



PROYECTO INTEGRADO

DESARROLLO DE APLICACIONES WEB

MAXIMUM:

NUEVA FORMA DE ALIMENTACIÓN

PARA TUS MASCOTAS

Autor: Eduardo Villar García

Tutor: Antonio Jesús de la Rosa Jiménez

Índice

1. Análisis de requisitos del proyecto.....	3
1.1 Descripción general de la aplicación.	3
1.2 Entidades principales.....	3
1.3 Requisitos en tecnologías.....	4
2.Herramientas utilizadas.....	5
3. Diseño.....	7
3.1 Diseño de datos.	7
3.2 Diseño de la aplicación.....	11
3.2.1 Diseño de backend.	11
3.2.2 Flujo de la aplicación en los procesos principales.	12
3.2.3 Diseño de la interfaz web/app.....	23
4. Implementación.....	25
4.1 Implementación de la BD.	25
4.2 Descripción de la estructura de ficheros y carpetas.....	25
5. Despliegue de la aplicación.....	26
5.1 Instrucciones para el despliegue de la aplicación.....	26
6. Conclusiones.	27
6.1 Futuras ampliaciones.	27
7. Bibliografía.	29

1. Análisis de requisitos del proyecto.

1.1 Descripción general de la aplicación.

La aplicación web desarrollada va dirigida para la venta de la nueva e innovadora línea de piensos para perros cuyo objetivo es cubrir las posibles necesidades de nuestras mascotas con cinco productos nuevos.

La idea de desarrollar esta aplicación surge de la necesidad de expandir y dar a conocer esta nueva línea de alimentos desarrollada por la empresa **Piensos Compuestos de Priego S.L** los cuales poseen más de 43 años de experiencia trabajando en el sector de la nutrición y la sanidad animal.

El proyecto permite la compra por parte de los clientes de los productos a través de un diseño amigable y una interfaz accesible desde dispositivos móviles y ordenadores.

También se ofrece al cliente la posibilidad de enviar un mensaje a través de un formulario de contacto para la consulta de dudas o cualquier tipo información que desee.

Posee un carrito de fácil acceso y un sistema de compra con mucha facilidad de uso, lo que permitirá una gran experiencia para el cliente.

Una vez realizada la compra, existen una interfaz “mis pedidos”, donde el usuario podrá ver los detalles referente a pedido y generar una factura de la compra realizada con los datos de la misma.

El administrador posee un apartado de “Blog” donde podrá subir noticias interesantes y la opción de los clientes de compartir comentarios respecto a estas.

También posee una vista de administración donde podrá monitorear diferentes opciones como el número de usuarios registrados, el estado de los pedidos y el cálculo de la venta mensual, aparte podrá crear, ver, modificar y eliminar las distintas entidades de la aplicación.

Explicado esto, podemos ver que existen principalmente 2 tipos de usuarios, y un poco por encima las distintas funcionalidades que poseen, ahora pasaremos a explicar en detalle cada parte del proceso hasta llegar a la presentación del proyecto terminado.

1.2 Entidades principales

A continuación, explicaré las diferentes entidades que conforman nuestra base de datos:

- **Usuarios:** Posee los datos personales del usuario, aparte de un apartado de dirección personal, que permitirá el autocompletado en el proceso de compra si lo posee actualizado.
- **Noticias:** Almacena las noticias creadas por el administrador.

- **Comentarios:** Almacena los comentarios de los usuarios relacionados con cada una de las noticias.
- **Productos:** Contiene toda la información correspondiente a los productos de la web.
- **Valoraciones:** Almacena las reseñas de los usuarios referente a cada uno de los productos.
- **Método Pago:** Almacena la información referente a los distintos métodos de pago.
- **Método Envío:** Almacena la información referente a los distintos métodos de envío.
- **Pedidos:** Se guarda toda la información referente a las compras de los clientes, con información de envío y facturación por separado.
- **Productos Pedido:** Almacena los productos relacionados con cada pedido, almacenando el precio unitario y la cantidad, para evitar que en caso de modificación del producto original, el precio del pedido no coincida con el de los productos.
- **Avisos:** Almacena los datos de los avisos enviados por los usuarios anónimos a través del formulario del home o de la pestaña “Contacto”.

