

Manual Técnico – DartNova

(Esta información puede cambiar a lo largo del desarrollo)

1. Introducción

El presente manual tiene como propósito proporcionar una guía exhaustiva para la instalación, configuración, operación, mantenimiento y resolución de problemas de la plataforma DartNova.

El manual está dirigido a:

Desarrolladores, que requieren conocer la arquitectura y dependencias.

Administradores de sistemas, que deben instalar, configurar y mantener la aplicación en entornos Windows con PostgreSQL y DevContainers.

Soporte técnico, que necesita diagnosticar y resolver problemas comunes en producción.

2. Arquitectura del Sistema

2.1 Vista General

La arquitectura de DartNova está basada en una arquitectura cliente-servidor con separación entre frontend y backend, soportada en contenedores para facilitar la portabilidad y escalabilidad.

2.2 Componentes

Cliente Web: desarrollado con HTML, CSS y JavaScript, ejecutado en navegadores modernos.

Cliente Móvil: desarrollado en Flutter, compatible con Android e iOS

3. Requisitos Técnicos

3.1 Software (Windows)

- Windows 10/11 Pro o Windows Server 2019/2022.
- Docker Desktop (última versión estable).
- Visual Studio Code con la extensión Remote – Containers.
- PostgreSQL 14+ instalado en contenedor o directamente en Windows.
- Git para control de versiones.

3.2 Dependencias dentro del DevContainer

- Ruby 3.2+

- Rails 7.x
- Node.js 18+
- Yarn
- Bundler
- PostgreSQL client (para interacción con la BD).

4. Instalación y Configuración

4.1 Clonar el repositorio

<https://github.com/jpso2406/ZONA-44-.git>

5. Operación Interna

5.1 Flujo Cliente

- Accede a la app/web.
- Explora menú → selecciona productos → agrega al carrito.
- Envía pedido → se registra en BD.
- Se envía confirmación vía correo.

5.2 Flujo Administrador

- Inicia sesión en /admin.
- Gestiona grupo, productos y promociones.
- Revisa pedidos en tiempo real.

5.3 Estructura de Base de Datos

Tabla	Descripción
users	Almacena clientes y administradores
products	Platillos con sus atributos
orders	Pedidos realizados
order_items	Detalle de productos por pedido

6. Resolución de Problemas

Error	Causa	Solución
500 en login	Llave secreta inválida	Regenerar con rails secret
BD no conecta	env mal configurado	Verificar credenciales
Flutter falla	SDK viejo	flutter upgrade
Docker no levanta	Puerto ocupado	Cambiar ports en docker-compose.yml

7. Seguridad

- Contraseñas: bcrypt.
- Comunicación: SSL/TLS en producción.
- Roles: cliente y administrador.
- PostgreSQL: habilitar **pg_hba.conf** solo para usuarios autorizados.
- Contenedores: no exponer puertos innecesarios.

8. Buenas Prácticas

- Separar entornos (dev, test, prod).
- Usar CI/CD para despliegues.
- Monitorear contenedores con Prometheus + Grafana.
- Mantener dependencias actualizadas.
- Ejecutar tests automáticos antes de cada reléase.

9. Apéndices

Comandos Rails

- rails c # Consola
- rails db:migrate # Migrar
- rails db:seed # Datos iniciales
- rails s # Servidor

Comandos Docker

- docker ps
- docker-compose up -d
- docker-compose down

10. Glosario

DevContainer: entorno aislado para desarrollo en VS Code.

JWT: autenticación para móviles.

Seed: carga inicial de datos.

SMTP: protocolo para envío de correos.

UML : acrónimo de Unified Modeling Language . Lenguaje de modelado para proceso de diseño de sistemas.

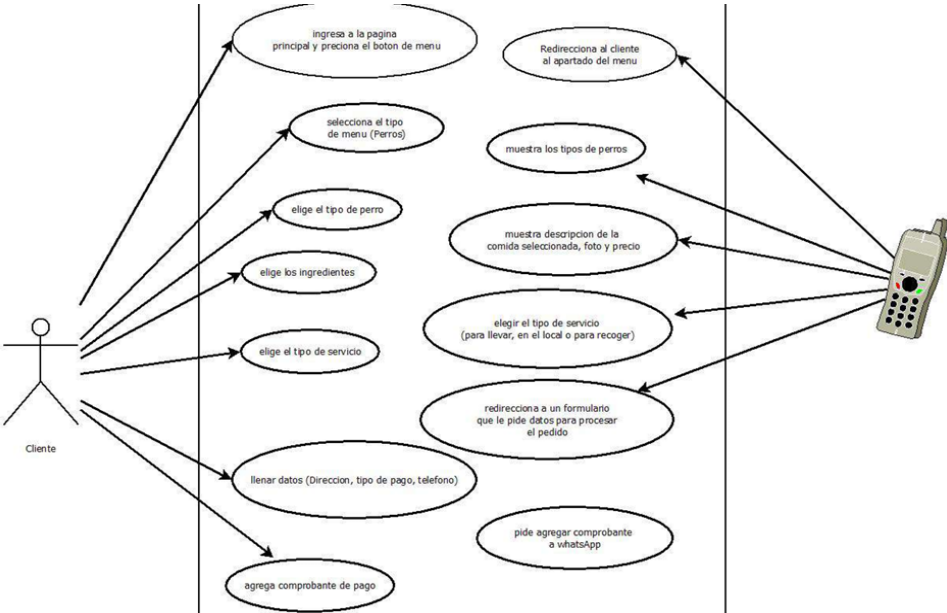
Database: conjunto de información o datos organizados y almacenados electrónicamente en un sistema informático, como un ordenador o la nube.

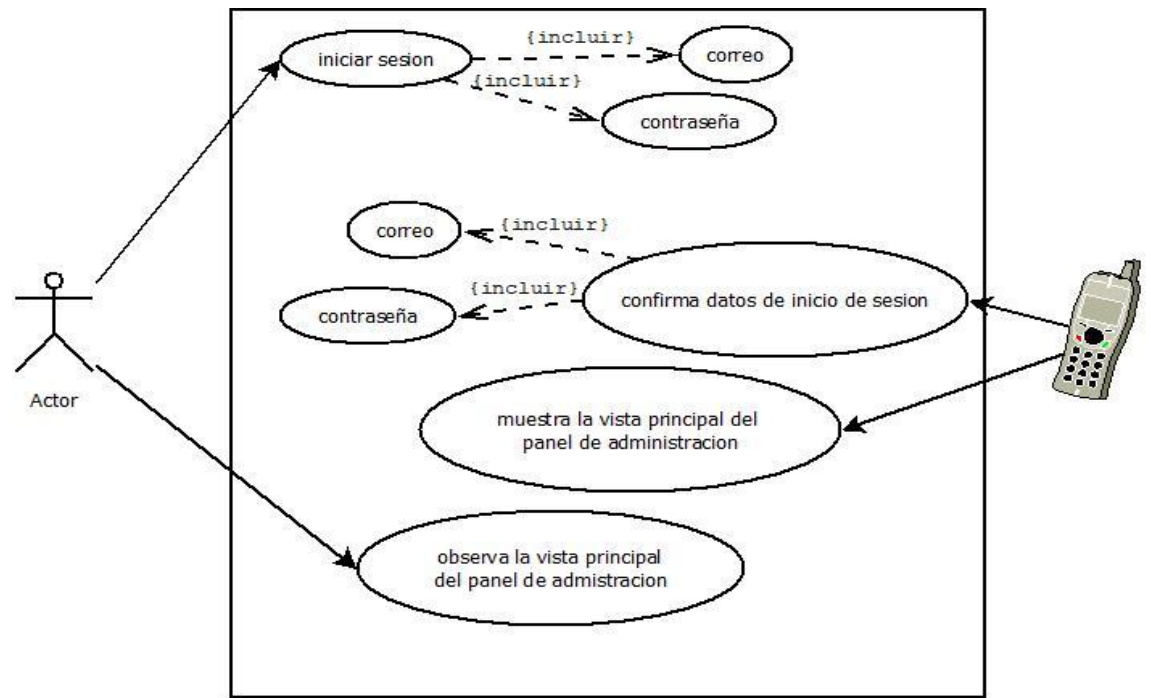
Ruby on rails: popular framework de código abierto para aplicaciones web de backend , escrito en el lenguaje de programación ruby.

Flutter : Es un kit de desarrollo de software (SDK) de código abierto creado por Google que permite crear aplicaciones para múltiples plataformas (iOS , Android, web, Escritorio)

Dependencias : Es un componente de software que tu aplicación necesita para poder funcionar como una biblioteca de software o un complemento

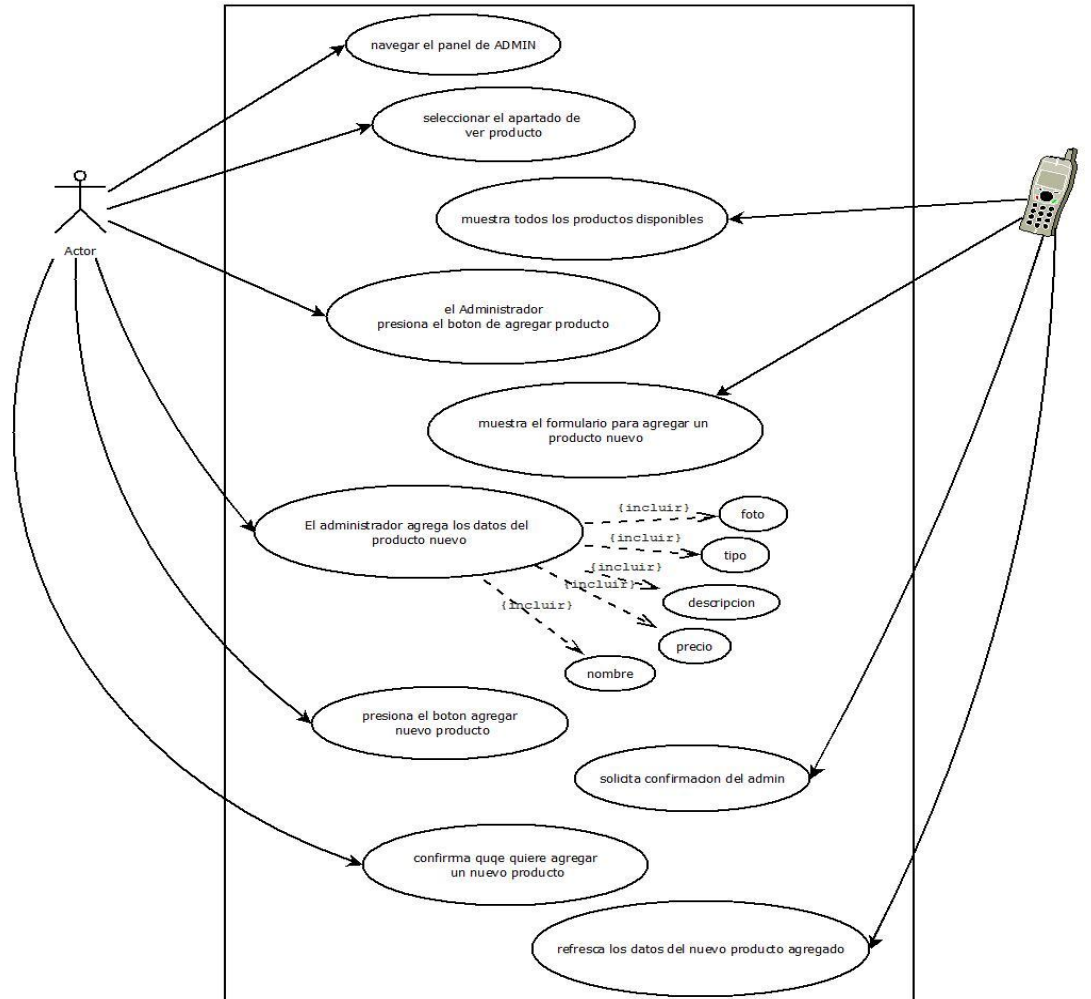
Active Récord: es la implementación del patrón Object-Relational Mapping (ORM) en Ruby on Rails, que permite interactuar con bases de datos relacionales utilizando objetos de Ruby en lugar de escribir SQL directame

