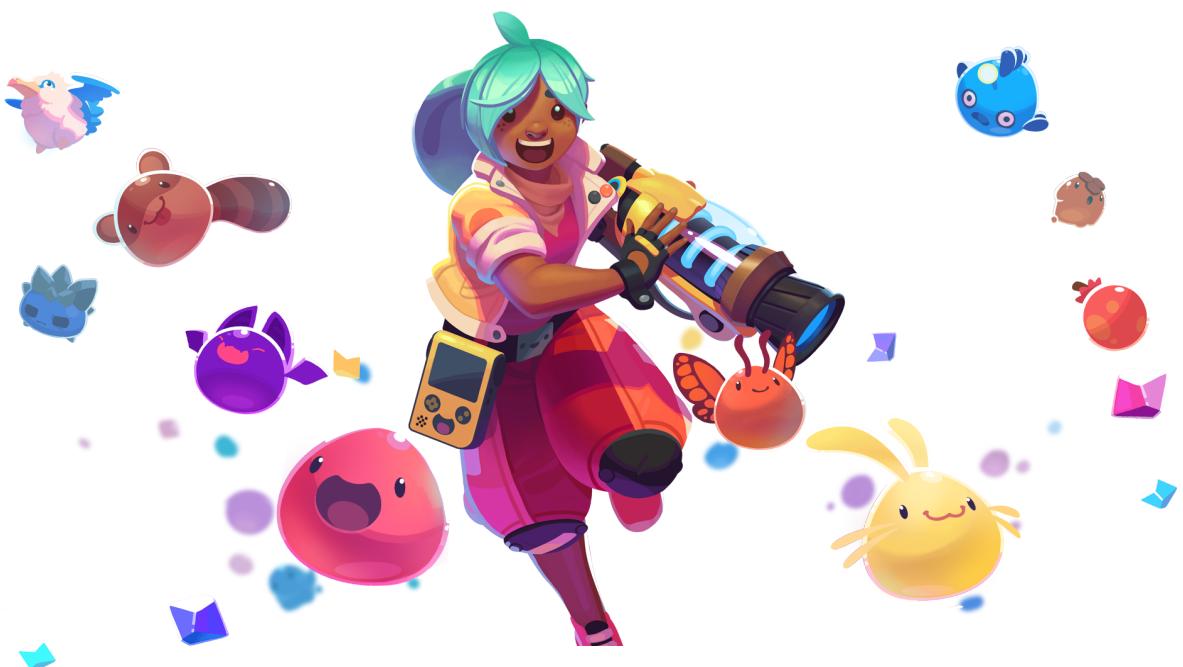


Slime Shop

Isaac Julián Pavón Ruiz
Francisco Jose Avendaño Coloma



Índice

PRESENTACIÓN	1
BASES DE DATOS	1
MODELO ENTIDAD/RELACIÓN	2
MODELO RELACIONAL	3
Pérdida semánticas	4
DDL	4
DML	4
Programación	4
Clases DAO	4
DIAGRAMA DE CLASES DAO	5
Clases DTO	6
DIAGRAMA DE CLASES DTO	7
Carrito	8
PANTALLA DEL CARRITO CON ARTÍCULOS INCLUIDOS	8
Facturación	9
PANTALLA DE PEDIDOS REALIZADOS LISTOS PARA FACTURAR	9
PANTALLA DE PEDIDOS FACTURADOS LISTOS PARA EMITIR FACTURA EN FORMATOS XML Y/O PDF	10
FACTURA EMITIDA EN FORMATO XML	11
FACTURA EMITIDA EN FORMATO PDF	12
Sesiones	13
PANTALLA DE INICIO DE ACCESO GENERAL	13
PANTALLA DE ACCESO GENERAL A PRODUCTOS	13
PANTALLA DE ACCESO GENERAL A CATEGORÍAS	14
PANTALLA DE ACCESO A SOBRE NOSOTROS	14
PANTALLA DE ACCESO A CONTACTO	15
PANTALLA DE CONTACTO CON APERTURA DE GESTOR DE CORREO	15
Atributos y otros datos en servlets	16
DIAGRAMA DE CLASES SERVLETS	16
DIAGRAMA DE CLASES UTILS	16
Frontend	17
Mockup	17
Primer desarrollo	17
Segundo desarrollo	17
Estructuración del proyecto	18
Backend	18
Frontend	18
CONCLUSIONES Y DIFICULTADES ENCONTRADAS	19
POSIBLES MEJORAS O AMPLIACIONES	20

PRESENTACIÓN

El proyecto consta de una tienda web, basada en el juego [slimerancher](#). Para ello se han utilizado las siguientes tecnologías:

1. [Base de datos](#)

Para la base de datos hemos utilizado Oracle DB dentro de un contenedor de Docker, ahorrándonos la implementación de un servidor dedicado.

2. [Para el backend](#)

El proyecto consta de server side rendering constituido en Java EE, utilizando servlets y las plantillas JSP.

3. Para el frontend

Junto a las plantillas JSP hemos utilizado Webcomponents, para poder crear componentes aislados y reutilizables a lo largo del proyecto.

