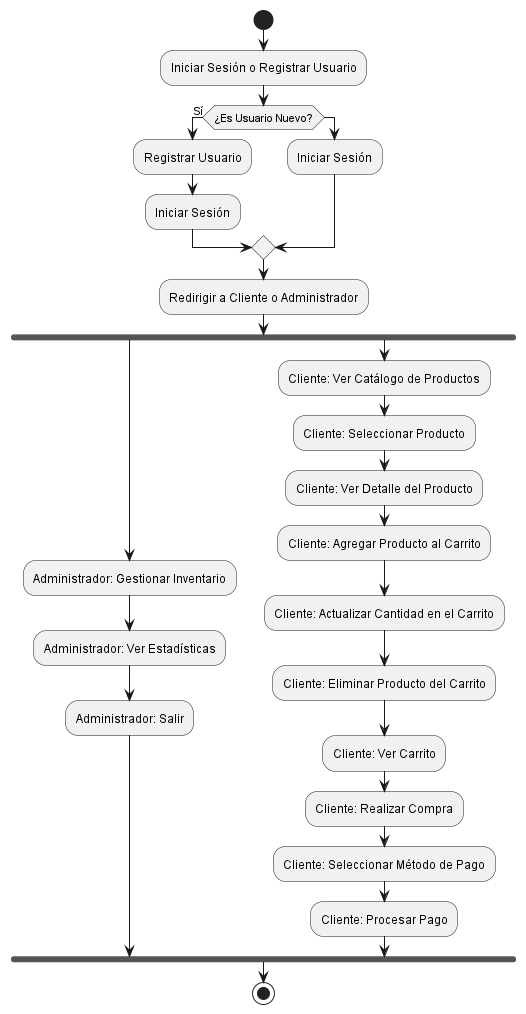
Los **paquetes** agrupan casos de uso relacionados bajo un mismo contexto o módulo funcional. En este diagrama tenemos los siguientes paquetes:

1. **Gestión de Usuario**: Este paquete agrupa los casos de uso relacionados con las acciones que el cliente realiza sobre su cuenta.
   1. **Registrar Usuario (UC8)**: Permite al cliente registrarse en el sistema.
   2. **Autenticar Usuario (UC9)**: Permite al cliente iniciar sesión en el sistema con sus credenciales.
   3. **Actualizar Datos del Usuario (UC10)**: Permite al cliente actualizar su información personal.
   4. **Cambiar Contraseña (UC11)**: Permite al cliente cambiar su contraseña.
2. **Gestión de Inventario**: Este paquete incluye los casos de uso que permiten gestionar el inventario de productos, y son accesibles solo para los administradores.
   1. **Gestionar Inventario (UC1)**: Permite al administrador agregar, eliminar o actualizar productos en el sistema.
   2. **Ver Productos (UC3)**: Permite tanto a administradores como a clientes visualizar el catálogo de productos.
3. **Carrito y Compras**: Este paquete agrupa los casos de uso relacionados con el proceso de compra y el carrito de compras.
   1. **Agregar Producto al Carrito (UC4)**: Permite al cliente agregar productos al carrito de compras.
   2. **Ver Carrito (UC5)**: Permite al cliente ver el contenido de su carrito de compras.
   3. **Realizar Compra (UC6)**: Permite al cliente realizar el proceso de compra de los productos en el carrito.
   4. **Procesar Pago (UC7)**: Permite al cliente procesar el pago por los productos adquiridos.
4. **Estadísticas y Reportes**: Este paquete está dirigido a los administradores y permite ver estadísticas relacionadas con las operaciones del sistema.
   1. **Ver Estadísticas (UC2)**: Permite al administrador visualizar estadísticas del sistema, como ventas, productos más vendidos, etc.

### **Relaciones entre casos de uso y actores:**

* **Cliente:**
  + El **Cliente** puede registrar una cuenta (UC8), autenticarla (UC9), actualizar sus datos (UC10), cambiar su contraseña (UC11), ver los productos disponibles (UC3), agregar productos al carrito (UC4), ver su carrito (UC5), realizar la compra (UC6) y procesar el pago (UC7).
* **Administrador:**
  + El **Administrador** tiene permisos para gestionar el inventario (UC1) y ver estadísticas (UC2).
* **Relaciones de inclusión (<<include>>):**
  + **Gestionar Inventario** (UC1) incluye **Ver Productos** (UC3): Esto indica que el administrador, al gestionar el inventario, puede ver los productos disponibles como parte de su acción.
  + **Agregar Producto al Carrito** (UC4) incluye **Ver Carrito** (UC5): El cliente al agregar un producto al carrito debe ver el carrito, por lo que estas acciones están relacionadas.
  + **Realizar Compra** (UC6) incluye **Procesar Pago** (UC7): Una vez que el cliente ha realizado la compra, debe procesar el pago, por lo que estas acciones se ejecutan de manera secuencial.

