# 

INSTITUTO PROFESIONAL DUOC

SEDE MAIPÚ

ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES

Carnes Ttamiña

| 1. Antecedentes Personales |
| --- |

| Nombre estudiante | **Benjamin Navarro / Cristopher Urbina / Vicente Guerra** |
| --- | --- |
| Rut | **21.267.181-9 / 21.328.053-8 / 20.825.188-0** |
| Carrera | **Ingenieria en Informatica** |
| Sede | **Maipú** |

Índice

[**1. Vista Lógica 3**](#_tsctmhmahexl)

[Componentes Principales 3](#_788d5nrau4yy)

[**2. Vista de desarrollo 5**](#_mhhu18phdovv)

[**3. Vista de Procesos 7**](#_un0ss056m187)

## 1. Vista Lógica

Nuestro sitio web de **Carnes Ttamiña** se organiza en módulos que representan las operaciones clave de la carnicería.

#### **Componentes Principales**

* **Gestión de Usuarios:** permite el registro, inicio y cierre de sesión. Cada usuario está vinculado a uno o más roles a través de la tabla intermedia usuario\_rol.
* **Gestión de Roles:** define los permisos y accesos del sistema. El rol *Administrador* puede gestionar productos e inventario; el rol *Cliente* puede navegar, comprar y revisar pedidos.
* **Gestión de Productos:** administra los artículos disponibles, con información detallada (nombre, categoría, precio, unidad de medida, imagen y stock).
* **Gestión de Clientes:** administra la información detallada de los clientes, como nombre o razón social, RUT, dirección, correo y teléfono. Distingue si el cliente es persona natural o restaurante, y se relaciona directamente con los pedidos registrados.
* **Catálogo y Navegación:** permite explorar los productos por categorías, aplicar filtros y buscar por nombre o palabra clave.
* **Carrito de Compras:** facilita la selección de productos, permitiendo agregar, eliminar y modificar ítems.
* **Gestión de Pedidos:** procesa las compras de los clientes y registra los pedidos en la base de datos Supabase.
* **Panel de Administración:** reservado a usuarios con rol de administrador; permite crear, modificar y eliminar productos directamente desde la interfaz.
* **Validación de Stock:** evita compras con stock insuficiente mediante consultas dinámicas antes de confirmar el pedido.

Entidades Principales (Modelo Relacional Simplificado)

| Entidad | Atributos principales | Descripción |
| --- | --- | --- |
| Rol | rol\_id (PK)  nombre | Define los niveles de acceso del sistema (Cliente, Administrador). |
| usuario\_rol | usuario\_id(PK)  rol\_id(FK) | Tabla intermedia que relaciona usuarios con roles. |
| usuario | usuario\_id (PK)  cliente\_id (FK)  email  hash\_password nombre  zona\_horaria  activo  creado\_en  auth\_user\_id | Representa a cada persona registrada en la plataforma. |
| producto | producto\_id (PK) nombre descripcion categoria unidad precio stock activo creado\_en imagen\_url | Almacena los datos de los productos disponibles para la venta. |
| pedido | pedido\_id (PK)  cliente\_id (FK)  fecha, estado , metodo\_pago  envío  subtotal  descuento  total, creado\_por (FK) actualizado\_en | Representa la compra efectuada por un usuario. |
| pedido \_item | item\_id (PK)  pedido\_id (FK)  producto\_id (FK) cantidad  precio\_unitario total\_linea, unidad | Detalla los productos y cantidades incluidos en cada pedido. |

## 2. Vista de desarrollo

El desarrollo del sistema web **“Carnes Ttamiña”** se implementó utilizando HTML, CSS y JavaScript como base para el frontend, junto con Supabase como servicio backend (Base de Datos PostgreSQL y autenticación).



**Tecnologías utilizadas:**

* **HTML5:** estructura y contenido del sitio web.
* **CSS3:** diseño visual y maquetación responsive.
* **JavaScript (ES6):** control de lógica del cliente, manipulación del DOM y consumo de la API de Supabase.
* **Supabase (PostgreSQL + Auth):** gestión de base de datos, autenticación y políticas de acceso.
* **LocalStorage:** almacenamiento de datos temporales del carrito en el navegador.
* **GitHub:** control de versiones y respaldo del código fuente.

**Organización del código:** El sistema se desarrolló de forma modular, separando responsabilidades por archivo.

* Los scripts en la carpeta **js/** controlan la lógica funcional (usuarios, roles, carrito y productos).
* Los estilos en **css/** permiten mantener una interfaz visual coherente y fácil de mantener.
* Los archivos HTML se encargan de estructurar las distintas secciones del sitio (inicio, catálogo, carrito, acceso, registro y administración).

**Integración con Supabase:** La conexión con Supabase se realiza mediante el archivo supabaseClient.js, donde se definen las claves del proyecto y los métodos de autenticación.  
 A través de las funciones de Supabase, el sistema puede:

* Validar usuarios registrados.
* Consultar roles asignados a través de la tabla usuario\_rol.
* Obtener y actualizar información de productos.
* Registrar pedidos y sus detalles en las tablas pedido y pedido\_item.

