

**Universidad ORT Uruguay Facultad  
de Ingeniería  
Escuela de Tecnología**

**OBLIGATORIO PROGRAMACIÓN 1**  
**documentación**



**[Facundo Robayna – 345655]**



**[Alan Langelan – 338517]**

**[M1E]**

**Docente: [Sergio Palay]**

**[Analista en Tecnologías de la Información]**

**[Fecha de entrega del documento (20-06-2024)]**

## **Índice**

Documento de análisis .....	4
Descripción General del Problema a Resolver .....	4
Tipos de Usuario del Sistema .....	4
Listado de Funcionalidades .....	5
Funcionalidades para Comprador .....	5
Funcionalidades para Administrador .....	5
Detalle de Funcionalidades .....	5
2.1 F01 – Registro de Comprador.....	6
2.2 F02 – Ingreso al Sistema.....	7
2.3 F03     Compra de Productos .....	9
2.4 F04     Ver Listado de Compras .....	11
2.5 F05 – Ver Productos en Oferta.....	12
2.6 F06     Ingreso al Sistema .....	13
2.7 F07     Listado y Aprobación de Compras.....	14
2.8 F08     Crear Productos.....	15
2.9 F09     Administrar Productos .....	16
2.10 F10 – Ver Informe de Ganancias .....	18

Casos de prueba .....	19
F-01: Registro de Comprador .....	19
02: Ingreso al Sistema.....	20
F-03: Compra de Productos .....	20
04: Listado de Compras .....	21
F-05: Ver Productos en Oferta .....	22
07: Listado y Aprobación de Compras .....	23
08: Crear Producto .....	24
09: Administrar Productos .....	25
10: Informe de Ganancias .....	26
Código.....	28
HTML.....	28
JS.....	33
CSS .....	87
Listado de información precargada.....	113
Administradores precargados .....	113
Compradores precargados.....	113
Productos precargados .....	114
Compras precargadas .....	115

## **Documento de análisis**

### **Descripción General del Problema a Resolver**

La tienda de artículos deportivos quiere vender sus productos en línea porque actualmente solo lo hace en su local físico. Necesitan un sistema de e-commerce para lograrlo. La idea es crear una plataforma donde los clientes puedan comprar artículos deportivos y los administradores puedan gestionar el inventario y las ventas.

### **Tipos de Usuario del Sistema**

1. Comprador
2. Administrador

# **Listado de Funcionalidades**

## **Funcionalidades para Comprador**

- F01 – Registro de Comprador – Usuario/s: Comprador
- F02 – Ingreso al Sistema – Usuario/s: Comprador
- F03 – Compra de Productos – Usuario/s: Comprador
- F04 – Ver Listado de Compras – Usuario/s: Comprador
- F05 – Ver Productos en Oferta – Usuario/s: Comprador

## **Funcionalidades para Administrador**

- F06 – Ingreso al Sistema – Usuario/s: Administrador
- F07 – Listado y Aprobación de Compras – Usuario/s: Administrador
- F08 – Crear Productos – Usuario/s: Administrador
- F09 – Administrar Productos – Usuario/s: Administrador
- F10 – Ver Informe de Ganancias – Usuario/s: Administrado

# **Detalle de Funcionalidades**

## 2.1 F01 – Registro de Comprador

### 2.1.1.

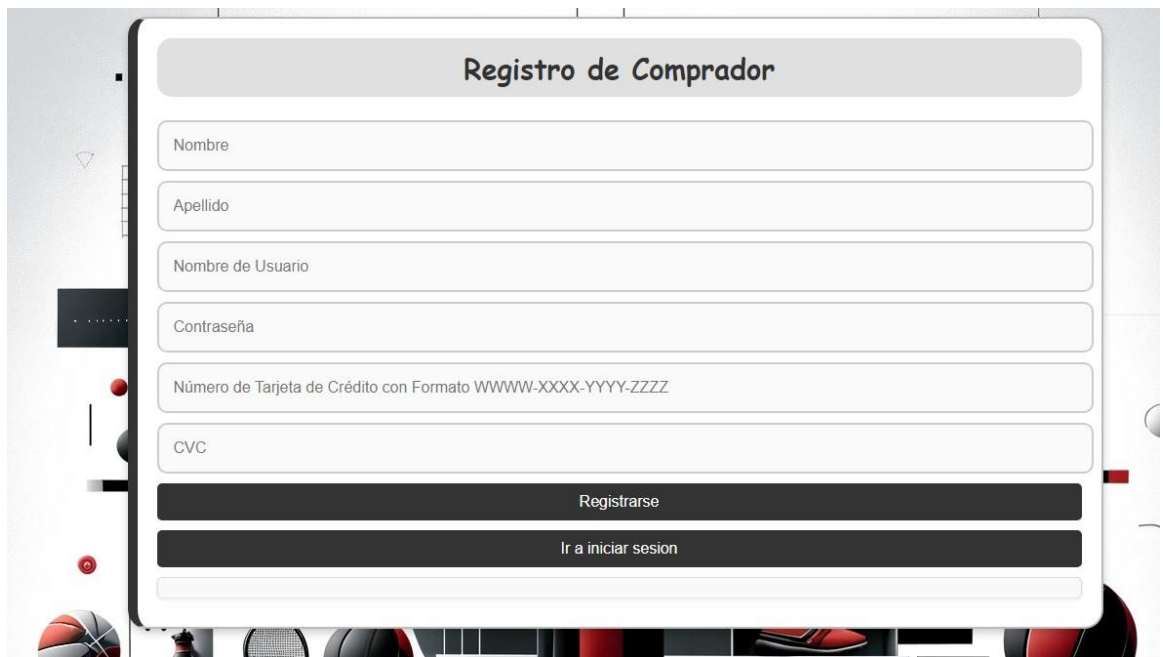
Acceso: Comprador

### 2.1.2.

Descripción: Permite a un nuevo usuario registrarse como comprador en la plataforma. Al completar el registro, el usuario recibe un crédito inicial de 3000 pesos que puede utilizar para realizar compras en la tienda.

### 2.1.3.

Interfaz de Usuario:



The image shows a user interface for a buyer registration form. The form is titled "Registro de Comprador" and is displayed on a screen. It contains several input fields for user information: "Nombre", "Apellido", "Nombre de Usuario", "Contraseña", "Número de Tarjeta de Crédito con Formato WWW-XXXX-YYYY-ZZZZ", and "CVC". Below the input fields are two buttons: "Registrarse" and "Ir a iniciar sesión". The form is set against a background that appears to be a sports-related image.

### 2.1.4.

Validaciones:

- Nombre: Obligatorio.

- Apellido: Obligatorio.
- Nombre de usuario: Obligatorio, único, case insensitive.
- Mensaje de error: "El nombre de usuario ya está en uso."
- Contraseña: Obligatorio, mínimo 5 caracteres, al menos una mayúscula, una minúscula y un número.
- Mensaje de error: "La contraseña debe tener al menos 5 caracteres, una mayúscula, una minúscula y un número."
- Número de tarjeta de crédito: Obligatorio, formato WWW-XXXX-YYYYZZZZ, validación Luhn.
- Mensaje de error: "El número de tarjeta de crédito no es válido."
- CVC: Obligatorio, 3 dígitos.
- Mensaje de error: "El CVC debe tener 3 dígitos."

## **2.2 F02 – Ingreso al Sistema**

### **2.2.1.**

Acceso: Comprador

### 2.2.2.

Descripción: Permite a los usuarios compradores registrados ingresar al sistema.

### 2.2.3.

Interfaz de Usuario:



### 2.2.4.

Validaciones:

- Nombre de usuario (Obligatorio)
- Mensaje de error: "Usuario incorrecto"
- Contraseña (Obligatorio)
- Mensaje de error: "Contraseña incorrecta"



## 2.3 F03 Compra de Productos

### 2.3.1.










Acceso: Comprador

### 2.3.2.

Descripción: Permite a los compradores adquirir productos disponibles en el catálogo.

### 2.3.3.

Interfaz de Usuario:

Compra de Productos								
Ver Productos								
ID Producto	Imagen	Nombre	Descripción	Precio	Stock	Cantidad	Oferta	Comprar
Prod_ID_0		Campera de Nacional	Campera del club mas grande del mundo	5000	5	<input type="text"/>	No	<button>Comprar</button>
Prod_ID_2		Short de Nacional	Short del club mas grande del mundo	1299	12	<input type="text"/>	No	<button>Comprar</button>
Prod_ID_3		Gorro de Nacional	Gorro del club mas grande del mundo	790	8	<input type="text"/>	Si	<button>Comprar</button>
Prod_ID_4		Medias de Nacional	Medias del club mas grande del mundo	350	6	<input type="text"/>	Si	<button>Comprar</button>
Prod_ID_5		Remera de Nacional	Remera del club mas grande del mundo	2500	10	<input type="text"/>	No	<button>Comprar</button>
Prod_ID_6		Buzo de Nacional	Buzo del club mas grande del mundo	4500	7	<input type="text"/>	No	<button>Comprar</button>
Prod_ID_7		Pantalón de Nacional	Pantalón del club mas grande del mundo	3200	4	<input type="text"/>	No	<button>Comprar</button>
Prod_ID_8		Camiseta de Nacional	Camiseta del club mas grande del mundo	2800	15	<input type="text"/>	No	<button>Comprar</button>
Prod_ID_9		Botines de Nacional	Botines del club mas grande del mundo	6000	3	<input type="text"/>	No	<button>Comprar</button>

#### **2.3.4.**

Validaciones:

- Cantidad a comprar: Obligatorio, mayor que 0.
- Mensaje de error: "Debe ingresar una cantidad válida."

## 2.4 F04 Ver Listado de Compras

### 2.4.1.

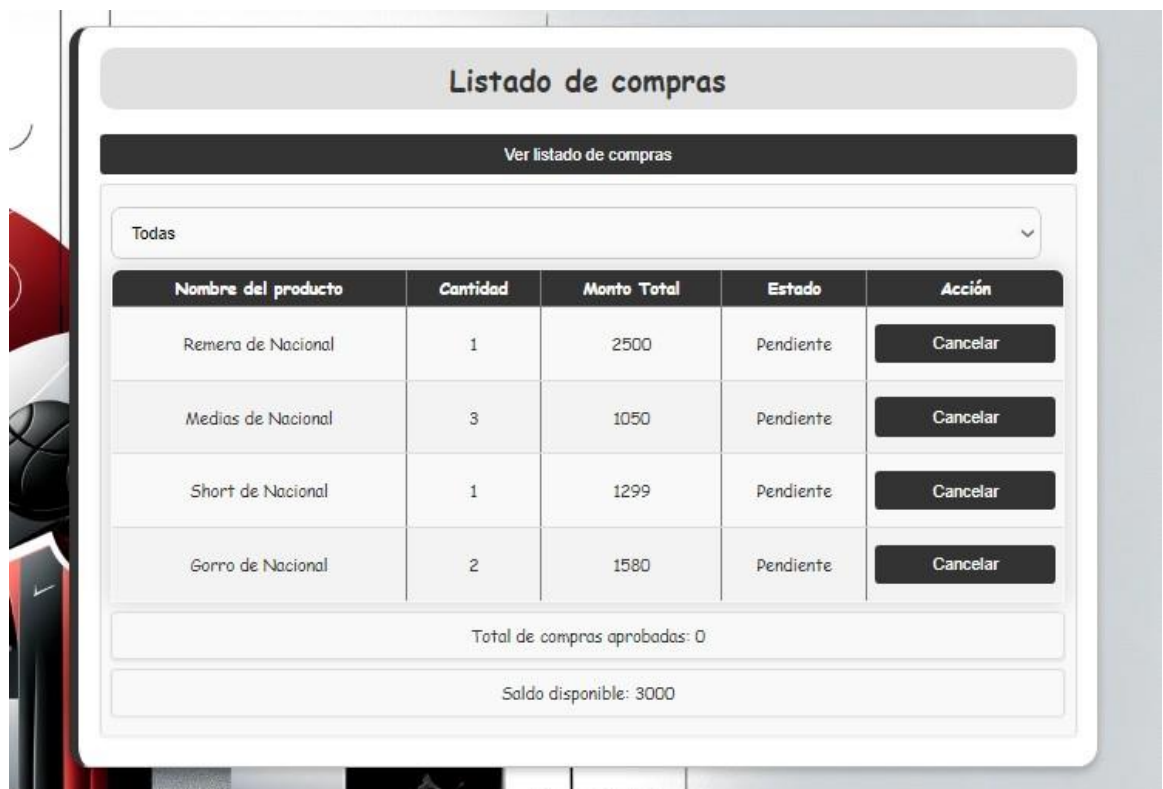
Acceso: Comprador

### 2.4.2.

Descripción: Permite a los compradores ver sus compras realizadas y el estado de cada una.