1.3 Requisitos en tecnologías.

La web ha sido diseñada para poder usarla tanto en ordenadores como en dispositivos móviles.

El diseño de la página ha sido 100% desarrollada por mí, tanto como imágenes como distribución de los diferentes componentes, dándole un diseño amigable para el público y generando un diseño atractivo para este.

La idea es el uso de la web a nivel nacional, permitiendo al cliente el acceso a nuestro producto desde cualquier lugar del país gracias a nuestros sistemas de envío.

2.Herramientas utilizadas.

Para desarrollar un proyecto de tal magnitud desde cero requiere un control de tareas y la mezcla de distintas herramientas para conseguir el resultado final, en el desarrollo de esta aplicación he hecho uso de distintas herramientas como *frameworks*, tanto en *backend* como *frontend*, así como librerías adicionales que nos han facilitado la vida en ciertos aspectos.

La base del proyecto está basada en el framework Laravel (v. 10.5.1), un conocido *framework* para el desarrollo en PHP, el cual me ha permitido hacer una gestión de la base de datos de una manera más sencilla haciendo uso de sus migraciones y factories para la generación de contenido de manera más rápida. También me ha permitido una gestión fácil de las sesiones de usuarios, tanto registros como login, haciendo uso *Blade/Breeze*.

En la base de datos he hecho uso de XAMPP, lo que me permitía un acceso fácil a PhpMyAdmin (v. 5.2.0) y así gestionarlo todo fácilmente desde mi navegador.

Una vez obtenidas ambas cosas, nos hacía falta un ambiente de trabajo, también conocido como un editor de código, para ello he optado hacer uso de *Visual Studio Code*.

Otra parte fundamental del desarrollo de una aplicación es el uso de una herramienta de control de versiones Git, que nos permite obtener nuestros progresos desde cualquier sitio y comprobar por versiones los cambios que ha sufrido nuestro código a lo largo del tiempo, en nuestro caso hemos hecho uso de *GitHub*.

Para el *frontend*, he hecho uso de diferentes frameworks, como *Tailwind* y *Bootstrap* permitiéndome una base de diseño bastante sólida modificada más tarde por CSS a nuestra necesidad.

También he hecho uso de *JavaScript* para mejorar tanto el backend como frontend, añadiendo distintas animaciones y control de ciertos aspectos funcionales.

Entre las librerías usadas cabe destacar:

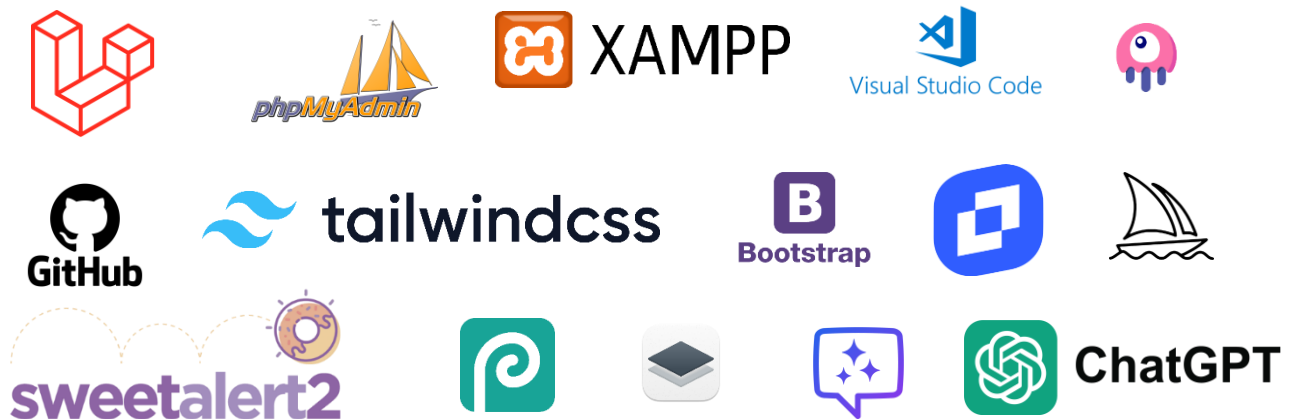
- **Livewire** (v. 2.12): es una librería de Laravel que me ha permitido crear vistas interactivas para el usuario haciendo uso de componentes *PHP* y *Blade* en lugar de *JavaScript*. He usado esta librería para aplicar filtros a diferentes apartados de la web y la creación del CRUD de las distintas entidades en el backend del administrador.
- **Dompdf** (v. 2.0.3): es una librería que facilita la generación de archivos PDF en aplicaciones web. Proporciona una sintaxis sencilla y familiar basada en *Blade*, lo que te permite diseñar las plantillas de tus