### **Diagrama Actividades:**



### **Inicio de sesión o Registro:**

* **Actividad**: El proceso comienza con el **Inicio de sesión** o la opción de **Registrar Usuario**.
* **Decisión**: Si el usuario es nuevo, se registra, de lo contrario, simplemente se inicia sesión.

### **Redirección según el rol:**

* **Redirección**: Después de que el usuario se autentica (ya sea registrado o logueado), el sistema lo redirige al flujo correspondiente según su rol.
* **Posibles roles**: El usuario puede ser un **Administrador** o un **Cliente**.

### **Flujo del Administrador (bifurcación):**

* **Gestión de Inventario**: El **Administrador** puede gestionar el inventario, como añadir, modificar o eliminar productos.
* **Ver Estadísticas**: También tiene acceso a las estadísticas del sistema, como ventas, productos populares, etc.
* **Salir**: Finalmente, el administrador puede cerrar su sesión y salir del sistema.

### **Flujo del Cliente (bifurcación):**

* **Ver Catálogo de Productos**: El **Cliente** tiene la opción de ver el catálogo de productos disponibles en el sistema.
* **Seleccionar Producto**: El cliente puede seleccionar productos específicos de ese catálogo.
* **Ver Detalle del Producto**: Al seleccionar un producto, el cliente puede ver más detalles sobre ese producto, como precio, descripción, imágenes, etc.
* **Agregar Producto al Carrito**: El cliente puede agregar productos al carrito de compras.
* **Actualizar Cantidad en el Carrito**: Si lo desea, puede modificar la cantidad de los productos ya agregados al carrito.
* **Eliminar Producto del Carrito**: También puede eliminar productos del carrito.
* **Ver Carrito**: El cliente puede revisar los productos que tiene en su carrito.
* **Realizar Compra**: El cliente puede proceder a realizar la compra de los productos en su carrito.
* **Seleccionar Método de Pago**: Debe elegir cómo desea pagar (tarjeta de crédito, PayPal, etc.).
* **Procesar Pago**: Finalmente, el cliente procesa el pago de la compra.

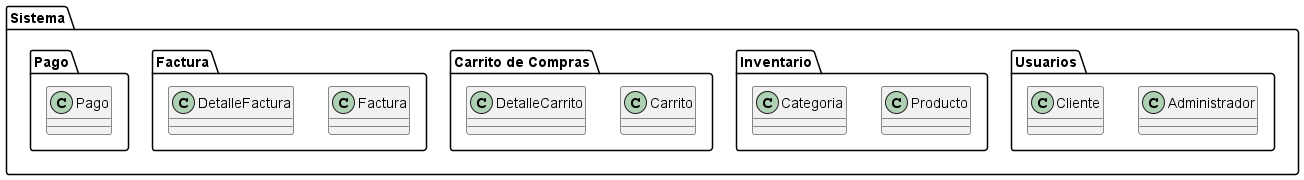
### **Paralelismo (fork y end fork):**

* **fork**: Esta es la parte más importante de este diagrama, ya que representa el inicio de dos actividades paralelas. Después de que el sistema redirige al usuario según su rol (Administrador o Cliente), ambas actividades pueden ejecutarse al mismo tiempo.
* **Administrador y Cliente**: Ambos roles realizan tareas de manera independiente. El **Administrador** tiene su flujo de trabajo que termina con la salida, mientras que el **Cliente** pasa por un proceso más largo de interacción con el catálogo de productos y el carrito.

### **Fin (stop):**

* Después de que las actividades para el **Administrador** y el **Cliente** se completan, el diagrama termina con el nodo de **stop**.

### **Diagrama de Paquetes:**



### **1. Paquete "Sistema"**

El paquete principal que engloba todo el sistema. Es el contenedor de los demás paquetes y define la estructura global de la aplicación. Dentro de este paquete se encuentran los sub-paquetes que organizan las funcionalidades en módulos.

plantuml

Copiar código

package "Sistema" {  
 ...  
}

### **2. Paquete "Usuarios"**

Este paquete contiene las clases que representan a los usuarios del sistema, es decir, las entidades que interactúan con la aplicación. Aquí tenemos dos tipos de usuarios: administradores y clientes.

* **Administrador**: Clase que representa a un usuario con privilegios especiales. Los administradores son responsables de la gestión de inventarios, productos y control de estadísticas.
* **Cliente**: Clase que representa a los usuarios finales que compran productos. Los clientes pueden agregar productos al carrito, realizar pagos, y ver detalles de su cuenta.

**Relaciones**:

* Administrador y Cliente son clases independientes pero dentro de un mismo contexto de **usuarios**.

plantuml

Copiar código

package "Usuarios" {  
 class Administrador  
 class Cliente  
}

### **3. Paquete "Inventario"**

Este paquete agrupa las clases relacionadas con la gestión del inventario de productos. Se encarga de la estructura y organización de los productos dentro del sistema.

* **Producto**: Representa un producto en el inventario. Cada producto tiene propiedades como nombre, descripción, precio, y cantidad disponible.
* **Categoria**: Define categorías a las que pueden pertenecer los productos (por ejemplo, "Electrónica", "Ropa", etc.).