### 2.4.3.

Interfaz de Usuario:



### 2.4.4.

Validaciones:

- Solo mostrar botón de cancelar para compras pendientes.

## 2.5 F05 – Ver Productos en Oferta

### 2.5.1.

Acceso: Comprador

### 2.5.2.

Descripción: Muestra a los compradores únicamente los productos en oferta.

### 2.5.3.

Interfaz de Usuario:

Ver Productos en Oferta								
Ver productos en oferta								
ID Producto	Imagen	Nombre	Descripción	Precio	Stock	Cantidad	Oferta	Comprar
Prod_ID_3		Gorro de Nacional	Gorro del club mas grande del mundo	790	8	<input type="text"/>	Si	<button>Comprar</button>
Prod_ID_4		Medias de Nacional	Medias del club mas grande del mundo	350	6	<input type="text"/>	Si	<button>Comprar</button>

### 2.5.4.

Validaciones:

- Cantidad a comprar: Obligatorio, mayor que 0.
- Mensaje de error: "Debe ingresar una cantidad válida."

## 2.6 F06 Ingreso al Sistema

### 2.6.1.

Acceso: Administrador

### 2.6.2.

Descripción: Permite a los administradores ingresar al sistema.

### 2.6.3.

Interfaz de Usuario:



### 2.6.4.

Validaciones:

- Nombre de usuario: Obligatorio, case insensitive. - Mensaje de error: "El nombre de usuario no existe."
- Contraseña: Obligatorio, case sensitive.
- Mensaje de error: "Contraseña incorrecta."

## 2.7 F07 Listado y Aprobación de Compras

### 2.7.1.

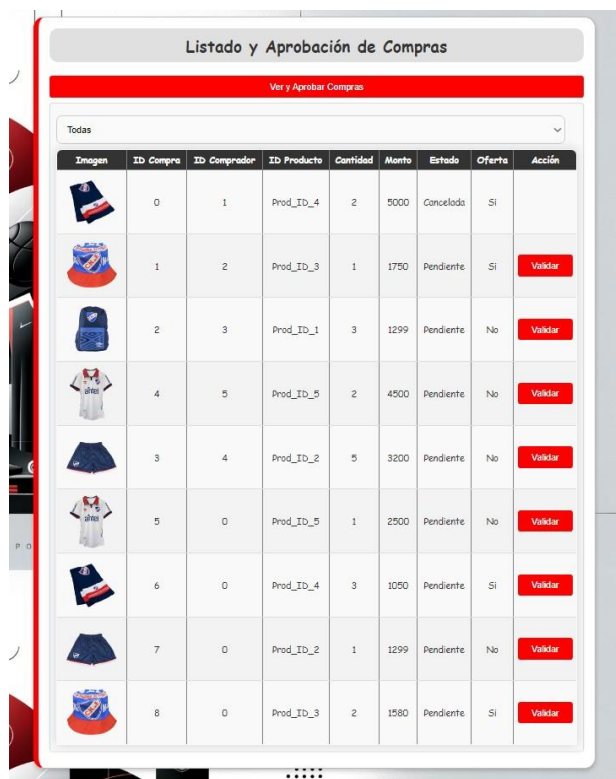
Acceso: Administrador










### 2.7.2.

Descripción: Permite a los administradores ver y aprobar o cancelar las compras realizadas por los compradores.

### 2.7.3.

Interfaz de Usuario:



Listado y Aprobación de Compras								
Ver y Aprobar Compras								
Todas								
Imagen	ID Compra	ID Comprador	ID Producto	Cantidad	Monto	Estado	Oferta	Acción
	0	1	Prod_ID_4	2	5000	Concloda	Si	
	1	2	Prod_ID_3	1	1750	Pendiente	Si	Validar
	2	3	Prod_ID_1	3	1299	Pendiente	No	Validar
	4	5	Prod_ID_5	2	4500	Pendiente	No	Validar
	3	4	Prod_ID_2	5	3200	Pendiente	No	Validar
	5	0	Prod_ID_5	1	2500	Pendiente	No	Validar
	6	0	Prod_ID_4	3	1050	Pendiente	Si	Validar
	7	0	Prod_ID_2	1	1299	Pendiente	No	Validar
	8	0	Prod_ID_3	2	1580	Pendiente	Si	Validar

### 2.7.4.

Validaciones:

- Verificar saldo y stock del producto antes de aprobar.

- Mensaje de error: "Saldo insuficiente" o "Stock insuficiente."

## 2.8 F08 Crear Productos

### 2.8.1.

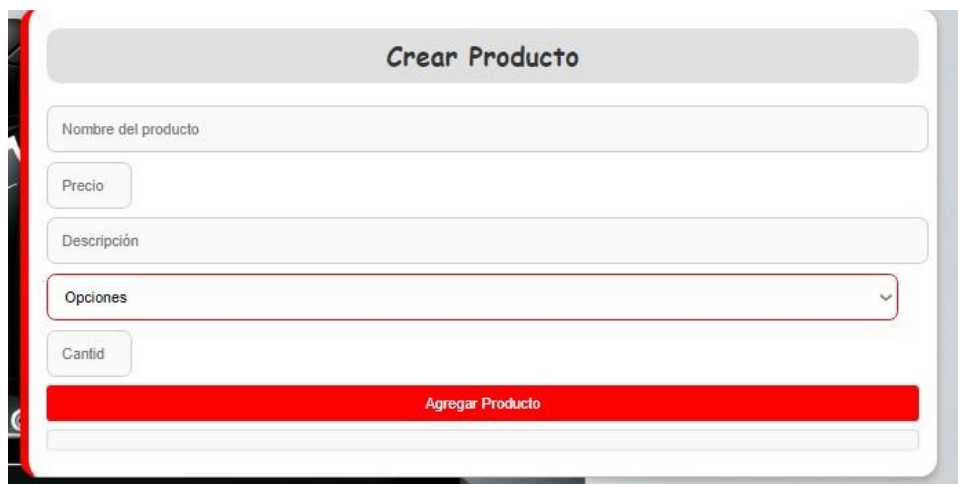
Acceso: Administrador

### 2.8.2.

Descripción: Permite a los administradores agregar nuevos productos al catálogo.

### 2.8.3.

Interfaz de Usuario:



The screenshot shows a web form titled "Crear Producto" (Create Product). The form contains several input fields: "Nombre del producto" (Product Name), "Precio" (Price), "Descripción" (Description), "Opciones" (Options) with a dropdown arrow, and "Cantidad" (Quantity). At the bottom of the form is a prominent red button labeled "Agregar Producto" (Add Product).

### 2.8.4.

Validaciones:

- Todos los campos: Obligatorios.
- Mensaje de error: "Este campo es obligatorio."
- Precio: Mayor que 0.

- 
- Mensaje de error: "El precio debe ser mayor que 0." - Stock: Mayor que 0.
- Mensaje de error: "El stock debe ser mayor que 0."

## **2.9 F09    Administrar Productos**

### **2.9.1.**

Acceso: Administrador











### **2.9.2.**

Descripción: Permite a los administradores modificar el stock, estado y oferta de los productos.

### **2.9.3.**

Interfaz de Usuario:



Administrar Productos								
Administrar Productos								
ID Producto	Imagen	Nombre	Descripción	Precio	Stock	Estado	Oferta	Acción
Prod_ID_0		Campera de Nacional	Campera del club mas grande del mundo	5000	<input type="text" value="5"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_1		Mochila de Nacional	Mochila del club mas grande del mundo	1750	<input type="text" value="9"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_2		Short de Nacional	Short del club mas grande del mundo	1299	<input type="text" value="12"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_3		Gorro de Nacional	Gorro del club mas grande del mundo	790	<input type="text" value="8"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_4		Medias de Nacional	Medias del club mas grande del mundo	350	<input type="text" value="6"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_5		Remera de Nacional	Remera del club mas grande del mundo	2500	<input type="text" value="10"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_6		Buzo de Nacional	Buzo del club mas grande del mundo	4500	<input type="text" value="7"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_7		Pantalón de Nacional	Pantalón del club mas grande del mundo	3200	<input type="text" value="4"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_8		Camiseta de Nacional	Camiseta del club mas grande del mundo	2800	<input type="text" value="15"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>
Prod_ID_9		Botines de Nacional	Botines del club mas grande del mundo	6000	<input type="text" value="3"/>	<input type="button" value="v"/>	<input type="button" value="v"/>	<a href="#">Modificar</a>

#### 2.9.4.

Validaciones:

- Stock: Mayor que 0.
- Mensaje de error: "El stock debe ser mayor que 0."

### 2.10 F10 – Ver Informe de Ganancias

#### 2.10.1.

Acceso: Administrador

#### 2.10.2.

Descripción: Permite a los administradores ver un informe de las ganancias y unidades vendidas.

