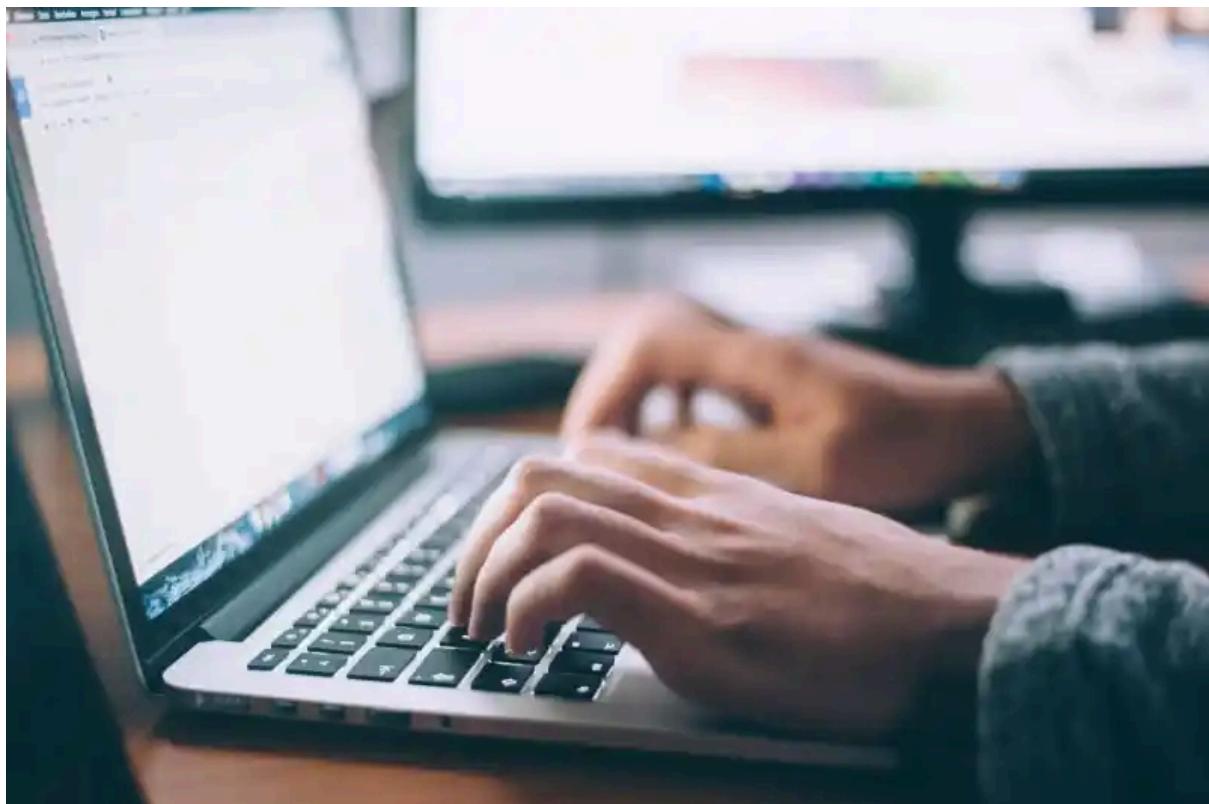


PROYECTO FINAL DE CICLO FORMATIVO

SISTEMA DE GESTIÓN DE UN SUPERMERCADO

CFGS DESARROLLO DE APLICACIONES MULTIPLATAFORMA



Autor: José Antonio Pérez de Prada
Tutora: María Fernanda Hoyuela de la Cueva
Fecha de entrega: 10 de junio de 2024

ÍNDICE

[Diagrama de casos de uso](#)

[Casos de uso de la aplicación](#)

[Diagrama de clases](#)

[Diagrama de Entidad-Relación](#)

[Diagrama de componentes](#)

[Casos de prueba](#)

[Implementación](#)

[Servidor REST con Spring Boot](#)

[Dependencias](#)

[Archivo application.properties de Spring Boot](#)

[Construcción de las tablas de la base de datos](#)

[Arquitectura de la aplicación servidor](#)

[Jerarquía de paquetes y clases](#)

[Repositorios y consultas JPQL](#)

[Data Transfer Objects \(DTO\)](#)

[Spring Security](#)

[API REST](#)

[Controladores](#)

[Control de las excepciones de los controladores](#)

[SPA con React](#)

[Comunicación con el servidor](#)

[Estructura de los componentes React del SPA cliente](#)

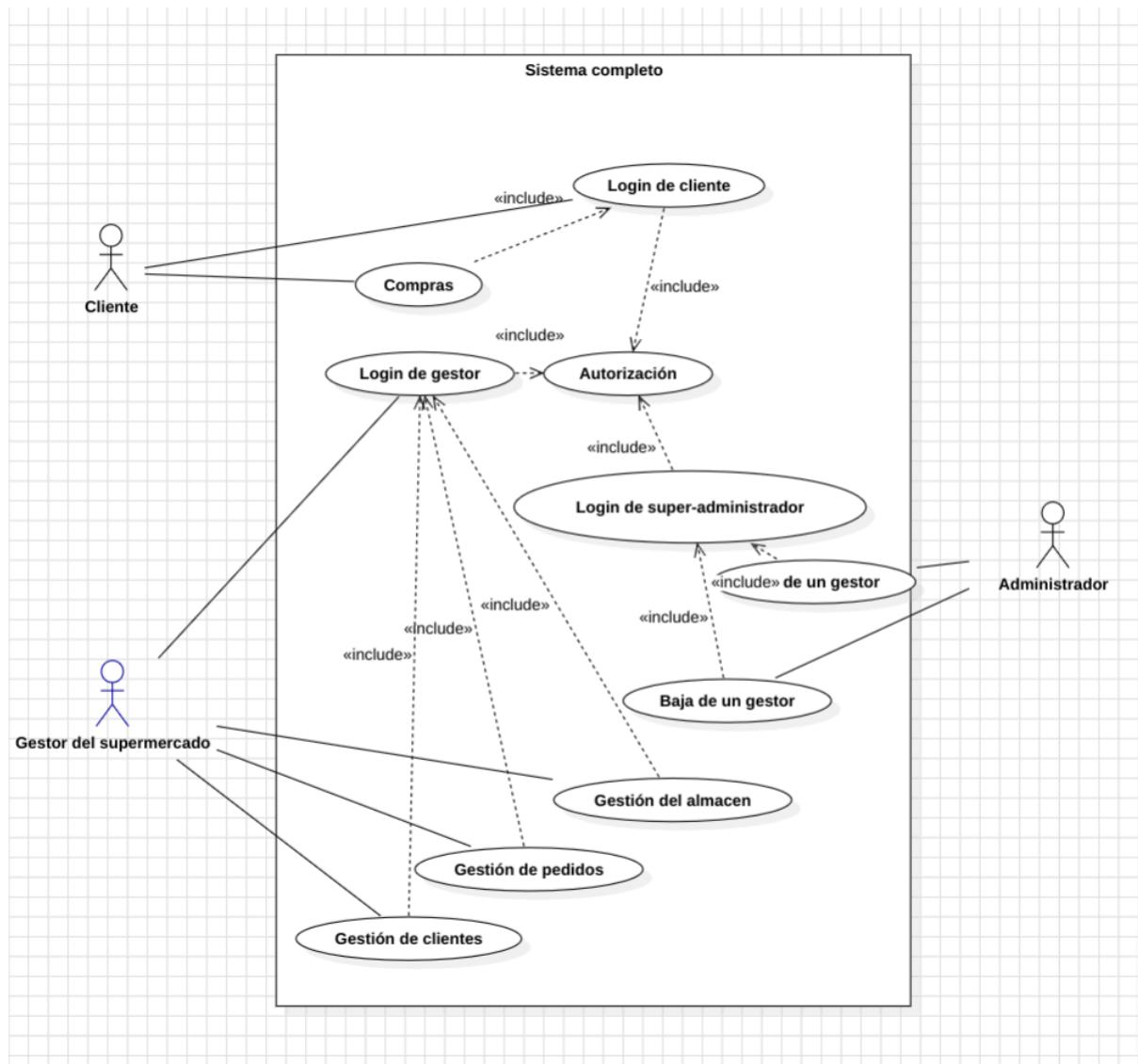
[Instrucciones para la compilación y empaquetado de la aplicación](#)

[Instrucciones para la puesta en marcha de la aplicación](#)

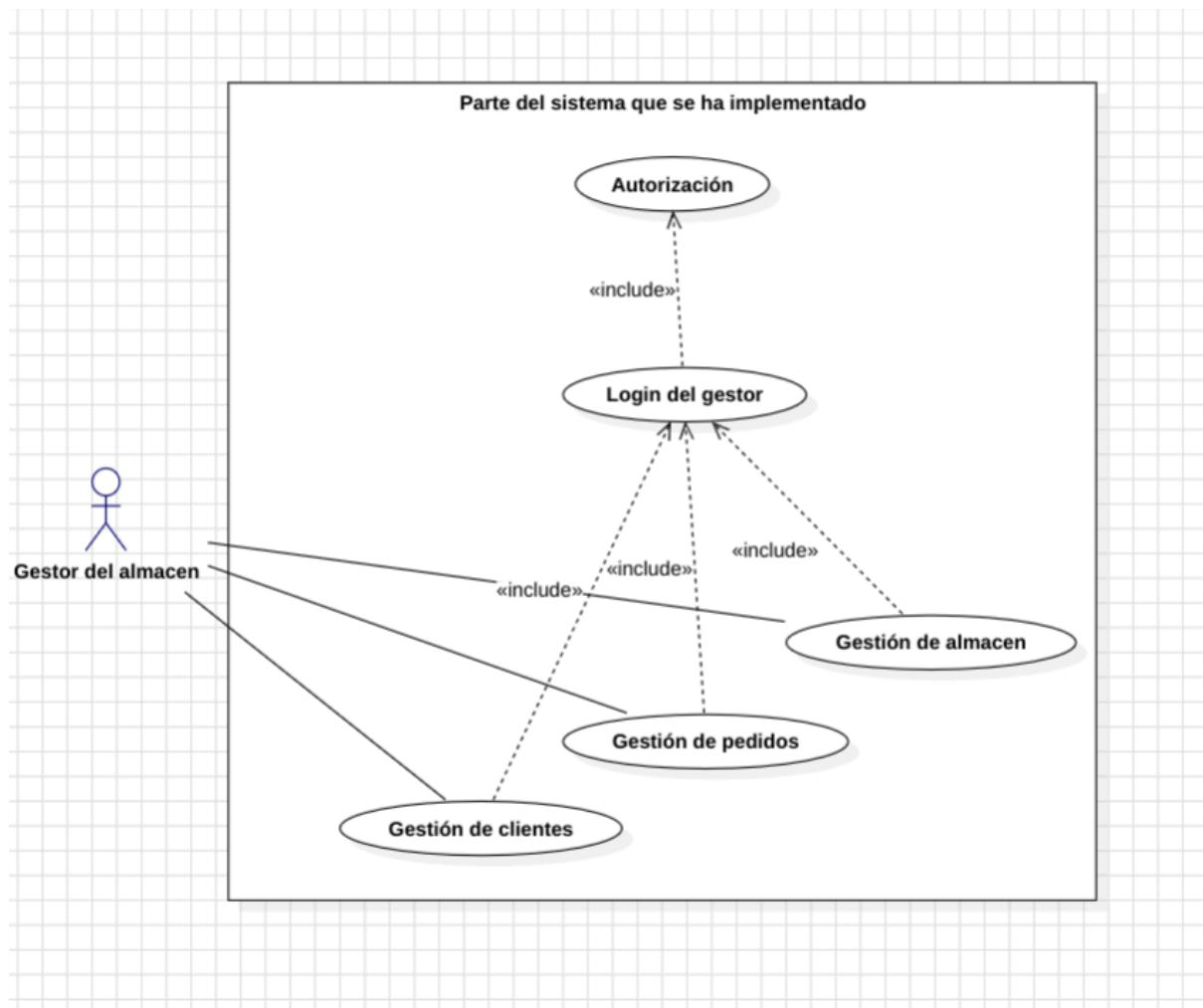
[Omisión de index.html en la ruta](#)

Diagrama de casos de uso

La aplicación desarrollada forma parte de un sistema de comercio electrónico para un supermercado. A continuación se muestran los casos de uso de dicho sistema.