documentos PDF de manera eficiente. Ha sido la herramienta perfecta para la generación de las facturas una vez completadas las compras.

- **Darryldecode/Cart** (v. 4.2.3): es una librería de Laravel, que me permite implementar un carrito de compras en tu aplicación web.
- Te ofrece la funcionalidad de añadir y gestionar productos en un carrito, calcular el total de la compra, aplicar descuentos, etc....
- **SweetAlert2**: es una librería de *JavaScript* que me ha permitido crear y mostrar alertas personalizadas en mi aplicación web. Esta librería facilita la mejora de la experiencia del usuario al proporcionar notificaciones visuales atractivas y fáciles de usar.
- **TinyMCE**: es un editor de texto enriquecido que me ha permitido añadirlo en ciertos campos, principalmente en el apartado de blog, dándome una libertad en la creación y diseño de las noticias.

Para el diseño de imágenes, he usado herramientas como Photopea, que es una herramienta gratuita con la mayoría de funcionalidades de Photoshop, aparte he hecho uso de varias IAs tanto para el diseño como desarrollo del proyecto.

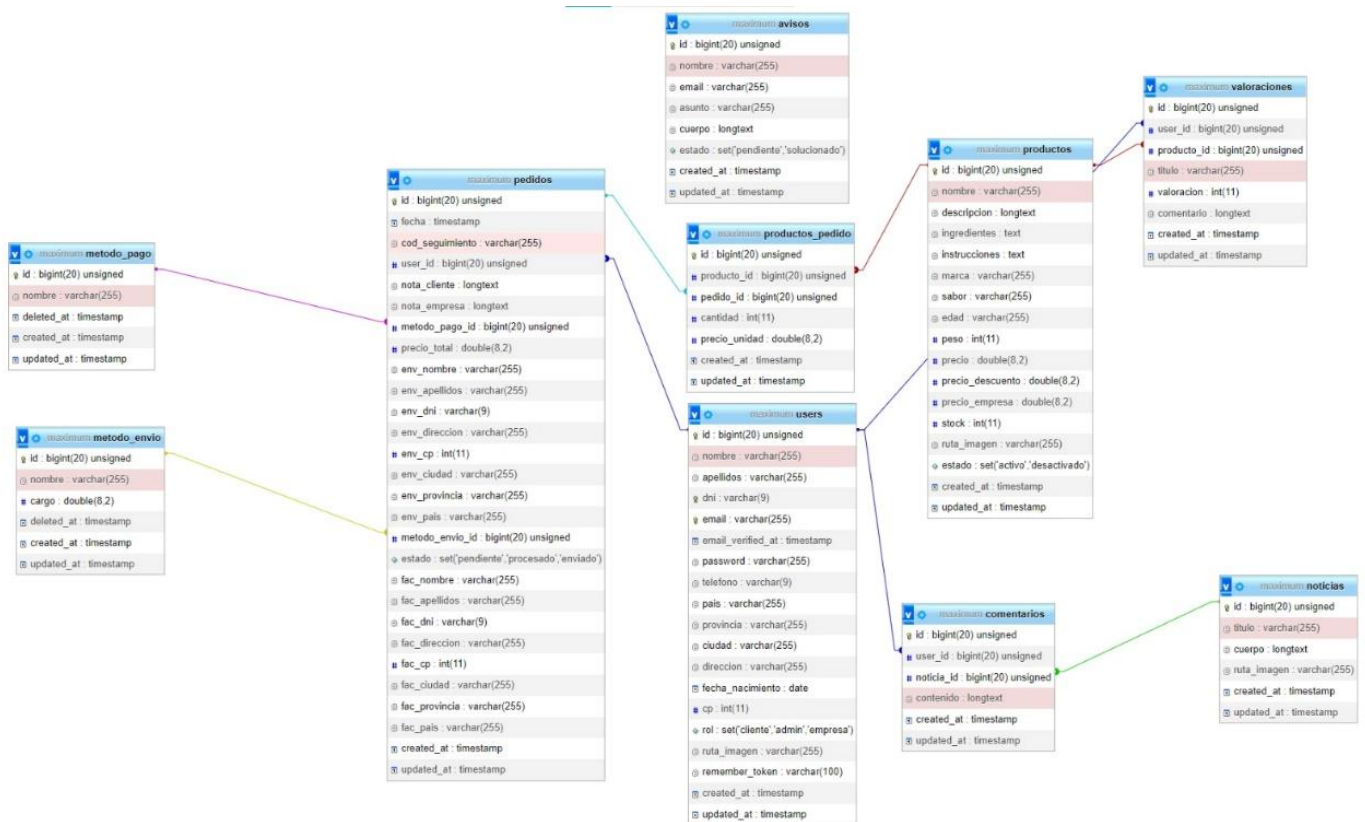
Entre ellas destacan, remove.bg (elimina el fondo de cualquier imagen), watermarkremover.io (elimina las marcas de agua de las imágenes), midjourney (generación de imágenes por texto), adobe firefly (permite edición de imágenes a través de IA), ChatGPT-4 (ofrece una ayuda en todos los ámbitos increíble), Rix (IA con la misma funcionalidad que ChatGPT pero centrada en los desarrolladores).