#### 2.10.3.

Interfaz de Usuario:



## Casos de prueba

### F-01: Registro de Comprador

**Precondiciones:** El usuario no debe estar registrado.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Fall a, P=pasa )
F01-T01	Registro correcto de comprador	1. Ingresa nombre. 2. Ingresa apellido. 3. Ingresa nombre de usuario. 4. Ingresa contraseña. 5. Ingresa número de tarjeta. 6. Ingresa CVC. 7. Presiona "Registrarse"	Nombre: Juan Apellido: Pérez Nombre usuario: jperez Contraseña: Password@123 Tarjeta: 5555-5555-5555-4444 CVC: 333	El usuario es registrado correctamente y puede iniciar sesión	El usuario es registrado correctamente y puede iniciar sesión.	P
F01-T02	Registro con nombre vacío	1. Deja el campo de nombre vacío. 2. Completa los otros campos. 3. Presiona "Registrarse".	Nombre: (vacío) Apellido: Pérez Nombre usuario: jperez Contraseña: Password@123 Tarjeta: 5555-5555-5555-4444 CVC: 333	Muestra mensaje de error indicando que el nombre no puede estar vacío..	Muestra mensaje de error indicando que el nombre no puede estar vacío	F

## 02: Ingreso al Sistema

**Precondiciones:** El usuario debe estar registrado.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F02-T01	Inicio de sesión correcto	1. Ingresa nombre de usuario. 2. Ingresa contraseña. 3. Presiona "Ingresar"	Nombre usuario: jperez Contraseña: Password@123	El usuario es autenticado correctamente y se muestra su perfil.	El usuario es autenticado correctamente y se muestra su perfil	P
F02-T02	Inicio de sesión con contraseña incorrecta	1. Ingresa nombre de usuario. 2. Ingresa contraseña incorrecta. 3. Presiona "Ingresar".	Nombre usuario: jperez Contraseña: wrongpass@123	Muestra mensaje de error indicando que la contraseña es incorrecta.	Muestra mensaje de error indicando que la contraseña es incorrecta.	F

## F-03: Compra de Productos

**Precondiciones:** El usuario debe estar autenticado y deben existir productos activos.

F-

ID	Escenario de Test	Pasos	Datos Utilizados	Resultado Esperado	Resultado Obtenido	Estado
F03-T0	Compra de producto correcto	1. Selecciona producto. 2. Ingresa cantidad válida. 3. Presiona "Comprar".	Producto: Campera de Nacional Cantidad: 1	La compra es realizada con éxito.	La compra es realizada con éxito.	P
F03-T02	Compra de producto con cantidad inválida	1. Selecciona producto. 2. Ingresa cantidad inválida. 3. Presiona "Comprar"	Producto: Campera de Nacional Cantidad: -1	Muestra mensaje de error.	Muestra mensaje de error.	F

#### 04: Listado de Compras

**Precondiciones:** El usuario debe estar autenticado y debe haber realizado compras

ID	Escenario de Test	Pasos	Datos Utilizados	Resultado Esperado	Resultado Obtenido	Estado
----	-------------------	-------	------------------	--------------------	--------------------	--------

F04-T0	Listar compras del usuario	1. Inicia sesión. 2. Presiona "Ver listado de compras"	Usuario: jperez	Se muestra el listado de compras del usuario autenticado.	Se muestra el listado de compras del usuario autenticado	P
F04-T02	Error al listar compras	1. Inicia sesión. 2. Presiona "Ver listado de compras".	Usuario: jperez	Muestra mensaje de error al listar las compras.	Muestra mensaje de error al listar las compras	F

#### F-05: Ver Productos en Oferta

**Precondiciones:** El usuario debe estar autenticado y deben existir productos en oferta.

ID	Escenario de Test	Pasos	Datos Utilizados	Resultado Esperado	Resultado Obtenido	Estado
F05-T01	Ver productos en oferta	1. Inicia sesión. 2. Presiona "Ver productos en oferta"	Usuario: jperez	Se muestra el listado de productos en oferta.	Se muestra el listado de productos en oferta	P
F05-T02	Error al ver productos en oferta	1. Inicia sesión. 2. Presiona "Ver productos en oferta"	Usuario: jperez	Muestra mensaje de error al ver los productos en oferta.	Se muestra el listado de productos en oferta (funciona bien, nunca da error)	P

**F-**

**07: Listado y Aprobación de Compras**

**Precondiciones:** El usuario debe estar autenticado como administrador y deben existir compras pendientes.

ID	Escenario de Test	Pasos	Datos Utilizados	Resultado Esperado	Resultado Obtenido	Estado
F07-T01	Ver y aprobar compras pendientes	1. Inicia sesión como administrador. 2. Presiona "Ver y Aprobar Compras"	Administrador: fRobayna	Se muestra el listado de compras pendientes y se pueden aprobar.	Se muestra el listado de compras pendientes y se pueden aprobar	P
F07-T02	Error al aprobar compras pendientes	1. Inicia sesión como administrador. 2. Presiona "Ver y Aprobar Compras"	Administrador: fRobayna	Muestra mensaje de error al aprobar compras pendientes.	Se muestra el listado de compras pendientes y se pueden aprobar (funciona bien, nunca da error)	P

F-

### 08: Crear Producto

**Precondiciones:** El usuario debe estar autenticado como administrador.

ID	Escenario de Test	Pasos	Datos Utilizados	Resultado Esperado	Resultado Obtenido	Estado
F08T01	Crear producto correctamente	1. Inicia sesión como administrador. 2. Ingresar nombre del producto. 3. Ingresar precio. 4. Ingresar descripción. 5. Selecciona imagen. 6. Ingresar cantidad de stock. 7. Presiona "Agregar Producto"	Nombre: Campera Nacional Precio: 5000 Descripción: Campera del club más grande del mundo Imagen: camperonNacional.png Stock: 5	El producto es creado correctamente y se muestra en el listado de productos.	El producto es creado correctamente y se muestra en el listado de productos.	P



**F-**

F08T02	Crear producto con datos inválidos	1. Inicia sesión como administrador. 2. Ingresar nombre del producto vacío. 3. Ingresar precio negativo. 4. Ingresar descripción vacía. 5. No seleccionar imagen. 6. Ingresar cantidad de stock negativa. 7. Presionar "Agregar Producto".	Nombre: (vacío) Precio: -5000 Descripción: (vacío) Imagen: opciones Stock: -5	Muestra mensaje de error indicando que los datos del producto son inválidos.	Muestra mensaje de error indicando que los datos del producto son inválidos.	F
--------	------------------------------------	--	---	--	--	---

## 09: Administrar Productos

**Precondiciones:** El usuario debe estar autenticado como administrador.

ID	Escenario de Test	Pasos	Datos Utilizados	Resultado Esperado	Resultado Obtenido	Estado
----	-------------------	-------	------------------	--------------------	--------------------	--------

**F-**

F09-T01	Administrar productos correctamente	1. Inicia sesión como administrador. 2. Presiona "Administrar Productos". 3. Modifica stock. 4. Modifica estado. 5. Modifica oferta. 6. Presiona "Modificar".	Nombre: Campera Nacional Nuevo Stock: 10 Nuevo Estado: pausado Nueva Oferta: true	El producto es modificado correctamente y se reflejan los cambios	El producto es modificado correctamente y se reflejan los cambios	P
F09-T02	Error al administrar productos	1. Inicia sesión como administrador. 2. Presiona "Administrar Productos".	Nombre: Campera Nacional Nuevo Stock: -10 Nuevo Estado: (vacío) Nueva Oferta: (vacío)	Muestra mensaje de error al modificar los productos.	Muestra mensaje de error al modificar los productos.	F

### **10: Informe de Ganancias**

**Precondiciones:** El usuario debe estar autenticado como administrador.

**F-**

ID	Escenario de Test	Pasos	Datos Utilizados	Resultado Esperado	Resultado Obtenido	Estado
F10-T01	Ver informe de ganancias correctamente	1. Inicia sesión como administrador. 2. Presiona "Informe de Ganancias"	Administrador: fRobayna	Se muestra el informe de ganancias correctamente	Se muestra el informe de ganancias correctamente	P
F10-T02	Error al ver informe de ganancias	1. Inicia sesión como administrador. 2. Presiona "Informe de Ganancias".	Administrador: fRobayna	Muestra mensaje de error al ver el informe de ganancias	Se muestra el informe de ganancias correctamente (funciona bien, nunca da error)	P

## Código

### HTML

```
<!DOCTYPE html>
<html lang="es">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Tienda deportiva</title>
  <link rel="stylesheet" href="estilos.css">
</head>

<body>

  <button id="btnCerrarSesion">Cerrar Sesión</button>
<button id="btnCarrito"></button>
  <button id="btnFlechaAtras"></button>

  <section id="F01">
    <h2>Registro de Comprador</h2>
    <input type="text" id="txtNombre" placeholder="Nombre">
    <div id="errorNombre" class="error"></div>
    <input type="text" id="txtApellido" placeholder="Apellido">
    <div id="errorApellido" class="error"></div>
    <input type="text" id="txtNombreUsuario" placeholder="Nombre de
Usuario">
    <div id="errorUsuario" class="error"></div>
    <input type="password" id="passContrasena"
placeholder="Contraseña">
    <div id="errorContrasena" class="error"></div>
    <input type="text" id="txtNumeroTarjeta" placeholder="Número de
Tarjeta de Crédito con Formato WWW-XXXX-YYYY-ZZZZ">
    <div id="errorTarjeta" class="error"></div>
    <input type="text" id="numCVC" placeholder="CVC">
    <div id="errorCVC" class="error"></div>
    <input type="button" id="btnRegistro" value="Registrarse">
    <input type="button" id="bntIraLogin" value="Ir a iniciar sesion">
    <p id="msgRegistro"></p>
  </section>
```

```

<section id="F02">
  <form id="formLogin">
    <h2>Ingreso al Sistema</h2>
    <input type="text" id="txtNombreUsuarioLogin"
placeholder="Nombre de Usuario">
    <input type="password" id="passContrasenaLogin"
placeholder="Contraseña">
    <input type="button" id="btnLogin" value="Ingresar">
<input type="button" id="btnIraRegistro" value="Ir a registrarse">
    <p id="msgLogin"></p>
  </form>
</section>

<section id="F03">
  <h2>Compra de Productos</h2>
  <input type="button" id="btnVerProductos" value="Ver Productos">
  <p id="msgVerProductos"></p>
</section>

<section id="F04">
  <h2>Listado de compras</h2>
  <input type="button" id="btnVerListadoCompras" value="Ver listado
de compras">
  <p id="msgListadoCompras"></p>
</section>

<section id="F05">
  <h2>Ver Productos en Oferta</h2>
  <input type="button" id="btnVerProductosOferta" value="Ver
productos en oferta">
  <p id="msgProductosOferta"></p>
</section>

<section id="F07">
  <h2>Listado y Aprobación de Compras</h2>
  <input type="button" id="btnVerAprobarCompras" value="Ver y
Aprobar Compras">
  <p id="msgAprobarCompras"></p>
</section>

<section id="F08">
  <h2>Crear Producto</h2>
  <input type="text" id="txtNombreProducto" placeholder="Nombre del
producto">

```



```

        <input type="number" id="numPrecioProducto" placeholder="Precio">
        <input type="text" id="txtDescripcionProducto"
placeholder="Descripción">
        <select id="slcCrearImagen">
            <option value="opciones">Opciones</option>
            <option value="./img/botinesNacional.png">Botines
Nacional</option>
            <option value="./img/buzoNacional.png">Buzo Nacional</option>
            <option value="./img/camisetaNacional.png">Camiseta
Nacional</option>
            <option value="./img/camperonNacional.png">Camperon
Nacional</option>
            <option value="./img/gorroNacional.png">Gorro
Nacional</option>
            <option value="./img/mediasNacional.png">Medias
Nacional</option>
            <option value="./img/mochilaNacional.png">Mochila
Nacional</option>
            <option value="./img/pantalonNacional.png">Pantalon
Nacional</option>
            <option value="./img/remeraNacional.png">Remera
Nacional</option>
            <option value="./img/shortNacional.png">Short
Nacional</option>
        </select>

        <input type="number" id="numCantidadStockProducto"
placeholder="Cantidad de stock">
        <input type="button" id="btnCrearProducto" value="Agregar
Producto">
        <p id="msgCrearProducto"></p>
    </section>

    <section id="F09">
        <h2>Administrar Productos</h2>
        <input type="button" id="btnAdministrarProducots"
value="Administrar Productos">
        <p id="msgAdministrarProductos"></p>
    </section>

    <section id="F10">
        <h2>Informe de Ganancias</h2>
        <input type="button" id="btnInformeGanancias" value="Informe de
Ganancias">
        <p id="msgInformeGanancias"></p>
    </section>