Los casos de uso que gestiona la aplicación de este proyecto son los que se presentan en el siguiente diagrama.



Casos de uso de la aplicación

Caso de Uso 01: LOGIN

<i>Identificador de Caso de Uso</i>	CU-01
<i>Nombre</i>	LOGIN
<i>Descripción</i>	Proceso de autorización y acceso a la web
<i>Actores</i>	Usuario
Secuencia normal	
<i>Actor</i>	<i>Software</i>
1. Introduce el nombre de usuario, la contraseña y pulsa en el botón de Log in	
	2. Se validan las credenciales y se genera un token de acceso
	3. Se accede al menú de opciones de la web
Excepciones	Software
Usuario y contraseña incorrectos	Muestra mensaje de error
Casos de uso relacionados	
<i>Precondición</i>	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios)
<i>Postcondición</i>	El usuario puede acceder al resto de la web

Caso de Uso 02: Mostrar clientes

Identificador de Caso de Uso	CU-02
Nombre	Mostrar clientes
Descripción	Proceso para mostrar el listado de clientes registrados en la web
Actores	Usuario
Secuencia normal	
Actor	Software
1. En el menú de inicio selecciona la opción de clientes	
	2. Se muestran los datos de todos los clientes registrados
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). El usuario debe tener un token de autenticación.
Postcondición	El usuario puede acceder a la línea de clientes (cliente, pedidos del cliente, pedido del cliente, detalles del pedido y detalle del pedido). Crear cliente.

Caso de Uso 03: Mostrar cliente

Identificador de Caso de Uso	CU-03
Nombre	Mostrar cliente
Descripción	Proceso para mostrar datos de un cliente y sus pedidos.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre uno de los clientes de la lista	
	2. Se muestran los datos del cliente.
	3. Se muestran los pedidos del cliente
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes.
Postcondición	El usuario puede acceder a la línea de clientes (cliente, pedidos del cliente, pedido del cliente, detalles del pedido y detalle del pedido). Crear pedido. Editar cliente. Eliminar cliente.

Caso de Uso 04: Mostrar pedido del cliente

Identificador de Caso de Uso	CU-04
Nombre	Mostrar pedido del cliente
Descripción	Proceso para mostrar los datos de un pedido y sus detalles.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre uno de los pedidos de la lista	
	2. Se muestran los datos del pedido.
	3. Se muestran los detalles del pedido.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes.
Postcondición	El usuario puede acceder a la línea de clientes (cliente, pedidos del cliente, pedido del cliente, detalles del pedido y detalle del pedido). Crear detalle Editar pedido. Eliminar pedido.

Caso de Uso 05: Mostrar detalle del pedido

Identificador de Caso de Uso	CU-05
Nombre	Mostrar detalle del pedido
Descripción	Proceso para mostrar los datos del detalle de un pedido
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre uno de los detalles de la lista	
	2. Se muestran los datos del detalle.
	3. Se muestran los detalles del pedido.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes.
Postcondición	El usuario puede acceder a la línea de clientes (cliente, pedidos del cliente, pedido del cliente, detalles del pedido y detalle del pedido). Editar detalle. Eliminar detalle.

Caso de Uso 06: Crear cliente

Identificador de Caso de Uso	CU-06
Nombre	Crear cliente
Descripción	Proceso para crear un nuevo cliente
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de crear cliente	
	2. Se muestra el formulario para crear un cliente.
3. Rellena los campos y le da al botón de añadir	
	4. Se vuelve a la pantalla con la lista de clientes, añadiendo el nuevo cliente.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
El nombre está vacío	Muestra mensaje de error
El apellido está vacío	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes.
Postcondición	Volver a la lista de clientes

Caso de Uso 07: Editar cliente

Identificador de Caso de Uso	CU-07
Nombre	Editar cliente
Descripción	Proceso para editar los datos de un cliente
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de editar cliente	
	2. Se muestra el formulario para editar el cliente con los datos actuales del cliente.
3. Rellena los campos y le da al botón de editar	
	4. Se vuelve a la pantalla con los datos del cliente actualizados.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
El nombre está vacío	Muestra mensaje de error
El apellido está vacío	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes. Debe haber entrado a través de la pantalla del cliente.
Postcondición	Volver a la pantalla del cliente.

Caso de Uso 08: Eliminar cliente

Identificador de Caso de Uso	CU-08
Nombre	Eliminar cliente
Descripción	Proceso para eliminar un cliente
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de eliminar cliente	
	2. Se muestran los datos del cliente para confirmar que se quiere eliminar.
3. Le da al botón de eliminar	
	4. Se vuelve a la lista de clientes.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes. Debe haber entrado a través de la pantalla del cliente.
Postcondición	Volver a la lista de clientes.

Caso de Uso 09: Crear pedido

Identificador de Caso de Uso	CU-09
Nombre	Crear pedido
Descripción	Proceso para crear un nuevo pedido
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de crear pedido	
	2. Se muestra el formulario para crear un pedido.
3. Rellena los campos y le da al botón de añadir	
	4. Se vuelve a la pantalla con la lista de pedidos en la pantalla del cliente, añadiendo el nuevo pedido.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La fecha está vacía	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes.
Postcondición	Volver a la pantalla del cliente

Caso de Uso 10: Editar pedido

Identificador de Caso de Uso	CU-10
Nombre	Editar pedido
Descripción	Proceso para editar los datos de un pedido
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de editar pedido	
	2. Se muestra el formulario para editar el pedido con los datos actuales del pedido.
3. Rellena los campos y le da al botón de editar	
	4. Se vuelve a la pantalla con los datos del pedido actualizados.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La fecha está vacía	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes. Debe haber entrado a través de la pantalla del pedido.
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 11: Eliminar pedido

Identificador de Caso de Uso	CU-11
Nombre	Eliminar pedido
Descripción	Proceso para eliminar un pedido
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de eliminar pedido	
	2. Se muestran los datos del pedido para confirmar que se quiere eliminar.
3. Le da al botón de eliminar	
	4. Se vuelve a la pantalla del cliente.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes. Debe haber entrado a través de la pantalla del pedido.
Postcondición	Volver a la pantalla del cliente.

Caso de Uso 12: Crear detalle

Identificador de Caso de Uso	CU-12
Nombre	Crear detalle
Descripción	Proceso para crear un nuevo detalle
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de crear detalle	
	2. Se muestra el formulario para crear un detalle.
3. Rellena los campos y le da al botón de añadir	
	4. Se vuelve a la pantalla con la lista de pedidos en la pantalla del pedido, añadiendo el nuevo detalle.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La cantidad está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes.
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 13: Editar detalle

Identificador de Caso de Uso	CU-13
Nombre	Editar detalle
Descripción	Proceso para editar los datos de un detalle
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de editar detalle	
	2. Se muestra el formulario para editar el pedido con los datos actuales del detalle.
3. Rellena los campos y le da al botón de editar	
	4. Se vuelve a la pantalla con los datos del detalle actualizados.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La fecha está vacía	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes y pedido. Debe haber entrado a través de la pantalla del detalle.
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 14: Eliminar detalle

Identificador de Caso de Uso	CU-14
Nombre	Eliminar detalle
Descripción	Proceso para eliminar un detalle
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de eliminar detalle	
	2. Se muestran los datos del detalle para confirmar que se quiere eliminar.
3. Le da al botón de eliminar	
	4. Se vuelve a la pantalla del pedido.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
Casos de uso relacionados	
Precondición	<p>El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios).</p> <p>Debe haber pasado previamente por la lista de clientes.</p> <p>Debe haber entrado a través de la pantalla del pedido.</p>
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 15: Mostrar pedidos por estados

Identificador de Caso de Uso	CU-15
Nombre	Mostrar pedidos por estados
Descripción	Proceso para mostrar el listado de pedidos según se quieran filtrar: todos los pedidos, solo los pendientes, los enviados y los entregados.
Actores	Usuario
Secuencia normal	
Actor	Software
1. En el menú de inicio selecciona la opción de pedidos	
	2. Se muestran los datos de todos los pedidos según el filtro que se utilice (todos, pendientes, enviados y entregados)
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). El usuario debe tener un token de autenticación.
Postcondición	El usuario puede acceder a la línea de pedidos (pedido, detalles del pedido, detalle del pedido).

Caso de Uso 16: Mostrar pedido

Identificador de Caso de Uso	CU-16
Nombre	Mostrar pedido
Descripción	Proceso para mostrar datos de un pedido y sus detalles.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre uno de los clientes de la lista	
	2. Se muestran los datos del pedido.
	3. Se muestran los detalles del pedido.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de pedidos.
Postcondición	El usuario puede acceder a la línea de pedidos (pedido, detalles del pedido, detalle del pedido). Crear detalle. Editar pedido. Eliminar pedido.

Caso de Uso 17: Mostrar detalle del pedido

Identificador de Caso de Uso	CU-17
Nombre	Mostrar detalle del pedido
Descripción	Proceso para mostrar los datos de un detalle del pedido.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre uno de los detalles de la lista	
	4. Se muestran los datos del detalle.
Excepciones	
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de clientes.
Postcondición	El usuario puede acceder a la línea de pedidos (pedido, detalles del pedido, detalle del pedido). Editar detalle. Eliminar detalle.

Caso Uso de 18: Crear pedido

Identificador de Caso de Uso	CU-18
Nombre	Crear pedido
Descripción	Proceso para crear un nuevo pedido
Actores	Usuario
Secuencia normal	
Actor	Software
5. Hace click sobre el botón de crear pedido	
	6. Se muestra el formulario para crear un pedido.
7. Rellena los campos y le da al botón de añadir	
	8. Se vuelve a la pantalla con la lista de pedidos en la pantalla del cliente, añadiendo el nuevo pedido.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La fecha está vacía	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de pedidos.
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 19: Editar pedido

Identificador de Caso de Uso	CU-19
Nombre	Editar pedido
Descripción	Proceso para editar los datos de un pedido
Actores	Usuario
Secuencia normal	
Actor	Software
5. Hace click sobre el botón de editar pedido	
	6. Se muestra el formulario para editar el pedido con los datos actuales del pedido.
7. Rellena los campos y le da al botón de editar	
	8. Se vuelve a la pantalla con los datos del pedido actualizados.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La fecha está vacía	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de pedidos.
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 20: Eliminar pedido

Identificador de Caso de Uso	CU-20
Nombre	Eliminar pedido
Descripción	Proceso para eliminar un pedido
Actores	Usuario
Secuencia normal	
Actor	Software
5. Hace click sobre el botón de eliminar pedido	
	6. Se muestran los datos del pedido para confirmar que se quiere eliminar.
7. Le da al botón de eliminar	
	8. Se vuelve a la lista de los pedidos.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de pedidos.
Postcondición	Volver a la lista de pedidos.

Caso de Uso 21: Crear detalle

Identificador de Caso de Uso	CU-21
Nombre	Crear detalle
Descripción	Proceso para crear un nuevo detalle
Actores	Usuario
Secuencia normal	
Actor	Software
5. Hace click sobre el botón de crear detalle	
	6. Se muestra el formulario para crear un detalle.
7. Rellena los campos y le da al botón de añadir	
	8. Se vuelve a la pantalla con la lista de pedidos en la pantalla del pedido, añadiendo el nuevo detalle.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La cantidad está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de pedidos.
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 22: Editar detalle

Identificador de Caso de Uso	CU-22
Nombre	Editar detalle
Descripción	Proceso para editar los datos de un detalle
Actores	Usuario
Secuencia normal	
Actor	Software
5. Hace click sobre el botón de editar detalle	
	6. Se muestra el formulario para editar el pedido con los datos actuales del detalle.
7. Rellena los campos y le da al botón de editar	
	8. Se vuelve a la pantalla con los datos del detalle actualizados.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
La fecha está vacía	Muestra mensaje de error
La dirección está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de pedidos. Debe haber entrado a través de la pantalla del detalle.
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 23: Eliminar detalle

Identificador de Caso de Uso	CU-23
Nombre	Eliminar detalle
Descripción	Proceso para eliminar un detalle
Actores	Usuario
Secuencia normal	
Actor	Software
5. Hace click sobre el botón de eliminar detalle	
	6. Se muestran los datos del detalle para confirmar que se quiere eliminar.
7. Le da al botón de eliminar	
	8. Se vuelve a la pantalla del pedido.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
Casos de uso relacionados	
Precondición	<p>El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios).</p> <p>Debe haber pasado previamente por la lista de clientes.</p> <p>Debe haber entrado a través de la pantalla del pedido.</p>
Postcondición	Volver a la pantalla del pedido.