**Relaciones**:

* No hay relaciones directas entre Producto y Categoria en este diagrama, pero un producto puede estar asociado a una categoría.

plantuml

Copiar código

package "Inventario" {  
 class Producto  
 class Categoria  
}

### **4. Paquete "Carrito de Compras"**

Este paquete está dedicado a la funcionalidad de carrito de compras, donde los clientes pueden agregar productos para realizar su compra.

* **Carrito**: Representa el carrito de compras de un cliente. Un carrito puede contener varios productos, y tiene atributos como el total de la compra.
* **DetalleCarrito**: Representa los detalles de los productos dentro del carrito, es decir, cuántos de cada producto están en el carrito.

**Relaciones**:

* El Carrito contiene varios DetalleCarrito. Un carrito puede tener múltiples detalles que representan productos específicos con su cantidad y precio.

plantuml

Copiar código

package "Carrito de Compras" {  
 class Carrito  
 class DetalleCarrito  
}

### **5. Paquete "Factura"**

Este paquete maneja la creación y gestión de facturas una vez que se completa una compra. Después de que un cliente realiza un pago, una factura se genera con los detalles de la compra.

* **Factura**: Representa la factura generada después de la compra. Contiene información sobre la compra, como el total y la fecha.
* **DetalleFactura**: Representa un producto dentro de una factura, similar al concepto de DetalleCarrito. Contiene la cantidad de productos comprados y su precio.

**Relaciones**:

* La relación **Factura "1" \*-- "\*" DetalleFactura** indica que una factura puede tener varios detalles, cada uno representando un producto específico.
* La relación **Producto "1" \*-- "\*" DetalleFactura** indica que un producto puede aparecer en varias facturas.

plantuml

Copiar código

package "Factura" {  
 class Factura  
 class DetalleFactura  
}

### **6. Paquete "Pago"**

Este paquete maneja los pagos realizados por los clientes al completar una compra.

* **Pago**: Representa el pago realizado por un cliente. Incluye información como el monto, el método de pago y el estado de la transacción.

**Relaciones**:

* **Pago** se asocia con un Cliente que realiza el pago. Cada cliente puede realizar múltiples pagos.

plantuml

Copiar código

package "Pago" {  
 class Pago  
}

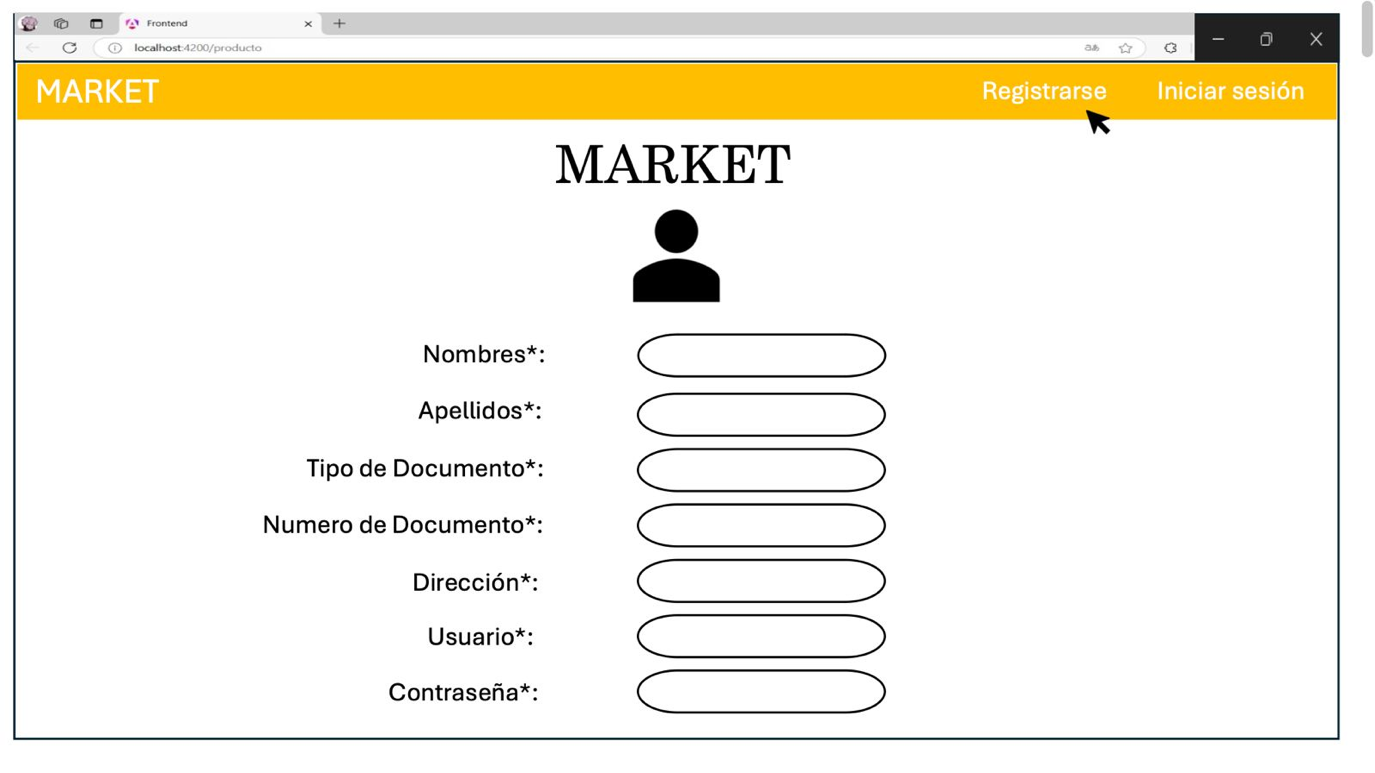
### **Resumen de las Relaciones entre los Paquetes:**