3. Diseño.

3.1 Diseño de datos.

Anteriormente os hablé un poco sobre las diferentes entidades que conforman mi aplicación, ahora os mostraré el diagrama entidad/relación:



Con este diagrama y ya con una idea en la cabeza de las entidades y sus relaciones detallaremos cada uno de los atributos de cada entidad:

- **Usuarios:** aquí se almacenará toda la información de los usuarios registrados. Entre los campos encontramos:
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - nombre: nombre del usuario. Tipo varchar.
 - apellidos: apellidos del usuario. Tipo varchar.
 - dni: dni del usuario y de uso único. Tipo varchar.
 - email: correo electrónico del usuario y de uso único. Tipo varchar.
 - email_verified_at: este campo indica en que fecha ha sido verificado el email del usuario. Tipo timestamp.
 - password: contraseña de la cuenta del usuario. Se encuentra cifrada. Tipo varchar.
 - teléfono: teléfono de contacto del usuario. Tipo varchar.
 - país: país de residencia del usuario. Tipo varchar.
 - provincia: provincia de residencia del usuario. Tipo varchar.
 - ciudad: ciudad de residencia del usuario. Tipo varchar.
 - dirección: dirección de residencia del usuario. Tipo varchar.
 - fecha_nacimiento: fecha de nacimiento del usuario. Tipo varchar.
 - cp: código postal de residencia del usuario. Tipo entero.
 - rol: rol que posee dicho usuario en la web, ya sea 'cliente', 'admin', 'empresa'. Dependiendo del rol poseerá diferentes permisos en la aplicación.
 - ruta_imagen: almacena la ruta de almacenamiento de la imagen del usuario. Tipo varchar.
- **Noticias:** almacena las noticias del momento registradas por el administrador. Entre los campos encontramos:
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - titulo: titulo de la noticia. Tipo varchar.
 - cuerpo: cuerpo de la noticia. Tipo longtext.
 - ruta_imagen: almacena la ruta de almacenamiento de la portada de la noticia.
- **Comentarios:** almacena los comentarios relacionados con las diferentes noticias, es una tabla generada de la relación N:M entre usuarios y comentarios. Entre los campos encontramos:
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - user_id: clave foránea procedente de la tabla usuarios, enlaza al usuario creador del comentario de la noticia con esta tabla. Tipo entero.