No	Descripción	Detalle
1	Alcance	DartNova es una plataforma de pedidos en línea para gastrobar Zona44 ofrece tanto una aplicación web como una aplicación móvil.
2	Descripción de procesos	<p style="text-align: center;">Caso de uso Dartnova</p> <p>1. Como realizar un pedido por la página web</p>  <p>El usuario ingresa a la página principal y presiona el botón menú</p> <ol style="list-style-type: none"> 1.1 la plataforma redirecciona al cliente al apartado del menú 1.2 El usuario selecciona el tipo de menú (Perros, Salchipapas, Hamburguesas etc.) 1.3 La plataforma muestra en pantalla el menú requerido por el cliente 1.4 El usuario elige el producto de su preferencia 1.5 La plataforma muestra una breve descripción del producto, imagen y precio 1.6 El usuario elige los ingredientes con los que quiere acompañar el producto (Pepperoni, Lechuga, Tomate etc.) 1.7 La plataforma le muestra dos opciones en pantalla, para llevar o comer aquí 1.8 El usuario elige el tipo de servicio 1.9 La plataforma lo redirecciona al formulario para el proceso de pago 2.0 El Usuario llena los campos requeridos 2.1 La plataforma pide enviar el comprobante por vía WhatsApp 2.2 Después de agregar el comprobante de pago la plataforma le mandara una notificación cuando el pedido esté listo



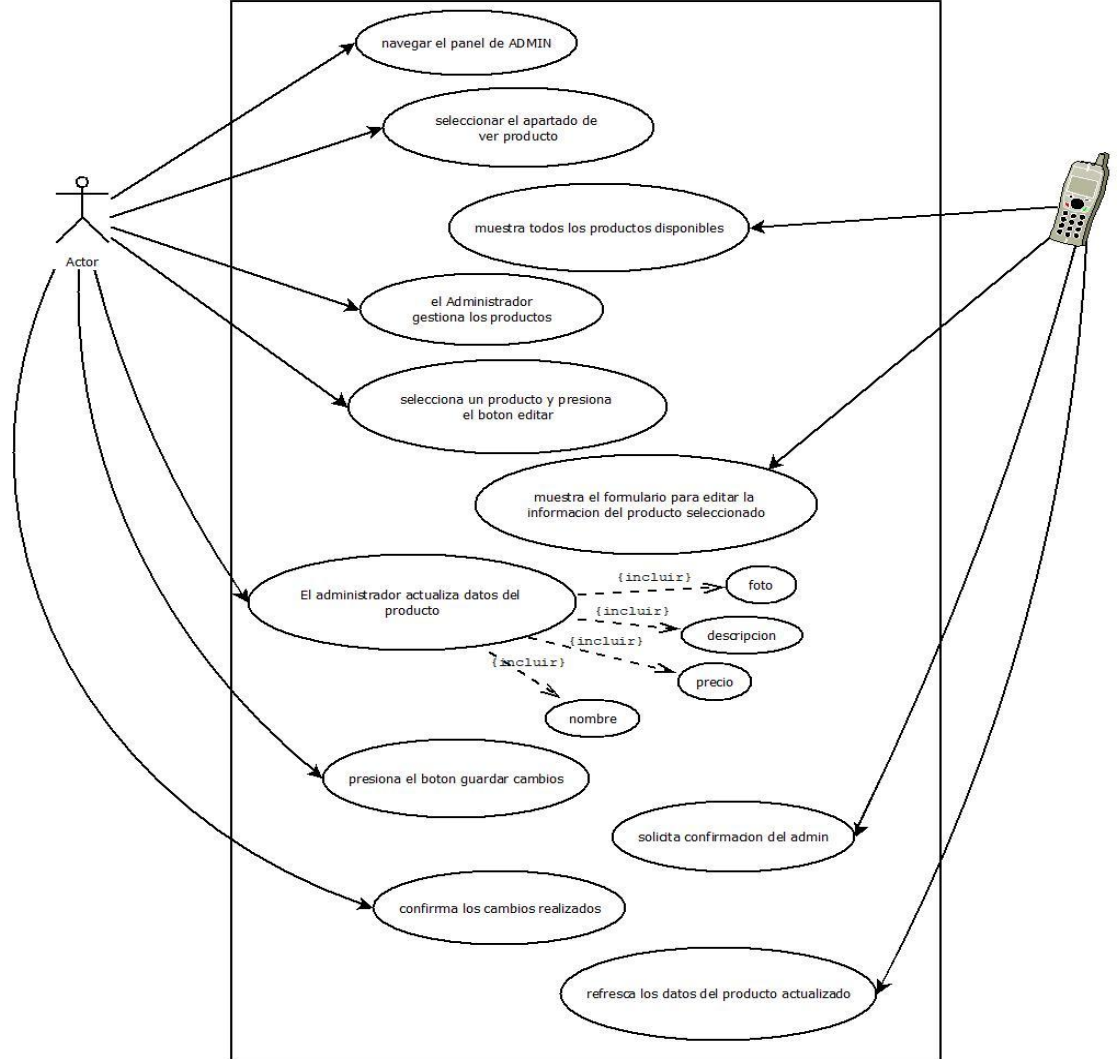
		NOMBRE: CASO 03- inicio de sesion admin	
		AUTOR: Santiago Filigrana	
		ACTOR: Admin y Panel de admistracion.	
		PRECONDICIÓN	
		1	El Admin debe tener correo para iniciar sesion.
		2	El Admin debe tener la contraseña para iniciar sesion.
		3	El Admin debe tener celular o computador con internet.
		FLUJO NORMAL	
		1	El Admin introduce su correo y contraseña para inicar sesion.
		2	El sistema resive y valida los datos de inicio de sesion.
		3	El sistema verifica que los datos son correctos y permite el ingreso al panel de admistracion.
		4	El sistema muestra el home y las opciones del panel de administracion.
		5	El Admin navega por el apartado de Administracion.
		FLUJO ALTERNATIVO	
		1.1	El Amin no ingreso correctamente el correo.
		2.1	El Amin no ingreso correctamente la contraseña.
		3.1	El sistema no valida datos.
		3.2	El sistema toma como incorrectos los datos.
		3.3	El sistema no deja ingresar al Admin .
		6.1	El sistema no carga la pantalla.
		7.1	El sistema no muestra la opcion del panel de admistracion.
		9.1	El sistema no deja ver productos.
		POSTCONDICIONES	
		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.
		2	El Admin puede navegar y usar correctamente el panel de administracion.

07 agregar producto



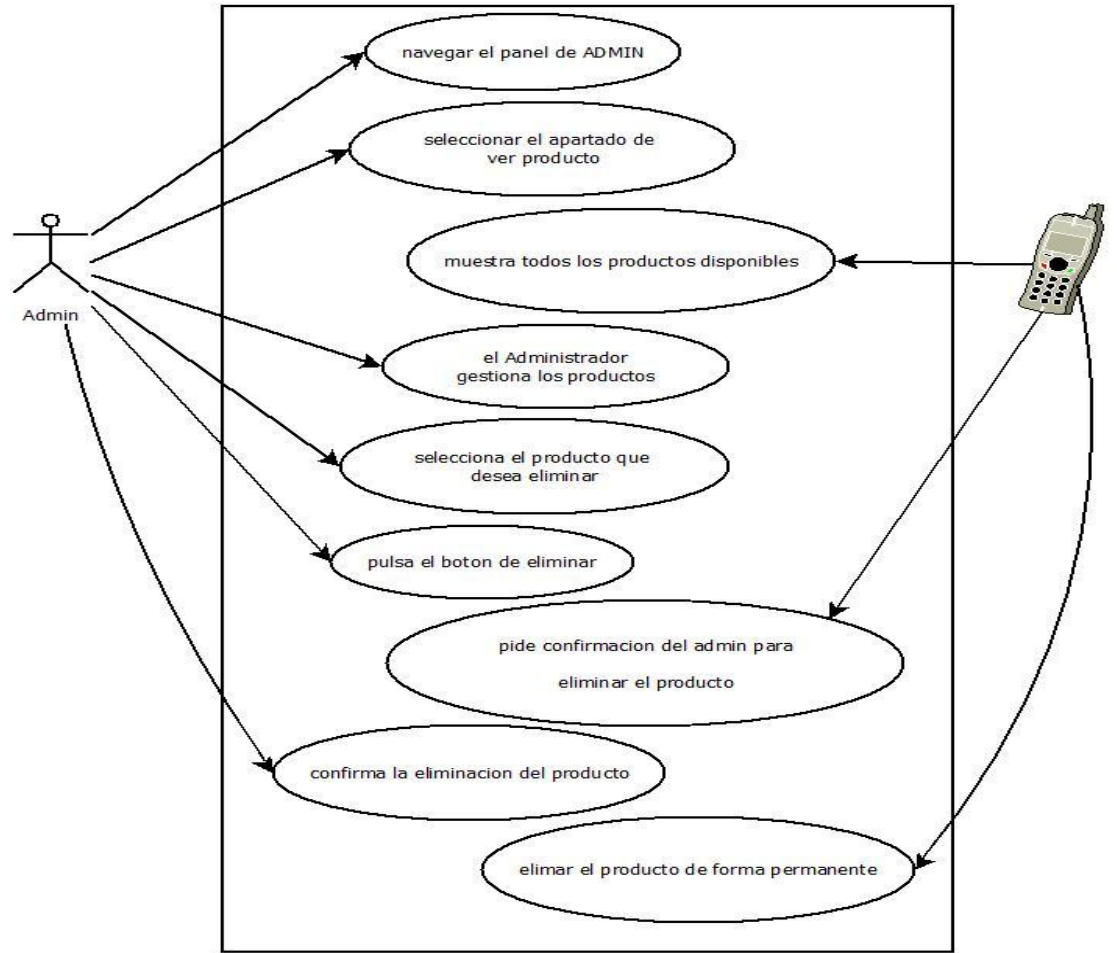
		NOMBRE: CASO 07 - editar producto	
		AUTOR: Santiago Filigrana	
		ACTOR: Admin y Panel de administracion.	
		PRECONDICIÓN	
		1	El Admin debe tener correo y contraseña para iniciar sesion.
		2	El Admin debe tener celular o computador con internet.
		3	El Admin debe tener la necesidad de agregar un nuevo producto.
		FLUJO NORMAL	
		1	El Admin despues de iniciar sesion correctamente se encuentra en el apartado principal del panel de administracion.
		2	El sistema muestra las diferentes opciones del panel de administrador.
		3	El Admin selecciona la vista de productos.
		4	El sistema muestra los productos disponibles ordenados por grupos.
		5	El Admin presiona el boton de agregar un nuevo productos.
		6	El sistema muestra un formulario con los datos para agregar un nuevo producto producto (nombre, precio, descripcion, tipo, foto del producto).
		7	el Admin agrega los datos en los campos necesarios para agregar un nuevo producto y pulsa el boton de "agregar nuevo producto".
		8	El Sistema le pide confirmacion al admin para agregar el producto.
		9	El Admin confirma que quiere agregar el nuevo producto.
		10	El sistema agrega el nuevo producto y los refleja en la pagina.
		FLUJO ALTERNATIVO	
		1.1	El Admin no ingreso correctamente el correo.
		2.1	El Admin no ingreso correctamente la contraseña.
		3.1	El sistema no valida datos.
		3.2	El sistema toma como incorrectos los datos.
		3.3	El sistema no deja ingresar al Admin .
		6.1	El sistema no carga la pantalla.
		7.1	El sistema no muestra la opcion de agregar productos en el panel de administracion.
		9.1	El sistema no permite agregar los productos.
		POSTCONDICIONES	
		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.
		2	El Admin puede navegar y usar correctamente el panel de administracion.
		3	El Admin puede agregar y guardar el nuevo producto de forma correcta.