Caso de Uso 24: Mostrar categorías

Identificador de Caso de Uso	CU-24
Nombre	Mostrar categorías
Descripción	Proceso para mostrar el listado de categorías
Actores	Usuario
Secuencia normal	
Actor	Software
1. En el menú de inicio selecciona la opción de categorías	
	2. Se muestran los datos de todos las categorías.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). El usuario debe tener un token de autenticación.
Postcondición	El usuario puede acceder a la línea de categorías (categoría, productos, producto). Crear categoría.

Caso de Uso 25: Mostrar categoría

Identificador de Caso de Uso	CU-25
Nombre	Mostrar categoría
Descripción	Proceso para mostrar datos de una categoría y sus productos.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre uno de las categorías de la lista	
	2. Se muestran los datos de la categoría.
	3. Se muestran los productos.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de categorías.
Postcondición	El usuario puede acceder a la línea de categorías (categoría, productos, producto). Crear producto. Editar categoría. Eliminar categoría.

Caso de Uso 26: Mostrar producto

Identificador de Caso de Uso	CU-26
Nombre	Mostrar producto
Descripción	Proceso para mostrar los datos de un producto.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre uno de los productos de la lista	
	2. Se muestran los datos del producto.
Excepciones	
El token de autenticación ha expirado	Muestra mensaje de error
No se encuentran datos	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de categorías.
Postcondición	El usuario puede acceder a la línea de categorías (categoría, productos, producto). Editar producto. Eliminar producto.

Caso de Uso 27: Crear categoría

Identificador de Caso de Uso	CU-27
Nombre	Crear categoría
Descripción	Proceso para crear una nueva categoría.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de crear categoría	
	2. Se muestra el formulario para crear una categoría.
3. Rellena los campos y le da al botón de añadir	
	4. Se vuelve a la pantalla de lista de categorías, añadiendo la nueva categoría.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
El nombre está vacío	Muestra mensaje de error
La descripción está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de categorías.
Postcondición	Volver a la lista de categorías.

Caso de Uso 28: Editar categoría

Identificador de Caso de Uso	CU-28
Nombre	Editar categoría
Descripción	Proceso para editar los datos de una categoría
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de editar categoría	
	2. Se muestra el formulario para editar la categoría con los datos actuales de la categoría.
3. Rellena los campos y le da al botón de editar	
	4. Se vuelve a la pantalla con los datos de la categoría actualizados.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
El nombre está vacío.	Muestra mensaje de error
La descripción está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de categorías.
Postcondición	Volver a la pantalla de la categoría.

Caso de Uso 29: Eliminar categoría

Identificador de Caso de Uso	CU-29
Nombre	Eliminar categoría
Descripción	Proceso para eliminar una categoría.
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de eliminar categoría.	
	2. Se muestran los datos de la categoría para confirmar que se quiere eliminar.
3. Le da al botón de eliminar	
	4. Se vuelve a la lista de las categorías.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de categorías.
Postcondición	Volver a la lista de categorías.

Caso de Uso 30: Crear producto

Identificador de Caso de Uso	CU-30
Nombre	Crear producto
Descripción	Proceso para crear un nuevo producto
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de crear producto	
	2. Se muestra el formulario para crear un producto.
3. Rellena los campos y le da al botón de añadir	
	4. Se vuelve a la pantalla con la lista de productos en la pantalla de la categoría, añadiendo el nuevo producto.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
El nombre está vacío	Muestra mensaje de error
El precio está vacío	Muestra mensaje de error
La cantidad está vacía	Muestra mensaje de error
La descripción está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios). Debe haber pasado previamente por la lista de categorías.
Postcondición	Volver a la pantalla de productos.

Caso de Uso 31: Editar producto

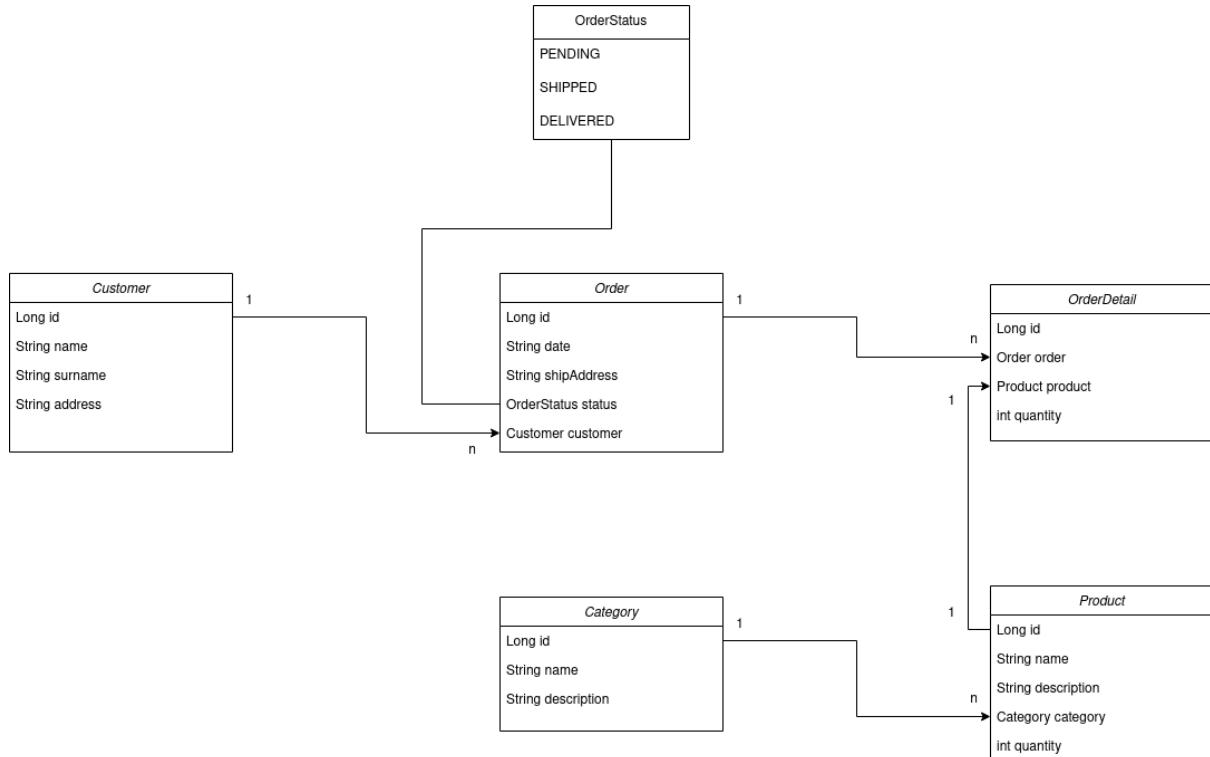
Identificador de Caso de Uso	CU-31
Nombre	Editar producto
Descripción	Proceso para editar los datos de un producto
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de editar producto	
	2. Se muestra el formulario para editar el producto con los datos actuales del producto.
3. Rellena los campos y le da al botón de editar	
	4. Se vuelve a la pantalla con los datos del producto actualizados.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
El nombre está vacío	Muestra mensaje de error
El precio está vacío	Muestra mensaje de error
La cantidad está vacía	Muestra mensaje de error
La descripción está vacía	Muestra mensaje de error
Casos de uso relacionados	
Precondición	<p>El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios).</p> <p>Debe haber pasado previamente por la lista de productos.</p> <p>Debe haber entrado a través de la pantalla del producto.</p>
Postcondición	Volver a la pantalla del producto.

Caso de Uso 32: Eliminar producto

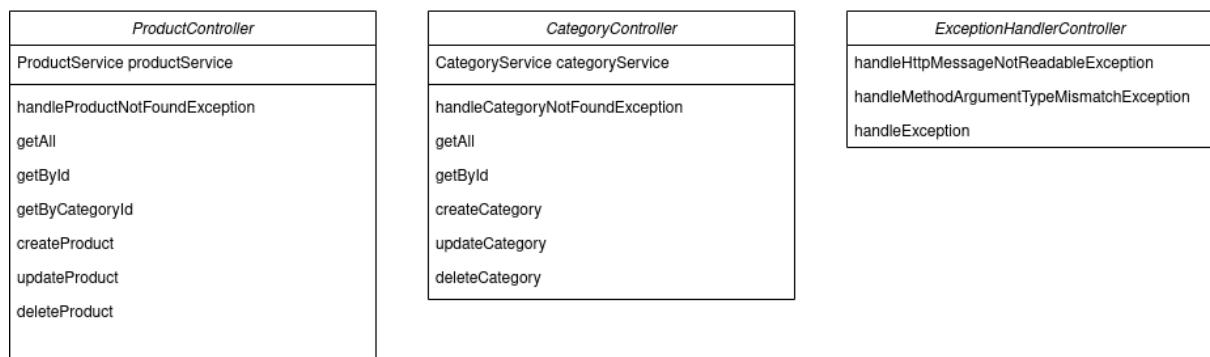
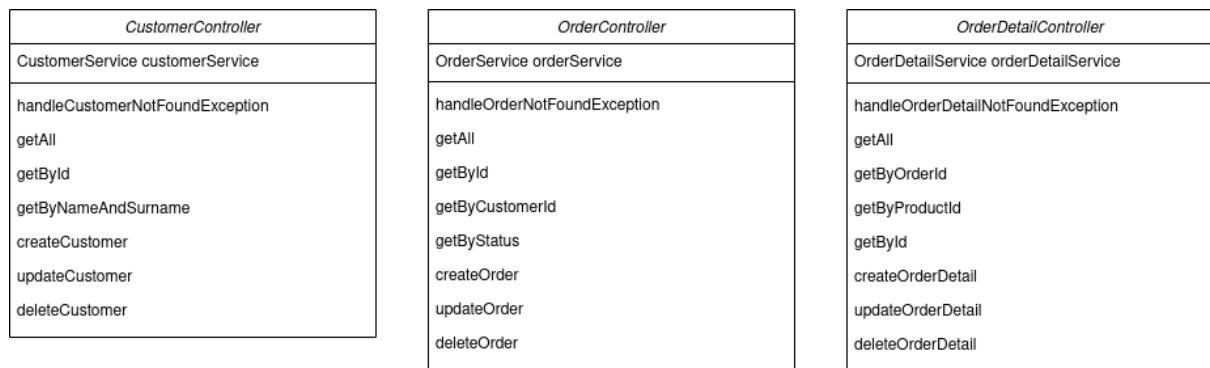
Identificador de Caso de Uso	CU-32
Nombre	Eliminar producto
Descripción	Proceso para eliminar un producto
Actores	Usuario
Secuencia normal	
Actor	Software
1. Hace click sobre el botón de eliminar producto	
	2. Se muestran los datos del producto para confirmar que se quiere eliminar.
3. Le da al botón de eliminar	
	4. Se vuelve a la pantalla de la categoría.
Excepciones	Software
El token de autenticación ha expirado	Muestra mensaje de error
Casos de uso relacionados	
Precondición	<p>El usuario debe estar registrado (desde otra aplicación que gestiona los usuarios).</p> <p>Debe haber pasado previamente por la lista de productos.</p> <p>Debe haber entrado a través de la pantalla del producto.</p>
Postcondición	Volver a la pantalla de la categoría.