- noticia_id: clave foránea procedente de la tabla noticia, enlaza la noticia con el comentario de esta tabla. Tipo entero.
- contenido: almacena el comentario del usuario. Tipo longtext.
- **Productos:** almacena la información relacionada con los productos. Entre los campos contiene:
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - nombre: nombre del producto. Tipo varchar.
 - descripcion: descripción del producto. Tipo longtext..
 - ingredientes: ingredientes del producto. Tipo longtext..
 - instrucciones: instrucciones del producto. Tipo longtext..
 - marca: marca del producto. Tipo varchar.
 - sabor: sabor del producto. Tipo varchar
 - edad: edad del producto. Tipo varchar
 - peso: peso del producto. Tipo int
 - precio: precio general del producto. Tipo float.
 - precio_descuento: precio descuento del producto. Tipo float.
 - precio_empresa: precio empresa del producto. Tipo float.
 - stock: stock del producto en almacén. Tipo float.
 - ruta_imagen: almacena la ruta de la imagen referente al producto.
 - estado: estado en el que se encuentra el producto, ya sea como activo o desactivado. Tipo set.
- **Valoraciones:**
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - user_id: clave foránea procedente de la tabla usuarios, enlaza al usuario creador de la valoración con esta tabla. Tipo entero.
 - producto_id: clave foránea procedente de la tabla productos, enlaza al producto con la valoración. Tipo entero.
 - titulo: titulo de la valoración. Tipo varchar.
 - valoracion: valoración que va del 1-5 estrellas. Tipo entero.
 - comentario: comentario de la valoración. Tipo longtext.
- **Método Pago:**
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - nombre: nombre del método de pago. Tipo varchar.
- **Método Envío:**
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - nombre: nombre del método de envío. Tipo varchar.
 - cargo: precio del método de envío. Tipo float.

- **Pedidos:**

- id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
- fecha: fecha en la que se ha realizado el pedido. Tipo date.
- cod_seguimiento: cod.seguimiento del pedido, es único. Tipo varchar.
- user_id: clave foránea procedente de la tabla usuarios, enlaza al usuario del pedido con esta tabla. Tipo entero.
- nota_cliente: nota que puede dejar el cliente sobre el pedido. Tipo longtext.
- nota_empresa: nota que puede dejar el administrador sobre el pedido. Tipo longtext.
- método_pago_id: clave foránea procedente de la tabla método pago con esta tabla. Tipo entero.
- precio_total: precio total del pedido. Tipo float.
- env_nombre: nombre de envío. Tipo varchar.
- env_apellidos: apellidos de envío. Tipo varchar.
- env_dni: dni de envío. Tipo varchar.
- env_direccion: direccion de envío. Tipo varchar.
- env_cp: cp de envío. Tipo int.
- env_ciudad: ciudad de envío. Tipo varchar.
- env_provincia: provincia de envío. Tipo varchar.
- env_pais: país de envío. Tipo varchar.
- metodo_envio_id: clave foránea procedente de la tabla método envío con esta tabla. Tipo entero.
- estado: estado del pedido. Tipo set.
- fac_nombre: nombre de facturación. Tipo varchar.
- fac_apellidos: apellidos de facturación. Tipo varchar.
- fac_dni: dni de facturación. Tipo varchar.
- fac_direccion: direccion de facturación. Tipo varchar.
- fac_cp: cp de facturación. Tipo entero.
- fac_ciudad: ciudad de facturación. Tipo varchar.
- fac_provincia: provincia de facturación. Tipo varchar.
- fac_pais: país de facturación. Tipo varchar.

- **Productos_pedido:**

- id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
- producto_id: clave foránea procedente de la tabla productos, enlaza al producto del pedido con esta tabla. Tipo entero.
- pedido_id: clave foránea procedente de la tabla pedidos, enlaza al pedido con los producto. Tipo entero.
- cantidad: cantidad de productos comprados en un pedido. Tipo entero.
- precio_unidad: precio unitario del producto a la hora de la compra. Tipo float.