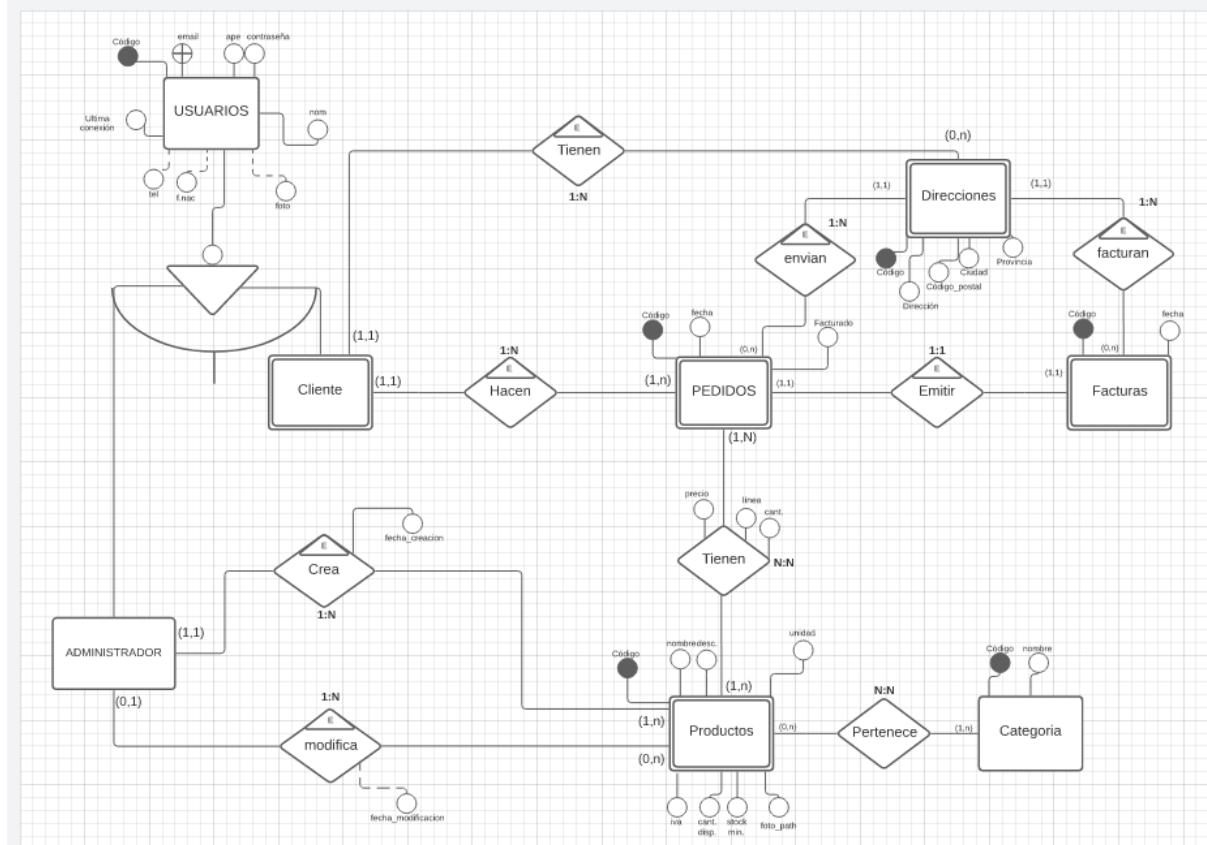
BASES DE DATOS

Para la constitución de la base de datos hemos usado los siguientes pasos:

1. [Modelo Entidad relación](#)
2. [Modelo Relacional](#)
3. [DDL \(Data Definition Language\)](#)
4. [DML \(Data Manipulation Language\)](#)

MODELO ENTIDAD/RELACIÓN

Para el modelo entidad relación hemos utilizado la aplicación web [LucidChart](#) y dejamos un esquema estático de la base de datos junto al [enlace al proyecto](#).



MODELO RELACIONAL

TIENDA_DIRECCION (codigo, direccion, codigo_postal, ciudad, provincia)

- PK: codigo
- FK: cliente -> CLIENTE.codigo
- NN (direccion, codigo_postal, ciudad, provincia)

TIENDA_USUARIO (codigo, email, nombre, apellido, contrasenya, ultimaConexion, tel, fecha_nacimiento, foto, direccion, tipo)

- PK: codigo
- UK: Email
- FK: direccion -> DIRECCIONES.codigo
- NN (nombre, apellidos, contra, ultima-con, direccion, tipo)

TIENDA_PEDIDO (codigo, codigo_usuario, fecha, estado)

- PK: codigo
- FK: codigo_usuario -> USUARIOS.codigo
- NN (codigo_usuario, fecha, estado)

TIENDA_PEDIDO (codigo, codigo_categoria, nombre, descripcion, precio, stock, imagen)

- PK: codigo
- FK: codigo_categoria -> CATEGORIAS.codigo
- NN (nombre, descripcion, precio, stock)

TIENDA_CATEGORIA (codigo, nombre)

- PK: codigo
- NN (nombre)

Pedido_Producto (codigo_pedido, codigo_producto, cantidad)

- PK: codigo_pedido, codigo_producto
- FK: codigo_pedido -> PEDIDOS.codigo
- FK: codigo_producto -> PRODUCTOS.codigo
- NN (codigo_pedido, codigo_producto, cantidad)
-

TIENDA_FACTURA (codigo, codigo_pedido, fecha)

- PK: codigo
- FK: codigo_pedido -> PEDIDOS.codigo
- NN (codigo_pedido, fecha)

Pérdida semánticas

1. Se pierden el ***1*** de categoría y pedidos.
2. Se pierden los ***1*** de administrador y productos.
3. Se pierden los ***1*** de cliente y dirección.

DDL

Para el DDL hemos tenido en cuenta algunas restricciones como la del tipo de usuario o los valores DEFAULT como los de las fotos, en el caso de que no se introduzca nada

Se puede ver una versión del mismo en el [Github del proyecto](#)

DML

Hemos introducido por defecto todos los Plorts del juego, así como su propia categoría, siendo este nuestro producto estrella, también todo tipo de alimentos.