* **Usuarios**: Este paquete define las clases Administrador y Cliente, que representan los roles de los usuarios dentro del sistema.
* **Inventario**: Gestiona los productos y categorías, lo que permite a los administradores organizar y modificar el inventario.
* **Carrito de Compras**: Permite a los clientes gestionar su carrito, agregando productos y determinando la cantidad deseada.
* **Factura**: Genera facturas basadas en las compras realizadas, asociando productos a detalles de facturación.
* **Pago**: Se encarga de procesar los pagos realizados por los clientes, vinculados con las compras realizadas.

## **Prototipo o Mockup.**

El propósito de este mockup es proporcionar una representación visual e interactiva de la interfaz de usuario del sistema de **Carrito de Compras e Inventario**. Este diseño muestra cómo se presentarán los productos y las interacciones entre los usuarios (clientes y administradores) dentro de la plataforma. El mockup permite visualizar de manera anticipada la disposición de los elementos en la pantalla, la estructura del sistema y cómo los usuarios interactuarán con las diversas funcionalidades del sistema.

**1. Registro de Usuario:**



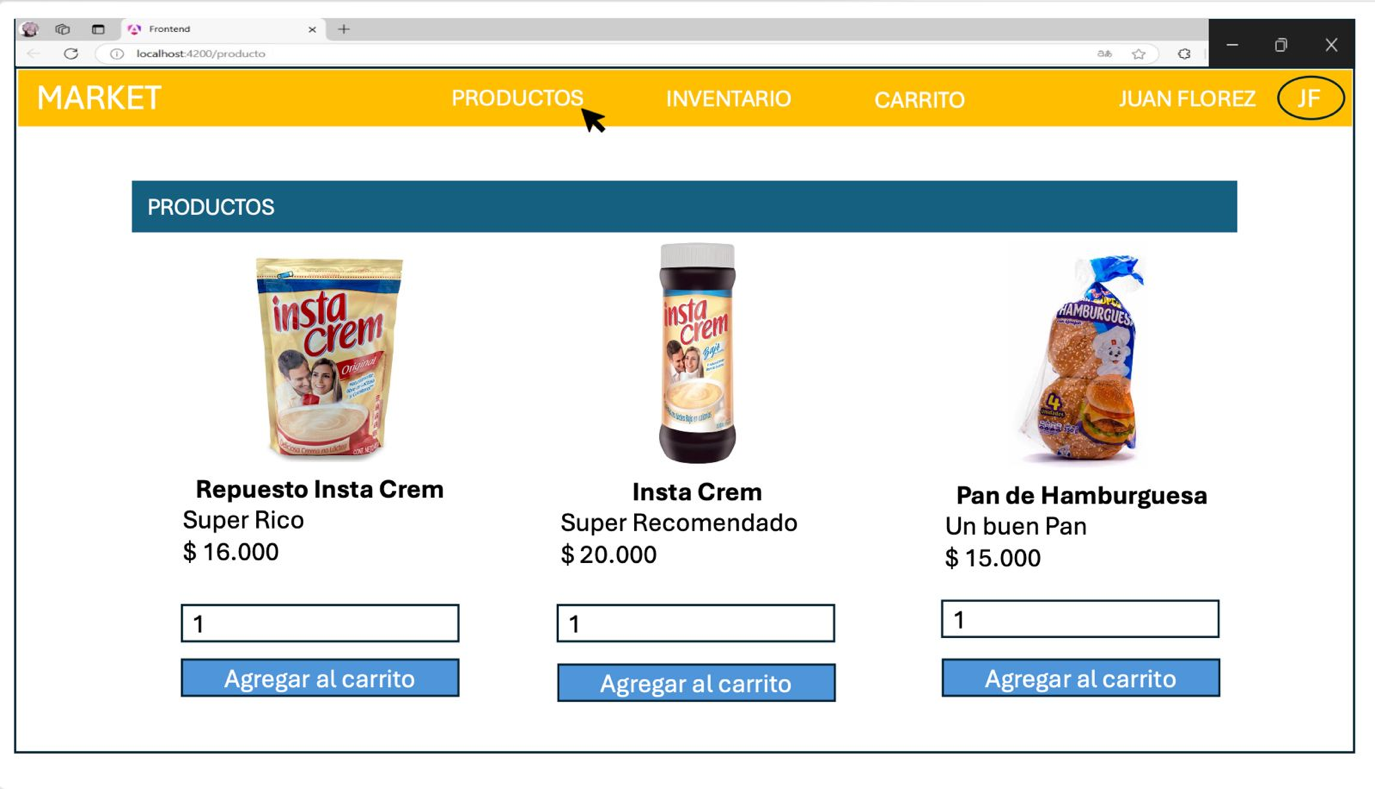
* La sección de **registro de usuario** permite a nuevos usuarios crear una cuenta para poder acceder a las funcionalidades del sistema. Al acceder a la pantalla de registro, los usuarios deben completar un formulario con los siguientes campos:
* **Nombre completo**.
* **Tipo de Documento (**para comunicaciones y validación)
* **Numero de Documento (**para comunicaciones y validación)
* **Dirección** (si es relevante para la compra, como en el caso de envíos).
* **Usuario** (para seguridad en el acceso).
* **Contraseña** (para seguridad en el acceso).
* Una vez que el usuario ingresa los datos, puede hacer clic en el botón **"Registrar"**, y el sistema validará los datos. Si todo es correcto, el usuario recibirá un mensaje de confirmación indicando que su cuenta ha sido creada exitosamente y podrá proceder a iniciar sesión.