```

```
<script src="obligatorio.js"></script>  
</body>  
</html>
```



## JS

```
// Contadores let
cantidadAdministradores = 0 let
cantidadCompradores = 0 let
cantidadProductos = 0 let
cantidadCompras = 0

// Variables let
usuarioConectado = null

// Clases class Sistema {    // Listas
(arrays)    listaAdministradores = new
Array()    listaCompradores = new
Array()    listaProductos = new Array()
listaCompras = new Array()

    //Metodos de la clase Sistema    existeAdmin(unNombre)
{        for (let unAdmin of this.listaAdministradores) {
if (unAdmin.adminUsuario == unNombre) return true
        }
return false
    }    hallarAdmin(unNombre) {        for (let
unAdmin of this.listaAdministradores) {
        if (unAdmin.adminUsuario == unNombre) return unAdmin
        }
return null
    }    existeComprador(unNombre) {        for (let unComprador of
this.listaCompradores) {        if (unComprador.compradorUsuario
== unNombre) return true        }
return
false
    }    hallarComprador(unNombre) {        for (let
unComprador of this.listaCompradores) {
```

```

        if (unComprador.compradorUsuario == unNombre) return
unComprador
    }
    return null
}
    hallarProducto(unId) {
        for (let unProducto of
this.listaProductos) {
            if (unProducto.productoId ==
unId) return unProducto
        }
        return null
    }
}

// Metodos para funcion validar compra
saldoComprador(compradorId) {
    for (let unComprador of
this.listaCompradores) {
        if (unComprador.compradorId
== compradorId) return unComprador.saldo
    }
    return null
}
    obtenerStock(productoId) {
        for (let
unProducto of this.listaProductos) {
            if
(unProducto.productoId == productoId) return
unProducto.stock
        }
        return null
    }
}
clienteCompra(comprasId) {
    for (let unaCompra of this.listaCompras)
{
        if (unaCompra.comprasId == comprasId)
return unaCompra.idComprador
    }
}
    productoCompra(comprasId) {
        for (let
unaC of this.listaCompras) {
            if (unaC.comprasId ==
comprasId) return unaC.idProducto
        }
    }
}
    montoCompra(comprasId) {
        for (let unaC of this.listaCompras)
{
            if (unaC.comprasId == comprasId) return unaC.monto
        }
    }
}

```



```

        }    }    cantidadCompra(comprasId) {        for (let
unaC of this.listaCompras) {        if (unaC.comprasId ==
comprasId) return unaC.cantidad        } return null
    }

    pausarProducto(productoId) {        for (let unProd of
this.listaProductos) {        if (unProd.productoId ==
productoId) unProd.estado ==
"pausado"
    }

    }    estadoProducto(productoId) {        for (let unProd of
this.listaProductos) {        if (unProd.productoId == productoId)
return unProd.estado        }
    }    actualizarCompra(comprasId, unEstado) {        for (let unaC
of this.listaCompras) {        if (unaC.comprasId == comprasId)
unaC.estado = unEstado        }    }    actualizarProducto(productoId,
unStock) {        for (let unProd of this.listaProductos) {        if
(unProd.productoId == productoId) unProd.stock = unStock        }

    }    modificarsaldoCliente(compradorId, nuevoValor) {        for
(let unCli of this.listaCompradores) {        if (unCli.compradorId ==
compradorId) unCli.saldo = nuevoValor        }
    }

}

```



```

class Administrador {
constructor(unAdminUsuario, unAdminPassword) {
this.adminId = cantidadAdministradores
cantidadAdministradores++
this.adminUsuario =
unAdminUsuario.toLowerCase() this.adminPassword
= unAdminPassword
}

} class Compradores { constructor(unCompradorNombre,
unCompradorApellido, unCompradorUsuario,
unCompradorPassword, unCompradorTarjeta, unCompradorCvc) {
this.compradorId = cantidadCompradores
cantidadCompradores++
this.compradorNombre = unCompradorNombre
this.compradorApellido = unCompradorApellido
this.compradorUsuario = unCompradorUsuario.toLowerCase()
this.compradorPassword = unCompradorPassword
this.compradorTarjeta = unCompradorTarjeta
this.compradorCvc = unCompradorCvc this.saldo = 3000
}

} class Productos { constructor(unNombre, unaDescripcion, unPrecio,
unStock, unaImagen) { this.productoId = "Prod_ID_" +
cantidadProductos cantidadProductos++
this.nombre = unNombre
this.descripcion = unaDescripcion
this.precio = unPrecio this.stock
= unStock this.imagen = unaImagen
this.oferta = false this.estado =
"activo"
}
} class
Compras {

```

```

        constructor(idComprador, idProducto, cantidad, monto) {
this.comprasId = cantidadCompras          cantidadCompras++
        this.idComprador          =
idComprador          this.idProducto =
idProducto          this.cantidad =
cantidad          this.monto = monto
this.estado = "Pendiente"

    }
}

// Objetos preCargados!!!!

// Sistema preCargado let
sistema = new Sistema()

// Admins preCargados let admin1 = new
Administrador("fRobayna", "12345") let admin2 = new
Administrador("alangelan", "12345") let admin3 = new
Administrador("mGonzalez", "12345") let admin4 = new
Administrador("cMartinez", "12345") let admin5 = new
Administrador("lFernandez", "12345")

// Compradores preCargados let comprador1 = new Compradores("Matias",
"Perez", "MatiasPro", "12345A",
"5555555555554444", "333") let comprador2 = new Compradores("Felipe",
"Rodriguez", "FeliRodriguez",
"12345B", "5105105105105100", "435") let comprador3 = new
Compradores("Juan", "Alpaca", "JuanitoKratos",
"12345C", "5105105105105108", "123") let comprador4 = new
Compradores("Ana", "Lopez", "AnaL123", "12345D",
"4012888888881881", "456") let comprador5 = new Compradores("Laura",
"Martinez", "LauMartinez",
"12345E", "4111111111111111", "789")

// Productos preCargados
let p1 = new Productos("Campera de Nacional", "Campera del club mas
grande del mundo", 5000, 5, "img/camperonNacional.png") let p2 = new
Productos("Mochila de Nacional", "Mochila del club mas grande del mundo",
1750, 9, "img/mochilaNacional.png")

```





```

let p3 = new Productos("Short de Nacional", "Short del club mas grande del
mundo", 1299, 12, "img/shortNacional.png") let p4 = new Productos("Gorro de
Nacional", "Gorro del club mas grande del mundo", 790, 8,
"img/gorroNacional.png") let p5 = new Productos("Medias de Nacional",
"Medias del club mas grande del mundo", 350, 6, "img/mediasNacional.png")
let p6 = new Productos("Remera de Nacional", "Remera del club mas grande
del mundo", 2500, 10, "img/remeraNacional.png") let p7 = new Productos("Buzo
de Nacional", "Buzo del club mas grande del mundo", 4500, 7,
"img/buzoNacional.png") let p8 = new Productos("Pantalón de Nacional",
"Pantalón del club mas grande del mundo", 3200, 4,
"img/pantalonNacional.png") let p9 = new Productos("Camiseta de Nacional",
"Camiseta del club mas grande del mundo", 2800, 15,
"img/camisetaNacional.png") let p10 = new Productos("Botines de Nacional",
"Botines del club mas grande del mundo", 6000, 3, "img/botinesNacional.png")

p2.estado = "pausado"

// Productos en oferta
p4.oferta =
true p5.oferta =
true

//Compras Precargadas
let compra1 = new Compras("1", "Prod_ID_4", 2, 5000,
"Pendiente") let compra2 = new Compras("2", "Prod_ID_3", 1, 1750,
"Pendiente") let compra3 = new Compras("3", "Prod_ID_1", 3, 1299,
"Pendiente") let compra4 = new Compras("4", "Prod_ID_2", 5, 3200,
"Pendiente") let compra5 = new Compras("5", "Prod_ID_5", 2, 4500,
"Pendiente")

compra1.estado = "Cancelada"
// Guardar datos preCargados

// Guardar admins preCargados sistema.listaAdministradores.push(admin1,
admin2, admin3, admin4, admin5 )
// Guardar compradores preCargados
sistema.listaCompradores.push(comprador1, comprador2,
comprador3, comprador4, comprador5 )

```



```

// Guardar productos preCargados
sistema.listaProductos.push(p1, p2, p3, p4, p5, p6, p7, p8, p9,
p10)
// Guardar compras preCargadas
sistema.listaCompras.push(compra1, compra2, compra3, compra5,
compra4)

// Funciones

inicio()
function inicio() {    ocultarTodo()
document.querySelector("#F02").style.display = "block"
    //          Escuchas          de          botones
document.querySelector("#btnIraRegistro").addEventListener("click",
irARegistro)
document.querySelector("#bntIraLogin").addEventListener("click", irALogin)
document.querySelector("#btnLogin").addEventListener("click",  hacerLogin)
document.querySelector("#btnCerrarSesion").addEventListener("click",
cerrarSesion)
document.querySelector("#btnVerProductos").addEventListener("click",
mostrarListadoProductos)
document.querySelector("#btnCarrito").addEventListener("click",
mostrarCarrito)
document.querySelector("#btnFlechaAtras").addEventListener("click",
flechaAtras)
document.querySelector("#btnRegistro").addEventListener("click",
hacerRegistro)
document.querySelector("#btnCrearProducto").addEventListener("click",
crearProducto)
document.querySelector("#btnVerAprobarCompras").addEventListener("click",
verCompras)
document.querySelector("#btnVerProductosOferta").addEventListener("click",
productosOfertas)
document.querySelector("#btnAdministrarProductos").addEventListener("click",
administrarProductos)
document.querySelector("#btnVerListadoCompras").addEventListener("click",
verListadoCompras)
document.querySelector("#btnInformeGanancias").addEventListener("click",
informeGanancias)

```



```

} function
ocultarTodo() {
    document.querySelector("#btnCerrarSesion").style.display =
"none"
    document.querySelector("#btnCarrito").style.display =
"none"
    document.querySelector("#btnFlechaAtras").style.display =
"none"
    document.querySelector("#F01").style.display = "none"
document.querySelector("#F02").style.display = "none"
document.querySelector("#F03").style.display = "none"
document.querySelector("#F04").style.display = "none"
document.querySelector("#F05").style.display = "none"
document.querySelector("#F07").style.display = "none"
document.querySelector("#F08").style.display = "none"
document.querySelector("#F09").style.display = "none"
document.querySelector("#F10").style.display = "none"
} function
mostrarAdmin() {
    document.querySelector("#btnCerrarSesion").style.display =
"block"
    document.querySelector("#F01").style.display = "none"
document.querySelector("#F02").style.display = "none"
document.querySelector("#F07").style.display = "block"
document.querySelector("#F08").style.display = "block"
document.querySelector("#F09").style.display = "block"
document.querySelector("#F10").style.display = "block"
} function mostrarComprador() {    ocultarTodo()
document.querySelector("#btnCerrarSesion").style.display = "block"
document.querySelector("#btnCarrito").style.display = "block"
document.querySelector("#F01").style.display = "none"
document.querySelector("#F02").style.display = "none"
document.querySelector("#F03").style.display = "block"
document.querySelector("#F05").style.display = "block"
}    function    mostrarCarrito()    {        ocultarTodo()
document.querySelector("#F04").style.display = "block"
document.querySelector("#btnFlechaAtras").style.display = "block"