- **Avisos:**
 - id: clave primaria de la tabla e identificación única de la entidad. Tipo entero.
 - nombre: nombre de la persona que envía el aviso. Tipo varchar.
 - email: email de la persona que envía el aviso. Tipo varchar.
 - asunto: asunto del aviso. Tipo varchar.
 - cuerpo: cuerpo del aviso. Tipo longtext.
 - estado: estado del aviso, principalmente indica si ha sido leído o no por el administrador. Tipo set.

3.2 Diseño de la aplicación.

3.2.1 Diseño de backend.

El desarrollo de mi proyecto está basado en la metodología Modelo-Vista-Controlador (MVC) haciendo uso extensivo de *Eloquent* y *Laravel*.

El uso de la metodología MVC nos permite una separación clara de responsabilidades y facilita la construcción del código.

El modelo representa la capa de datos, la vista se encarga de la presentación de la interfaz al usuario y el controlador coordina la interacción entre el modelo y la vista.

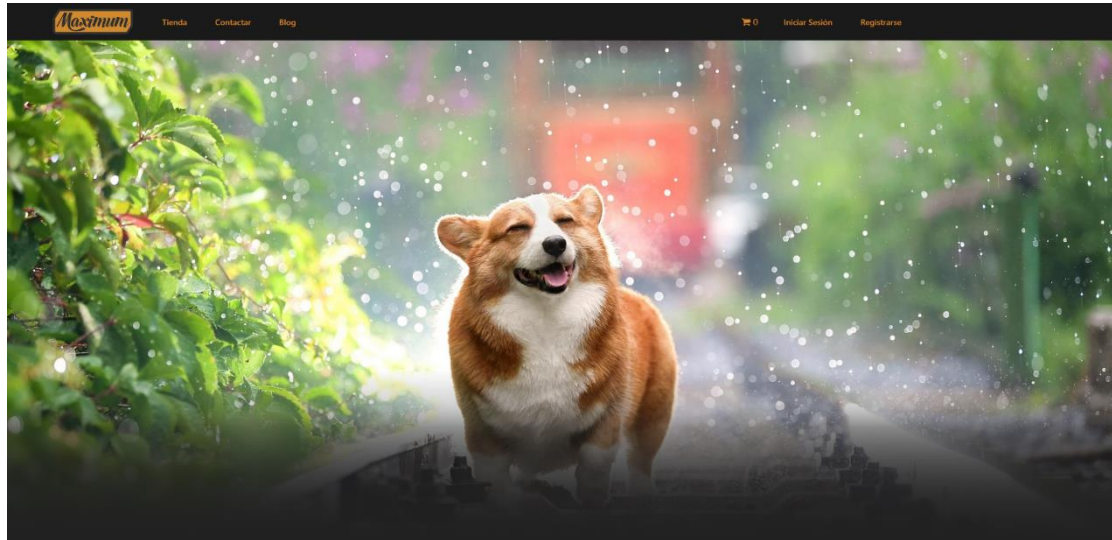
Eloquent es el ORM (Object-Relational Mapping) incluido en *Laravel*. Me ha proporcionado una forma sencilla de interactuar con la base de datos haciendo uso de objetos en lugar de consultas SQL directas.

Unas de las funcionalidades más importante que posee es la posibilidad de obtener registros relacionados en otras tablas a un objeto de forma que no hace falta interactuar con la tabla ajena.

Por ejemplo, en mi caso me ha sido posible a través del método relacional definido en el modelo sobre la tabla 'Pedidos' obtener los registros relacionados a ese usuario, lo que ha reducido el número de consultas realizadas de manera manual en muchos apartados.

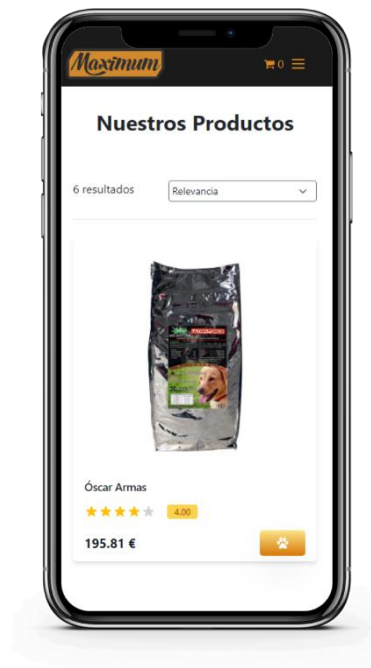
3.2.2 Flujo de la aplicación en los procesos principales.