06 editar producto



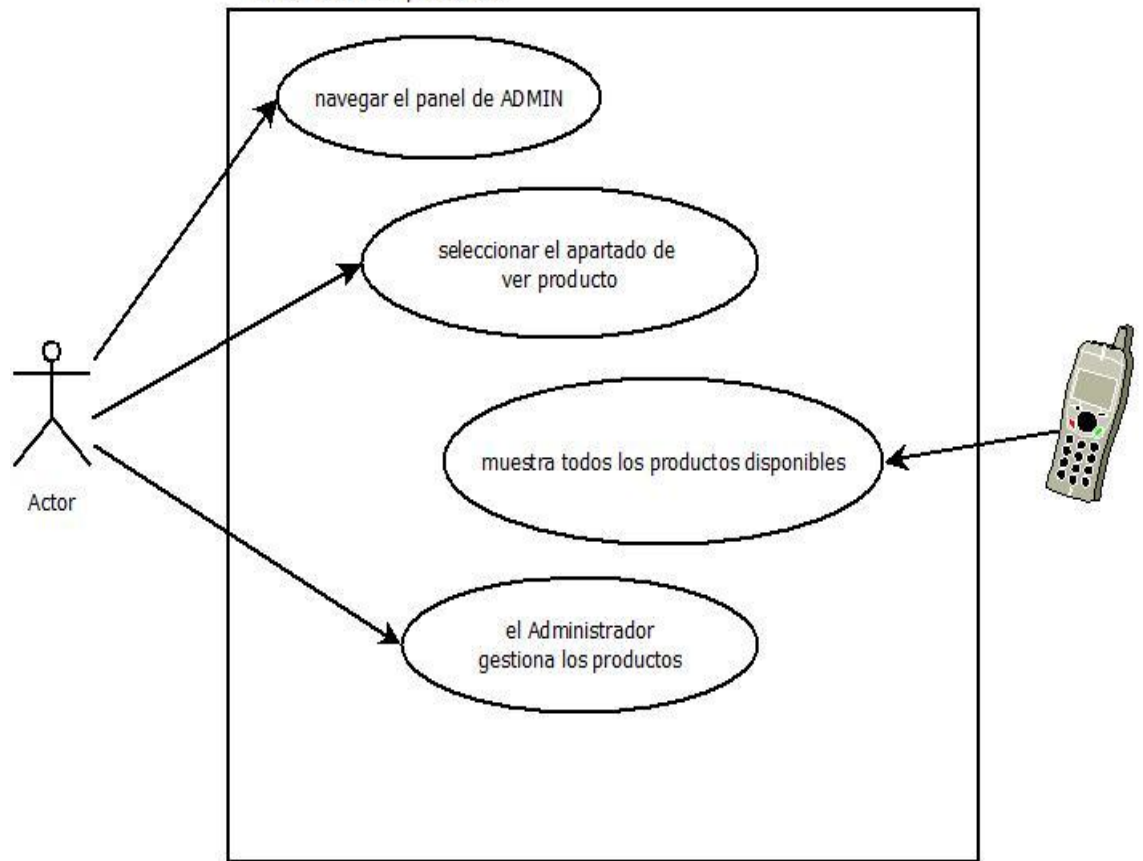
		<table><tr><td>NOMBRE:</td><td>CASO 06 - editar producto</td></tr><tr><td>AUTOR:</td><td>Santiago Filigrana</td></tr><tr><td>ACTOR:</td><td>Admin y Panel de admistracion.</td></tr><tr><td colspan="2">PRECONDICIÓN</td></tr><tr><td>1</td><td>El Admin debe tener correo y contraseña para iniciar sesion.</td></tr><tr><td>2</td><td>El Admin debe tener celular o computador con internet.</td></tr><tr><td>3</td><td>El Admin debe tener la necesidad de actualizar los datos de algun producto.</td></tr><tr><td colspan="2">FLUJO NORMAL</td></tr><tr><td>1</td><td>El Admin despues de iniciar sesion correctamente se encuentra en el apartado principal del panel de administracion.</td></tr><tr><td>2</td><td>El sistema muestra las diferentes opciones del panel de administrador.</td></tr><tr><td>3</td><td>El Admin selecciona la vista de productos.</td></tr><tr><td>4</td><td>El sistema muestra los productos disponibles ordenados por grupos.</td></tr><tr><td>5</td><td>El Admin navega por el apartado de productos.</td></tr><tr><td>6</td><td>El Admin selecciona el producto que desea editar.</td></tr><tr><td>7</td><td>El Admin pulsa el boton de editar.</td></tr><tr><td>8</td><td>El sistema le muestra un formulario en el que puede editar el producto (nombre, precio, descripcion, foto del producto).</td></tr><tr><td>9</td><td>el Admin edita los campos que considera necesarios y pulsa el boton de "guardar cambios".</td></tr><tr><td>10</td><td>El Sistema le pide confirmacion al admin para actualizar los datos del producto.</td></tr><tr><td>11</td><td>El Admin confirma que quiere actualizar los datos del producto.</td></tr><tr><td>12</td><td>El sistema aactualiza los datos del producto y los refleja en la pagina.</td></tr><tr><td colspan="2">FLUJO ALTERNATIVO</td></tr><tr><td>1.1</td><td>El Admin no ingreso correctamente el correo.</td></tr><tr><td>2.1</td><td>El Admin no ingreso correctamente la contraseña.</td></tr><tr><td>3.1</td><td>El sistema no valida datos.</td></tr><tr><td>3.2</td><td>El sistema toma como incorrectos los datos.</td></tr><tr><td>3.3</td><td>El sistema no deja ingresar al Admin .</td></tr><tr><td>6.1</td><td>El sistema no carga la pantalla.</td></tr><tr><td>7.1</td><td>El sistema no muestra la opcion de productos en el panel de admistracion.</td></tr><tr><td>9.1</td><td>El sistema no permite editar los productos.</td></tr><tr><td colspan="2">POSTCONDICIONES</td></tr><tr><td>1</td><td>El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.</td></tr><tr><td>2</td><td>El Admin puede navegar y usar correctamente el panel de administracion.</td></tr><tr><td>3</td><td>El Admin puede editar y guardar los cambios de forma correcta el producto seleccionado.</td></tr></table>	NOMBRE:	CASO 06 - editar producto	AUTOR:	Santiago Filigrana	ACTOR:	Admin y Panel de admistracion.	PRECONDICIÓN		1	El Admin debe tener correo y contraseña para iniciar sesion.	2	El Admin debe tener celular o computador con internet.	3	El Admin debe tener la necesidad de actualizar los datos de algun producto.	FLUJO NORMAL		1	El Admin despues de iniciar sesion correctamente se encuentra en el apartado principal del panel de administracion.	2	El sistema muestra las diferentes opciones del panel de administrador.	3	El Admin selecciona la vista de productos.	4	El sistema muestra los productos disponibles ordenados por grupos.	5	El Admin navega por el apartado de productos.	6	El Admin selecciona el producto que desea editar.	7	El Admin pulsa el boton de editar.	8	El sistema le muestra un formulario en el que puede editar el producto (nombre, precio, descripcion, foto del producto).	9	el Admin edita los campos que considera necesarios y pulsa el boton de "guardar cambios".	10	El Sistema le pide confirmacion al admin para actualizar los datos del producto.	11	El Admin confirma que quiere actualizar los datos del producto.	12	El sistema aactualiza los datos del producto y los refleja en la pagina.	FLUJO ALTERNATIVO		1.1	El Admin no ingreso correctamente el correo.	2.1	El Admin no ingreso correctamente la contraseña.	3.1	El sistema no valida datos.	3.2	El sistema toma como incorrectos los datos.	3.3	El sistema no deja ingresar al Admin .	6.1	El sistema no carga la pantalla.	7.1	El sistema no muestra la opcion de productos en el panel de admistracion.	9.1	El sistema no permite editar los productos.	POSTCONDICIONES		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.	2	El Admin puede navegar y usar correctamente el panel de administracion.	3	El Admin puede editar y guardar los cambios de forma correcta el producto seleccionado.
NOMBRE:	CASO 06 - editar producto																																																																			
AUTOR:	Santiago Filigrana																																																																			
ACTOR:	Admin y Panel de admistracion.																																																																			
PRECONDICIÓN																																																																				
1	El Admin debe tener correo y contraseña para iniciar sesion.																																																																			
2	El Admin debe tener celular o computador con internet.																																																																			
3	El Admin debe tener la necesidad de actualizar los datos de algun producto.																																																																			
FLUJO NORMAL																																																																				
1	El Admin despues de iniciar sesion correctamente se encuentra en el apartado principal del panel de administracion.																																																																			
2	El sistema muestra las diferentes opciones del panel de administrador.																																																																			
3	El Admin selecciona la vista de productos.																																																																			
4	El sistema muestra los productos disponibles ordenados por grupos.																																																																			
5	El Admin navega por el apartado de productos.																																																																			
6	El Admin selecciona el producto que desea editar.																																																																			
7	El Admin pulsa el boton de editar.																																																																			
8	El sistema le muestra un formulario en el que puede editar el producto (nombre, precio, descripcion, foto del producto).																																																																			
9	el Admin edita los campos que considera necesarios y pulsa el boton de "guardar cambios".																																																																			
10	El Sistema le pide confirmacion al admin para actualizar los datos del producto.																																																																			
11	El Admin confirma que quiere actualizar los datos del producto.																																																																			
12	El sistema aactualiza los datos del producto y los refleja en la pagina.																																																																			
FLUJO ALTERNATIVO																																																																				
1.1	El Admin no ingreso correctamente el correo.																																																																			
2.1	El Admin no ingreso correctamente la contraseña.																																																																			
3.1	El sistema no valida datos.																																																																			
3.2	El sistema toma como incorrectos los datos.																																																																			
3.3	El sistema no deja ingresar al Admin .																																																																			
6.1	El sistema no carga la pantalla.																																																																			
7.1	El sistema no muestra la opcion de productos en el panel de admistracion.																																																																			
9.1	El sistema no permite editar los productos.																																																																			
POSTCONDICIONES																																																																				
1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.																																																																			
2	El Admin puede navegar y usar correctamente el panel de administracion.																																																																			
3	El Admin puede editar y guardar los cambios de forma correcta el producto seleccionado.																																																																			

05 eliminar producto.



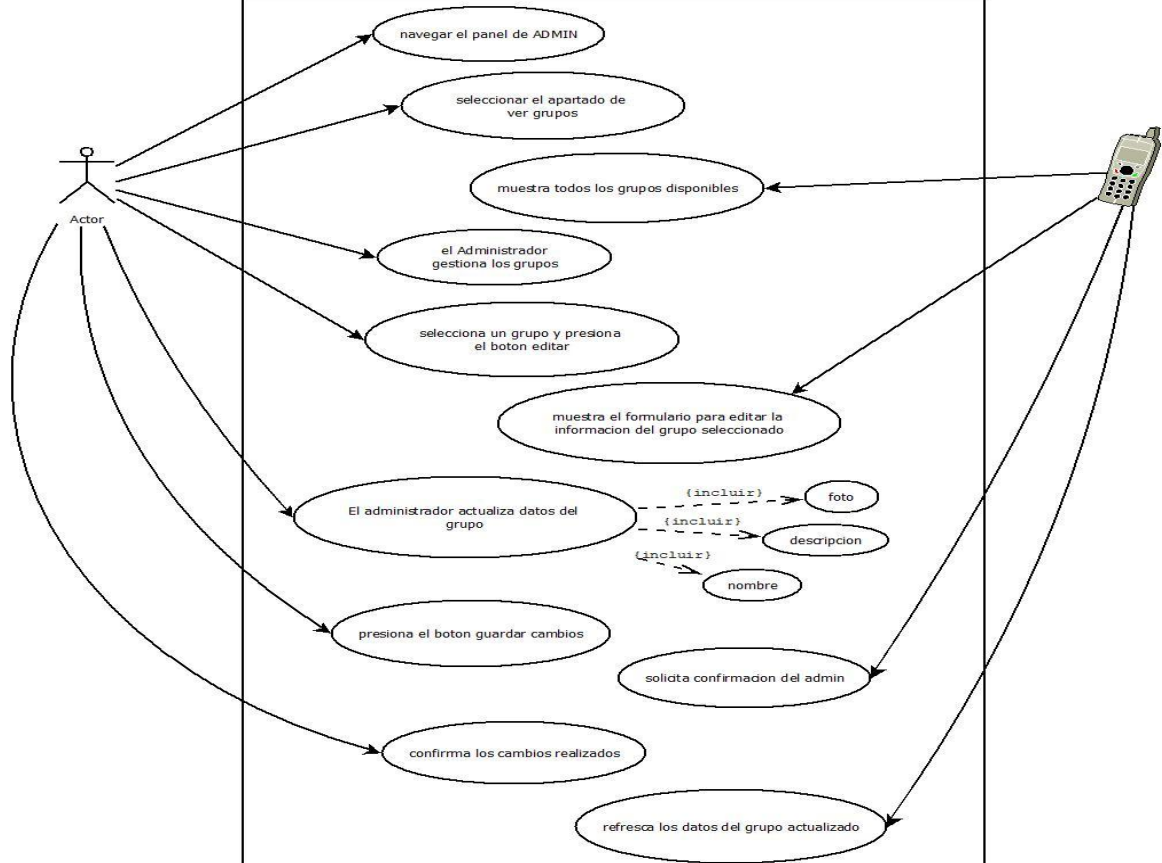
		NOMBRE:	CASO 09 - eliminar grupo de comida
		AUTOR:	Santiago Filigrana
		ACTOR:	Admin y Panel de administracion.
		PRECONDICIÓN	
		1	El Admin debe tener correo y contraseña para iniciar sesion.
		2	El Admin debe tener celular o computador con internet.
		3	El Admin debe tener la necesidad de eliminar algun grupo de comida.
		FLUJO NORMAL	
		1	El Admin despues de iniciar sesion correctamente se encuentra en el apartado principlal del panel de administracion.
		2	El sistema muestra las diferentes opciones del panel de administrador.
		3	El Admin selecciona la vista de grupos.
		4	El sistema muestra los grupos disponibles.
		5	El Admin navega por el apartado de grupos.
		6	El Admin selecciona el grupo que desea eliminar.
		7	El Admin pulsa el boton de eliminar.
		8	El sistema le pide que confirme si quiere eliminar el grupo.
		9	El Admin confirma que quiere eliminar el grupo de forma permanente.
		10	El sistema elimia de forma permanente el grupo.
		FLUJO ALTERNATIVO	
		1.1	El Admin no ingreso correctamente el correo.
		2.1	El Admin no ingreso correctamente la contraseña.
		3.1	El sistema no valida datos.
		3.2	El sistema toma como incorrectos los datos.
		3.3	El sistema no deja ingresar al Admin .
		6.1	El sistema no carga la vista del panel de administracion.
		7.1	El sistema no muestra la opcion de grupos en el panel de administracion.
		9.1	El sistema no permite borrar grupos.
		POSTCONDICIONES	
		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.
		2	El Admin puede navegar y usar correctamente el panel de administracion.
		3	El Admin puede elimar de forma correcta el grupo seleccionado.