```



```

        document.querySelector("#btnCerrarSesion").style.display = "block"
    }
    function flechaAtras() {
        document.querySelector("#btnCerrarSesion").style.display = "block"
        document.querySelector("#btnCarrito").style.display = "block"
        document.querySelector("#F01").style.display = "none"
        document.querySelector("#F02").style.display = "none"
        document.querySelector("#F03").style.display = "block"
        document.querySelector("#F05").style.display = "block" }

function irARegistro() {
document.querySelector("#F01").style.display = "block"
document.querySelector("#F02").style.display = "none"
}
function irALogin() {
document.querySelector("#F01").style.display = "none"
document.querySelector("#F02").style.display = "block"
}
function hacerLogin() {
    let nombreUsuario =
document.querySelector("#txtNombreUsuarioLogin").value.toLowerCase()
let pass = document.querySelector("#passContrasenaLogin").value
    if (sistema.existeAdmin(nombreUsuario)) {
        let
elAdmin = sistema.hallarAdmin(nombreUsuario)
        if
(elAdmin.adminPassword == pass) {
            usuarioConectado = elAdmin
            mostrarAdmin()
        } else {
            alert("Error de Inicio de Sesión. Por favor,
verifica que tu nombre de usuario y contraseña sean correctos e inténtalo
nuevamente.")
        }
    } else {
        if
(sistema.existeComprador(nombreUsuario)) {
            let elComprador
            if
            = sistema.hallarComprador(nombreUsuario)
            if
            (elComprador.compradorPassword == pass) {
                usuarioConectado = elComprador
            }
        }
    }
}

```





```

        mostrarComprador()
    } else {
        alert("Error de Inicio de Sesión. Por favor, verifica que tu nombre de usuario y contraseña sean correctos e inténtalo nuevamente.")
    } } else {
        alert("Error de Inicio de Sesión. Por favor, verifica que tu nombre de usuario y contraseña sean correctos e inténtalo nuevamente.")
    }
}

function cerrarSesion() {
    ocultarTodo()
    document.querySelector("#F02").style.display = "block"
    document.querySelector("#formLogin").reset()
    usuarioConectado = null
}

function mostrarListadoProductos() {
    let miTabla = ""
    miTabla += `
        <table border="1">
            <thead>
                <td> ID Producto </td>
                <td> Imagen </td>
                <td> Nombre </td>
                <td> Descripcion </td>
                <td> Precio </td>
                <td> Stock </td>
                <td> Cantidad </td>
                <td> Oferta </td>
                <td> Comprar </td>
            </thead>
            `
    for (let unProducto of sistema.listaProductos) {
        if (unProducto.estado === "activo") {
            miTabla += `
                <tr>
                    <td>${unProducto.productoId}</td>
                    <td></td>
                    <td>${unProducto.nombre}</td>
                    <td>${unProducto.descripcion}</td>
            `
        }
    }
}

```



```

        <td>${unProducto.precio}</td>
        <td>${unProducto.stock}</td>
</td> <input type="number"
id="numCantidad${unProducto.productoId}"> </td>
        <td> ${convertir(unProducto.oferta)} </td>
<td> <input type="button" value="Comprar" class="botonAccion"
id="boton${unProducto.productoId}"> </td>
    </tr>`
    }
    }
    miTabla += `</table>`
document.querySelector("#msgVerProductos").innerHTML = miTabla
    let listaInputs = document.getElementsByClassName("botonAccion")
for (let unInput of listaInputs) {
unInput.addEventListener("click", hacerCompra)
}
}

function hacerCompra() {
    let idBoton = this.id
    let idProd = idBoton.substring(5) //traer el stock
    en una variable let objProducto =
    sistema.hallarProducto(idProd) let stock =
    objProducto.stock
    let idCantidad = `#numCantidad${idProd}` let
    cantidadInput = document.querySelector(idCantidad) let
    cantidad = Number(cantidadInput.value) if (cantidad <=
    0 || stock < cantidad) { alert("Error de compra")
    return
    } else {
        let idUsuario =
        usuarioConectado.compradorId
        let precio = precioProducto(idProd) if (precio !== null)
        {
            let monto = cantidad * precio let miCompra = new
            Compras(idUsuario, idProd, cantidad, monto)
            sistema.listaCompras.push(miCompra) alert("La compra fue
            realizada con éxito")
        }
    }
}

```



```

    // else {
    //     alert("Producto no encontrado.")
    // }
    // } else {
    //     alert("Cantidad no válida.")
    // }
}

function precioProducto(idProd) {
    for (let unProducto of sistema.listaProductos)
    {
        if (unProducto.productoId == idProd) return unProducto.precio
    }

    return null
}

function hacerRegistro() {
    let tarjetaDeCredito =
document.querySelector("#txtNumeroTarjeta").value    let nombre =
document.querySelector("#txtNombre").value            let apellido =
document.querySelector("#txtApellido").value          let contraseña =
document.querySelector("#passContrasena").value      let cvc =
document.querySelector("#numCVC").value               let usuario =
document.querySelector("#txtNombreUsuario").value.toLowerCase()
    //Errores de ingreso
document.querySelector("#errorNombre")               let errorNombre =
document.querySelector("#errorApellido")             let errorApellido =
document.querySelector("#errorUsuario")              let errorUsuario =
document.querySelector("#errorContrasena")           let errorContrasena =
document.querySelector("#errorTarjeta")              let errorTarjeta =
document.querySelector("#errorCVC")                  let errorCVC =
document.querySelector("#errorCVC")                  let msgRegistro =
document.querySelector("#msgRegistro")

    // Reiniciar mensajes de error y clases
errorNombre.innerHTML = ""
errorNombre.style.display = "none"
errorApellido.innerHTML = ""
errorApellido.style.display = "none"
errorUsuario.innerHTML = ""
errorUsuario.style.display = "none"
errorContrasena.innerHTML = ""
errorContrasena.style.display = "none"

```



```

        errorTarjeta.innerHTML = ""      errorTarjeta.style.display
= "none"      errorCVC.innerHTML = ""      errorCVC.style.display
= "none"      msgRegistro.innerHTML = ""
msgRegistro.style.display = "none"
errorNombre.classList.remove("error", "correct")
errorApellido.classList.remove("error", "correct")
errorUsuario.classList.remove("error", "correct")
errorContrasena.classList.remove("error", "correct")
errorTarjeta.classList.remove("error", "correct")
errorCVC.classList.remove("error", "correct")
msgRegistro.classList.remove("error", "correct", "success")
    document.querySelector("#txtNombre").classList.remove("input-error",
"input-correct")
document.querySelector("#txtApellido").classList.remove("input-error",
"input-correct")
document.querySelector("#txtNombreUsuario").classList.remove("inputerror",
"input-correct")
document.querySelector("#passContrasena").classList.remove("inputerror",
"input-correct")
document.querySelector("#txtNumeroTarjeta").classList.remove("inputerror",
"input-correct")
document.querySelector("#numCVC").classList.remove("input-error",
"input-correct")
    let hayError =
false
    if (nombre.trim() === "") {      errorNombre.innerHTML = "El nombre
no puede estar vacío."      errorNombre.style.display = "block"
errorNombre.classList.add("error")
document.querySelector("#txtNombre").classList.add("input-error")
hayError = true
    } else if (!validarNombre(nombre)) {      errorNombre.innerHTML =
"Nombre inválido."      errorNombre.style.display = "block"
errorNombre.classList.add("error")
document.querySelector("#txtNombre").classList.add("input-error")
hayError = true
    } else {      errorNombre.innerHTML = "Correcto"
errorNombre.style.display = "block"
errorNombre.classList.add("correct")
document.querySelector("#txtNombre").classList.add("inputcorrect")
    }
}

```





```

        if (apellido.trim() === "") {
            errorApellido.innerHTML =
            "El apellido no puede estar vacío."
            errorApellido.style.display
            = "block"
            errorApellido.classList.add("error")
            document.querySelector("#txtApellido").classList.add("inputerror")
            hayError = true
        } else if (!validarApellido(apellido)) {
            errorApellido.innerHTML
            = "Apellido inválido."
            errorApellido.style.display
            = "block"
            errorApellido.classList.add("error")
            document.querySelector("#txtApellido").classList.add("inputerror")
            hayError = true
        } else {
            errorApellido.innerHTML = "Correcto"
            errorApellido.style.display
            = "block"
            errorApellido.classList.add("correct")
            document.querySelector("#txtApellido").classList.add("inputcorrect")
        } if (usuario.trim() === "") {
            errorUsuario.innerHTML =
            "El nombre de usuario no puede estar vacío."
            errorUsuario.style.display = "block"
            errorUsuario.classList.add("error")
            document.querySelector("#txtNombreUsuario").classList.add("inputerror")
            hayError = true
        } else if (validarUsuario(usuario)) {
            errorUsuario.innerHTML
            = "El nombre de usuario ya existe."
            errorUsuario.style.display =
            "block"
            errorUsuario.classList.add("error")
            document.querySelector("#txtNombreUsuario").classList.add("inputerror")
            hayError = true
        } else {
            errorUsuario.innerHTML = "Correcto"
            errorUsuario.style.display
            = "block"
            errorUsuario.classList.add("correct")
            document.querySelector("#txtNombreUsuario").classList.add("inputcorrect")
        } if (contraseña.trim()
        === "") {

```



```

        errorContrasena.innerHTML = "La contraseña no puede estar vacía."
errorContrasena.style.display = "block"
errorContrasena.classList.add("error")
document.querySelector("#passContrasena").classList.add("inputerror")
hayError = true
    } else if (!esValidaPass(contraseña)) {
errorContrasena.innerHTML = "Contraseña inválida. Debe tener al menos 5
caracteres, una mayúscula, una minúscula, un número y un carácter
especial."
        errorContrasena.style.display = "block"
errorContrasena.classList.add("error")
document.querySelector("#passContrasena").classList.add("inputerror")
hayError = true
    } else {
        errorContrasena.innerHTML = "Correcto"
errorContrasena.style.display = "block"
errorContrasena.classList.add("correct")
document.querySelector("#passContrasena").classList.add("inputcorrect")
    }
    if (tarjetaDeCredito.trim() === "") {
errorTarjeta.innerHTML = "El número de tarjeta de crédito no puede estar
vacío."
        errorTarjeta.style.display = "block"
errorTarjeta.classList.add("error")
document.querySelector("#txtNumeroTarjeta").classList.add("inputerror")
hayError = true
    } else if (!validarTarjeta(tarjetaDeCredito)) {
errorTarjeta.innerHTML = "Tarjeta de crédito inválida."
errorTarjeta.style.display = "block"
errorTarjeta.classList.add("error")
document.querySelector("#txtNumeroTarjeta").classList.add("inputerror")
hayError = true
    } else {
        errorTarjeta.innerHTML = "Correcto"
errorTarjeta.style.display = "block"
errorTarjeta.classList.add("correct")
document.querySelector("#txtNumeroTarjeta").classList.add("inputcorrect")
    }
    if (cvc.trim()
=== "") {