**Buenas prácticas aplicadas:**

* Uso de variables y funciones reutilizables para evitar duplicación de código.
* Separación de la lógica de negocio (JavaScript) del diseño visual (CSS y HTML).
* Validación de roles en el frontend para restringir el acceso a las páginas de administración.
* Implementación de control de errores en las consultas a Supabase.
* Persistencia del carrito mediante localStorage para mantener la experiencia del usuario.

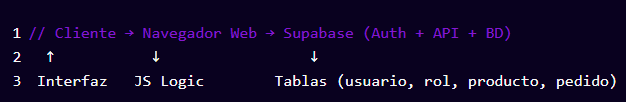
## 3. Vista de Procesos

En la vista de procesos se describe cómo interactúan los distintos componentes del sistema en tiempo de ejecución.

En el proyecto “Carnes Ttamiña”, los procesos principales se desarrollan entre el usuario **(cliente o administrador)**, el navegador web y Supabase como backend.

**Flujo general del sistema:**

1. El usuario accede al sitio web desde su navegador.
2. Al iniciar sesión, el sistema envía las credenciales a Supabase Auth.
3. Supabase valida los datos y devuelve la sesión activa.
4. El navegador obtiene el rol del usuario desde la tabla usuario\_rol.
5. Según el rol, se muestran u ocultan funciones del menú (por ejemplo, acceso al panel de administración).
6. El usuario puede navegar, agregar productos al carrito y procesar su pedido.
7. Antes de confirmar una compra, el sistema consulta el stock actual del producto.
8. Los datos del pedido y sus ítems se registran en las tablas pedido y pedido\_item.
9. El administrador puede ver, actualizar o eliminar productos directamente desde su panel.



## 

## **Procesos clave:**

## Autenticación de usuario.

## Asignación y validación de roles.

## Gestión de productos y stock.

## Registro de pedidos.

## Persistencia de carrito.

## 

## 

## 

## 4.Vista física

## 

## 5. Escenarios

**Escenario 1:** Compra de productos por un cliente

**Actor principal:** Cliente registrado

**Descripción:** El cliente accede al sitio, selecciona productos, los añade al carrito y confirma la compra.

**Flujo Principal**

1. El cliente inicia sesión mediante Supabase Auth.
2. Explora el catálogo de productos (vista catálogo).
3. Agrega productos al carrito, donde se calculan subtotales y cantidades.
4. El sistema consulta la base de datos para validar disponibilidad de stock.
5. Al confirmar, se registran los datos en las tablas pedido y pedido\_item.
6. Supabase responde con éxito y se muestra un mensaje de confirmación.

**Validación arquitectónica:**

* Verifica la comunicación entre **Frontend (HTML/JS)**, **Supabase Auth** y **Database (PostgreSQL)**.
* Comprueba la integridad del flujo de compra y el control de stock.
* Valida la persistencia del carrito y el registro de pedidos.

**Escenario 2:** Administración de productos por el empleado

**Actor principal:** Administrador (rol Empleado)

**Descripción:** El administrador gestiona el inventario y los productos disponibles desde el panel de control.

**Flujo Principal:**

1. Inicia sesión con rol de Administrador.
2. Accede al menú de gestión (menu.html).
3. Visualiza la lista de productos con su stock y precios actuales.
4. Modifica o agrega nuevos productos mediante el formulario correspondiente.
5. Supabase actualiza los registros en la tabla producto.
6. El catálogo principal se actualiza automáticamente para los clientes.

**Validación arquitectónica:**

* Verifica la **asignación de roles** y restricciones de acceso.
* Confirma la correcta integración entre **interfaz administrativa**, **Supabase Database** y **políticas RLS**.
* Asegura que los cambios se reflejan dinámicamente en el frontend.

**Escenario 3:** Control de stock automático

**Actor principal:** Administrador (rol Empleado)

**Descripción:** El administrador gestiona el inventario y los productos disponibles desde el panel de control.

**Flujo principal:**

1. Cuando un cliente confirma una compra, el sistema consulta la cantidad disponible en la tabla producto.
2. Si el stock es suficiente, se descuenta la cantidad correspondiente.
3. Si no hay stock suficiente, se muestra un mensaje de error al usuario y la operación se cancela.
4. El administrador puede revisar posteriormente el inventario desde el panel de control.

**Validación arquitectónica:**

* Comprueba la **lógica de negocio del sistema** y su independencia del rol del usuario.
* Demuestra el **control de consistencia de datos** entre pedidos y productos.
* Garantiza la **integridad referencial** en las tablas pedido\_item y producto.

**Escenario 4 (opcional):** Registro y gestión de usuarios

**Actor principal:** Nuevo usuario

**Descripción:** Un visitante se registra en la plataforma para realizar compras.

**Flujo principal:**

1. El usuario completa el formulario de registro.
2. Supabase Auth crea el nuevo registro y genera un ID único.
3. El sistema asigna automáticamente el rol “Cliente” en la tabla usuario\_rol.
4. El usuario puede iniciar sesión y comenzar a comprar.

**Validación arquitectónica:**

* Valida la integración entre **autenticación y base de datos relacional**.
* Comprueba la asignación automática de roles.
* Garantiza la seguridad del proceso de registro.