04 ver o enlistar productos.



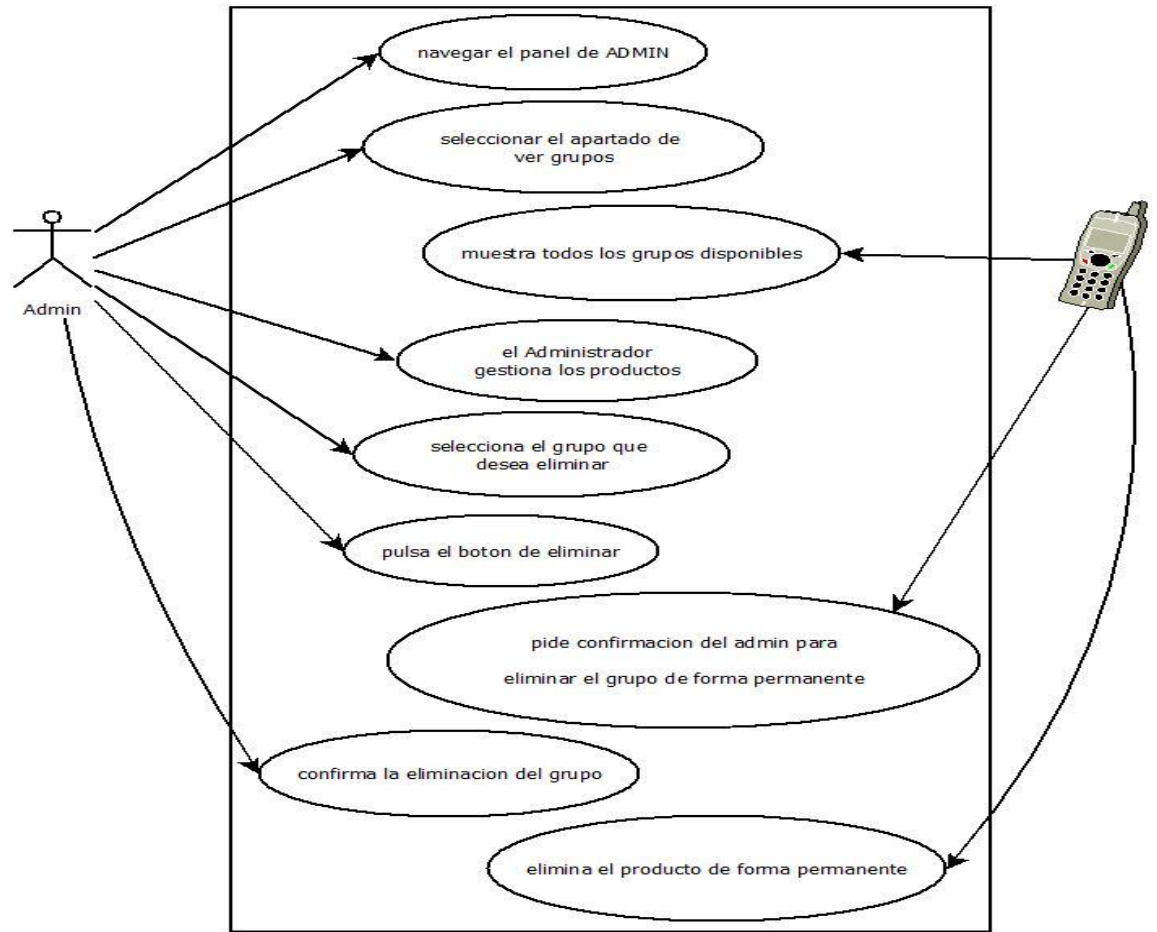
		NOMBRE: CASO 04- ver o enlistar productos	
		AUTOR: Santiago Filigrana	
		ACTOR: Admin y Panel de admistracion.	
		PRECONDICIÓN	
		1	El Admin debe tener correo para iniciar sesion.
		2	El Admin debe tener la contraseña para iniciar sesion.
		3	El Admin debe tener celular o computador con internet.
		FLUJO NORMAL	
		1	El Admin despues de iniciar sesion correctamente se encuentra en el apartada princiapal del panel de administracion.
		2	El sistema muestra las diferentes opciones del panel de administrador.
		3	El Admin selecciona la vista de productos.
		4	El sistema muestra los productos disponibles ordenados por grupos.
		5	El Admin navega por el apartado de productos.
		FLUJO ALTERNATIVO	
		1.1	El Admin no ingreso correctamente el correo.
		2.1	El Admin no ingreso correctamente la contraseña.
		3.1	El sistema no valida datos.
		3.2	El sistema toma como incorrectos los datos.
		3.3	El sistema no deja ingresar al Admin .
		6.1	El sistema no carga la pantalla.
		7.1	El sistema no muestra la opcion del panel de admistracion.
		9.1	El sistema no deja ver productos.
		POSTCONDICIONES	
		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.
		2	El Admin puede navegar y usar correctamente el panel de administracion.

10 editar grupo



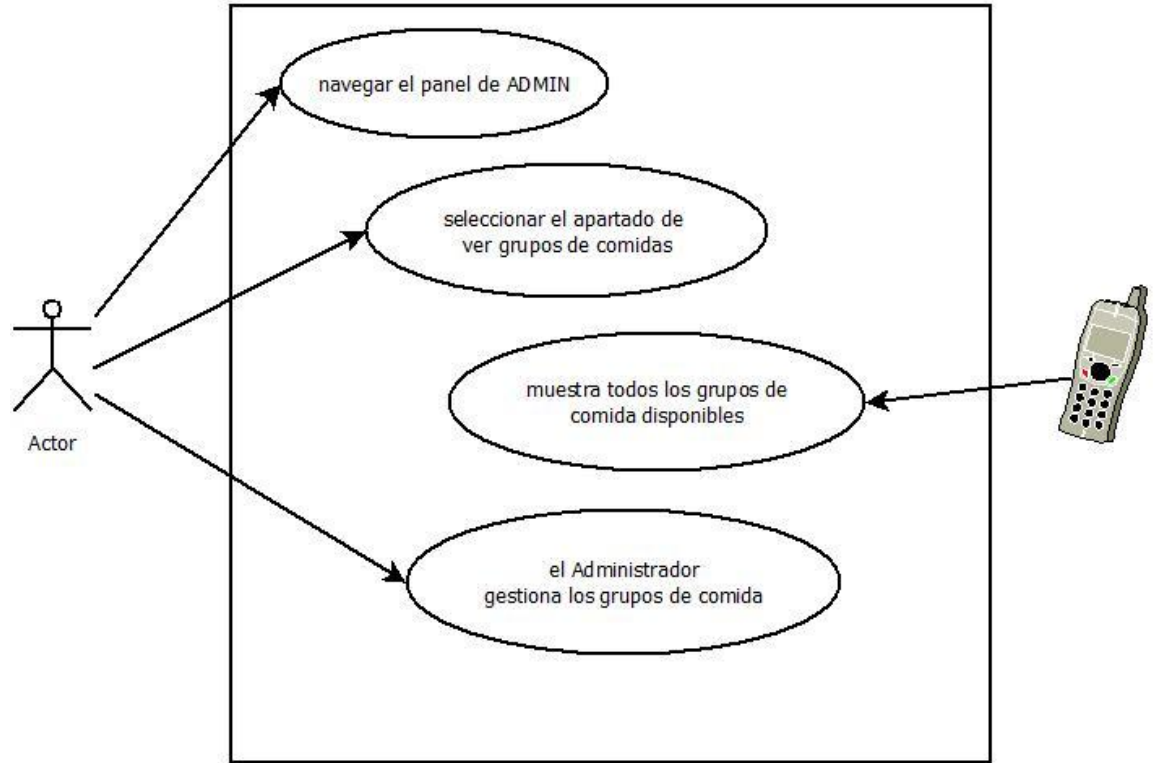
		NOMBRE:	CASO 10 - editar grupo
		AUTOR:	Santiago Filigrama
		ACTOR:	Admin y Panel de administracion.
		PRECONDICIÓN	
		1	El Admin debe tener correo y contraseña para iniciar sesion.
		2	El Admin debe tener celular o computador con internet.
		3	El Admin debe tener la necesidad de actualizar o editar los datos de un grupo.
		FLUJO NORMAL	
		1	El Admin despues de iniciar sesion correctamente se encuentra en el apartado princiapal del panel de administracion.
		2	El sistema muestra las diferentes opciones del panel de administrador.
		3	El Admin selecciona la vista de grupos.
		4	El sistema muestra los grupos disponibles.
		5	El Admin navega por el apartado de grupos.
		6	El Admin selecciona el grupo que desea editar.
		7	El Admin pulsa el boton de editar.
		8	El sistema le muestra un formulario en el que puede editar el grupo (nombre, descripcion, foto del grupo).
		9	el Admin edita los campos que considera necesarios y pulsa el boton de "guardar cambios".
		10	El Sistema le pide confirmacion al admin para actualizar los datos del grupo.
		11	El Admin confirma que quiere actualizar los datos del grupo.
		12	El sistema actualiza los datos del grupo y los refleja en la pagina.
		FLUJO ALTERNATIVO	
		1.1	El Admin no ingreso correctamente el correo.
		2.1	El Admin no ingreso correctamente la contraseña.
		3.1	El sistema no valida datos.
		3.2	El sistema toma como incorrectos los datos.
		3.3	El sistema no deja ingresar al Admin .
		6.1	El sistema no carga la pantalla de grupos.
		7.1	El sistema no muestra la opcion de grupos en el panel de administracion.
		9.1	El sistema no permite editar los grupos.
		POSTCONDICIONES	
		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.
		2	El Admin puede navegar y usar correctamente el panel de administracion.
		3	El Admin puede editar y guardar los cambios de forma correcta del grupo seleccionado.

09 eliminar grupo.