Se puede ver una versión del mismo en el [Github del proyecto](#)

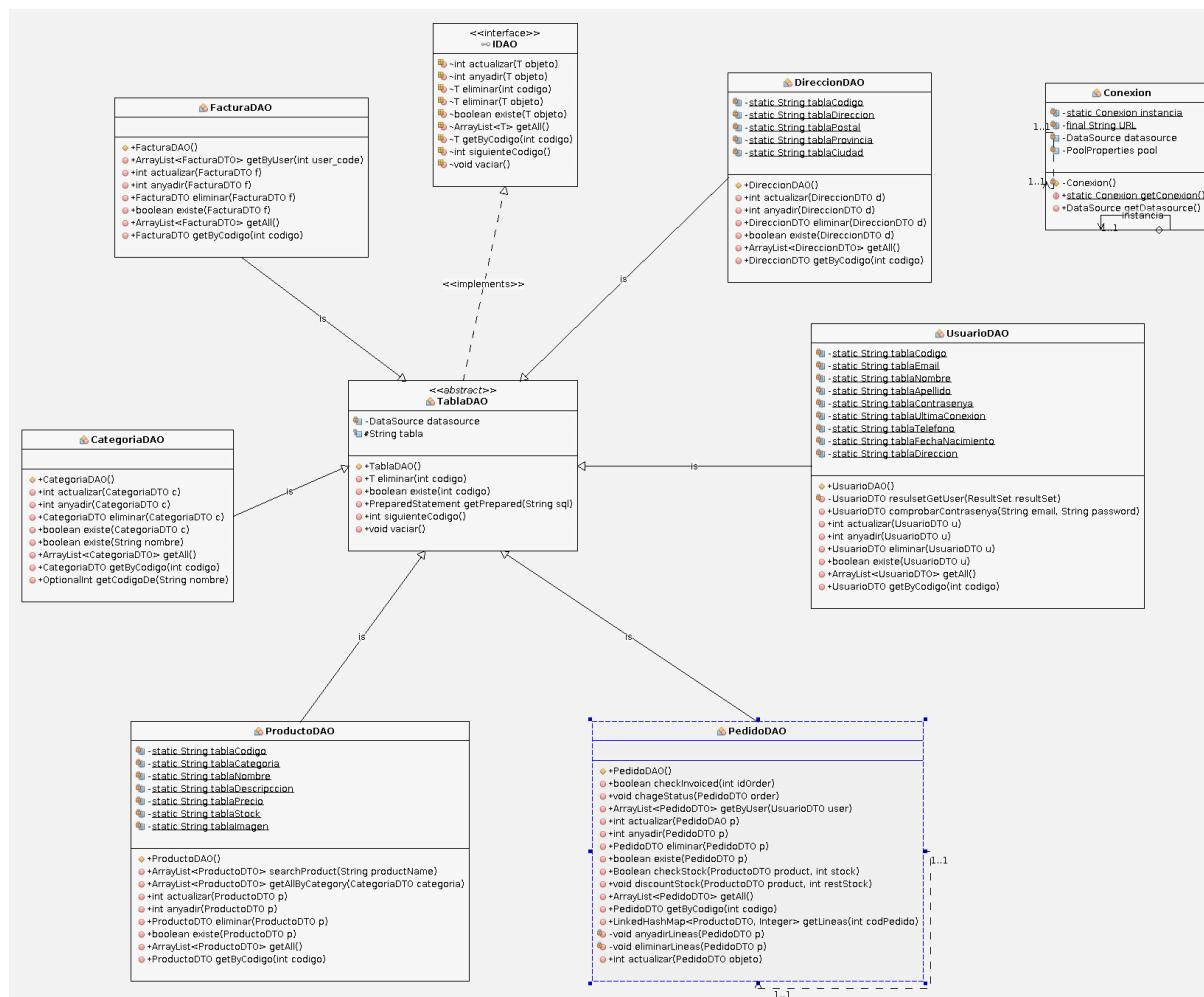
Programación

Para el apartado de backend del proyecto hemos intentado controlar toda las peticiones y tratamientos de datos a través de los servlets, para así usar solo las templates de JSP para mostrar datos o reutilizar vistas y archivos estáticos del FrontEnd.

Clases DAO

Para controlar las llamadas a la base de datos, hemos utilizado clases con ese mismo fin llamadas DAO. Esos datos obtenidos van a las clases dedicadas llamadas DAO, donde almacenamos la información para su posterior tratamiento.