Diagrama de clases

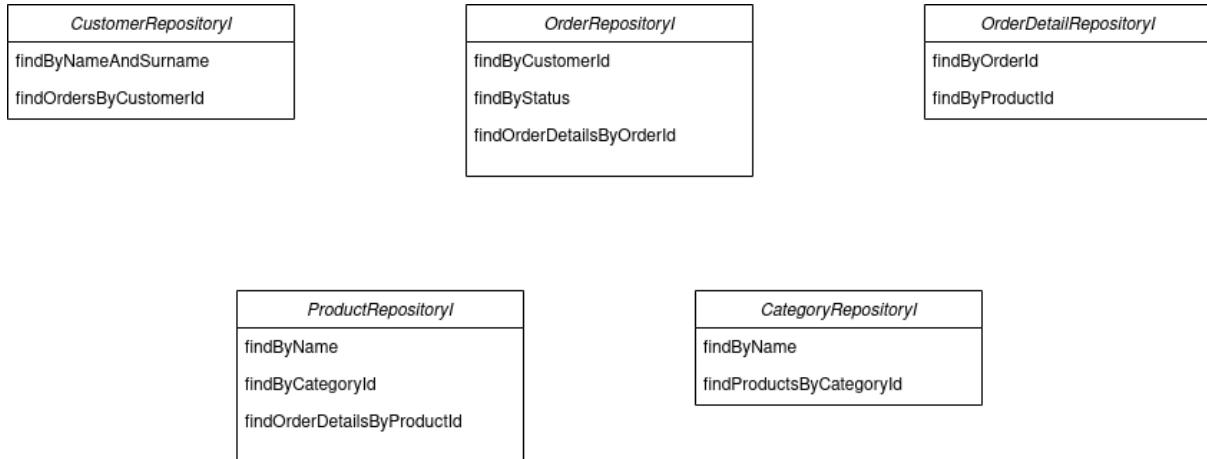
Modelo



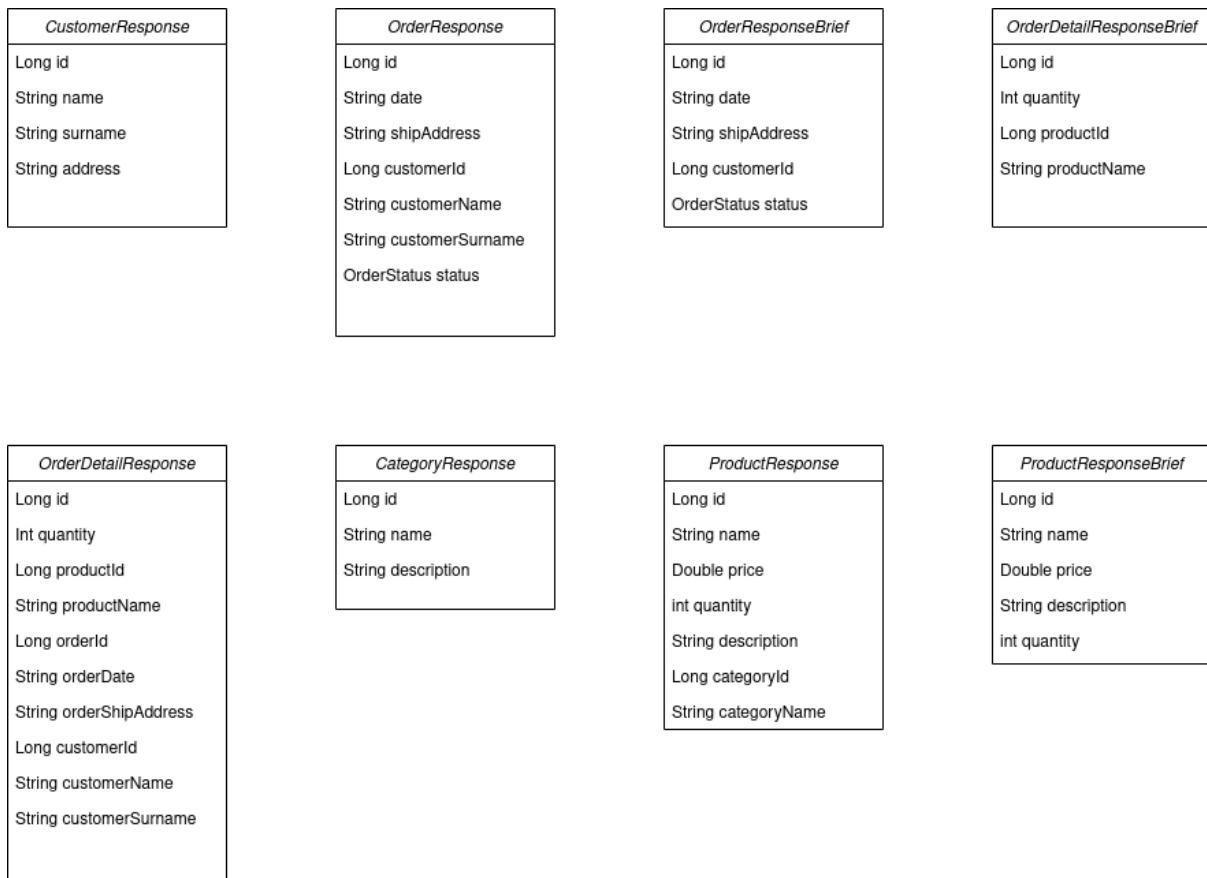
Controlador



Repositorio



DTO: Respuesta



DTO: Petición

<i>CustomerRequest</i>	<i>OrderRequest</i>	<i>OrderDetailRequest</i>
String name	String date	Long orderId
String surname	String shipAddress	Long productId
String address	Long customerId	Int quantity
	OrderStatus status	
<i>ProductRequest</i>	<i>CategoryRequest</i>	
String name	String name	
Double price	String description	
String description		
Long categoryId		
int quantity		

Servicio

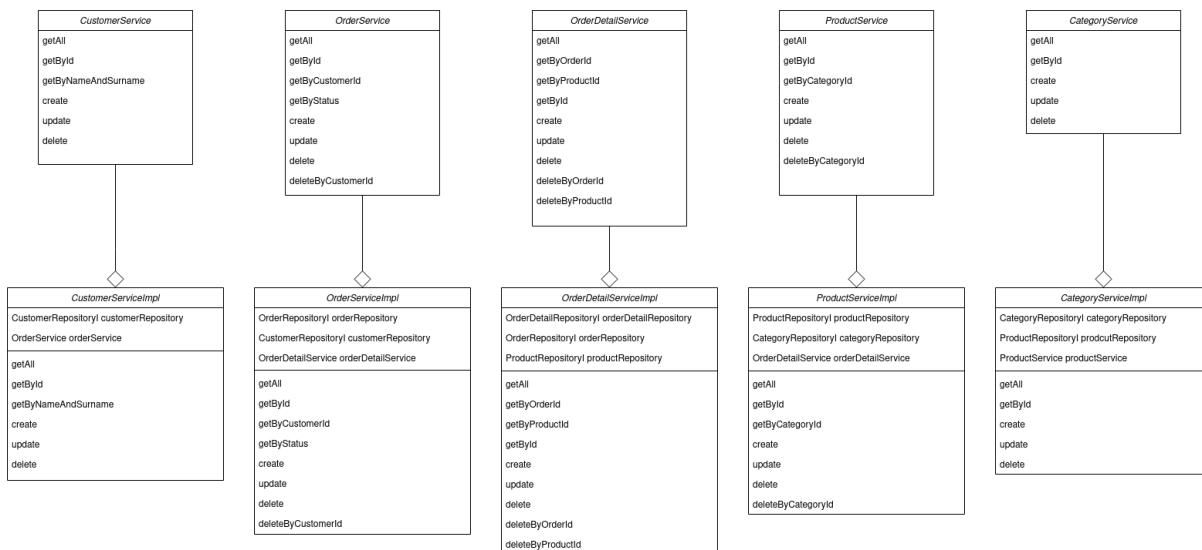


Diagrama de Entidad-Relación

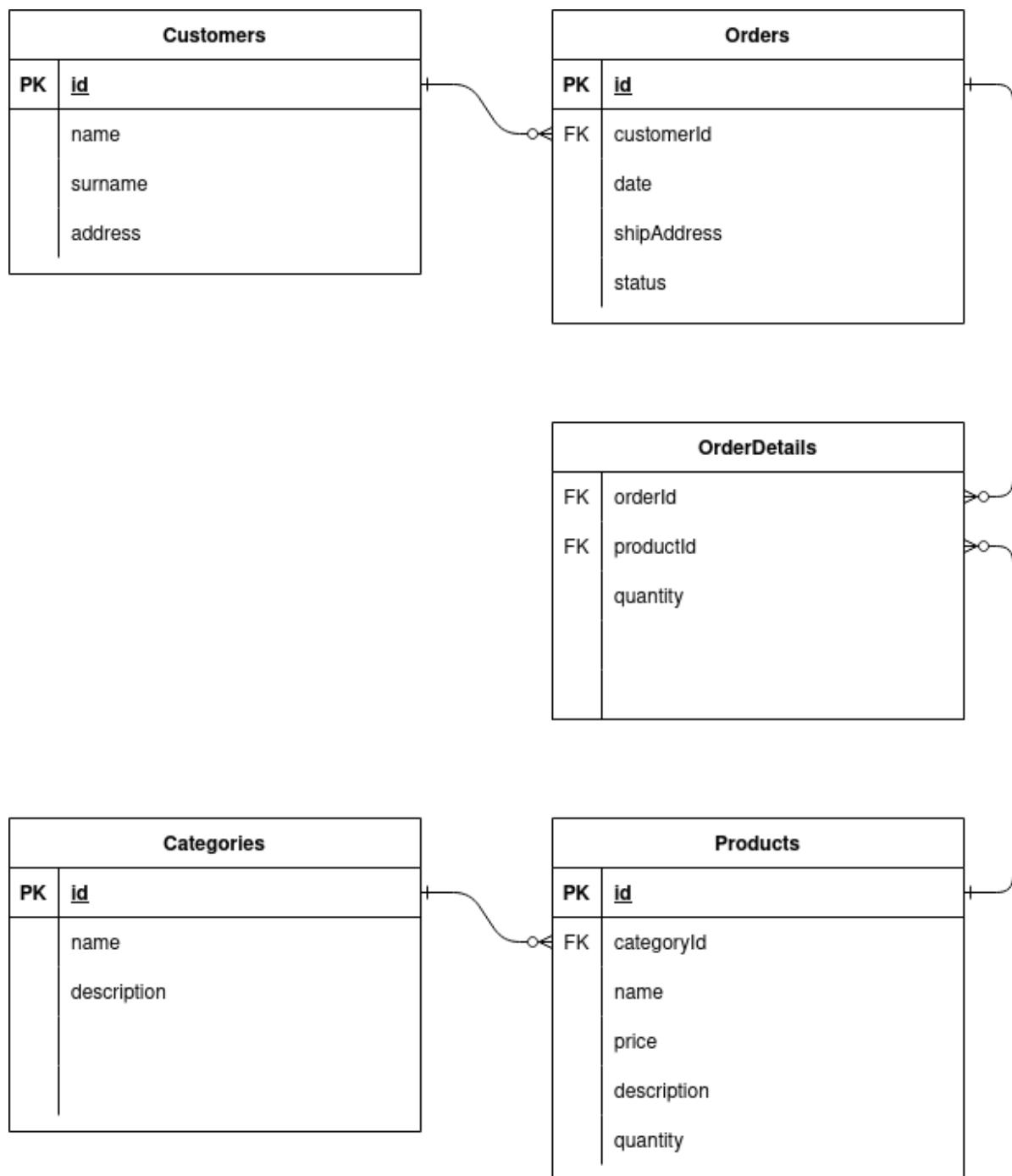
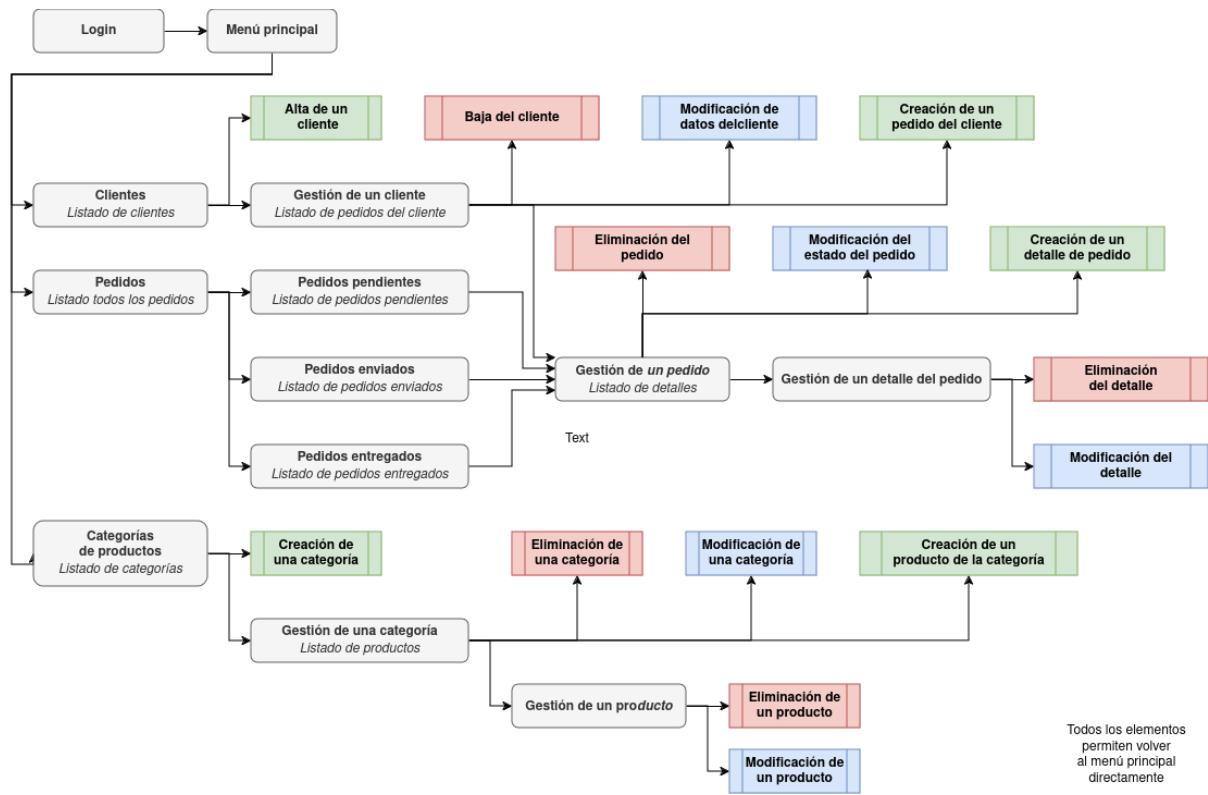