		NOMBRE:	CASO 09 - elimiar grupo de comida
		AUTOR:	Santiago Filigrana
		ACTOR:	Admin y Panel de admistracion.
		PRECONDICIÓN	
		1	El Admin debe tener correo y contraseña para iniciar sesion.
		2	El Admin debe tener celular o computador con internet.
		3	El Admin debe tener la necesidad de eliminar algun grupo de comida.
		FLUJO NORMAL	
		1	El Admin despues de iniciar sesion correctamente se encuentra en el apartado princiapal del panel de administracion.
		2	El sistema muestra las diferentes opciones del panel de administrador.
		3	El Admin selecciona la vista de grupos.
		4	El sistema muestra los grupos disponibles.
		5	El Admin navega por el apartado de grupos.
		6	El Admin selecciona el grupo que desea eliminar.
		7	El Admin pulsa el boton de eliminar.
		8	El sistema le pide que confirme si quiere eliminar el grupo.
		9	El Admin confirma que quiere eliminar el grupo de forma permanente.
		10	El sistema elimia de forma permanente el grupo.
		FLUJO ALTERNATIVO	
		1.1	El Admin no ingreso correctamente el correo.
		2.1	El Admin no ingreso correctamente la contraseña.
		3.1	El sistema no valida datos.
		3.2	El sistema toma como incorrectos los datos.
		3.3	El sistema no deja ingresar al Admin .
		6.1	El sistema no carga la vista del panel de administracion.
		7.1	El sistema no muestra la opcion de grupos en el panel de admistracion.
		9.1	El sistema no permite borrar grupos.
		POSTCONDICIONES	
		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.
		2	El Admin puede navegar y usar correctamente el panel de administracion.
		3	El Admin puede elimiar de forma correcta el grupo seleccionado.

08 ver o enlistar grupos de comidas.



		NOMBRE:	CASO 08 - ver o enlistar grupos
		AUTOR:	Santiago Filigrana
		ACTOR:	Admin y Panel de administracion.
		PRECONDICIÓN	
		1	El Admin debe tener correo y contraseña para iniciar sesion.
		2	El Admin debe tener celular o computador con internet.
		3	El Admin debe tener la necesidad de ver los grupos de comida disponibles.
		FLUJO NORMAL	
		1	El Admin despues de iniciar sesion correctamente se encuentra en el apartada principal del panel de administracion.
		2	El sistema muestra las diferentes opciones del panel de administrador.
		3	El Admin selecciona la vista de grupos.
		4	El sistema muestra los grupos de comida disponibles.
		5	El Admin navega por el apartado de grupos.
		FLUJO ALTERNATIVO	
		1.1	El Admin no ingreso correctamente el correo.
		2.1	El Admin no ingreso correctamente la contraseña.
		3.1	El sistema no valida datos.
		3.2	El sistema toma como incorrectos los datos.
		3.3	El sistema no deja ingresar al Admin .
		6.1	El sistema no carga la pantalla.
		7.1	El sistema no muestra la vista de grupos.
		9.1	El sistema no deja ver los grupos.
		POSTCONDICIONES	
		1	El sistema resive y valida correctamente los datos y da el ingreso al panel de administracion.
		2	El Admin puede navegar y usar correctamente el panel de administracion.
		3	El Admin puede ingresar y ver los grupos disponibles de forma correcta.

El diagrama de clases del sistema es el siguiente:

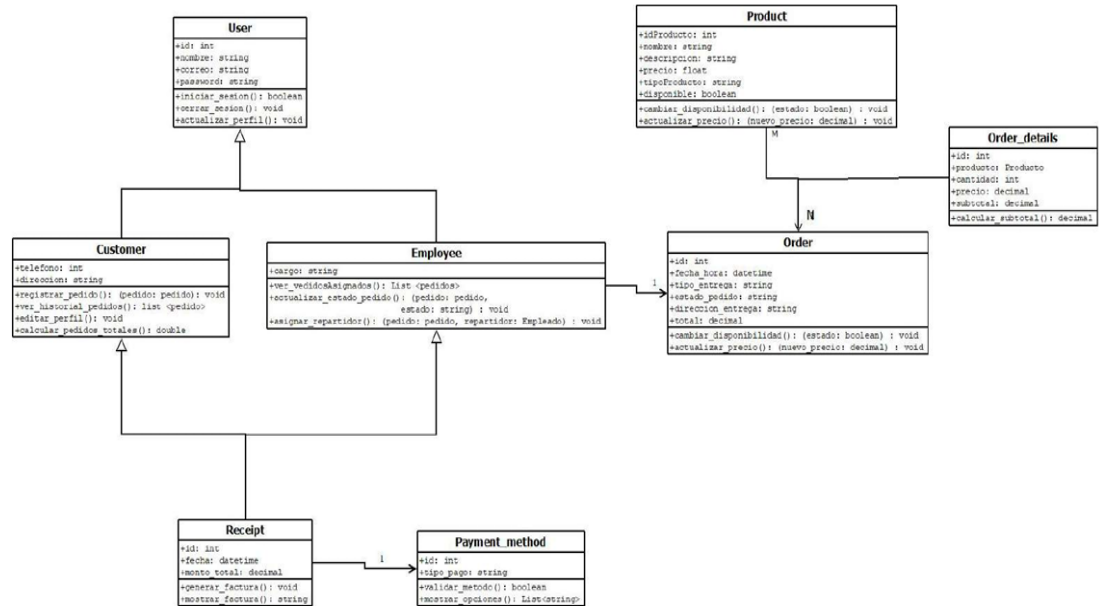


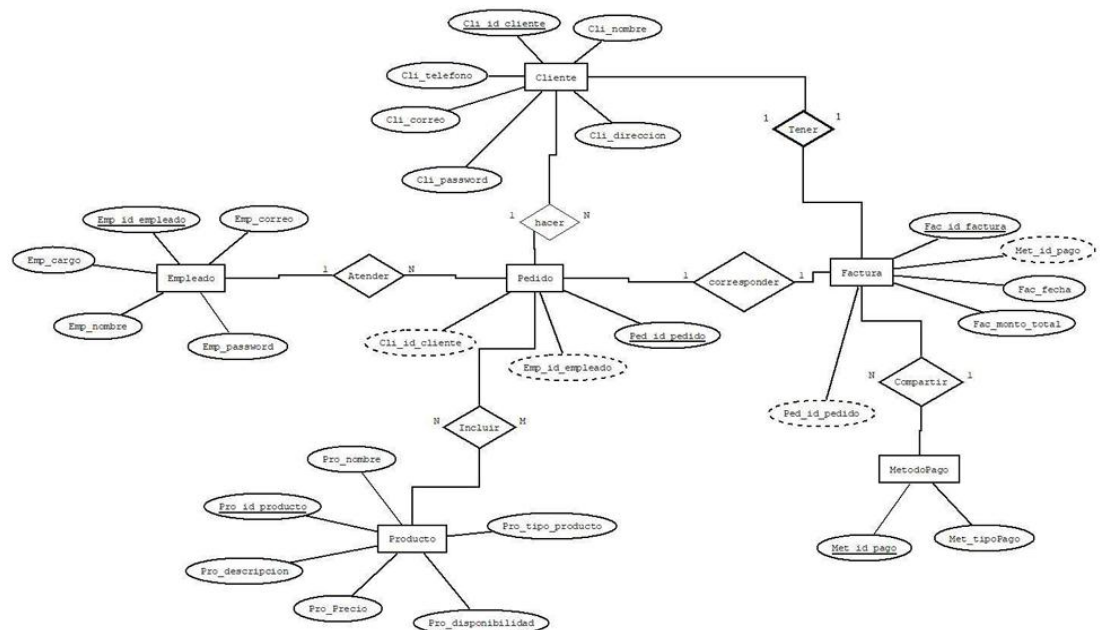
Diagrama de Clases (Modelo principal en Rails)

Clase	Atributos / Relaciones
Usuario	id, nombre, email, rol, password_digest 1—* Carrito 1—* Pedido
Categoría	id, nombre 1—* Plato
Plato	id, nombre, precio, imagen belongs_to Categoría
Carrito	id belongs_to Usuario 1—* CarritoItem
CarritoItem	id, cantidad belongs_to Carrito belongs_to Plato
Pedido	id, fecha, estado belongs_to Usuario 1—* PedidoItem
PedidoItem	id, cantidad belongs_to Pedido belongs_to Plato

4

Modelo relacional de la base de datos

El modelo relacional de la base de datos citas se muestra a continuación:



- Modelo dividido por tabla

Tabla: usuarios

Columna
id (PK)
nombre
email (UNIQUE)
password_digest
rol (cliente / admin)

Tabla: categorias

Columna
id (PK)
nombre

Tabla: platos

Columna
id (PK)
nombre
precio
imagen
categoria_id (FK → categorias.id)

Tabla: carritos

Columna
id (PK)
usuario_id (FK → usuarios.id)

Tabla: pedidos

Columna
id (PK)
fecha
estado (recibido, en preparación, terminado, entregado)
usuario_id (FK → usuarios.id)

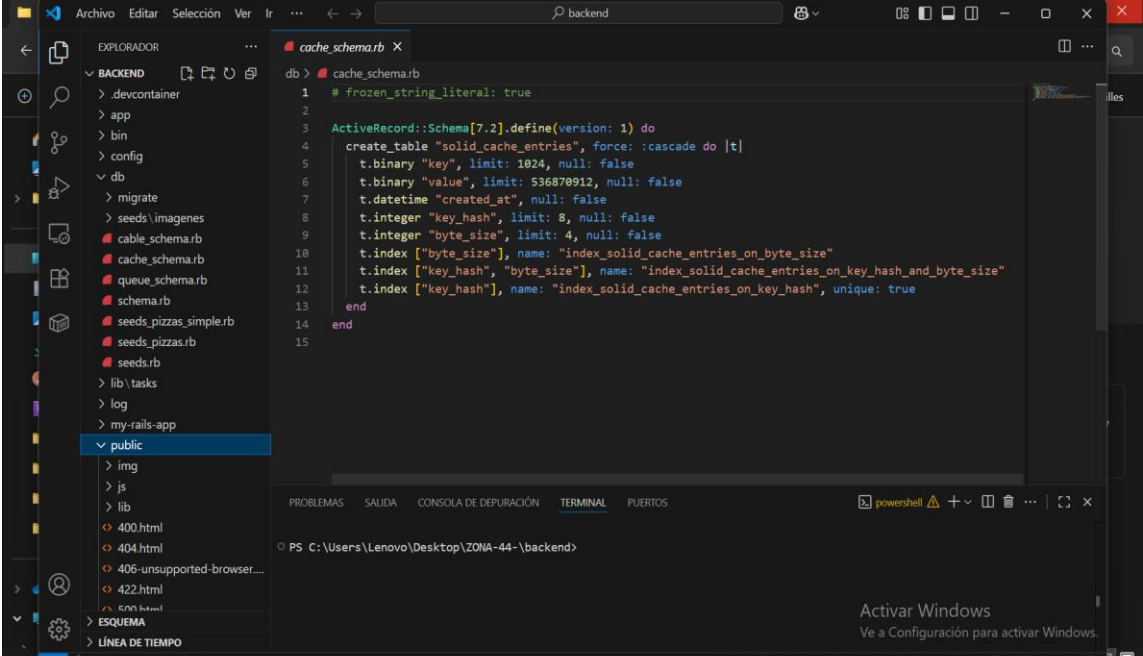
5	Descripción de la Plataforma	<p>Servidor de Aplicaciones (Backend): construido en Ruby on Rails, ejecutándose en un contenedor DevContainer.</p> <p>Base de Datos: PostgreSQL 14+, encargada de almacenar información de usuarios, productos, pedidos y transacciones.</p> <p>Correo (SMTP): utilizado para notificaciones automáticas a clientes y administradores.</p> <p>Contenedores (Docker): aseguran la ejecución aislada del backend y base de datos.</p> <p>Para instalar el servidor de la aplicación se requiere lo siguiente:</p> <table border="1"> <thead> <tr> <th>Componente</th><th>Mínimo</th><th>Recomendación</th></tr> </thead> <tbody> <tr> <td>CPU</td><td>2 NUCLEOS</td><td>4 NUCLEOS O MÁS</td></tr> <tr> <td>RAM</td><td>4 GB</td><td>8 – 16 GB</td></tr> <tr> <td>ALMACENAMIENTO</td><td>50 GB HDD</td><td>100 GB SSD</td></tr> <tr> <td>RAM</td><td>10 Mbps</td><td>100 Mbps con redundancia</td></tr> </tbody> </table> <p>Arquitectura</p> <p>El sistema sigue el patrón MVC (Modelo-Vista-Controlador) propio de Ruby on Rails:</p> <ul style="list-style-type: none"> • Modelos: representan las entidades del negocio (usuarios, categorías, platos, carritos, pedidos). • Controladores: procesan las solicitudes HTTP y devuelven respuestas en JSON. • Vistas: aunque no se utilizan directamente en producción, Rails permite vistas HTML para pruebas internas. <p>Funcionalidades Principales</p> <ol style="list-style-type: none"> 1. Usuarios <ul style="list-style-type: none"> ○ Registro e inicio de sesión. ○ Roles: cliente y administrador. ○ Seguridad con contraseñas encriptadas. 2. Categorías y Platos <ul style="list-style-type: none"> ○ Creación, actualización y eliminación de categorías. ○ CRUD completo de platos (nombre, precio, imagen, categoría). 3. Carrito de Compras <ul style="list-style-type: none"> ○ Asociación a cada usuario. ○ Agregar, modificar o eliminar productos. ○ Cálculo automático de totales. 	Componente	Mínimo	Recomendación	CPU	2 NUCLEOS	4 NUCLEOS O MÁS	RAM	4 GB	8 – 16 GB	ALMACENAMIENTO	50 GB HDD	100 GB SSD	RAM	10 Mbps	100 Mbps con redundancia
Componente	Mínimo	Recomendación															
CPU	2 NUCLEOS	4 NUCLEOS O MÁS															
RAM	4 GB	8 – 16 GB															
ALMACENAMIENTO	50 GB HDD	100 GB SSD															
RAM	10 Mbps	100 Mbps con redundancia															