```



```

        errorCVC.innerHTML = "El CVC no puede estar vacío."
errorCVC.style.display = "block"          errorCVC.classList.add("error")
document.querySelector("#numCVC").classList.add("input-error")
hayError = true
    } else if (!validarCvc(cvc)) {          errorCVC.innerHTML = "CVC
inválido. Debe ser un número de 3 dígitos."
errorCVC.style.display = "block"
errorCVC.classList.add("error")
document.querySelector("#numCVC").classList.add("input-error")
hayError = true
    } else {                              errorCVC.innerHTML = "Correcto"
errorCVC.style.display = "block"          errorCVC.classList.add("correct")
document.querySelector("#numCVC").classList.add("input-correct")      }
if (!hayError) {                          msgRegistro.innerHTML = "Usuario creado con
éxito, ya puede iniciar sesión"          msgRegistro.style.display =
"block"                                  msgRegistro.classList.add("success")
        let nuevoUsuario = new Compradores(nombre, apellido, usuario,
contraseña,                                tarjetaDeCredito,          cvc)
sistema.listaCompradores.push(nuevoUsuario)
    }
}

function esValidaPass(unaContraseña) {
    if (unaContraseña.length >= 5 &&
contieneUnaMayuscula(unaContraseña)
&& contieneUnaMinuscula(unaContraseña) && contieneUnDigito(unaContraseña)
&& contieneCaracterRaro(unaContraseña)) {
return true    } else {
    return false
    }
}

function
contieneUnaMayuscula(unaContraseña) {    let
contador = 0    for (let unCaracter of
unaContraseña) {

```



```

        if (unCaracter == unCaracter.toUpperCase() && unCaracter != " ")
return true    }    return false
} function contieneUnaMinuscula(unaContraseña) {    let contador = 0
for (let unCaracter of unaContraseña) {    if (unCaracter ==
unCaracter.toLowerCase() && unCaracter != " ") return true    }    return
false
} function contieneUnDigito(unaContraseña) {    let contador =
0    for (let unCaracter of unaContraseña) {    if
(!isNaN(unCaracter) && unCaracter != " ") return true
    }    return
false
} function
contieneCaracterRaro(unaContraseña) {

    // Todos los que quieras aceptar como caracter
let conjunto = "@_ -#!.:"
    for (let unCaracter of unaContraseña) {    if
(conjunto.includes(unCaracter)) return true    }
return false
} function
quitarGuionesTarjeta(tarjeta) {
    let tarjetaLimpia =
""
    for (let i of tarjeta) {
if (i != "-") {
tarjetaLimpia += i
    }    }
return tarjetaLimpia

```





```

} function validarTarjeta(nro) {
nro = quitarGuionesTarjeta(nro)
let suma = 0 let impar = true
  for (let i = 0; i < nro.length; i++)
  {
    let n = Number(nro[i]) if
(impar && nro != "") { n = n *
2 if (n > 9) { n
= n - 9
    }

  } suma
= suma + n
impar = !impar
  } return (suma %
10 == 0)
} function
validarNombre(unNombre) {
  if (unNombre != "")
{
  return
true
  }
return false

} function
validarApellido(unApellido) {
  if (unApellido != "")
{
  return
true
  }
return false

}

```

```

function validarCvc(unCvc) {
    if (unCvc < 1000 && unCvc.length == 3)
    {
        return
true
    }
return false

} function
validarUsuario(unUsuario) {
    for (let i of sistema.listaCompradores)
    {
        if (unUsuario == i.compradorUsuario.toLowerCase() && unUsuario !=
        "") {
            return true
        }
    }
return false
} function crearProducto() {
    let nombre =
document.querySelector("#txtNombreProducto").value
    let
descripcion =
document.querySelector("#txtDescripcionProducto").value
    let
precio =
Number(document.querySelector("#numPrecioProducto").value)
    let stock =
Number(document.querySelector("#numCantidadStockProducto").value)
    let
imagen = document.querySelector("#slcCrearImagen").value
    if
(validarNombre(nombre) && validarDescripcion(descripcion) &&
validadPrecio(precio) && validarStock(stock) && validarImagen(imagen)) {
        let nuevoProducto = new Productos(nombre, descripcion, precio, stock,
imagen)
        sistema.listaProductos.push(nuevoProducto)
        alert("Producto creado con
exito")
    } else {
        alert("Error
de datos")
    }
}

```



```

}                                function
validarDescripcion(unDesc) {    if
(unDesc !== "") {              return true
    }                          return
false
}                                function
validarPrecio(unPrecioValidar) {    if
(unPrecioValidar > 0) {          return
true
    }                          return
false
}                                function
validarStock(unStockValidar) {    if
(unStockValidar > 0) {          return
true
    }                          return
false
}  function
validarImagen(unImagenValidar) {    if
(unImagenValidar !== "opciones") {
return true
    }                          return
false
}  function
verCompras() {    let
miTabla = `
    <select id="filtroComprasAdmin">
        <option value="todas">Todas</option>
        <option value="Aprobada">Aprobadas</option>
        <option value="Cancelada">Canceladas</option>
        <option value="Pendiente">Pendientes</option>
    </select>
    <div id="contenedorComprasAdmin">
        ${generarTablaComprasAdmin("todas")}
    </div> `;
document.querySelector("#msgAprobarCompras").innerHTML = miTabla;

```



```

        document.querySelector("#filtroComprasAdmin").addEventListener("change",
        filtrarComprasAdmin);
        agregarEventosValidarCompra();
    }
    function
    generarTablaComprasAdmin(filtro) {
        let
        miTabla = `
            <table border="1">
                <thead>
                    <tr>
                        <td>Imagen</td>
                        <td>ID Compra</td>
                        <td>ID Comprador</td>
                        <td>ID Producto</td>
                        <td>Cantidad</td>
                        <td>Monto</td>
                        <td>Estado</td>
                        <td>Oferta</td>
                        <td>Acción</td>
                    </tr>
                </thead>
                <tbody id="tablaComprasAdmin">
                    `;
                    for (let unaCompra of sistema.listaCompras) {
                        if
                        (filtro === "todas" || unaCompra.estado.toLowerCase() ===
                        filtro.toLowerCase()) {
                            miTabla += `
                                <tr>
                                    <td></td>
                                    <td>${unaCompra.comprasId}</td>
                                    <td>${unaCompra.idComprador}</td>
                                    <td>${unaCompra.idProducto}</td>
                                    <td>${unaCompra.cantidad}</td>
                                    <td>${unaCompra.monto}</td>
                                    <td>${unaCompra.estado}</td>
                                    <td>${convertir(encontrarOferta(unaCompra.idProducto))}</td>
                                <td> <input type="button" value="Validar"
                                class="botonValidar" id="boton${unaCompra.comprasId}"> </td>
                                </tr>`;
                                if (unaCompra.estado === "Pendiente") {
                                    miTabla
                                    += `
                                        <td> <input type="button" value="Validar"
                                        class="botonValidar" id="boton${unaCompra.comprasId}"> </td>
                                        </tr>`;
                                }
                            }
                        }
                    }
                }
            }

```



```

    }      miTabla
+= `
</tbody>
</table>`;
return miTabla;
}      function filtrarComprasAdmin() {      let filtro =
document.querySelector("#filtroComprasAdmin").value;      let
contenedorCompras =
document.querySelector("#contenedorComprasAdmin");
contenedorCompras.innerHTML = generarTablaComprasAdmin(filtro);
agregarEventosValidarCompra();
}      function agregarEventosValidarCompra() {      let botonesValidar =
document.getElementsByClassName("botonValidar");      for (let
unaValidacion of botonesValidar) {
unaValidacion.addEventListener("click", validarCompra);      }
}

function encontrarImagen(idProducto) {      for (let
unProducto of sistema.listaProductos) {      if
(unProducto.productoId == idProducto) {
return unProducto.imagen;
      }      }
return null;
}      function convertir(unValor) {
return unValor ? "Sí" : "No";
}

function validarCompra() {      let
idBoton = this.id;      let idCompra =
idBoton.substring(5);
      let idCliente = sistema.clienteCompra(idCompra);
let monto = sistema.montoCompra(idCompra);      let
idProducto = sistema.productoCompra(idCompra);      let
cantidad = sistema.cantidadCompra(idCompra);      let
stock = sistema.obtenerStock(idProducto);

```





```

    let estado = sistema.estadoProducto(idProducto);
let saldo = sistema.saldoComprador(idCliente);
    let
nuevoEstado;
    if (estado === "activo" && saldo >= monto && stock >= cantidad)
{
    let nuevoStock = stock - cantidad;        let nuevoSaldo =
saldo - monto;
    if (nuevoStock == 0) {
sistema.pausarProducto(idProducto);
    }
    nuevoEstado = "aprobada";
sistema.actualizarCompra(idCompra, nuevoEstado);
sistema.actualizarProducto(idProducto, nuevoStock);
sistema.modificarsaldoCliente(idCliente, nuevoSaldo);
totalAprobadas++;

    } else {
        nuevoEstado = "cancelada";
sistema.actualizarCompra(idCompra, nuevoEstado);
    }
verCompras();
}

function encontrarImagen(unIdProducto) {    for (let
unProducto of sistema.listaProductos) {        if
(unProducto.productoId == unIdProducto) {
return unProducto.imagen
        }
    }
    return null
}
function encontrarOferta(unIdProducto) {    for
(let unProducto of sistema.listaProductos) {        if
(unProducto.productoId == unIdProducto) {
return unProducto.oferta
        }
    }
    return null
}

```



```

function productosOfertas()
{
    let miTabla = ""
    miTabla += `
        <table border="1">
            <thead>
                <td> ID Producto </td>
                <td> Imagen </td>
                <td> Nombre </td>
                <td> Descripcion </td>
                <td> Precio </td>
                <td> Stock </td>
                <td> Cantidad </td>
                <td> Oferta </td>
                <td> Comprar </td>
            </thead>
            `
            for (let unProducto of sistema.listaProductos) {
if (unProducto.estado === "activo" && unProducto.oferta) {
miTabla += ` <tr>
                <td>${unProducto.productoId}</td>
                <td></td>
                <td>${unProducto.nombre}</td>
                <td>${unProducto.descripcion}</td>
                <td>${unProducto.precio}</td>
                <td>${unProducto.stock}</td>
                <td> <input type="number"
id="numCantidad${unProducto.productoId}"> </td>
                <td> ${convertir(unProducto.oferta)} </td>
                <td> <input type="button" value="Comprar" class="botonAccion"
id="boton${unProducto.productoId}"> </td>
            </tr>`
            }
            }
            miTabla += `</table>`
document.querySelector("#msgProductosOferta").innerHTML = miTabla
    let listaInputs =
document.getElementsByClassName("botonAccion")
    for (let unInput
of listaInputs) {
        unInput.addEventListener("click",
hacerCompra)
    }
}

function convertir(unValor)
{

```

```

        if (unValor == true) return "Si"
    if (unValor == false) return "No"

}
function
administrarProductos() {
    let
    miTabla = ""
    miTabla += `
        <table border="1">
            <thead>
                <tr>
                    <td>ID Producto</td>
                    <td>Imagen</td>
                    <td>Nombre</td>
                    <td>Descripcion</td>
                    <td>Precio</td>
                    <td>Stock</td>
                    <td>Estado</td>
                    <td>Oferta</td>
                    <td>Acción</td>
                </tr>
            </thead>
            `
            for (let unProducto of
sistema.listaProductos) {
                miTabla += `<tr>
                    <td>${unProducto.productoId}</td>
                    <td></td>
                    <td>${unProducto.nombre}</td>
                    <td>${unProducto.descripcion}</td>
                    <td>${unProducto.precio}</td>
                    <td><input type="number" id="txtStock${unProducto.productoId}"
value="${unProducto.stock}"></td>
                    <td><select id="slcEstado${unProducto.productoId}">
                        <option value="activo" ${unProducto.estado ==
"activo" ? "selected" : ""}>activo</option>
                        <option value="pausado" ${unProducto.estado ==
"pausado" ? "selected" : ""}>pausado</option>
                    </select></td>
                    <td><select
id="slcOferta${unProducto.productoId}">
                        <option value="true" ${unProducto.oferta ?
"selected" : ""}>Sí</option>