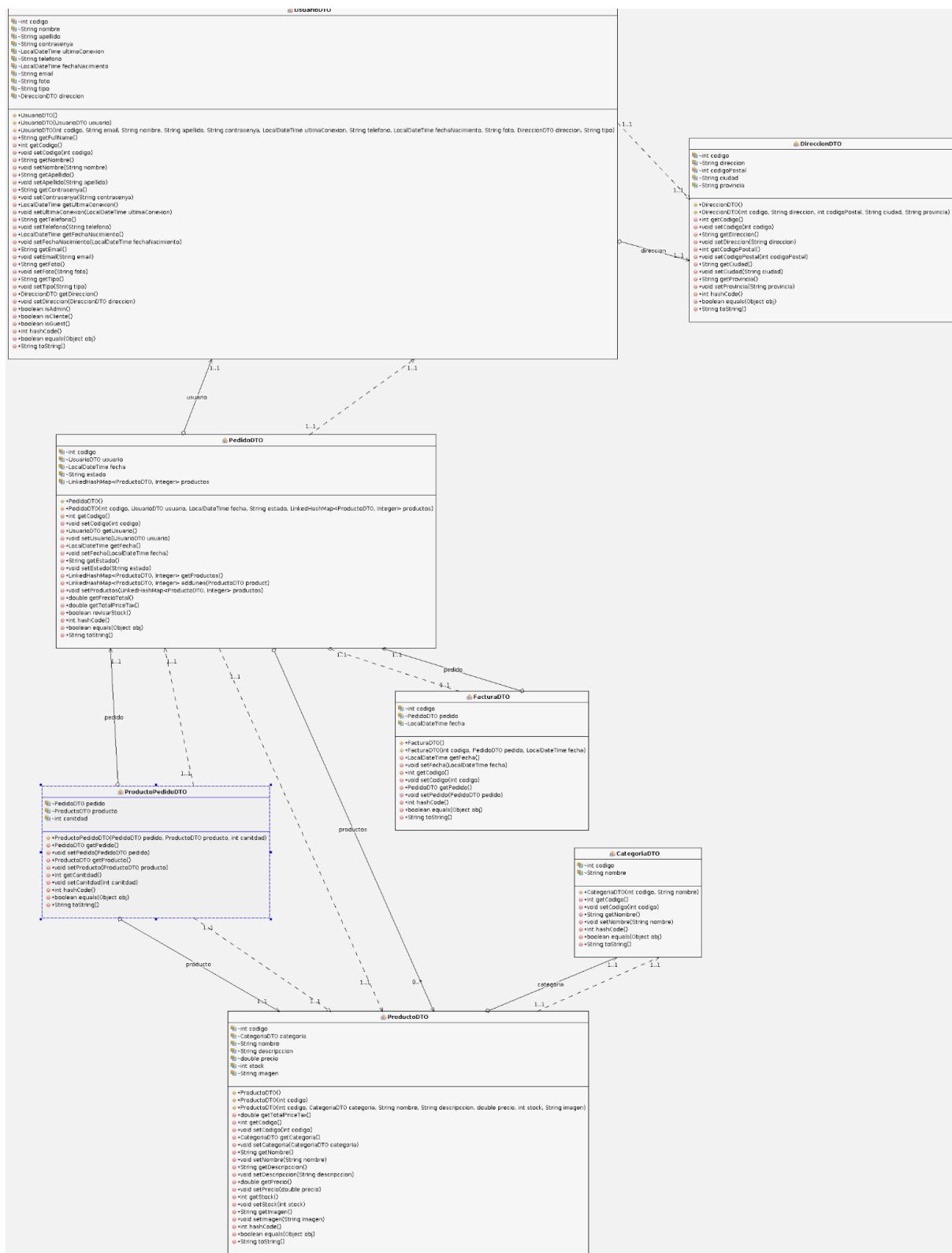
DIAGRAMA DE CLASES DAO



Clases DTO

Siendo las más importantes del proyecto, las clases de UsuarioDTO usado a lo largo del proyecto a través de sesiones para validar y controlar los accesos a la web, así como las diferentes compras o acciones en la base de datos derivadas. La clase PedidoDTO, que hemos utilizado tanto para el pedido como para el carrito.

DIAGRAMA DE CLASES DTO



Carrito

El carrito del proyecto tiene una mención especial ya que hemos obviado el almacenamiento de este en la base de datos. Nuestro carrito de compras solo es válido durante la sesión del usuario, que se convierte en un pedido en firme a la hora de confirmación del mismo. Para la confirmación del carrito se siguen una serie de pasos:

1. Se comprueba de que hay stock suficiente para poder crear el pedido
2. Si hay stock suficiente pasamos a la creación del pedido, descontando las unidades pedidas de la base de datos.
3. En el caso de lo contrario se devuelve un mensaje de error al usuario en la vista del carrito de la compra.

PANTALLA DEL CARRITO CON ARTÍCULOS INCLUIDOS



Facturación

Una vez realizado el pedido este se podrá facturar, y generar los pdf y xml pertinentes.

PANTALLA DE PEDIDOS REALIZADOS LISTOS PARA FACTURAR

The screenshot displays a mobile application interface for managing orders. At the top, there is a navigation bar with tabs: Principal, Productos, Categorías, Mis Compras, Carrito, and Salir. Below the navigation bar, there is a list of four orders, each represented by a white rounded rectangle with a shadow. Each order card contains the following information:

- Pedido n°1**: Total: 35.0 ⚡ Estado: EN PROCESO Pedido el: 30/05/2023 1u.
- Pedido n°2**: Total: 160.0 ⚡ Estado: EN PROCESO Pedido el: 30/05/2023 1u.
- Pedido n°3**: Total: 14.0 ⚡ Estado: ENVIADO Pedido el: 30/05/2023 1u.
- Pedido n°4**: Total: 47.0 ⚡ Estado: EN PROCESO Pedido el: 04/06/2023 3u.

Each order card features a pink rounded rectangle button labeled "Facturar". At the bottom of the screen, there is a dark blue footer bar with white text links: Principal, Productos, Categorías, Sobre Nosotros, Contacta, and a logo consisting of three overlapping circles.

PANTALLA DE PEDIDOS FACTURADOS LISTOS PARA EMITIR FACTURA EN FORMATOS XML Y/O PDF

The screenshot displays a mobile application interface for managing invoices. At the top, there is a navigation bar with tabs: Principal, Productos, Categorías, Mis Compras, and Carrito. On the far right of the bar is a red button labeled "Salir". Below the navigation bar, there is a list of five invoices, each represented by a white rounded rectangle. Each invoice entry includes the invoice number, the date it was factured, and two buttons for generating the document in XML or PDF format.

Factura n°	Facturado el:	XML	PDF
Factura n°1	2023-05-30T17:27:46	XML	PDF
Factura n°3	2023-04-16T00:00	XML	PDF
Factura n°4	2023-04-15T00:00	XML	PDF
Factura n°5	2023-06-04T13:44:32	XML	PDF

At the bottom of the screen, there is a dark blue footer bar containing links: Principal, Productos, Categorías, Sobre Nosotros, and Contacta.