		<p>4. Pedidos</p> <ul style="list-style-type: none">○ Generación de pedidos a partir del carrito.○ Registro de fecha y estado.○ Control de flujo de estado: recibido → en preparación → terminado → entregado. <p>5. API REST</p> <ul style="list-style-type: none">○ Endpoints para gestionar usuarios, categorías, platos, carritos y pedidos.○ Respuestas en formato JSON para consumo en la app móvil Flutter.
--	--	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

6

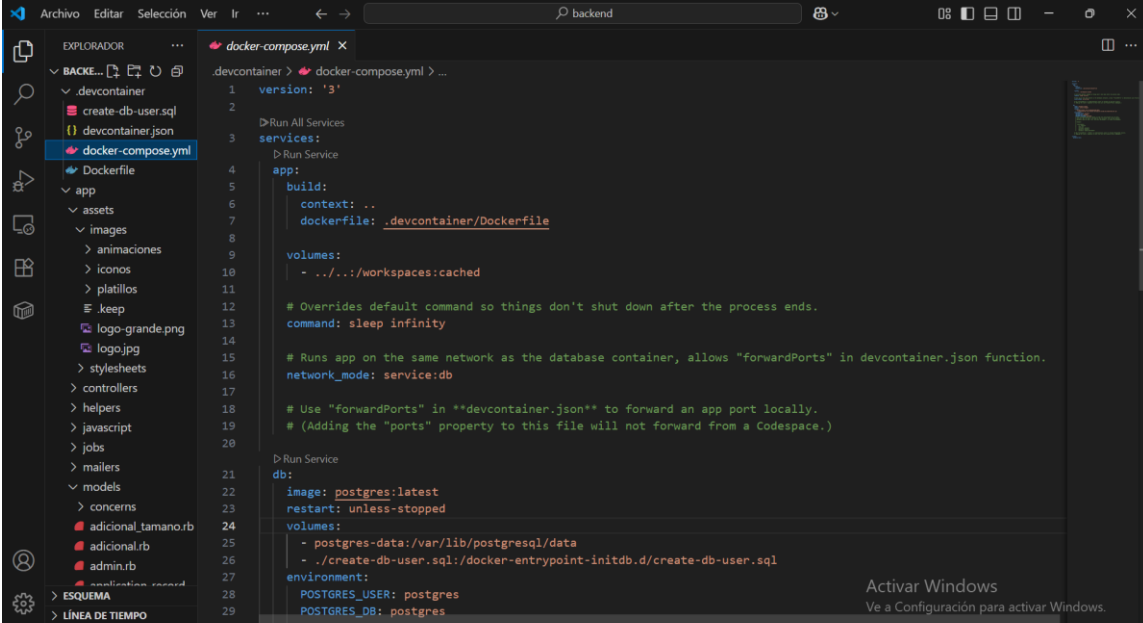
Documentación
del
código fuente

Descripción de la aplicación:



```
1 # frozen_string_literal: true
2
3 ActiveRecord::Schema[7.2].define(version: 1) do
4   create_table "solid_cache_entries", force: :cascade do |t|
5     t.binary "key", limit: 1024, null: false
6     t.binary "value", limit: 536870912, null: false
7     t.datetime "created_at", null: false
8     t.integer "key_hash", limit: 8, null: false
9     t.integer "byte_size", limit: 4, null: false
10    t.index ["byte_size"], name: "index_solid_cache_entries_on_byte_size"
11    t.index ["key_hash", "byte_size"], name: "index_solid_cache_entries_on_key_hash_and_byte_size"
12    t.index ["key_hash"], name: "index_solid_cache_entries_on_key_hash", unique: true
13  end
14 end
15
```

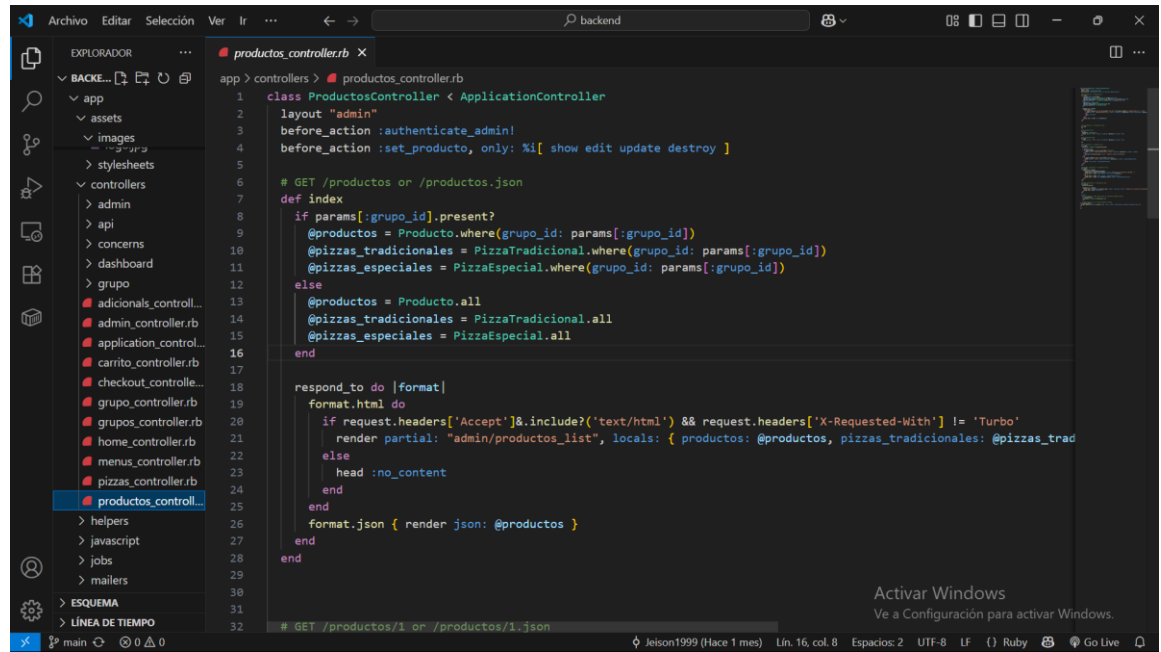
Codigo docker-compose.yml



```
1 version: '3'
2
3 services:
4   app:
5     build:
6       context: ..
7       dockerfile: .devcontainer/Dockerfile
8
9     volumes:
10      - ../workspaces:cached
11
12     # Overrides default command so things don't shut down after the process ends.
13     command: sleep infinity
14
15     # Runs app on the same network as the database container, allows "forwardPorts" in devcontainer.json function.
16     network_mode: service:db
17
18     # Use "forwardPorts" in **devcontainer.json** to forward an app port locally.
19     # (Adding the "ports" property to this file will not forward from a Codespace.)
20
21   db:
22     image: postgres:latest
23     restart: unless-stopped
24
25     volumes:
26      - postgres-data:/var/lib/postgresql/data
27      - ./create-db-user.sql:/docker-entrypoint-initdb.d/create-db-user.sql
28
29     environment:
30       POSTGRES_USER: postgres
31       POSTGRES_DB: postgres

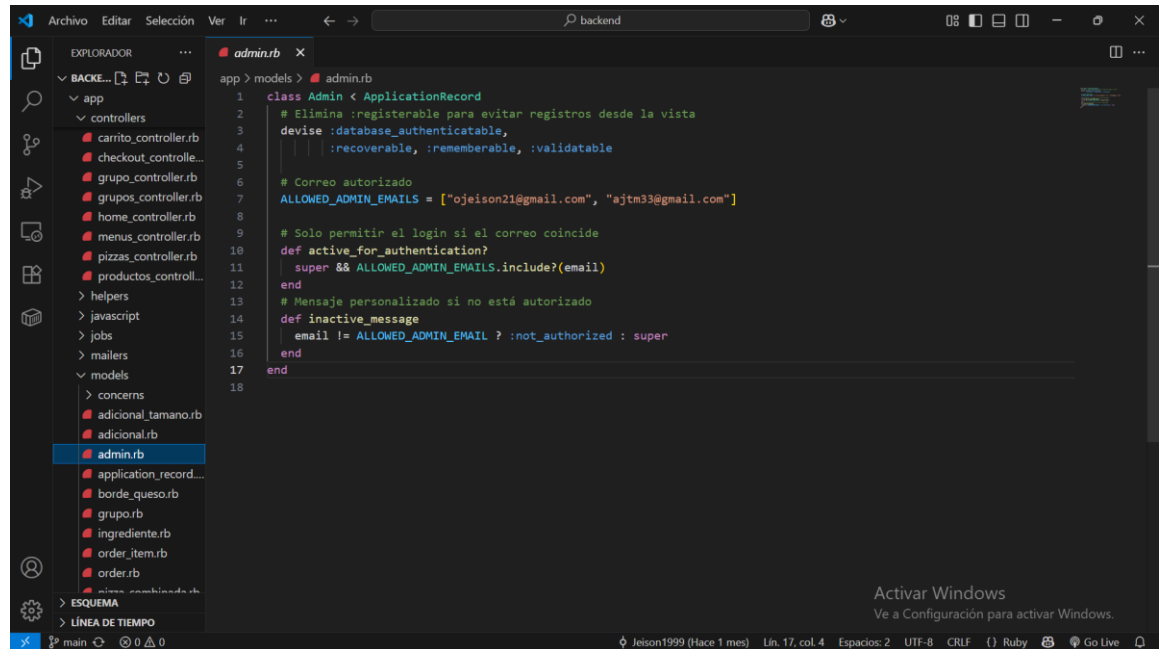
```