### **2. Inicio de Sesión:**



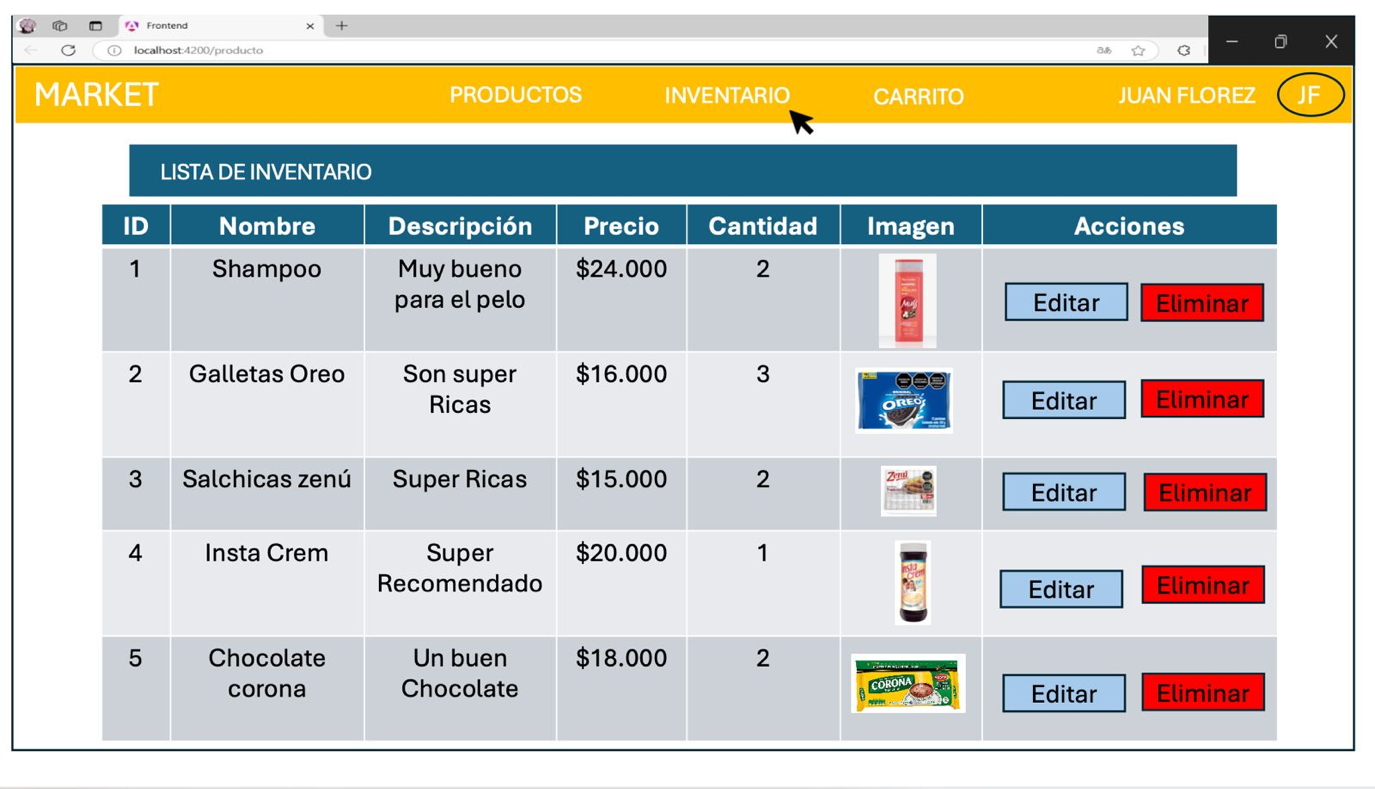
* En la pantalla de **inicio de sesión**, los usuarios ya registrados ingresan su **Usuario** y **contraseña** para acceder a su cuenta. El formulario está diseñado de forma sencilla y clara, con botones de **"Iniciar sesión".**
* Si los datos son correctos, el sistema redirige al usuario a la página principal del sistema, mostrando el **catálogo de productos** o el **panel de administración**, dependiendo del rol (Cliente o Administrador).