FACTURA EMITIDA EN FORMATO XML

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE factura [
    <!-- Factura -->
    <!-- Emisor -->
    <!-- Destinatario -->
    <!-- Lineas -->
]>
<factura codigo_factura="5">
    <fecha>
        <dia>4</dia>
        <mes>6</mes>
        <año>2023</año>
        <hora>13</hora>
        <minuto>44</minuto>
        <segundo>32</segundo>
    </fecha>
    <direccion_facturacion>
        <direccion>Calle de los plorts 123</direccion>
        <poblacion>Rancho Slimy</poblacion>
        <provincia>Estados Unidos</provincia>
    </direccion_facturacion>
    <pedido_asociado codigo_pedido="4">
        <cliente>Beatrix LeBeau</cliente>
        <direccion_envio>
            <direccion>Calle de los plorts 123</direccion>
            <poblacion>Rancho Slimy</poblacion>
            <provincia>Estados Unidos</provincia>
        </direccion_envio>
        <lineas>
            <producto codigo_producto="1">
                <nombre>Plort rosa</nombre>
                <precio_unitario>7.0</precio_unitario>
                <iva> 15% </iva>
                <cantidad_productos>1</cantidad_productos>
                <total_linea>7.0</total_linea>
                <total_linea_con_iva>8.05</total_linea_con_iva>
            </producto>
            <producto codigo_producto="2">
                <nombre>Plort Roca</nombre>
                <precio_unitario>16.0</precio_unitario>
                <iva> 15% </iva>
                <cantidad_productos>1</cantidad_productos>
                <total_linea>16.0</total_linea>
                <total_linea_con_iva>18.4</total_linea_con_iva>
            </producto>
            <producto codigo_producto="3">
                <nombre>Plort Atigrado</nombre>
                <precio_unitario>24.0</precio_unitario>
                <iva> 15% </iva>
                <cantidad_productos>1</cantidad_productos>
                <total_linea>24.0</total_linea>