Cuando el usuario accede a nuestra web lo primero que encontrará será el *home*, esta pantalla mostrará el siguiente diseño, en ella visualizaremos información, un formulario de contacto y un carrusel de productos.

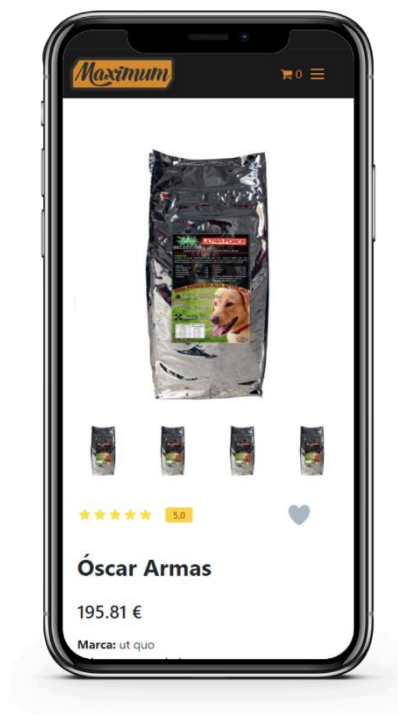
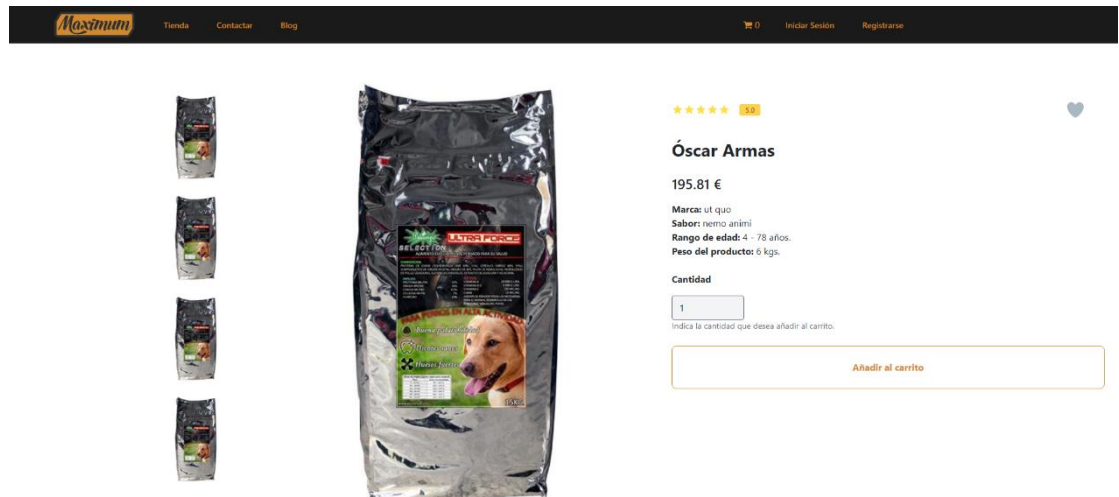


En el navbar nos encontramos diferentes apartados como:

- **Tienda:** en esta vista se mostrarán los diferentes productos aplicando una paginación y unos filtros de ordenación según precio.