Codigo productos_controller.rb



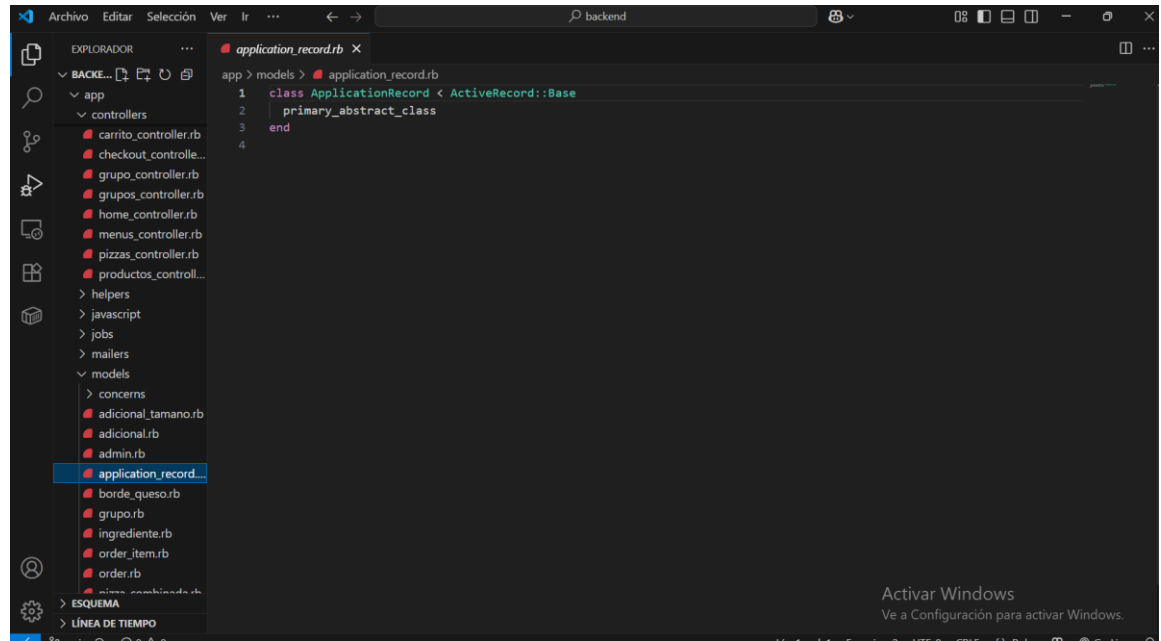
```
1 class ProductosController < ApplicationController
2   layout "admin"
3   before_action :authenticate_admin!
4   before_action :set_producto, only: [:show, :edit, :update, :destroy]
5
6   # GET /productos or /productos.json
7   def index
8     if params[:grupo_id].present?
9       @productos = Producto.where(grupo_id: params[:grupo_id])
10      @pizzas_tradicionales = PizzaTradicional.where(grupo_id: params[:grupo_id])
11      @pizzas_especiales = PizzaEspecial.where(grupo_id: params[:grupo_id])
12     else
13       @productos = Producto.all
14       @pizzas_tradicionales = PizzaTradicional.all
15       @pizzas_especiales = PizzaEspecial.all
16     end
17
18     respond_to do |format|
19       format.html do
20         if request.headers["Accept"]&.include?('text/html') && request.headers["X-Requested-With"] != 'Turbo'
21           render partial: "admin/productos_list", locals: { productos: @productos, pizzas_tradicionales: @pizzas_tradicionales, pizzas_especiales: @pizzas_especiales }
22         else
23           head :no_content
24         end
25       end
26       format.json { render json: @productos }
27     end
28   end
29
30   # GET /productos/1 or /productos/1.json
```

Codigo modelo admin



```
1 class Admin < ApplicationRecord
2   # Elimina :registerable para evitar registros desde la vista
3   devise :database_authenticatable,
4         :recoverable, :rememberable, :validatable
5
6   # Correo autorizado
7   ALLOWED_ADMIN_EMAILS = ["ojeison21@gmail.com", "ajtm33@gmail.com"]
8
9   # Solo permitir el login si el correo coincide
10  def active_for_authentication?
11    super && ALLOWED_ADMIN_EMAILS.include?(email)
12  end
13
14  # Mensaje personalizado si no está autorizado
15  def inactive_message
16    email != ALLOWED_ADMIN_EMAIL ? :not_authorized : super
17  end
18 end
```

Codigo clase ApplicationRecord

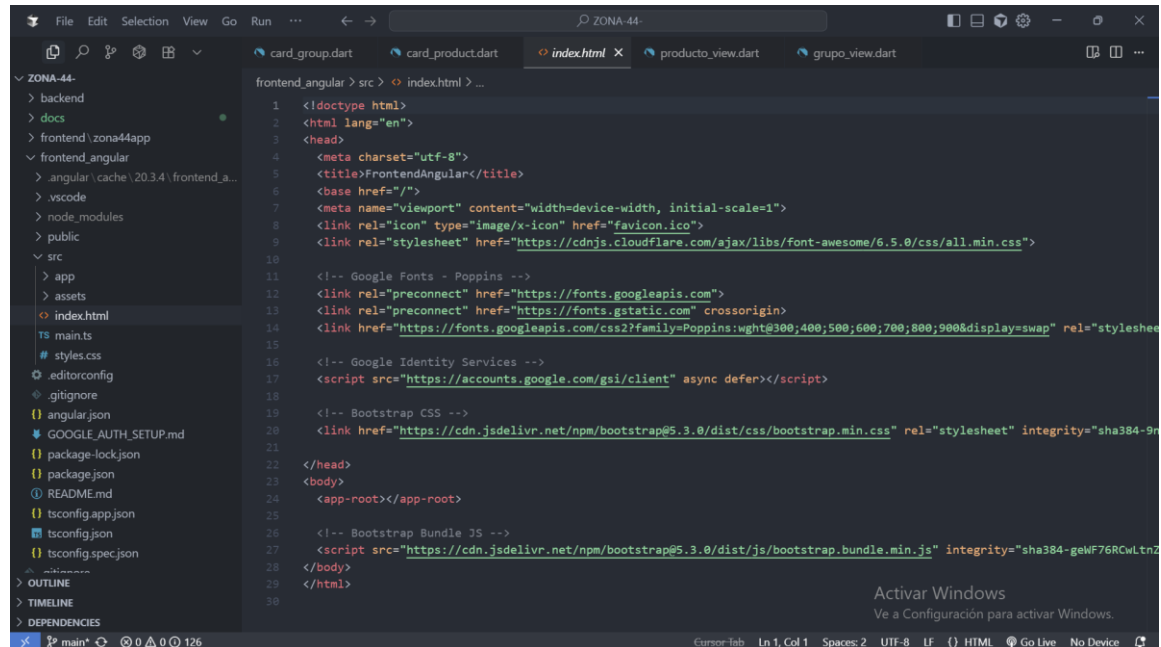


The screenshot shows a VS Code editor window with the file explorer on the left and the code editor on the right. The file explorer shows a project structure with folders like 'app', 'controllers', 'models', and 'views'. The 'models' folder is expanded, showing several files including 'application_record.rb'. The code editor displays the content of 'application_record.rb', which is a class definition for ApplicationRecord, inheriting from ActiveRecord::Base. The class has a primary_abstract_class attribute.

```
1 class ApplicationRecord < ActiveRecord::Base
2   primary_abstract_class
3 end
4
```

Framerwork Angular

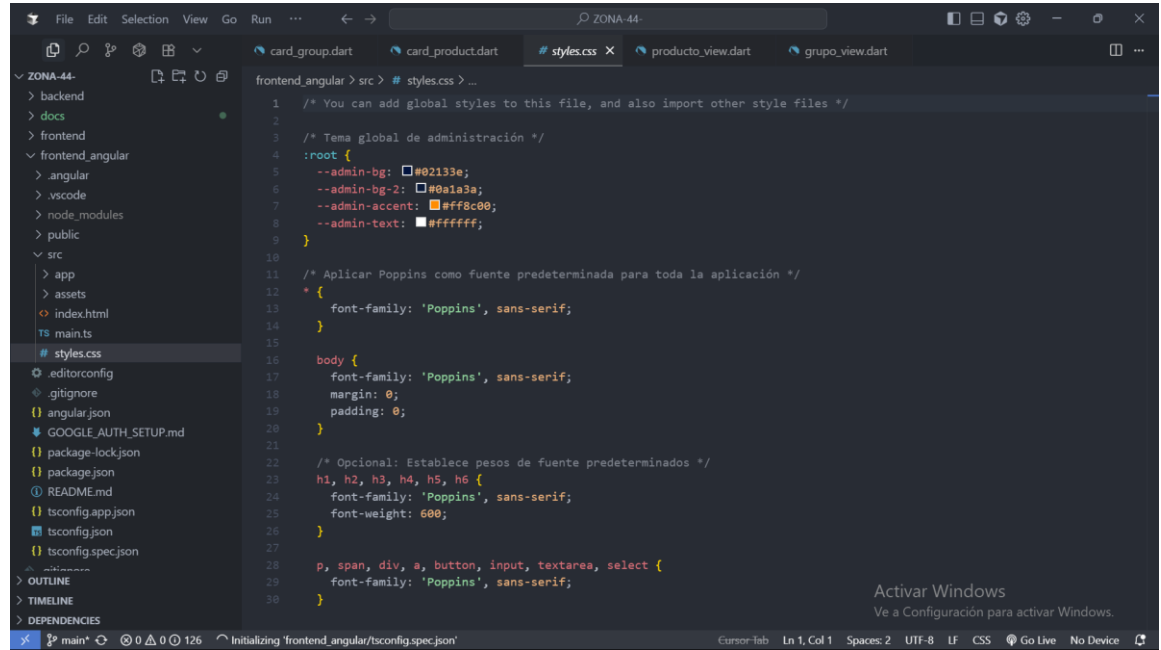
Index.html principal



The screenshot shows a VS Code editor window with the file explorer on the left and the code editor on the right. The file explorer shows a project structure with folders like 'src', 'assets', and 'index.html'. The 'index.html' file is selected, and its content is displayed in the code editor. The code is an HTML document for an Angular application, including meta tags for charset, viewport, and title, as well as links for fonts, CSS, and JavaScript.

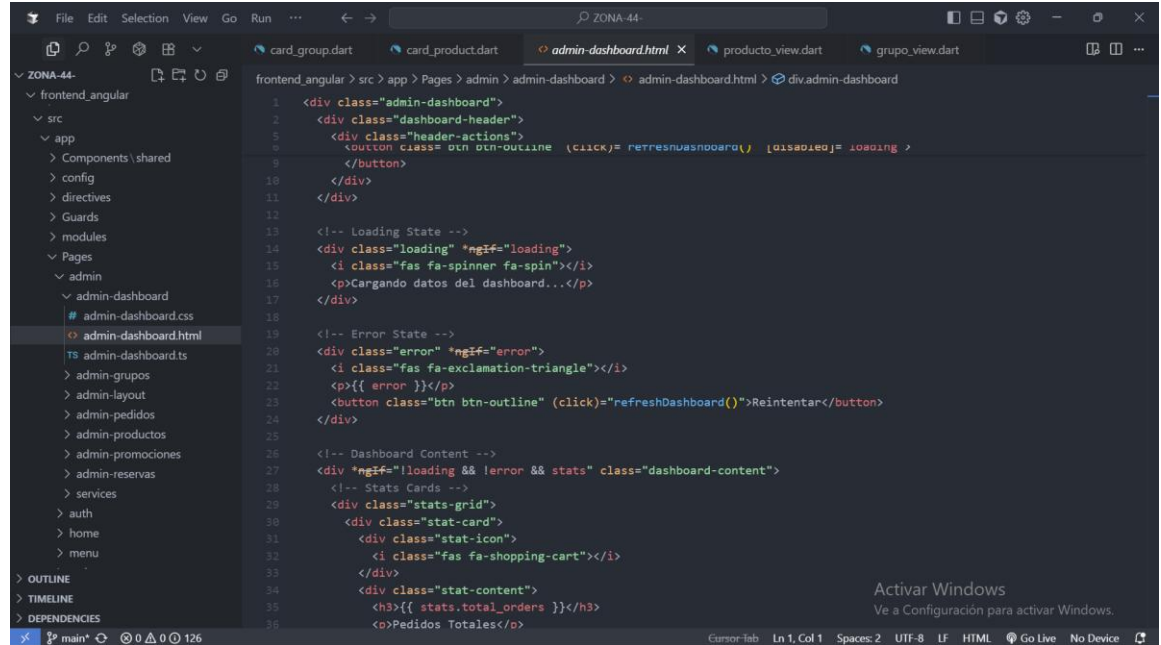
```
1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8">
5   <title>FrontendAngular</title>
6   <base href="/">
7   <meta name="viewport" content="width=device-width, initial-scale=1">
8   <link rel="icon" type="image/x-icon" href="favicon.ico">
9   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.0/css/all.min.css">
10
11   <!-- Google Fonts - Poppins -->
12   <link rel="preconnect" href="https://fonts.googleapis.com">
13   <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
14   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700;800;900&display=swap" rel="stylesheet">
15
16   <!-- Google Identity Services -->
17   <script src="https://accounts.google.com/gsi/client" async defer></script>
18
19   <!-- Bootstrap CSS -->
20   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-9n"
21
22 </head>
23 <body>
24   <app-root></app-root>
25
26   <!-- Bootstrap Bundle JS -->
27   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-geWf76RCwLtn2"
28 </body>
29 </html>
```