                <total_linea_con_iva>27.6</total_linea_con_iva>
            </producto>
        </lineas>
    </pedido_asociado>
    <base_imponible>47.0</base_imponible>
    <total>70.5</total>
</factura>
```

FACTURA EMITIDA EN FORMATO PDF

Factura nº 5

Datos del cliente:

Nº cliente: 1

Nombre: Beatrix LeBeau

E-mail: beatrix@rancher.com

Teléfono: 123456789

Datos de la factura:

Fecha: 04-06-2023 13:44:32

Dirección de facturación: Beatrix LeBeau - Calle de los plorts 123 (Estados Unidos)

Dirección de envío: Calle de los plorts 123 - Rancho Slimy (Estados Unidos)

Líneas del pedido:

CÓDIGO	NOMBRE	PRECIO	I.V.A.	CANTIDAD	IMPORTE
1	Plort rosa	7,00 €	15%	1	9,26 €
2	Plort Roca	16,00 €	15%	1	21,16 €
3	Plort Atigrado	24,00 €	15%	1	31,74 €

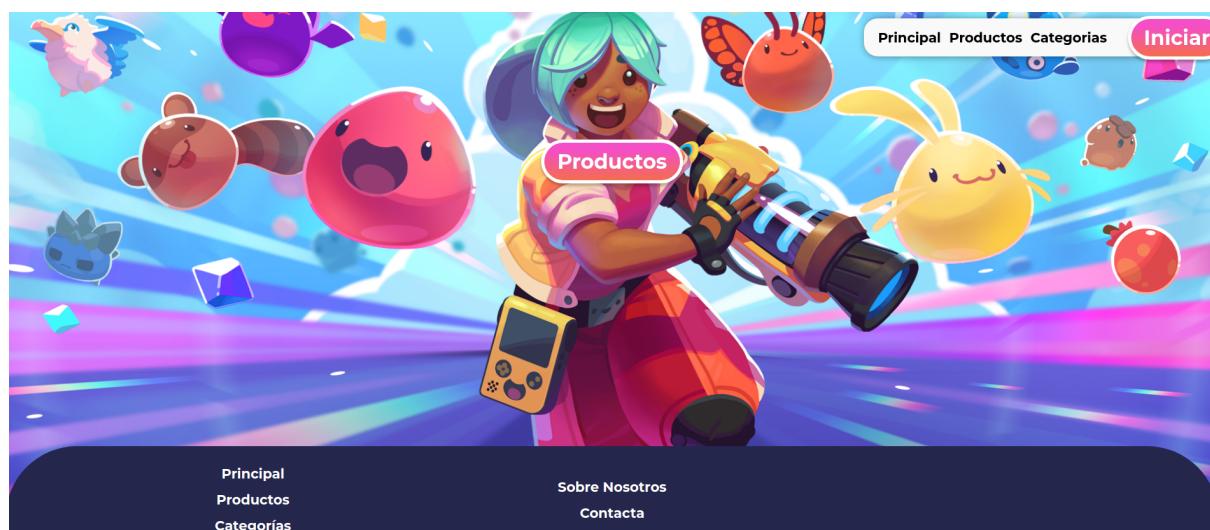
BASE IMPONIBLE: 47,00 €

TOTAL FACTURA: 70,50 €

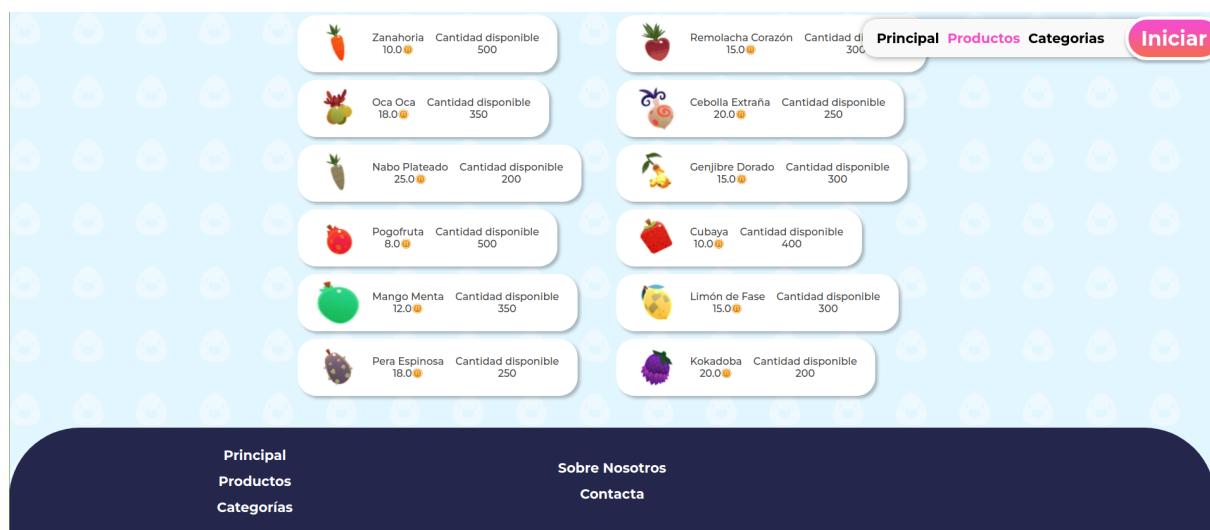
Sesiones

Para los datos de la sesión, siempre seguimos el mismo procedimiento: comprobamos que el dato de la sesión no sea nulo, y en el caso de ser cierto creamos una clase DTO, correspondiente vacía, usando una forma adaptada del concepto Java Beans.

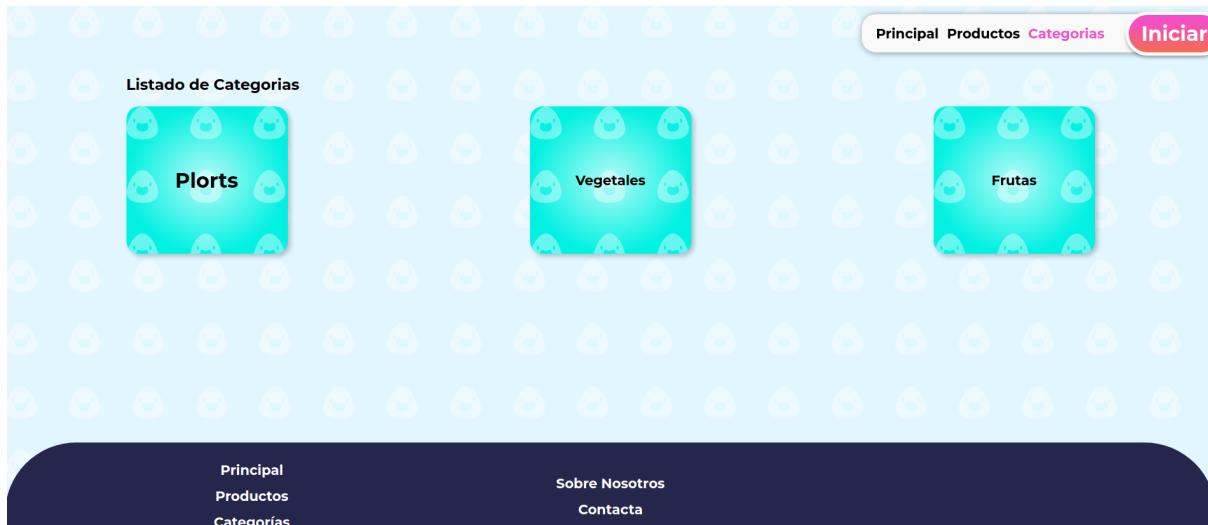
PANTALLA DE INICIO DE ACCESO GENERAL



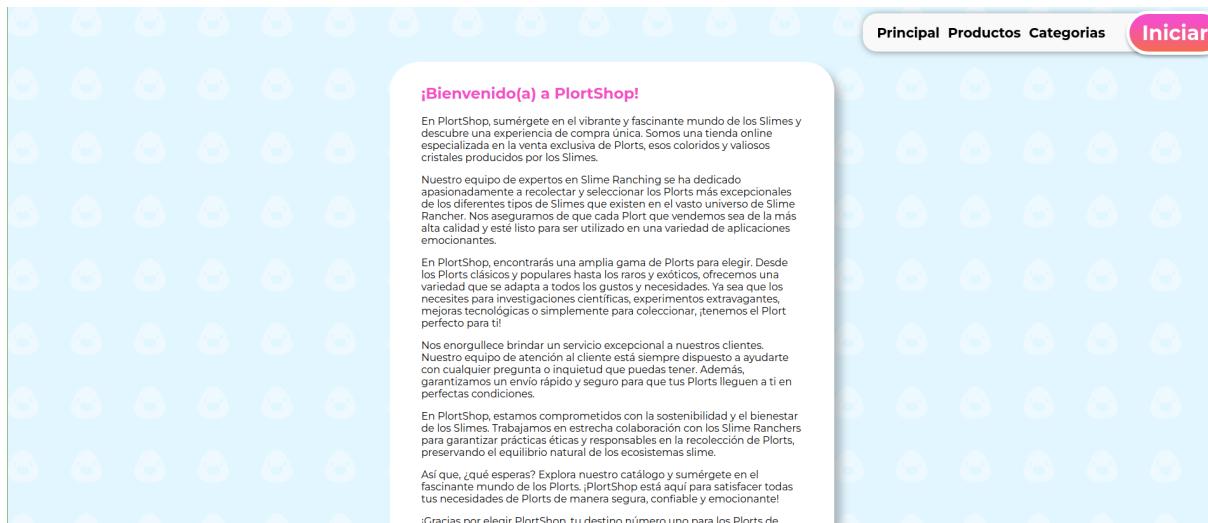
PANTALLA DE ACCESO GENERAL A PRODUCTOS



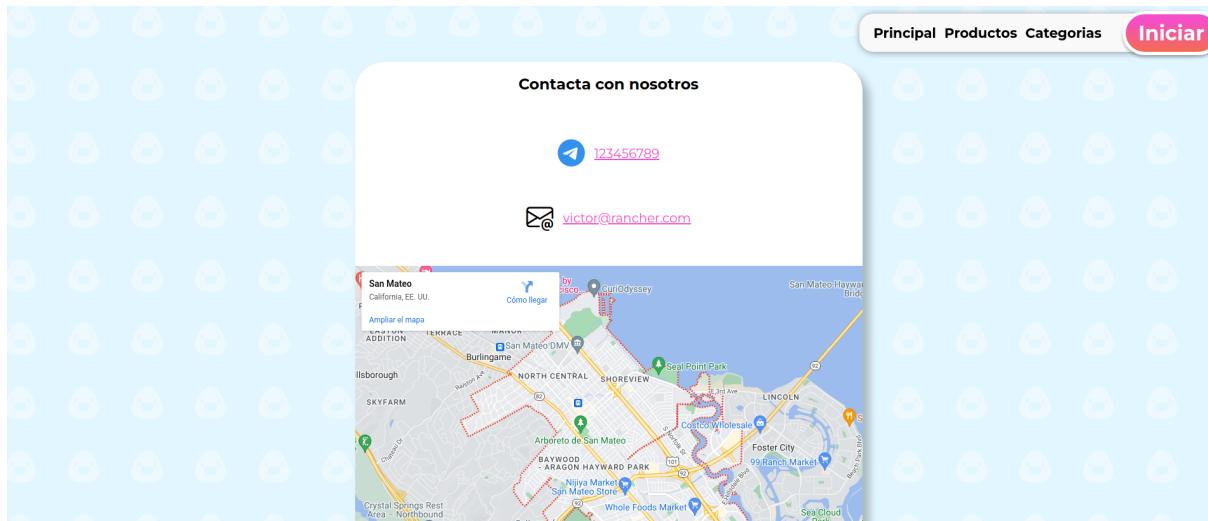
PANTALLA DE ACCESO GENERAL A CATEGORÍAS



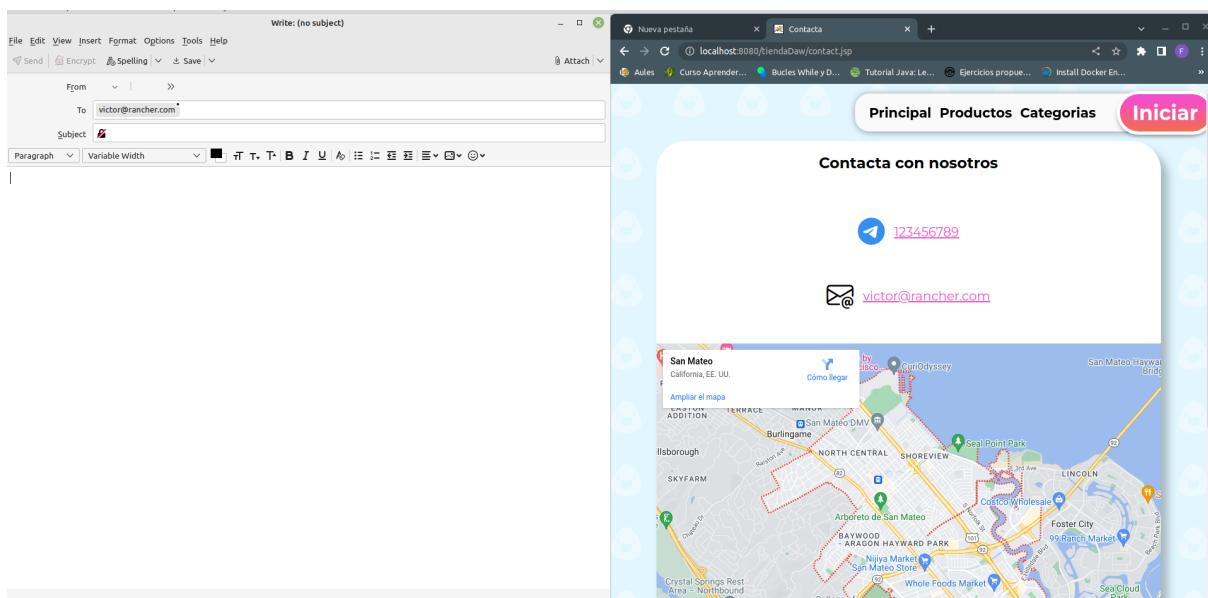
PANTALLA DE ACCESO A SOBRE NOSOTROS



PANTALLA DE ACCESO A CONTACTO



PANTALLA DE CONTACTO CON APERTURA DE GESTOR DE CORREO



Atributos y otros datos en servlets

Para el tratamiento de los atributos y datos que se manejan en la aplicación por parte del usuario, comprobamos la existencia de estos atributos. Si no existen, creemos vacíos o devolvemos la navegación en caso de ser oportuno para el envío de estos correctamente. Por último, validamos si el tipo de dato correspondiente es el que se necesita. En caso contrario, devolvemos la navegación con un mensaje de error en el frontend.