Diagrama de componentes



[Diseño de figma](#)

Casos de prueba

CU-01 LOGIN		
Objetivo probado	Requisitos probados	Pruebas a realizar
Log in correcto y obtención del token de Spring Security.	Se comprueba que al iniciar sesión se devuelve el token para que se utilice en el resto de peticiones.	Acceder a la web, introducir un usuario y contraseña correcto y comprobar si accede al menú de inicio.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire y muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.
Log in incorrecto.	No se permitirá acceder al resto de la web si el log in no es correcto.	<p>Se tendrá que intentar acceder a la web de las siguientes maneras:</p> <ul style="list-style-type: none">- Nombre y contraseña vacíos.- Nombre correcto y contraseña vacía.- Nombre incorrecto y contraseña vacía.- Nombre vacío y contraseña correcta.- Nombre vacío y contraseña incorrecta.- Nombre y contraseña incorrectos.- Nombre y contraseña correctos.

CU-02 Mostrar clientes		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los clientes de la lista con todos los datos indicados en dicha lista.	Se comprueba que al acceder a la página de la lista de clientes, aparecen los datos de id, nombre y apellido y dirección.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-03 Mostrar cliente		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los datos del cliente.	Se comprueba que al acceder a la página de un cliente en concreto, aparecen los datos de id, nombre y apellido y dirección, la lista de pedidos del cliente con un id, fecha, dirección de envío y estado.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio y seleccionar uno de los clientes de la lista.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-04 Mostrar pedido del cliente</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los datos del pedido.	Se comprueba que al acceder a la página de un pedido en concreto del cliente, aparecen los datos de id, nombre y apellido del cliente, estado del pedido, el importe total, fecha y dirección de envío y estado, además de los detalles con su id, el nombre del producto, la cantidad y el importe.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los clientes de la lista y luego seleccionar uno de los pedidos de la lista.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-05 Mostrar detalle del pedido</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los datos del detalle.	Se comprueba que al acceder a la página de un detalle en concreto del pedido, aparecen los datos de id, id del pedido, el nombre del producto, la cantidad y el importe.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los clientes de la lista, luego seleccionar uno de los pedidos de la lista y por último uno de la lista de detalles.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-06 Crear cliente</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se crea el cliente correctamente.	Se comprueba que al crear un cliente, este tiene tanto su id, como su nombre y apellido y su dirección.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar el botón de crear cliente y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-07 Editar cliente</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se edita el cliente correctamente.	Se comprueba que al editar un cliente, este tiene tanto su id, como su nombre y apellido y su dirección modificados.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar un cliente, seleccionar la opción de editar cliente y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-08 Eliminar cliente		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se elimina el cliente correctamente.	Se comprueba que al eliminar un cliente, este no aparece en la lista de clientes.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar un cliente, seleccionar la opción de eliminar cliente y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-09 Crear pedido		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se crea el pedido correctamente.	Se comprueba que al crear un pedido, este tiene tanto su id, como su fecha, su dirección y con el estado PENDING.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los clientes y seleccionar el botón de crear pedido y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-10 Editar pedido		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se edita el pedido correctamente.	Se comprueba que al editar un pedido, este tiene tanto su id, como su fecha, su dirección y estado.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar un cliente, seleccionar un pedido, seleccionar la opción de editar pedido y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-11 Eliminar pedido		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se elimina el pedido correctamente.	Se comprueba que al eliminar un pedido, este no aparece en la lista de pedidos del cliente.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar un cliente, seleccionar un pedido del cliente, seleccionar la opción de eliminar pedido y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-12 Crear detalle		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se crea el detalle correctamente.	Se comprueba que al crear un detalle, este tiene tanto su id, como el nombre del producto, la cantidad y el importe calculado en función del precio y la cantidad del producto.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los clientes, seleccionar uno de los pedido y seleccionar el botón de crear detalle y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-13 Editar detalle		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se edita el detalle correctamente.	Se comprueba que al editar un detalle, este tiene tanto su id, como el nombre del producto, la cantidad y el importe calculado en función del precio y la cantidad del producto.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los clientes, seleccionar uno de los pedido, seleccionar uno de los detalles y seleccionar el botón de editar detalle y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-14 Eliminar detalle</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se elimina el detalle correctamente.	Se comprueba que al eliminar un detalle, este no aparece en la lista de detalles del pedido.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los clientes, seleccionar uno de los pedido, seleccionar uno de los detalles y seleccionar el botón de eliminar detalle y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-15 Mostrar pedidos</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los pedidos con sus correspondientes filtros según el estado que tengan (todos, pendientes, enviados o entregados).	Se comprueba que al acceder a la página de la lista de pedidos, aparecen los datos de cada pedido a excepción del nombre del cliente y que sus estados corresponden a los de los filtros a excepción del de todos los pedidos.	Acceder a la pantalla de los pedidos a través del menú de acciones que se encuentra en el menú de inicio.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-16 Mostrar pedido</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los datos del pedido.	Se comprueba que al acceder a la página de un pedido en concreto del cliente, aparecen los datos de id, nombre y apellido del cliente, estado del pedido, el importe total, fecha y dirección de envío y estado, además de los detalles con su id, el nombre del producto, la cantidad y el importe.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los pedidos de la lista.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-17 Mostrar detalle del pedido</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los datos del detalle.	Se comprueba que al acceder a la página de un detalle en concreto del pedido, aparecen los datos de id, id del pedido, el nombre del producto, la cantidad y el importe.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los pedidos de la lista y por último uno de la lista de detalles.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-18 Editar pedido</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se edita el pedido correctamente.	Se comprueba que al editar un pedido, este tiene tanto su id, como su fecha, su dirección y estado.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar un pedido, seleccionar la opción de editar pedido y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-19 Eliminar pedido</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se elimina el pedido correctamente.	Se comprueba que al eliminar un pedido, este no aparece en la lista de pedidos del cliente.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar un pedido, seleccionar la opción de eliminar pedido y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-20 Crear detalle</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se crea el detalle correctamente.	Se comprueba que al crear un detalle, este tiene tanto su id, como el nombre del producto, la cantidad y el importe calculado en función del precio y la cantidad del producto.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los pedidos y seleccionar el botón de crear detalle y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-21 Editar detalle</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se edita el detalle correctamente.	Se comprueba que al editar un detalle, este tiene tanto su id, como el nombre del producto, la cantidad y el importe calculado en función del precio y la cantidad del producto.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los pedidos, seleccionar uno de los detalles y seleccionar el botón de editar detalle y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-22 Eliminar detalle</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se elimina el detalle correctamente.	Se comprueba que al eliminar un detalle, este no aparece en la lista de detalles del pedido.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar uno de los pedidos, seleccionar uno de los detalles y seleccionar el botón de eliminar detalle y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-23 Mostrar categorías</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos las categorías de la lista con todos los datos indicados en dicha lista.	Se comprueba que al acceder a la página de la lista de categorías, aparecen los datos de id, nombre y descripción.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-24 Mostrar categoría</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los datos de la categoría.	Se comprueba que al acceder a la página de un cliente en concreto, aparecen los datos de id, nombre y descripción, la lista de productos de la categoría con un id, nombre, precio, cantidad y descripción.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio y seleccionar una de las categorías de la lista.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-25 Mostrar producto de la categoría</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se muestran todos los datos del producto.	Se comprueba que al acceder a la página de un producto en concreto de la categoría, aparecen los datos de id, nombre, precio, cantidad y descripción.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar una de las categorías de la lista y luego seleccionar uno de los productos de la lista.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-26 Crear categoría		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se crea la categoría correctamente.	Se comprueba que al crear una categoría, esta tiene tanto su id, como su nombre y su descripción.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar el botón de crear categoría y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-27 Editar categoría		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se edita la categoría correctamente.	Se comprueba que al editar una categoría, esta tiene tanto su id, como su nombre y su descripción.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar una categoría, seleccionar el botón de editar categoría y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-28 Eliminar cliente		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se elimina la categoría correctamente.	Se comprueba que al eliminar una categoría, este no aparece en la lista de categorías.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar una categoría, seleccionar el botón de eliminar categoría y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

CU-29 Crear producto		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se crea el producto correctamente.	Se comprueba que al crear un producto, este tiene tanto su id, como su nombre, su precio, su cantidad y su descripción.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar una de las categorías, seleccionar el botón de crear producto y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-30 Editar pedido</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se edita el producto correctamente.	Se comprueba que al editar un producto, este tiene tanto su id, como su nombre, su precio, su cantidad y su descripción.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar una de las categorías, seleccionar uno y seleccionar el botón de editar producto y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

<i>CU-31 Eliminar producto</i>		
Objetivo probado	Requisitos probados	Pruebas a realizar
Comprobar que se elimina el producto correctamente.	Se comprueba que al eliminar un producto, este no aparece en la lista de productos de la categoría.	Acceder a la pantalla de los clientes a través del menú de acciones que se encuentra en el menú de inicio, seleccionar una de las categorías, seleccionar uno y seleccionar el botón de eliminar producto y llenar los campos.
Error de no autorizado si el token expira.	El token está configurado para caducar a los 60 minutos.	Esperar a que el token expire e intentar acceder a alguna de las opciones del menú, haciendo que muestre un mensaje de error que indique su expiración, pidiendo que se vuelva a iniciar sesión.