### **3. Catálogo de Productos (Pestaña Productos):**



* En la **pestaña de productos**, los usuarios pueden explorar todos los productos disponibles en la tienda. Esta sección muestra una **rejilla de productos**, donde cada producto incluye:
* **Imagen del producto**.
* **Nombre del producto**.
* **Descripción del Producto.**
* **Precio unitario**.
* Un botón de **"Agregar al Carrito"** para añadir el producto a su carrito de compras.

### **4. Inventario (Pestaña Inventario):**



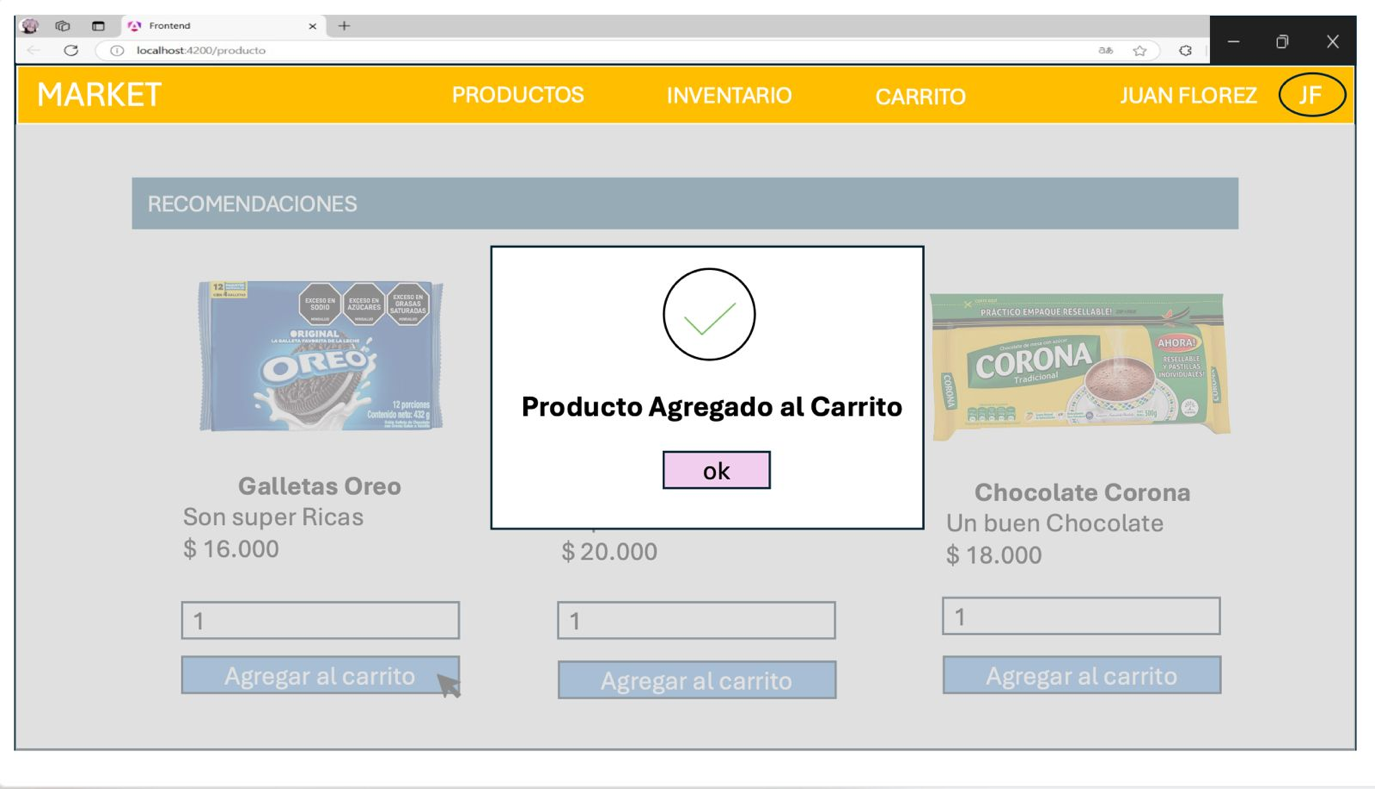
* La **pestaña de inventario** es accesible solo para los **administradores**, quienes pueden gestionar los productos disponibles. Esta sección muestra una lista de todos los productos almacenados en el sistema, incluyendo:
* **Nombre del producto**.
* **Precio**.
* **Cantidad en stock**.
* **Imagen del Producto.**
* Los administradores tienen opciones para:
* **Agregar un nuevo producto**, lo que les permite ingresar nombre, precio, cantidad y categoría.
* **Editar productos existentes**, actualizando datos como precio o cantidad.
* **Eliminar productos** que ya no estén disponibles o que hayan sido descontinuados.
* Esta sección está diseñada para facilitar la gestión del inventario y mantener actualizados los productos disponibles en la tienda.