DIAGRAMA DE CLASES SERVLETS

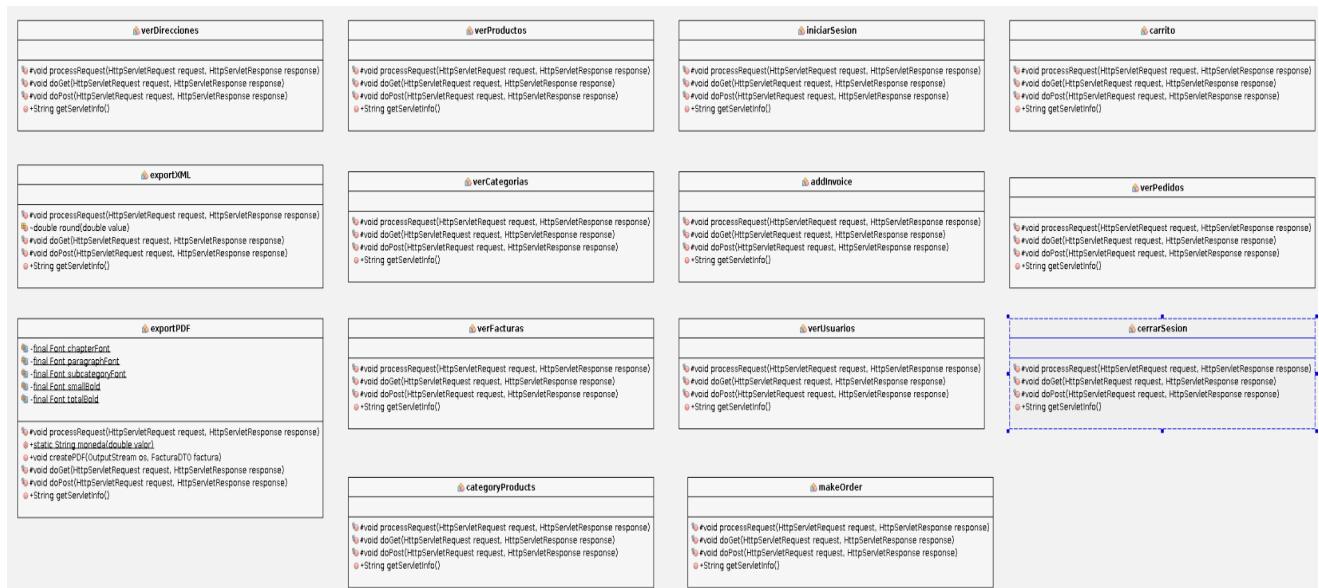
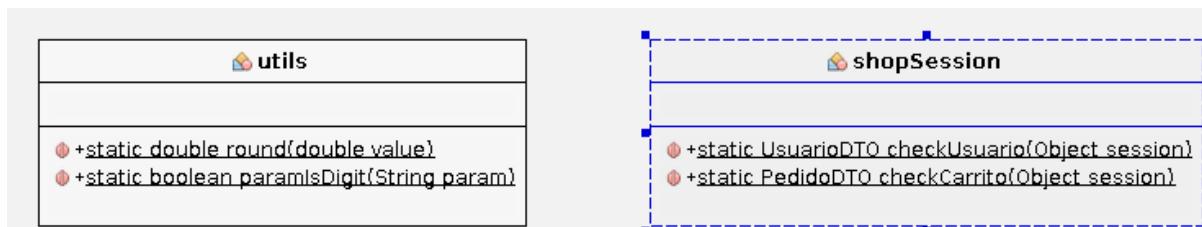


DIAGRAMA DE CLASES UTILS



Frontend

El frontend ha sido desarrollado en dos versiones principales, a partir del mismo mockup.

Mockup

Para el mockup se ha utilizado la herramienta de Figma para el diseño intentando emular el principio de diseño atómico, que a grandes rasgos consta en crear elementos independientes para poder crear conjuntos compuestos de estos más simples, dando autonomía y escalabilidad al proyecto, simulando los principios de clases y funciones usando en el backend.

Dentro del mockup podemos ver cada elemento del diseño separado en componentes que luego usaremos para montar el programa final.

[Enlace al mockup](#)

[Enlace al live Preview del mockup](#)

Primer desarrollo

Para el primer desarrollo se consideró el montaje del proyecto usando el framework de JavaScript Astro.js, ya que permite la generación de proyectos SSR, es decir Server Side Rendering, contenidos estáticos, generados en el servidor, pudiendo simular el entorno de JSP y Java, antes de que este mismo finalice.

[Más información acerca de Astro.js](#)

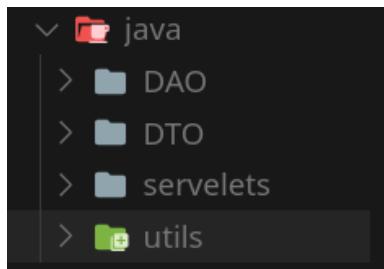
[Versión de Astro](#)

Segundo desarrollo

Para el segundo desarrollo fue necesario la fusión de los apartados del mockup junto con los jsp usando lo aprendido en la versión de astro, ya que pudimos distribuir y reutilizar tanto las vistas como los elementos de una manera más homogénea. Para ello intentamos convertir los componentes usados en Figma en [WebComponents](#).

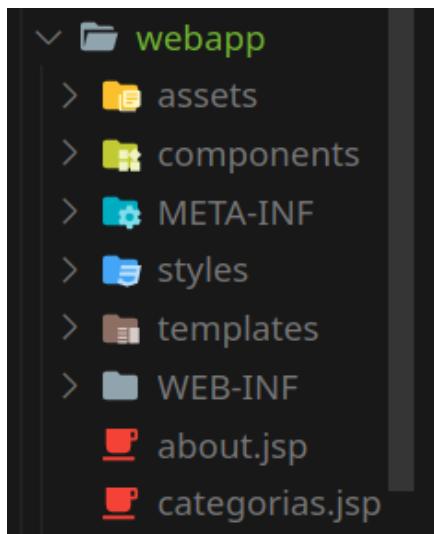
Estructuración del proyecto

Backend



Hemos dividido el proyecto en cuatro partes, dos para los DAO y DTO mencionados anteriormente, una parte dedicada a los SERVLETS y por último una clase dedicada a la verificación de sesiones y datos.