Implementación

Servidor REST con Spring Boot

Dependencias

Dependencias generales

groupId	artifactId
org.springframework.boot	spring-boot-starter-web
	spring-boot-devtools
	spring-boot-starter-test
org.projectlombok	lombok

Dependencias de Spring Security con JWT

groupId	artifactId
org.springframework.boot	spring-boot-starter-security
org.springframework.security	spring-security-test
io.jsonwebtoken	jjwt-api
	jjwt-impl
	jjwt-jackson

Dependencias relacionadas con las bases de datos

Hay que elegir entre MySQL y H2. En cada caso es necesario adaptar también el archivo application.properties.

groupId	artifactId
org.springframework.boot	spring-boot-starter-data-jpa
com.mysql	mysql-connector-j
com.h2database	h2

Archivo application.properties de Spring Boot

En este archivo configuramos:

- El acceso a la base de datos.
- El nivel de logging (utilizado principalmente en la capa de controladores)

Durante el desarrollo hemos utilizado la siguiente versión:

```
logging.level.org.springframework.web: INFO
logging.level.org.hibernate: ERROR

spring.datasource.url=jdbc:mysql://localhost:3306/japdp_cfgs_dam_tf?createDatabaseIfNotExist=true
spring.datasource.username=*****
spring.datasource.password=*****

spring.jpa.hibernate.ddl-auto=create-drop
spring.jpa.show-sql=true
```

Durante las pruebas, las tablas de la base de datos son inicializadas con datos de prueba por la clase de `DemoDatabaseLoader` que implementa la interfaz `CommandLineRunner` con anotación `@Component`.

En producción el archivo a utilizar sería el siguiente:

```
logging.level.org.springframework.web: INFO
logging.level.org.hibernate: ERROR

spring.datasource.url=jdbc:mysql://localhost:3306/japdp_cfgs_dam_tf
spring.datasource.username=*****
spring.datasource.password=*****
```

Construcción de las tablas de la base de datos

Gracias a JPA, las instrucciones para la construcción de las tablas de la base de datos pueden generarse de forma automática. Aquí las tenemos:

```
create table t_category (
    id bigint not null auto_increment,
    description varchar(255),
    name varchar(255),
    primary key (id)
) engine=InnoDB;

create table t_customer (
    id bigint not null auto_increment,
    address varchar(255), name varchar(255),
    surname varchar(255),
    primary key (id)
) engine=InnoDB;

create table t_order (
    customer_id bigint, id bigint not null auto_increment,
    date varchar(255),
    ship_address varchar(255),
    status enum ('PENDING', 'SHIPPED', 'DELIVERED') not null,
    primary key (id)
) engine=InnoDB;

create table t_orderdetail (
    quantity integer not null,
    id bigint not null auto_increment,
    order_id bigint,
    product_id bigint,
    primary key (id)
) engine=InnoDB;

create table t_product (
    price float(53) not null,
    quantity integer not null,
    category_id bigint,
    id bigint not null auto_increment,
    description varchar(255),
    name varchar(255),
    primary key (id)
) engine=InnoDB;

create table t_user (
    id bigint not null auto_increment,
    creation_date varchar(255),
    description varchar(255),
    email varchar(255),
    password varchar(255),
    username varchar(255),
    role enum ('ADMIN', 'USER') not null,
    primary key (id)
) engine=InnoDB;
```

```
alter table t_user add constraint
  UK_i6qjjoe560mee5ajdg7v1o6mi
  unique (email);

alter table t_user add constraint
  UK_jhib4legehrm4yscx9t3lirqi
  unique (username);

alter table t_order add constraint
  FKesy3n2gc3fa0s3trrk3tvyy9a
  foreign key (customer_id)
  references t_customer (id);

alter table t_orderdetail add constraint
  FKnjshvjicfnhhppqn7cta1jgbu
  foreign key (order_id)
  references t_order (id);

alter table t_orderdetail add constraint
  FK2x36oaauhp7p60nt1xdg0tpy
  foreign key (product_id)
  references t_product (id);

alter table t_product add constraint
  FKp17nkwpqnnxh5iax87dc58sp3
  foreign key (category_id)
  references t_category (id);
```

Arquitectura de la aplicación servidor

Se ha utilizado el patrón **Controlador-Servicio-Persistencia** que permite una clara separación de las responsabilidades de la aplicación:

- La capa **Controlador** es la única responsable de exponer la funcionalidad que puede ser consumida por entidades externas.
- La capa de **Servicio** es la que se encarga de la lógica empresarial.
- La capa de **Persistencia** es la responsable de modelar, almacenar y recuperar los datos.

Jerarquía de paquetes y clases

Lanzador de la aplicación

```
japdp.damtf
├── Main
├── ForwardToIndexConfig
└── DemoDatabaseLoader (Solo para pruebas)
```

Lógica de la aplicación

```
japdp.damtf.application.controller
├── CategoryController
├── CustomerController
├── ExceptionHandlerController
├── OrderController
├── OrderDetailController
└── ProductController

japdp.damtf.application.service
├── CategoryService
├── CategoryServiceImpl
├── CustomerService
├── CustomerServiceImpl
├── OrderDetailsService
├── OrderDetailServiceImpl
├── OrderService
├── OrderServiceImpl
├── ProductService
└── ProductServiceImpl
```

```
japdp.damtf.application.persistence
└── model
    ├── Category
    ├── Customer
    ├── OrderDetail
    ├── Order
    ├── OrderStatus
    └── Product
└── repository
    ├── CategoryRepository
    ├── CustomerRepository
    ├── OrderDetailRepository
    ├── OrderRepository
    └── ProductRepository

japdp.damtf.application.dto
└── request
    ├── CategoryRequest
    ├── CustomerRequest
    ├── OrderDetailRequest
    ├── OrderRequest
    └── ProductRequest
└── response
    ├── CategoryResponse
    ├── CustomerResponse
    ├── OrderDetailResponseBrief
    ├── OrderDetailResponse
    ├── OrderResponseBrief
    ├── OrderResponse
    ├── ProductResponseBrief
    └── ProductResponse

japdp.damtf.application.exception
└── CategoryNotFoundException
└── CustomerNotFoundException
└── OrderDetailNotFoundException
└── OrderNotFoundException
└── ProductNotFoundException
```

Seguridad de la aplicación

```
japdp.damtf.security
  └── authentication
      ├── AuthenticationController
      ├── AuthenticationService
      └── dto
          ├── AuthenticationResponseDTO
          ├── LoginRequestDTO
          └── RegisterRequestDTO
  └── configuration
      ├── AuthenticationManagerConfiguration
      ├── AuthenticationProviderConfiguration
      ├── PasswordEncoderConfiguration
      ├── SecurityFilterChainConfiguration
      └── UserDetailsServiceImpl
  └── jwt
      ├── JwtAuthenticationFilter
      └── JwtService
  └── persistence
      ├── model
          ├── Role
          └── User
      └── repository
          └── UserRepository
```

Repositorios y consultas JPQL

Aunque Spring Data permite la construcción de métodos de consulta derivada, he optado por indicar explícitamente la consulta JPQL asociada a cada método. A continuación se muestran las extensiones de la interfaz JpaRepository de la aplicación en las que pueden verse las consultas utilizadas:

CategoryRepository

```
@Repository
public interface CategoryRepository extends JpaRepository<Category, Long> {

    @Query("SELECT c FROM Category c WHERE c.name=:name")
    public Optional<Category> findByName(String name);

    @Query("SELECT p FROM Product p JOIN p.category c WHERE c.id=:id")
    public List<Product> findProductsByCategoryId(long id);

}
```

ProductRepository

```
@Repository
public interface ProductRepository extends JpaRepository<Product, Long> {

    @Query("SELECT p FROM Product p WHERE p.name=:name")
    public Optional<Product> findByName(String name);

    @Query("SELECT p FROM Product p JOIN p.category c WHERE c.id=:id")
    public List<Product> findByCategoryId(long id);

    @Query("SELECT od FROM OrderDetail od JOIN od.product p WHERE p.id=:id")
    public List<OrderDetail> findOrderDetailsByProductId(long id);

}
```

CustomerRepository

```
@Repository
public interface CustomerRepository extends JpaRepository<Customer, Long> {

    @Query("SELECT c FROM Customer c WHERE c.name=:name AND c.surname=:surname")
    public Optional<Customer> findByNameAndSurname(String name, String surname);

    @Query("SELECT o FROM Order o JOIN o.customer c WHERE c.id=:id")
    public List<Order> findOrdersByCustomerId(long id);
}
```

OrderRepository

```
@Repository
public interface OrderRepository extends JpaRepository<Order, Long> {

    @Query("SELECT o FROM Order o JOIN o.customer c WHERE c.id=:id")
    public List<Order> findByCustomerId(long id);

    @Query("SELECT o FROM Order o WHERE o.status=:status")
    public List<Order> findByStatus(OrderStatus status);

    @Query("SELECT od FROM OrderDetail od JOIN od.order o WHERE o.id=:id")
    public List<OrderDetail> findOrderDetailsByOrderId(long id);

}
```

OrderDetailRepository

```
@Repository
public interface OrderDetailRepository extends JpaRepository<OrderDetail, Long> {

    @Query("SELECT od FROM OrderDetail od JOIN od.order o WHERE o.id=:id")
    public List<OrderDetail> findByOrderId(long id);

    @Query("SELECT od FROM OrderDetail od JOIN od.product p WHERE p.id=:id")
    public List<OrderDetail> findByProductId(long id);
}
```

Data Transfer Objects (DTO)

En los DTO se utilizan las siguientes anotaciones Lombok:

```
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
```

Los DTO se clasifican en dos categorías:

- DTO de petición (*request*)
- DTO de respuesta (*response*)

DTO Request

```
public class CategoryRequest {

    private String name;
    private String description;

}
```

```
public class ProductRequest {

    private String name;
    private double price;
    private String description;
    private long categoryId;
    private int quantity;

}
```

```
public class CustomerRequest {

    private String name;
    private String surname;
    private String address;

}
```

```
public class OrderRequest {  
    private String date;  
    private String shipAddress;  
    private long customerId;  
    private OrderStatus status;  
}
```

```
public class OrderDetailRequest {  
    private long orderId;  
    private long productId;  
    private int quantity;  
}
```

DTO Response

En estos DTO tenemos un constructor que construye el objeto a partir del objeto del modelo.

```
public class CategoryResponse {  
    private long id;  
    private String name;  
    private String description;  
  
    public CategoryResponse(Category category) {  
        ...  
    }  
}
```

```
public class CustomerResponse {  
    private long id;  
    private String name;  
    private String surname;  
    private String address;  
  
    public CustomerResponse(Customer customer) {  
        ...  
    }  
}
```

En `ProductResponse` se incluye información de la categoría a la que el producto pertenece.

```
public class ProductResponse {  
  
    private long id;  
    private String name;  
    private double price;  
    private String description;  
    private long categoryId;  
    private String categoryName;  
    private int quantity;  
  
    public ProductResponse(Product product) {  
        ...  
    }  
}
```

En `OrderResponse` se incluye información del cliente al que el pedido pertenece.

```
public class OrderResponse {  
  
    private long id;  
    private String date;  
    private String shipAddress;  
    private OrderStatus status;  
  
    private long customerId;  
    private String customerName;  
    private String customerSurname;  
  
    public OrderResponse(Order order) {  
        ...  
    }  
}
```

En OrderDetailResponse se incluye información del pedido al que el detalle pertenece.

```
public class OrderDetailResponse {  
  
    private long id;  
    private int quantity;  
  
    private long productId;  
    private String productName;  
    private double productPrice;  
  
    private long orderId;  
    private String orderDate;  
    private String orderShipAddress;  
  
    private long customerId;  
    private String customerName;  
    private String customerSurname;  
  
    public OrderDetailResponse(OrderDetail orderDetail) {  
        ...  
    }  
  
}
```

Para la presentación en tablas no es necesaria toda la información de algunos de los anteriores, así que están disponibles versiones abreviadas:

```
public class ProductResponseBrief {  
  
    private long id;  
    private String name;  
    private double price;  
    private String description;  
    private int quantity;  
  
    public ProductResponseBrief(Product product) {  
        ...  
    }  
  
}
```

```
public class OrderDetailResponseBrief {  
    private long id;  
    private int quantity;  
  
    private long productId;  
    private String productName;  
    private double productPrice;  
  
    public OrderDetailResponseBrief(OrderDetail orderDetail) {  
        ...  
    }  
}
```