```



```

        <option value="false" ${!unProducto.oferta ?
"selected" : ""}>No</option>
        </select></td>
        <td><input type="button" value="Modificar"
class="botonModificar" id="boton${unProducto.productoId}"></td>
</tr>`
    }
    miTabla += `</table>`
document.querySelector("#msgAdministrarProductos").innerHTML = miTabla
    let listaInputs =
document.getElementsByClassName("botonModificar");
    for (let unInput
of listaInputs) {
        unInput.addEventListener("click", hacerCambios);
    }
    function hacerCambios() {
        let
idBoton = this.id;
        let idProd =
idBoton.substring(5);
        let idStock =
`#txtStock${idProd}`;
        let idEstado =
`#slcEstado${idProd}`;
        let idOferta =
`#slcOferta${idProd}`;
        let nuevoStock = document.querySelector(idStock)?.value;
        let
nuevoEstado = document.querySelector(idEstado)?.value;
        let
nuevaOferta = document.querySelector(idOferta)?.value === 'true';
        if (nuevoStock !== null && nuevoEstado !== null && nuevaOferta !==
null) {
            for (let unProducto of sistema.listaProductos) {
                if (unProducto.productoId == idProd) {
                    unProducto.stock
= Number(nuevoStock);
                    unProducto.estado = nuevoEstado;
                    unProducto.oferta = nuevaOferta;
                }
            }
        }
        administrarProductos();
        mostrarListadoProductos();
        productosOfertas();
    } else {
        console.error("No se pudieron obtener los datos para
el producto con ID:", idProd);
    }
    alert("Producto modificado con éxito");
    agregarEventosValidarCompra(); // Agregar los eventos nuevamente

```





```

}

let totalAprobadas = 0 function
verListadoCompras() {      let
contenidoHTML = `
    <select id="filtroCompras">
        <option value="todas">Todas</option>
        <option value="Aprobada">Aprobadas</option>
        <option value="Cancelada">Canceladas</option>
        <option value="Pendiente">Pendientes</option>
    </select>
    <div id="contenedorCompras">
        ${generarTablaCompras("todas")}
    </div>
`
document.querySelector("#msgListadoCompras").innerHTML = contenidoHTML

document.querySelector("#filtroCompras").addEventListener("change",
filtrarCompras)      agregarEventosCancelarCompra()
}                      function
generarTablaCompras(filtro) {      let
tabla = `
    <table border="1">
        <thead>
            <tr>
                <td>Nombre del producto</td>
                <td>Cantidad</td>
                <td>Monto Total</td>
                <td>Estado</td>
                <td>Acción</td>
            </tr>
        </thead>
        <tbody id="tablaCompras">
            `

            for (let unaCompra of sistema.listaCompras) {                      if
            (unaCompra.idComprador === usuarioConectado.compradorId) {          let
            producto = encontrarProducto(unaCompra.idProducto)                  if
            (producto) {                      let montoTotal = unaCompra.cantidad *
            producto.precio                      if (filtro === "todas" ||
            unaCompra.estado.toLowerCase()
            === filtro.toLowerCase()) {
            tabla += `

```



```

        <tr>
            <td>${producto.nombre}</td>
            <td>${unaCompra.cantidad}</td>
            <td>${montoTotal}</td>
            <td>${unaCompra.estado}</td>
            <td>${unaCompra.estado === "Pendiente" ? `<input
type="button" value="Cancelar" class="botonCancelar"
id="boton${unaCompra.comprasId}">` : ``}</td>
        </tr>
    `
    }
    if
    (unaCompra.estado === "Aprobada") {
        totalAprobadas += montoTotal
    }
    }
    }
    }
    tabla += `
        </tbody>
    </table>
    <p>Total de compras aprobadas: ${totalAprobadas}</p>
    <p>Saldo disponible: ${usuarioConectado.saldo}</p>
    `
    return
}
tabla
} function encontrarProducto(idProducto) {
    for
    (let unProducto of sistema.listaProductos) {
        if
        (unProducto.productoId === idProducto) {
            return unProducto
        }
    }
    return null
} function cancelarCompra() {
    let idBoton = this.id
    let idCompra = parseInt(idBoton.replace("boton", ""))
    for (let unaCompra of sistema.listaCompras) {
        if (unaCompra.comprasId === idCompra) {
            if
            (unaCompra.estado === "Pendiente") {
                {
                unaCompra.estado = "Cancelada"
                alert("Compra cancelada con éxito")
                filtrarCompras()
                return
            }
        }
    }
}

```



```

    }
  }
}

function filtrarCompras() {
  let filtro =
document.querySelector("#filtroCompras").value
  let contenedorCompras =
document.querySelector("#contenedorCompras")
contenedorCompras.innerHTML =
generarTablaCompras(filtro)
agregarEventosCancelarCompra()
}

function agregarEventosCancelarCompra() {
  let botonesCancelar =
document.getElementsByClassName("botonCancelar")
  for (let unBoton of
botonesCancelar) {
    unBoton.addEventListener("click",
cancelarCompra)
  }
}

function unidadesVendidas(idProducto) {
  let
total = 0;
  for (let unaCompra of sistema.listaCompras) {
    if
(unaCompra.idProducto === idProducto && unaCompra.estado ===
"aprobada") {
      total +=
unaCompra.cantidad;
    }
  }
return total;
}

function devolverNombreArticulo(productoId) {
  for (let unProducto of sistema.listaProductos) {
    if (unProducto.productoId === productoId) {
      return unProducto.nombre;
    }
  }
return null;
}

function informeGanancias() {
  document.querySelector("#msgInformeGanancias").innerHTML = "";
  let
aprobados = new Set(); // Utilizar Set para evitar duplicados
  let
total = 0;
  for (let unaCompra of sistema.listaCompras) {
    if
(unaCompra.estado === "aprobada") {
      if
(!aprobados.has(unaCompra.idProducto)) {
        let totalVendido
= unidadesVendidas(unaCompra.idProducto);

```

```
document.querySelector("#msgInformeGanancias").innerHTML
+= `${devolverNombreArticulo(unaCompra.idProducto)} -- ${totalVendido}
<br>`;
aprobados.add(unaCompra.idProducto);
        }
        total +=
unaCompra.monto;
    }
}
document.querySelector("#msgInformeGanancias").innerHTML += `<hr>
Ganancia total: ${total}`;
}
```



## CSS

```
#btnCerrarSesion {  
  position:    fixed;  
  bottom: 10px;
```





```
color: red;          color: white;
border: none;        padding: 10px
20px;               cursor: pointer;
font-size: 16px;     z-index:
1000;                width: auto;
transition: transform 0.3s;
}

#btnCerrarSesion:hover {
background-color: darkred;
transform: scale(1.1);
}

#btnCarrito {        position: fixed;
top: -30px;          right: -30px;
background-color: transparent;
border: none;        cursor: pointer;
z-index: 1000;       width: 200px;
height: 200px;       transition:
transform 0.3s;
}

#btnCarrito img {
width: 100%;
height: auto;
}
```

```
#btnCarrito:hover {  
  transform: scale(1.1);  
}
```



```
#btnFlechaAtras {  
  position: fixed; top: -30px; right: -30px;  
  background-color: transparent; border: none;  
  cursor: pointer; z-index: 1000;
```



```

        width: 200px;      height:
200px;      transition: transform
0.3s;
    }

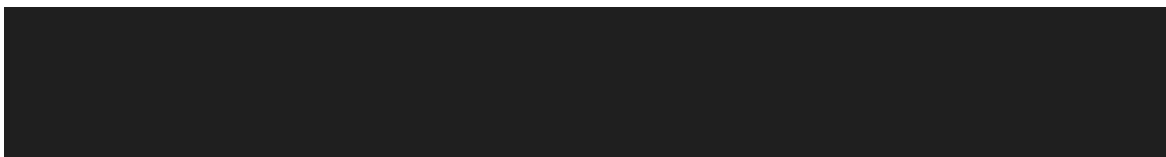
#btnFlechaAtras img {
width:      100%;
height: auto;
}


#btnFlechaAtras:hover {
transform: scale(1.1);
} body {    font-family: 'Comic Sans MS', 'Comic Sans',
cursive;    background-color: #f5f5f5;    margin: 0;
padding: 20px;    display: flex;    flex-direction:
column;    align-items: center;

    /* Fondo de imagen */    background-image:
url('img/fondoTienda.png'); /* Cambia 'img/fondo.jpg' por la
ruta de tu imagen */
} section {    background-color: #fff;
border: 2px solid #bbb;    border-radius:
20px;    box-shadow: 0 4px 8px rgba(0, 0, 0,
0.2);    margin: 20px;    padding: 20px;

```

```
width: 80%;      max-width:
1000px;          transition:
transform 0.3s;  overflow:
hidden;
```





```
    box-sizing: border-box;
}      section:hover {
transform: scale(1.05);
}  section
h2 {
```

```
margin-top: 0;      font-
size: 1.8em;      color: #333;
text-align:        center;
background-color:  #e0e0e0;
color: #333;      padding: 10px;
border-radius: 15px;
}      input[type="text"],
input[type="password"],
input[type="number"] {
display: block;      width:
calc(100% - 22px);      margin:
10px 0;      padding: 15px;
border: 2px solid #ccc;
border-radius: 10px;      font-
size: 1em;      background-
color: #fafafa;
}      input[type="button"], button {
display: block;      width: 100%;
margin: 10px 0;      padding: 15px;
border: none;      border-radius: 10px;
font-size: 1.2em;      cursor: pointer;
transition: background-color 0.3s;
}
```



```
    section#F01
input[type="button"], section#F02
input[type="button"], section#F03
input[type="button"], section#F04
input[type="button"], section#F05
input[type="button"] {
background-color: #333;
color: #fff;
}  section#F01
input[type="button"]:hover, section#F02
input[type="button"]:hover,
```



```

section#F03 input[type="button"]:hover,
section#F04 input[type="button"]:hover,
section#F05 input[type="button"]:hover {
background-color: #555;
} section#F06
input[type="button"], section#F07
input[type="button"], section#F08
input[type="button"], section#F09
input[type="button"], section#F10
input[type="button"],
#btnModificarStock1,
#btnModificarStock2,
#btnModificarStock3,
#btnPausar1,
#btnPausar2,
#btnActivar3,
#btnAsignarOferta1,
#btnAsignarOferta3,
#btnQuitarOferta2 {
background-color: red;
color: #fff;
} section#F06
input[type="button"]:hover, section#F07
input[type="button"]:hover, section#F08
input[type="button"]:hover, section#F09
input[type="button"]:hover, section#F10
input[type="button"]:hover,
#btnModificarStock1:hover,
#btnModificarStock2:hover,
#btnModificarStock3:hover,
#btnPausar1:hover,
#btnPausar2:hover,