### **5. Imagen de Agregado con Éxito:**



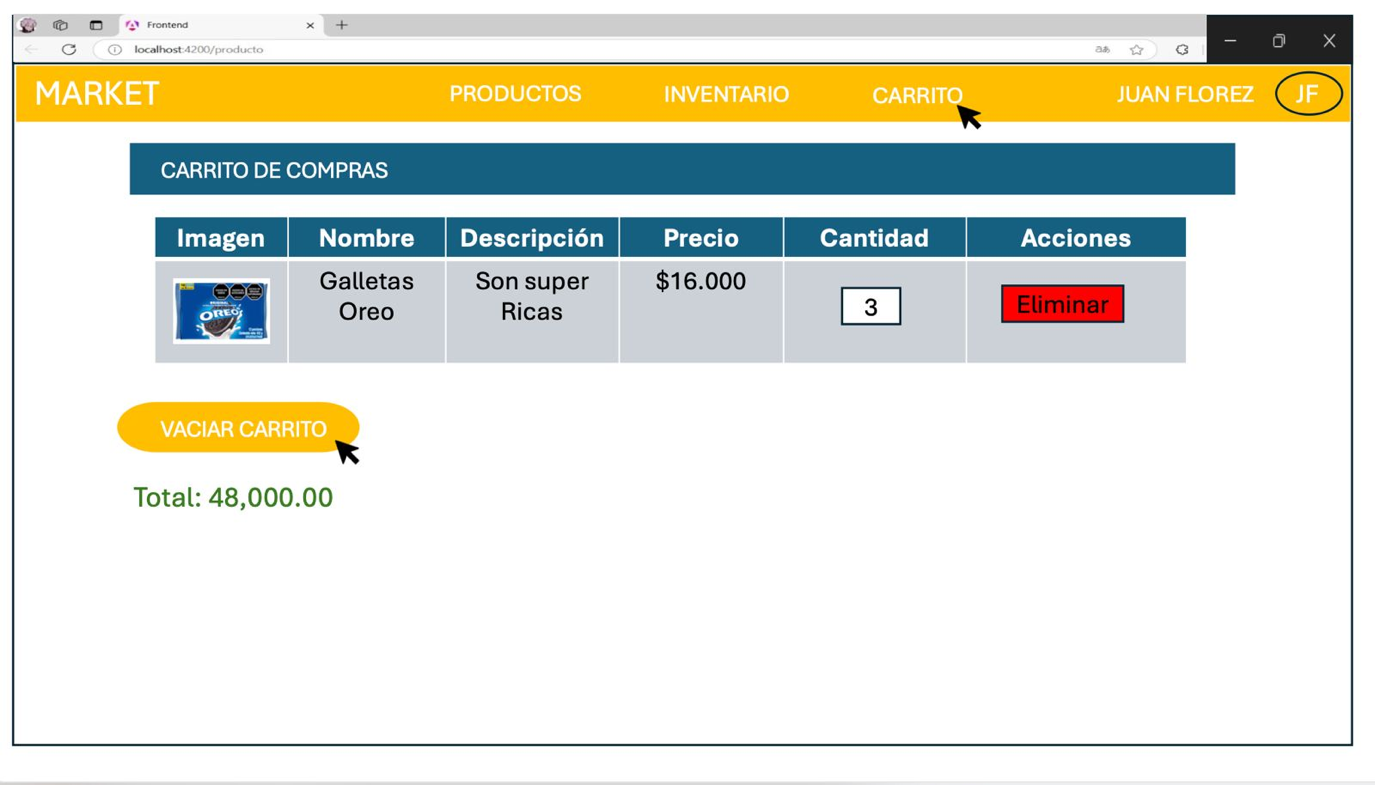
* Cuando un producto es exitosamente agregado al carrito, el sistema muestra una **notificación visual** (como un mensaje emergente o una pequeña imagen) que confirma que el producto ha sido añadido correctamente al carrito. Esta notificación suele incluir un mensaje como:
* **"Producto agregado al carrito"**.
* Un **ícono o imagen de éxito**, como un check verde o una miniatura del producto.
* Esto proporciona al usuario una confirmación inmediata de que su acción fue exitosa y facilita la experiencia de compra al hacerla más clara y transparente.