Spring Security

Hemos adaptado las clases proporcionadas por D. Joaquín Borrego para gestionar la seguridad de la aplicación mediante JWT. La configuración de la cadena de filtros se realiza en la clase `SecurityFilterChainConfiguration` que se presenta a continuación:

```
@Configuration
@EnableWebSecurity
@RequiredArgsConstructor
public class SecurityFilterChainConfiguration {

    @Autowired
    private final JwtAuthenticationFilter jwtAuthenticationFilter;

    @Autowired
    private final AuthenticationProvider authenticationProvider;

    @Bean
    public SecurityFilterChain securityFilterChain(HttpSecurity httpSecurityFilterChainBuilder)
        throws Exception {

        return httpSecurityFilterChainBuilder

            .csrf(csrfConfigurer -> csrfConfigurer.disable())

            .authorizeHttpRequests(registry ->
                registry
                    .requestMatchers("/auth/**", "/react/**").permitAll()
                    .anyRequest().authenticated()

            .authenticationProvider(authenticationProvider)

            .cors(Customizer.withDefaults())

            .sessionManagement(httpSecuritySessionManagementConfigurer ->
                httpSecuritySessionManagementConfigurer
                    .sessionCreationPolicy(SessionCreationPolicy.STATELESS))

            .addFilterBefore(jwtAuthenticationFilter,
                            UsernamePasswordAuthenticationFilter.class)

            .build();
    }
}
```

Tenemos dos tipos de rutas:

- De acceso público:
 - /auth/** : controladores encargados de la autorización.
 - /react/** : SPA cliente programado en React.
- De acceso restringido
 - Todas las demás.

API REST

La aplicación tiene los *endpoints* que se muestran en la siguiente tabla. Se han coloreado en verde aquellos que implican un *join* de tablas.

Mapping	Método	Acción
<code>/category</code>	GET	Obtiene todas las categorías
	POST	Crea una categoría
<code>/category/{id}</code>	GET	Obtiene la categoría por <code>id</code>
	PUT	Actualiza la categoría por <code>id</code>
	DELETE	Elimina la categoría por <code>id</code>

Mapping	Método	Acción
<code>/customer</code>	GET	Obtiene todos los clientes
	POST	Da de alta a un cliente
<code>/customer/{id}</code>	GET	Obtiene el cliente por <code>id</code>
	PUT	Actualiza el cliente por <code>id</code>
	DELETE	Elimina el cliente por <code>id</code>
<code>/customer/{name}/{surname}</code>	GET	Obtiene el cliente por <code>name</code> y <code>surname</code>

Mapping	Método	Acción
<code>/product</code>	GET	Obtiene todos los productos
	POST	Da de alta a un producto
<code>/product/{id}</code>	GET	Obtiene el producto por <code>id</code>
	PUT	Actualiza el producto por <code>id</code>
	DELETE	Elimina el producto por <code>id</code>
<code>/product/bycategory/{id}</code>	GET	Obtiene los productos de una categoría dada por <code>id</code>

Mapping	Método	Acción
<code>/order</code>	GET	Obtiene todos los pedidos
	POST	Crea un pedido
<code>/order/{id}</code>	GET	Obtiene el pedido por <code>id</code>
	PUT	Actualiza los datos y el estado del pedido por <code>id</code>
	DELETE	Elimina el pedido por <code>id</code>
<code>/order/bystatus/{status}</code>	GET	Obtiene los pedidos según por <code>status</code>
<code>/order/bycustomer/{id}</code>		Obtiene los pedidos de un cliente por su <code>id</code>

Mapping	Método	Acción
<code>/orderdetail</code>	GET	Obtiene todos los productos
	POST	Da de alta a un producto
<code>/orderdetail/{id}</code>	GET	Obtiene el producto por <code>id</code>
	PUT	Actualiza el producto por <code>id</code>
	DELETE	Elimina el producto por <code>id</code>
<code>/orderdetail/byorder/{id}</code>	GET	Obtiene los productos de un pedido por su <code>id</code>

Controladores

A continuación se muestran los controladores REST.

CategoryController

```
@CrossOrigin
@RestController
@RequestMapping("/category")
public class CategoryController {

    private static final Logger LOGGER =
        LoggerFactory.getLogger(CategoryController.class);

    @Autowired
    CategoryService categoryService;

    @ExceptionHandler(CategoryNotFoundException.class)
    @ResponseStatus(HttpStatus.NOT_FOUND)
    public String handleCategoryNotFoundException(Exception e) {
        LOGGER.info("Categoría no encontrada: {}", e.getMessage());
        return "Category no encontrado: " + e;
    }

    @GetMapping
    @ResponseStatus(HttpStatus.OK)
    public List<CategoryResponse> getAll() {
        LOGGER.info("Solicitadas todas las categorías");
        return categoryService.getAll();
    }

    @GetMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public CategoryResponse getById(@PathVariable long id) {
        LOGGER.info("Solicitada la categoría con id={}", id);
        return categoryService.getById(id);
    }

    @PostMapping
    @ResponseStatus(HttpStatus.CREATED)
    public CategoryResponse createCategory(@RequestBody CategoryRequest categoryRequest) {
        LOGGER.info("Solicitada la creación de una categoría: {}",
                   categoryRequest.getName());
        return categoryService.create(categoryRequest);
    }

    @PutMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public CategoryResponse updateCategory(@PathVariable long id,
                                           @RequestBody CategoryRequest categoryRequest) {
        LOGGER.info("Solicitada la actualización de la categoría con id={}",
                   categoryRequest.getName());
        return categoryService.update(id, categoryRequest);
    }

    @DeleteMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public void deleteCategory(@PathVariable long id) {
        LOGGER.info("Solicitada la eliminación de la categoría con id={}", id);
        categoryService.delete(id);
    }
}
```

ProductController

```
@CrossOrigin
@RestController
@RequestMapping("/product")
public class ProductController {

    private static final Logger LOGGER = LoggerFactory.getLogger(ProductController.class);

    @Autowired
    ProductService productService;

    @ExceptionHandler(ProductNotFoundException.class)
    @ResponseStatus(HttpStatus.NOT_FOUND)
    public String handleProductNotFoundException(Exception e) {
        LOGGER.info("Producto no encontrado: {}", e.getMessage());
        return "Producto no encontrado: " + e;
    }

    @GetMapping
    @ResponseStatus(HttpStatus.OK)
    public List<ProductResponse> getAll() {
        LOGGER.info("Solicitados todos los productos");
        return productService.getAll();
    }

    @GetMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public List<ProductResponseBrief> getByCategoryId(@PathVariable long id) {
        LOGGER.info("Solicitado los productos de la categoría id={}", id);
        return productService.getByCategoryId(id);
    }

    @GetMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public ProductResponse getById(@PathVariable long id) {
        LOGGER.info("Solicitado el producto con id={}", id);
        return productService.getById(id);
    }

    @PostMapping
    @ResponseStatus(HttpStatus.CREATED)
    public ProductResponse createProduct(@RequestBody ProductRequest productRequest) {
        LOGGER.info("Solicitada la creación de un producto de la categoría con id={}",
                    productRequest.getCategoryId());
        return productService.create(productRequest);
    }

    @PutMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public ProductResponse updateProduct(@PathVariable long id,
                                         @RequestBody ProductRequest productRequest) {
        LOGGER.info("Solicitada la actualización del producto con id={}", id);
        return productService.update(id, productRequest);
    }

    @DeleteMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public void deleteProduct(@PathVariable long id) {
        LOGGER.info("Solicitada la eliminación del producto con id={}", id);
        productService.delete(id);
    }
}
```

CustomerController

```
@CrossOrigin
@RestController
@RequestMapping("/customer")
public class CustomerController {

    private static final Logger LOGGER = LoggerFactory.getLogger(CustomerController.class);

    @Autowired
    CustomerService customerService;

    @ExceptionHandler(CustomerNotFoundException.class)
    @ResponseStatus(HttpStatus.NOT_FOUND)
    public String handleCustomerNotFoundException(Exception e) {
        LOGGER.info("Cliente no encontrado: {}", e.getMessage());
        return "Cliente no encontrado: " + e;
    }

    @GetMapping
    @ResponseStatus(HttpStatus.OK)
    public List<CustomerResponse> getAll() {
        LOGGER.info("Solicitados todos los clientes");
        return customerService.getAll();
    }

    @GetMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public CustomerResponse getById(@PathVariable long id) {
        LOGGER.info("Solicitado el cliente con id={}", id);
        return customerService.getById(id);
    }

    @GetMapping("/{name}/{surname}")
    @ResponseStatus(HttpStatus.OK)
    public CustomerResponse getCustomerByNameAndSurname(@PathVariable String name,
                                                       @PathVariable String surname) {
        LOGGER.info("Solicitado el cliente {}-{}", name, surname);
        return customerService.getByNameAndSurname(name, surname);
    }

    @PostMapping
    @ResponseStatus(HttpStatus.CREATED)
    public CustomerResponse createCustomer(@RequestBody CustomerRequest customerRequest) {
        LOGGER.info("Solicitada la creación del cliente {}-{}", customerRequest.getName(),
                    customerRequest.getSurname());
        return customerService.create(customerRequest);
    }

    @PutMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public CustomerResponse updateCustomer(@PathVariable long id,
                                           @RequestBody CustomerRequest customerRequest) {
        LOGGER.info("Solicitada la actualización del cliente con id={}, id");
        return customerService.update(id, customerRequest);
    }

    @DeleteMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public void deleteCustomer(@PathVariable long id) {
        LOGGER.info("Solicitada la eliminación del cliente con id={}", id);
        customerService.delete(id);
    }
}
```

OrderController

```
@CrossOrigin
@RestController
@RequestMapping("/order")
public class OrderController {

    private static final Logger LOGGER = LoggerFactory.getLogger(OrderController.class);

    @Autowired
    OrderService orderService;

    @ExceptionHandler(OrderNotFoundException.class)
    @ResponseStatus(HttpStatus.BAD_REQUEST)
    public String handleOrderNotFoundException(Exception e) {
        LOGGER.info("Pedido no encontrado: {}", e.getMessage());
        return "Pedido no encontrado: " + e;
    }

    @GetMapping
    @ResponseStatus(HttpStatus.OK)
    public List<OrderResponseBrief> getAll() {
        LOGGER.info("Solicitados todos los pedidos");
        return orderService.getAll();
    }

    @GetMapping("/bycustomer/{id}")
    @ResponseStatus(HttpStatus.OK)
    public List<OrderResponseBrief> getByCustomerId(@PathVariable long id) {
        LOGGER.info("Solicitado el pedido del cliente con id={}", id);
        return orderService.getByCustomerId(id);
    }

    @GetMapping("/bystatus/{status}")
    @ResponseStatus(HttpStatus.OK)
    public List<OrderResponseBrief> getByStatus(@PathVariable OrderStatus status) {
        LOGGER.info("Solicitados los pedidos con estado: {}", status);
        return orderService.getByStatus(status);
    }

    @GetMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public OrderResponse getById(@PathVariable long id) {
        LOGGER.info("Solicitado el pedido con id={}", id);
        return orderService.getById(id);
    }

    @PostMapping
    @ResponseStatus(HttpStatus.CREATED)
    public OrderResponse create(@RequestBody OrderRequest orderRequest) {
        LOGGER.info("Solicitada la creación de un pedido del cliente con id={},",
                    orderRequest.getCustomerId());
        return orderService.create(orderRequest);
    }

    @PutMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public OrderResponse update(@PathVariable long id,
                               @RequestBody OrderRequest orderRequest) {
        LOGGER.info("Solicitada la actualización del pedido con id={}, id");
        return orderService.update(id, orderRequest);
    }

    @DeleteMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public void delete(@PathVariable long id) {
        LOGGER.info("Solicitada la eliminación del pedido con id={}", id);
        orderService.delete(id);
    }
}
```