Styles.css principal



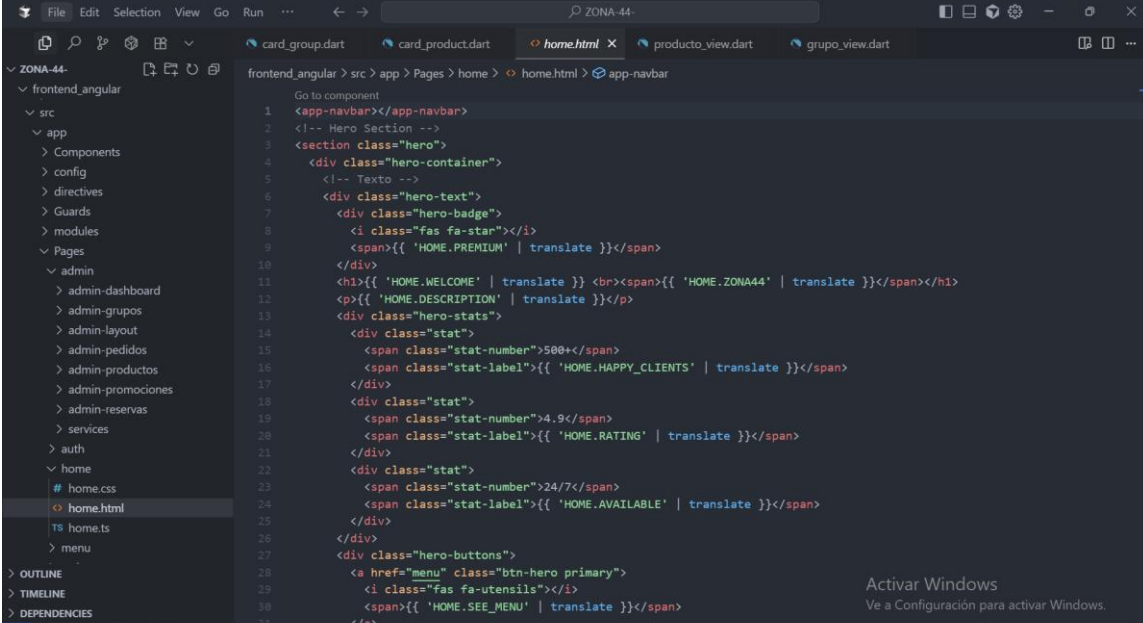
```
1 /* You can add global styles to this file, and also import other style files */
2
3 /* Tema global de administración */
4 :root {
5   --admin-bg: #021330;
6   --admin-bg-2: #0a1a3a;
7   --admin-accent: #fff8c0;
8   --admin-text: #ffffff;
9 }
10
11 /* Aplicar Poppins como fuente predeterminada para toda la aplicación */
12 * {
13   font-family: 'Poppins', sans-serif;
14 }
15
16 body {
17   font-family: 'Poppins', sans-serif;
18   margin: 0;
19   padding: 0;
20 }
21
22 /* Opcional: Establece pesos de fuente predeterminados */
23 h1, h2, h3, h4, h5, h6 {
24   font-family: 'Poppins', sans-serif;
25   font-weight: 600;
26 }
27
28 p, span, div, a, button, input, textarea, select {
29   font-family: 'Poppins', sans-serif;
30 }
```

Vista dashboard principal admin



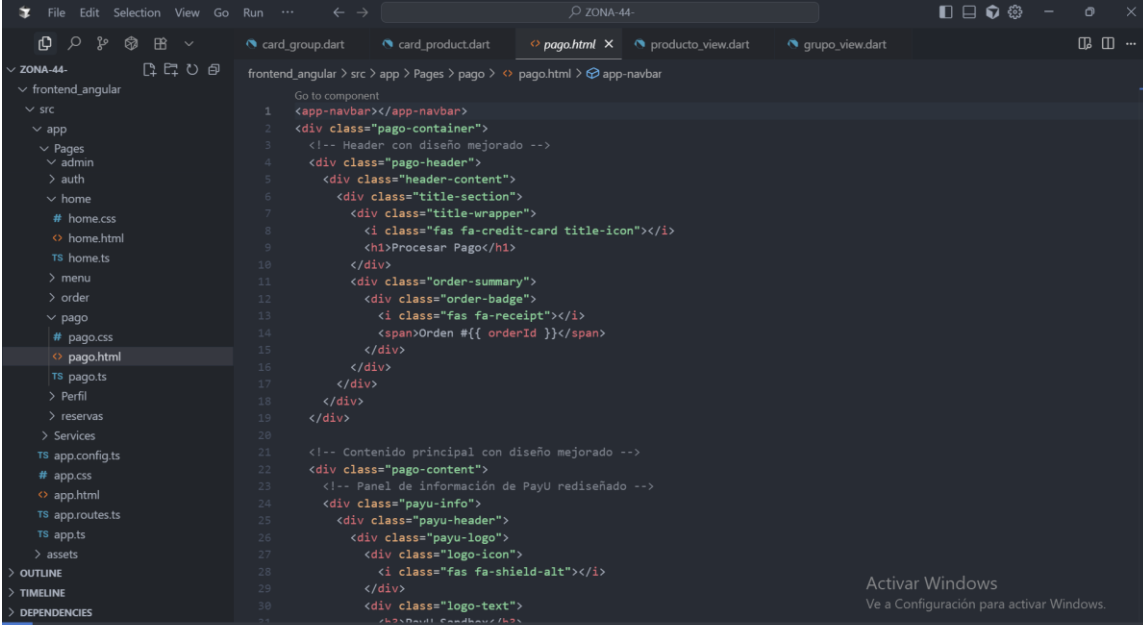
```
1 <div class="admin-dashboard">
2   <div class="dashboard-header">
3     <div class="header-actions">
4       <button class="btn btn-outline" (click)="refreshDashboard()" [disabled]="loading">
5
6     </div>
7   </div>
8
9   <!-- Loading State -->
10   <div class="loading" *ngIf="loading">
11     <i class="fas fa-spinner fa-spin"></i>
12     <p>Cargando datos del dashboard...</p>
13   </div>
14
15   <!-- Error State -->
16   <div class="error" *ngIf="error">
17     <i class="fas fa-exclamation-triangle"></i>
18     <p>{{ error }}</p>
19     <button class="btn btn-outline" (click)="refreshDashboard()">Reintentar</button>
20   </div>
21
22   <!-- Dashboard Content -->
23   <div *ngIf="loading && !error && stats" class="dashboard-content">
24     <!-- Stats Cards -->
25     <div class="stats-grid">
26       <div class="stat-card">
27         <i class="fas fa-shopping-cart"></i>
28       </div>
29       <div class="stat-content">
30         <h3>{{ stats.total_orders }}</h3>
31         <p>Pedidos Totales</p>
32       </div>
33     </div>
34   </div>
35 </div>
```

Home principal (Inicio)



```
1 <app-navbar></app-navbar>
2 <!-- Hero Section -->
3 <section class="hero">
4   <div class="hero-container">
5     <!-- Texto -->
6     <div class="hero-text">
7       <div class="hero-badge">
8         <i class="fas fa-star"></i>
9         <span>{{ 'HOME.PREMIUM' | translate }}</span>
10      </div>
11      <h1>{{ 'HOME.WELCOME' | translate }} <br><span>{{ 'HOME.ZONA44' | translate }}</span></h1>
12      <p>{{ 'HOME.DESCRPTION' | translate }}</p>
13      <div class="hero-stats">
14        <div class="stat">
15          <span class="stat-number">500</span>
16          <span class="stat-label">{{ 'HOME.HAPPY_CLIENTS' | translate }}</span>
17        </div>
18        <div class="stat">
19          <span class="stat-number">4.9</span>
20          <span class="stat-label">{{ 'HOME.RATING' | translate }}</span>
21        </div>
22        <div class="stat">
23          <span class="stat-number">24/7</span>
24          <span class="stat-label">{{ 'HOME.AVAILABLE' | translate }}</span>
25        </div>
26      </div>
27      <div class="hero-buttons">
28        <a href="#menu" class="btn-hero primary">
29          <i class="fas fa-utensils"></i>
30          <span>{{ 'HOME.SEE_MENU' | translate }}</span>
31        </a>
32      </div>
33    </div>
34  </section>
35 </div>
```

Pago carrito Payu



```
1 <app-navbar></app-navbar>
2 <div class="pago-container">
3   <!-- Header con diseño mejorado -->
4   <div class="pago-header">
5     <div class="header-content">
6       <div class="title-section">
7         <div class="title-wrapper">
8           <i class="fas fa-credit-card title-icon"></i>
9           <h1>Procesar Pago</h1>
10        </div>
11      </div>
12      <div class="order-summary">
13        <div class="order-badge">
14          <i class="fas fa-receipt"></i>
15          <span>Orden #{{ orderId }}</span>
16        </div>
17      </div>
18    </div>
19  </div>
20  <!-- Contenido principal con diseño mejorado -->
21  <div class="pago-content">
22    <!-- Panel de información de PayU rediseñado -->
23    <div class="payu-info">
24      <div class="payu-header">
25        <div class="payu-logo">
26          <div class="logo-icon">
27            <i class="fas fa-shield-alt"></i>
28          </div>
29          <div class="logo-text">
30            <h2>PayU</h2>
31          </div>
32        </div>
33      </div>
34    </div>
35  </div>
```

Estructura de directorios:

Directorio	Descripción
Controlador	Contiene la clase controlador
Modelo	Contiene las clases que están asociadas con la base de datos.
Vista	Contiene los directorios que implementan la vista de la aplicación.
App/assets/stylesheets	Contiene la definición de los archivos css o estilo en cascada
App/assets/controllers	Contiene los formularios de la aplicación web
App/assets/images	Contiene las imágenes usadas en la plataforma
Gemfile	Contiene las gemas utilizadas

Relación de programas

Programa	Lenguaje	Descripción
models/promoción.rb	Ruby	Implementar promociones en el inicio
models/pizza_tradicional.rb	Ruby	Implementar el grupo pizza
models/grupo.rb	Ruby	Crear grupos para los diferentes productos
app/assets/stylesheets/style	CSS	Contiene los estilos utilizados en todas las interfaces Html.rb
models/ingrediente_producto.rb	Ruby	Implementa ingredientes para los productos

7.

Diccionario de Datos

Grupos

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único del grupo
nombre	string	NO	SÍ	Nombre del grupo/categoría
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Productos

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único del producto
nombre	string	NO	SÍ	Nombre del producto
descripcion	text	SÍ	NO	Descripción del producto
categoria	string	SÍ	NO	Categoría (pizza, hamburguesa, bebida, etc.)
grupo_id	bigint	NO	NO	Relación con grupo
precio	integer	SÍ	NO	Precio base
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Tamaños

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único del tamaño
nombre	string	NO	SÍ	Nombre del tamaño (Personal, Mediano, etc.)
slices	integer	SÍ	NO	Número de porciones
tamano_cm	integer	SÍ	NO	Tamaño en centímetros
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Producto Tamaños

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único
producto_id	bigint	NO	NO	Relación con producto
tamano_id	bigint	NO	NO	Relación con tamaño
precio	integer	NO	NO	Precio para ese tamaño
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Adicionales

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único
nombre	string	NO	SÍ	Nombre del adicional
precio	integer	SÍ	NO	Precio del adicional
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Promociones

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único
titulo	string	NO	NO	Título de la promoción
descripcion	text	SÍ	NO	Descripción de la promoción
imagen	ActiveStorage	SÍ	NO	Imagen asociada
activa	boolean	NO	NO	Si la promoción está activa
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Usuarios

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único
email	string	NO	SÍ	Correo electrónico
encrypted_password	string	NO	NO	Contraseña cifrada
admin	boolean	NO	NO	Si es administrador
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Pedidos

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único
usuario_id	bigint	SÍ	NO	Relación con usuario
total	integer	NO	NO	Total del pedido
estado	string	NO	NO	Estado (pendiente, pagado, entregado, etc.)
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Pedido Productos

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único
pedido_id	bigint	NO	NO	Relación con pedido
producto_id	bigint	NO	NO	Relación con producto
cantidad	integer	NO	NO	Cantidad solicitada
precio_unitario	integer	NO	NO	Precio unitario al momento del pedido
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización

Pagos

Campo	Tipo de Dato	Nulo	Único	Descripción
id	bigint	NO	SÍ	Identificador único
pedido_id	bigint	NO	NO	Relación con pedido
metodo	string	NO	NO	Método de pago (PayU, efectivo, etc.)
estado	string	NO	NO	Estado del pago (pendiente, aprobado, fallido)
referencia	string	SÍ	NO	Referencia de PayU
respuesta	text	SÍ	NO	Respuesta completa de PayU (JSON/texto)
created_at	datetime	NO	NO	Fecha de creación
updated_at	datetime	NO	NO	Fecha de actualización