Frontend



Para el apartado de frontend tenemos:

- el directorio assets, donde se encuentran las imágenes y los audios del proyecto.
- El directorio components, donde se encuentran los componentes simples y compuestos de WebComponents.
- El directorio styles con los css generales.
- El directorio templates con las plantillas del header, head y footer.

CONCLUSIONES Y DIFICULTADES ENCONTRADAS

A través de la combinación de HTML, CSS y Java, hemos logrado crear un frontend atractivo y funcional, respaldado por un backend sólido y eficiente.

El desarrollo del frontend en HTML y CSS nos ha permitido crear una interfaz de usuario intuitiva y agradable, brindando una experiencia de uso fluida y atractiva para nuestros usuarios.

En cuanto al backend desarrollado en Java, hemos tratado de construir una base sólida y confiable, capaz de manejar de manera eficiente las solicitudes y procesos requeridos por la aplicación.

Hemos encontrado algunas dificultades relacionadas con el diseño de la base de datos y la conexión de la misma con la aplicación que han sido solucionadas gracias al apoyo de profesores y compañeros.

En resumen, el proyecto de desarrollo de aplicaciones web ha sido un desafío emocionante y enriquecedor para nosotros. A través de la aplicación de los conocimientos impartidos por nuestros profesores, hemos creado una aplicación web completa y funcional que destaca por su diseño atractivo y su robustez en el backend.

POSIBLES MEJORAS O AMPLIACIONES

La mejora del proyecto es sustancial, principalmente traer el sistema de trading y especulación de plorts, tanto individual como global.

También poder gestionar el intercambio de más newbucks por tecnología.

Pensamos que se podría implementar la opción de gestionar el alquiler de las granjas de slimes.

Y por último, se podría añadir la opción de creación de clientes y la administración de opciones en general.