OrderDetailController

```
@CrossOrigin
@RestController
@RequestMapping("/orderdetail")
public class OrderDetailController {

    private static final Logger LOGGER =
        LoggerFactory.getLogger(OrderDetailController.class);

    @Autowired
    OrderDetailsService orderDetailService;

    @ExceptionHandler(ProductNotFoundException.class)
    @ResponseStatus(HttpStatus.NOT_FOUND)
    public String handleOrderDetailNotFoundException(Exception e) {
        LOGGER.info("Detalle de pedido no encontrado: {}", e.getMessage());
        return "Detalle de pedido no encontrado: " + e;
    }

    @GetMapping
    @ResponseStatus(HttpStatus.OK)
    public List<OrderDetailResponseBrief> getAll() {
        LOGGER.info("Solicitados todos los detalles de pedido");
        return orderDetailService.getAll();
    }

    @GetMapping("/byorder/{id}")
    @ResponseStatus(HttpStatus.OK)
    public List<OrderDetailResponseBrief> getByOrderId(@PathVariable long id) {
        LOGGER.info("Solicitado los detalles del pedido con id={}", id);
        return orderDetailService.getByOrderId(id);
    }

    @GetMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public OrderDetailResponse getById(@PathVariable long id) {
        LOGGER.info("Solicitado el detalle de pedido con id={}", id);
        return orderDetailService.getById(id);
    }

    @PostMapping
    @ResponseStatus(HttpStatus.CREATED)
    public OrderDetailResponse create(@RequestBody OrderDetailRequest orderDetailRequest)
    {
        LOGGER.info("Solicitada la creación de un detalle del pedido con id={}",
                    orderDetailRequest.getOrderId());
        return orderDetailService.create(orderDetailRequest);
    }

    @PutMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public OrderDetailResponse updateOrderDetail(@PathVariable long id,
                                                @RequestBody OrderDetailRequest orderDetailRequest) {
        LOGGER.info("Solicitada la actualización del detalle de pedido con id={}", id);
        return orderDetailService.update(id, orderDetailRequest);
    }

    @DeleteMapping("/{id}")
    @ResponseStatus(HttpStatus.OK)
    public void deleteOrderDetail(@PathVariable long id) {
        LOGGER.info("Solicitada la eliminación del detalle de pedido con id={}", id);
        orderDetailService.delete(id);
    }
}
```

Control de las excepciones de los controladores

Cada controlador gestiona excepciones específicas. Para controlar las excepciones genéricas de la aplicación utilizamos una clase con la anotación `@RestControllerAdvice`.

`ExceptionHandlerController`

```
@RestControllerAdvice
public class ExceptionHandlerController {

    private static final Logger LOGGER =
        LoggerFactory.getLogger(ExceptionHandlerController.class);

    @ExceptionHandler(HttpMessageNotReadableException.class)
    @ResponseStatus(HttpStatus.BAD_REQUEST)
    public String handleHttpMessageNotReadableException(Exception e) {
        LOGGER.info("JSON Mal formado: {}",e.getMessage());
        return "JSON Mal formado: " + e;
    }

    @ExceptionHandler(MethodArgumentTypeMismatchException.class)
    @ResponseStatus(HttpStatus.BAD_REQUEST)
    public String handleMethodArgumentTypeMismatchException(Exception e) {
        LOGGER.info("Tipo del argumento del método erróneo: {}",e.getMessage());

        return "Tipo del argumento del método erróneo: " + e;
    }

    @ExceptionHandler(Exception.class)
    @ResponseStatus(HttpStatus.INTERNAL_SERVER_ERROR)
    public String handleException(Exception e) {
        LOGGER.info("Genérico: {}",e.getMessage());
        return "Genérico: " + e;
    }

}
```

SPA con React

Comunicación con el servidor

Las funciones que gestionan la comunicación con el API REST se encuentran en el módulo `Network.js`. Para solicitar autorización se utiliza su función `getAuth`:

```
async function getAuth(credentials) {
  const fetchConfig = {
    method: "POST",
    headers: { "Content-Type": "application/json", "Sec-Fetch-Site": "cross-site" },
    mode: "cors",
    body: JSON.stringify(credentials)
  };
  const response = await fetch(serverUrl + "/auth/login", fetchConfig);
  if (!response.ok) {
    throw new Error("response:" + response.status + " " + response.statusText);
  }
  const token = await response.json();
  tokenStore.setToken(token.token);
}
```

A continuación podemos ver algunas de las funciones que contiene el módulo.

```
async function getOrdersByStatus(status) {
  const token = tokenStore.getToken();
  const fetchConfig = {
    method: "GET",
    headers: {
      "Content-Type": "application/json",
      "Authorization": "Bearer " + token
    }
  };
  const response = await fetch(serverUrl + "/order/bystatus/" + status, fetchConfig);
  if (!response.ok) throw new Error("El token ha expirado, por favor cierre sesión");
  const data = await response.json();
  return data;
}
```

```

async function postOrder(order) {
  const token = tokenStore.getToken();
  const fetchConfig = {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      "Authorization": "Bearer " + token
    },
    body: JSON.stringify(order)
  };
  const response = await fetch(serverUrl + "/order", fetchConfig);
  if (!response.ok) throw new Error("El token ha expirado, por favor cierre sesión");
  const data = await response.json();
  return data;
}

```

Estructura de los componentes React del SPA cliente

A continuación se muestra el componente `ProductPage` como ejemplo de la estructura general que presentan los componentes de la aplicación.

```

import { useState, useEffect } from 'react';
import * as network from '../Network.js';
import * as stack from "../Stack.js";
import * as tokenStore from "../TokenStore.js";

function ProductPage(props) {

  const productId = props.productId;

  const [product, setProduct] = useState(null);
  const [error, setError] = useState(null);

  /* Asynchronous data loading */

  async function asyncAction() {
    try {
      const product = await network.getProduct(productId);
      setProduct(product);
    }
    catch (err) {
      setError(err);
    }
  }

  function setupEffect() {
    asyncAction();
  }

  useEffect(setupEffect, []);
}

```

```

/* GUI Event handlers */

function onClickEditProduct(ev) {
    const data = {
        id: product.id,
        categoryId: product.categoryId,
        name: product.name,
        description: product.description,
        price: product.price,
        quantity: product.quantity
    }
    stack.go({ name: "EditProductPage", data: data });
}

async function onClickDeleteProduct(ev) {
    const data = {
        id: product.id,
        name: product.name,
        description: product.description,
        price: product.price,
        quantity: product.quantity
    }
    stack.go({ name: "DeleteProductPage", data: data });
}

function onClickGoBack(ev) {
    stack.goBack();
}

```

```

/* GUI Event handlers */

function onClickEditProduct(ev) {
    const data = {
        id: product.id,
        categoryId: product.categoryId,
        name: product.name,
        description: product.description,
        price: product.price,
        quantity: product.quantity
    }
    stack.go({ name: "EditProductPage", data: data });
}

async function onClickDeleteProduct(ev) {
    const data = {
        id: product.id,
        name: product.name,
        description: product.description,
        price: product.price,
        quantity: product.quantity
    }
    stack.go({ name: "DeleteProductPage", data: data });
}

function onClickGoBack(ev) {
    stack.goBack();
}

```

Instrucciones para la compilación y empaquetado de la aplicación

Desde el directorio base del proyecto

1. Construir el SPA React:

```
cd client  
npm run build -- --base=/react  
mv dist react
```

2. Colocar el SPA React en el directorio estático del servidor Spring Boot:

```
mv react ../server/src/main/resources/static/
```

3. Creación del JAR del servidor:

```
cd ../server  
mvn clean package
```

Con esto se crea el archivo `cfgs-dam-tf.jar` en el subdirectorio `target`.

```
target/cfgs-dam-tf.jar
```

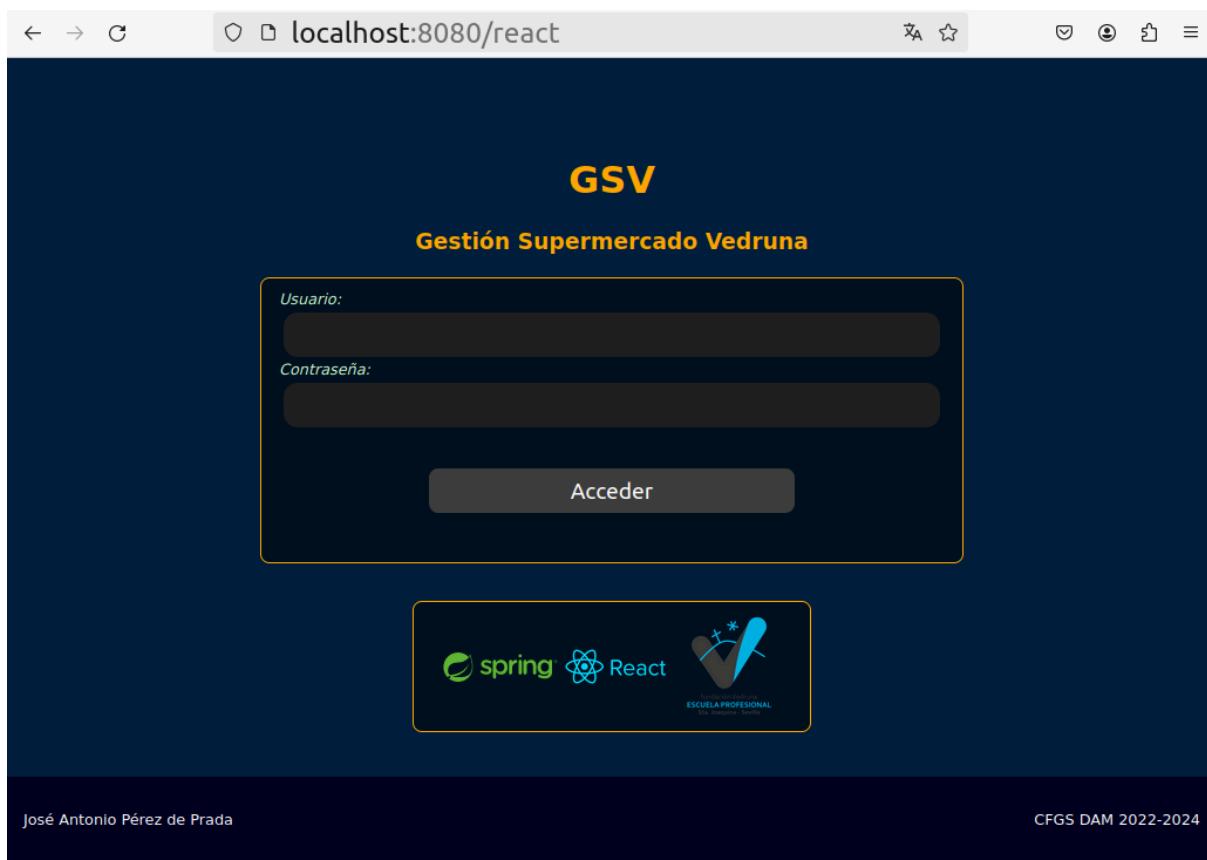
Instrucciones para la puesta en marcha de la aplicación

Si no movemos el archivo JAR, localmente podemos ejecutar la aplicación mediante

```
java -jar target/cfgs-dam-tf.jar
```

Para visualizar la aplicación debemos abrir la siguiente dirección mediante un navegador:

```
http://localhost:8080/react/index.html
```



El usuario administrador es `jose` y la contraseña inicial es `321`.

Omisión de `index.html` en la ruta

En principio, con la configuración REST que hemos realizado, para acceder al SPA React es necesario indicar la ruta

```
http://localhost:8080/react/index.html
```

Para que no sea necesario tener que escribir la parte final podemos introducir la siguiente clase que implementa la interfaz `WebMvcConfigurer`:

```
@Configuration
public class ForwardToIndexConfig implements WebMvcConfigurer {

    @Override
    public void addViewControllers(ViewControllerRegistry registry) {
        String viewName = "forward:/react/index.html";
        registry.addViewController("/react").setViewName(viewName);
        registry.addViewController("/react/").setViewName(viewName);
    }

}
```