### **6. Desde Productos Agregado al Carrito (Pestaña Productos):**



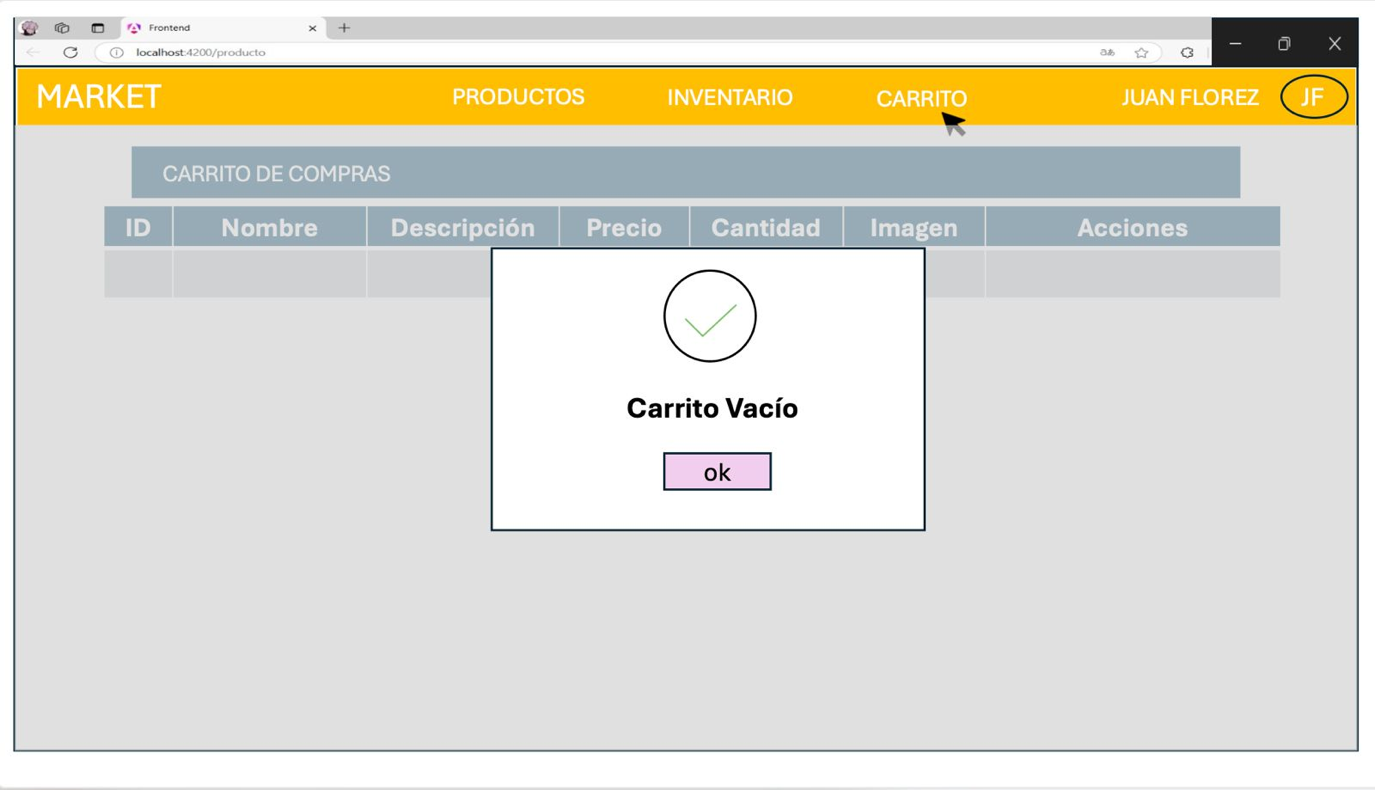
* Al hacer clic en el botón **"Agregar al Carrito"** en la **pestaña de productos**, el producto se añade al **Carrito de Compras** del usuario. El sistema muestra una **notificación** de que el producto ha sido añadido y puede actualizar la visualización del carrito en tiempo real, permitiendo que el usuario vea cuántos productos tiene en su carrito sin necesidad de ir a la pestaña del carrito.

### **7. Carrito con los Productos Agregados (Pestaña Carrito):**



* En la **pestaña de carrito**, los usuarios pueden ver todos los productos que han agregado a su carrito de compras. Esta sección incluye:
* **Nombre del producto**.
* **Cantidad seleccionada**.
* **Precio unitario**.
* **Total por producto** (precio por cantidad).
* **Total general** de la compra.
* Los usuarios pueden modificar la **cantidad** de cada producto directamente en el carrito o **eliminar productos** que ya no deseen. Al final de la página, hay un botón para proceder al **pago**.

### **8. Eliminar Producto o Vaciar Carrito (En la Pestaña Carrito):**



* Dentro de la **pestaña de carrito**, los usuarios tienen la opción de **eliminar productos** del carrito o **vaciar completamente el carrito**. Esto puede hacerse con botones dedicados junto a cada producto, como:
* **Eliminar**: Borra el producto seleccionado del carrito.
* **Vaciar Carrito**: Elimina todos los productos del carrito a la vez.
* La interacción es fácil de realizar y está diseñada para permitir que el usuario administre su carrito de manera eficiente antes de proceder al pago.