```



```
#btnActivar3:hover,
```



```
#btnAsignarOferta1:hover,  
#btnAsignarOferta3:hover,  
#btnQuitarOferta2:hover {  
background-color: darkred; }  
  
#btnCancelarProducto2 { background-  
color: red;  
}  
  
#btnCancelarProducto2:hover {  
background-color: darkred;
```



```

}

#btnAsignarOferta1,
#btnAsignarOferta3 {
background-color: #333;
}

#btnAsignarOferta1: hover,
#btnAsignarOferta3: hover,
#btnQuitarOferta2: hover {
background-color: #555;
}

#btnQuitarOferta2 {
background-color: #333;
}

#btnQuitarOferta2: hover {
background-color: #555;
} section#F01, section#F02,
section#F03, section#F04,
section#F05 { border-left:
10px solid #333;
} section#F06, section#F07,
section#F08, section#F09,
section#F10 { border-left:
10px solid red;
}

```



```
}  
  
p {  
    border-bottom: 1px solid #ddd;  
    padding: 10px 0;  
}    p:last-child {  
border-bottom: none;  
}  
  
#estadoAprobado1,
```



```

#estadoAprobado3      {
color: green;
}

#estadoPendiente2 {
    color: red;
}

#precio1, #precio2 {
color: green;    font-
weight: bold;
} p {    color: #333;
font-size:    1em;
text-align: center;
}

#msgVerProductos      {
display: flex;    justify-
content: center;    width:
100%;    overflow-x: auto;
margin-top: 20px;
} table {    width: 100%;    border-collapse: collapse;
margin: 0 auto;    font-size: 1em;    font-family:
'Comic Sans MS', 'Comic Sans', cursive; min-width: 400px;

```



```
border-radius: 10px 10px 0 0; overflow:
hidden;
    box-shadow: 0 0 20px rgba(0, 0, 0, 0.15);
}
table    thead    tr    {
background-color:    #333;
color:    white;    text-
align:    left;    font-
weight:    bold;
}
```



```

table th, table td {
padding: 6px 7px;
text-align: center;
} table tbody tr { border-
bottom: 1px solid #dddddd;
} table tbody tr:nth-of-
type(even) { background-color:
#f3f3f3;
} table tbody tr:last-of-type {
border-bottom: 2px solid #333;
} table tbody tr.active-row
{ font-weight: bold;
color: #333;
} table img { max-
width: 100px;
height: auto;
display: block;
margin: 0 auto;
}
input[type="number"] {
width: 60px;
} input[type="button"] {
background-color:
#333;

```



```
color: white; border:  
none; padding: 10px  
20px; cursor: pointer;
```







```
font-size: 16px;  
border-radius: 5px;    transition: background-  
color 0.3s, transform 0.3s;
```

```

}
input[type="button"]:hover {
background-color: #555;
transform: scale(1.1);
}

/* Estilo para los mensajes correctos */
.correct { color: green;
font-weight: bold; margin-
top: 5px; margin-bottom:
10px; background-color:
#d4edda; border: 1px solid
#c3e6cb; padding: 5px;
border-radius: 5px; font-
size: 0.9em; text-align:
left; display: none;
}

/* Estilo para los mensajes de error */
.error { color: red;
font-weight: bold; margin-
top: 5px; margin-bottom:
10px; background-color:
#f8d7da; border: 1px solid
#f5c6cb; padding: 5px;
border-radius: 5px; font-
size: 0.9em; text-align:
left; display: none;
}

/* Estilo para los mensajes de éxito */
.success {
color: blue;

```

```

font-weight: bold; margin-
top: 5px; margin-bottom: 10px;

```

```
background-color: #d1ecf1;  
border: 1px solid #bee5eb;  
padding: 5px; border-  
radius: 5px;
```

```

        font-size: 2.1em;
text-align:    left;
display: none;
}

/* Estilo para los campos de entrada con errores */
.input-error {    border:
2px solid red;
}

/* Estilo para los campos de entrada correctos */
.input-correct {    border:
2px solid green;
}

/* Estilo general para los elementos select cuando están en foco */
select:focus {    border-color: #333;    background-color: #e0e0e0;
outline: none;    height: 40px; /* Ajusta la altura del select */
width: 200px; /* Ajusta el ancho del select */    font-size: 16px;
/* Ajusta el tamaño de la fuente del select */    padding: 5px; /*
Añade padding para mejorar la legibilidad */ }

/* Estilo general para los elementos select cuando están en hover */
select:hover {    border-color: #555;    background-color: #f0f0f0;
height: 40px; /* Ajusta la altura del select */    width: 200px; /*
Ajusta el ancho del select */    font-size: 16px; /* Ajusta el tamaño
de la fuente del select */    padding: 5px; /* Añade padding para
mejorar la legibilidad */ }

/* Estilo específico para los elementos select dentro de las secciones de
administración */ section#F08 select, section#F09 select {    background-
color: #fff;    border-color: red;    height: 25px; /* Reduce la altura
del select */ width: 120px; /* Reduce el ancho del select */

```



```

    font-size: 12px; /* Reduce el tamaño de la fuente del select */
padding: 1px; /* Reduce el padding para mejorar la legibilidad */ }

/* Estilo específico para los elementos select dentro de las secciones
de administración cuando están en foco */ section#F08 select:focus,
section#F09 select:focus {    border-color: darkred;    background-
color: #ffe6e6;    height: 25px; /* Reduce la altura del select */
width: 120px; /* Reduce el ancho del select */    font-size: 12px; /*
Reduce el tamaño de la fuente del select */    padding: 1px; /* Reduce
el padding para mejorar la legibilidad */ }

/* Estilo específico para los elementos select dentro de las secciones
de administración cuando están en hover */ section#F08 select:hover,
section#F09 select:hover {    border-color: darkred;    background-
color: #ffc000;    height: 25px; /* Reduce la altura del select */
width: 120px; /* Reduce el ancho del select */    font-size: 12px; /*
Reduce el tamaño de la fuente del select */    padding: 1px; /* Reduce
el padding para mejorar la legibilidad */ }

/* Estilo general para los párrafos */ p {    color:
#333;    font-size: 1em;    text-align: center;
margin: 10px 0;    padding: 10px;    border-
radius: 5px;    background-color: #f9f9f9;
border: 1px solid #ddd;    box-shadow: 0 2px 4px
rgba(0, 0, 0, 0.1);    transition: background-
color 0.3s, color 0.3s;
}

/* Estilo para los párrafos en hover
*/ p:hover {    background-color:
#e0e0e0;

```



```
    color: #000;
}

/* Estilo para los mensajes de error dentro de párrafos */
p.error {    background-color: #f8d7da;    border-color:
#f5c6cb;    color: #721c24;
}

/* Estilo para los mensajes de éxito dentro de párrafos */
p.success {    background-color: #d4edda;    border-color:
#c3e6cb;    color: #155724;
}

/* Estilo para los mensajes informativos dentro de párrafos */
p.info {    background-color: #d1ecf1;    border-color:
#bee5eb;    color: #0c5460;
}

/* Estilo para los mensajes de advertencia dentro de párrafos */
p.warning {    background-color: #fff3cd;    border-color:
#ffeeba;    color: #856404;
}
```



## Listado de información precargada

### Administradores precargados

1. **Usuario:** fRobayna
  - **Contraseña:** 12345
2. **Usuario:** alangelan
  - **Contraseña:** 12345
3. **Usuario:** mGonzalez
  - **Contraseña:** 12345
4. **Usuario:** cMartinez
  - **Contraseña:** 12345
5. **Usuario:** lFernandez
  - **Contraseña:** 12345

### Compradores precargados

1. **Nombre:** Matias Perez
  - **Usuario:** MatiasPro
  - **Contraseña:** 12345A
  - **Tarjeta de Crédito:** 5555555555554444
  - **Código de Seguridad:** 333
2. **Nombre:** Felipe Rodríguez
  - **Usuario:** FeliRodriguez
  - **Contraseña:** 12345B
  - **Tarjeta de Crédito:** 5105105105105100
  - **Código de Seguridad:** 435
3. **Nombre:** Juan Alpaca
  - **Usuario:** JuanitoKratos
  - **Contraseña:** 12345C
  - **Tarjeta de Crédito:** 5105105105105108
  - **Código de Seguridad:** 123
4. **Nombre:** Ana Lopez
  - **Usuario:** AnaL123
  - **Contraseña:** 12345D
  - **Tarjeta de Crédito:** 4012888888881881
  - **Código de Seguridad:** 456
5. **Nombre:** Laura Martinez
  - **Usuario:** LauMartinez
  - **Contraseña:** 12345E
  - **Tarjeta de Crédito:** 4111111111111111
  - **Código de Seguridad:** 789

## Productos precargados

1. **Nombre:** Campera de Nacional
  - **Descripción:** Campera del club más grande del mundo
  - **Precio:** 5000
  - **Stock:** 5
  - **Imagen:** img/camperonNacional.png
2. **Nombre:** Mochila de Nacional
  - **Descripción:** Mochila del club más grande del mundo
  - **Precio:** 1750
  - **Stock:** 9
  - **Imagen:** img/mochilaNacional.png
3. **Nombre:** Short de Nacional
  - **Descripción:** Short del club más grande del mundo
  - **Precio:** 1299
  - **Stock:** 12
  - **Imagen:** img/shortNacional.png
4. **Nombre:** Gorro de Nacional
  - **Descripción:** Gorro del club más grande del mundo
  - **Precio:** 790
  - **Stock:** 8
  - **Imagen:** img/gorroNacional.png
5. **Nombre:** Medias de Nacional
  - **Descripción:** Medias del club más grande del mundo
  - **Precio:** 350
  - **Stock:** 6
  - **Imagen:** img/mediasNacional.png
6. **Nombre:** Remera de Nacional
  - **Descripción:** Remera del club más grande del mundo
  - **Precio:** 2500
  - **Stock:** 10
  - **Imagen:** img/remeraNacional.png
7. **Nombre:** Buzo de Nacional
  - **Descripción:** Buzo del club más grande del mundo
  - **Precio:** 4500
  - **Stock:** 7
  - **Imagen:** img/buzoNacional.png
8. **Nombre:** Pantalón de Nacional
  - **Descripción:** Pantalón del club más grande del mundo
  - **Precio:** 3200
  - **Stock:** 4
  - **Imagen:** img/pantalonNacional.png
9. **Nombre:** Camiseta de Nacional
  - **Descripción:** Camiseta del club más grande del mundo
  - **Precio:** 2800
  - **Stock:** 15
  - **Imagen:** img/camisetaNacional.png
10. **Nombre:** Botines de Nacional
  - **Descripción:** Botines del club más grande del mundo
  - **Precio:** 6000

- **Stock:** 3
- **Imagen:** img/botinesNacional.png

### **Compras precargadas**

1. **ID de Compra:** 1
  - **ID de Producto:** Prod\_ID\_4
  - **Cantidad:** 2
  - **Precio Unitario:** 5000
  - **Estado:** Pendiente
2. **ID de Compra:** 2
  - **ID de Producto:** Prod\_ID\_3
  - **Cantidad:** 1
  - **Precio Unitario:** 1750
  - **Estado:** Pendiente
3. **ID de Compra:** 3
  - **ID de Producto:** Prod\_ID\_1
  - **Cantidad:** 3
  - **Precio Unitario:** 1299
  - **Estado:** Pendiente
4. **ID de Compra:** 4
  - **ID de Producto:** Prod\_ID\_2
  - **Cantidad:** 5
  - **Precio Unitario:** 3200
  - **Estado:** Pendiente
5. **ID de Compra:** 5
  - **ID de Producto:** Prod\_ID\_5
  - **Cantidad:** 2
  - **Precio Unitario:** 4500
  - **Estado:** Pendiente