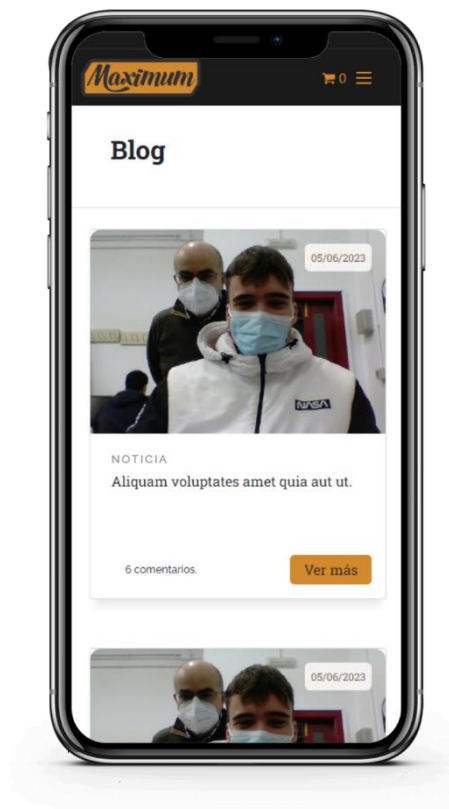
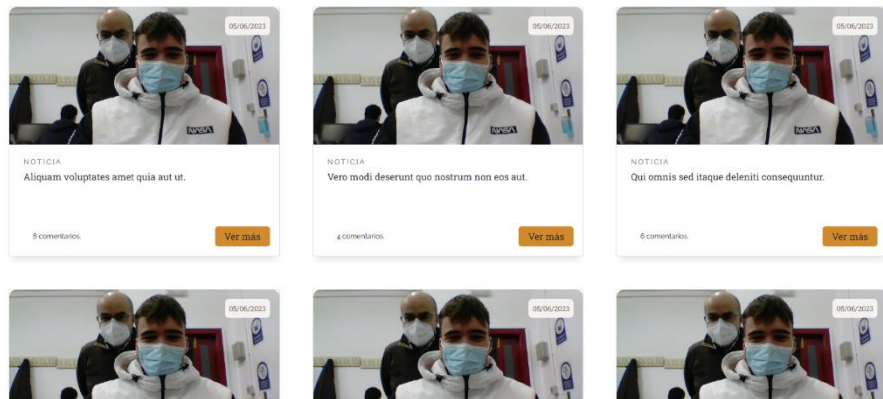
Desde esta vista tenemos acceso a la vista individual de los productos, mostrándose así:



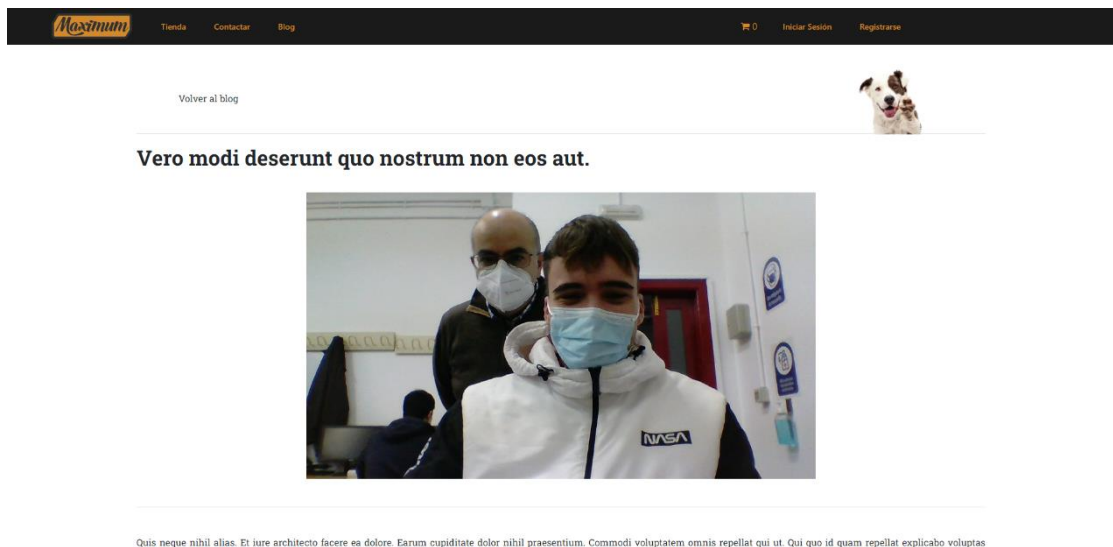
- **Blog:** en este apartado obtenemos una vista de diferentes noticias mostrando una vista modo ‘viñeta’ donde mostramos un resumen de esta, aparte, tenemos visión de las vistas de las noticias de manera individual.



Blog



Al presionar el botón ‘Ver más’, podemos ver la vista individual de la noticia correspondiente:



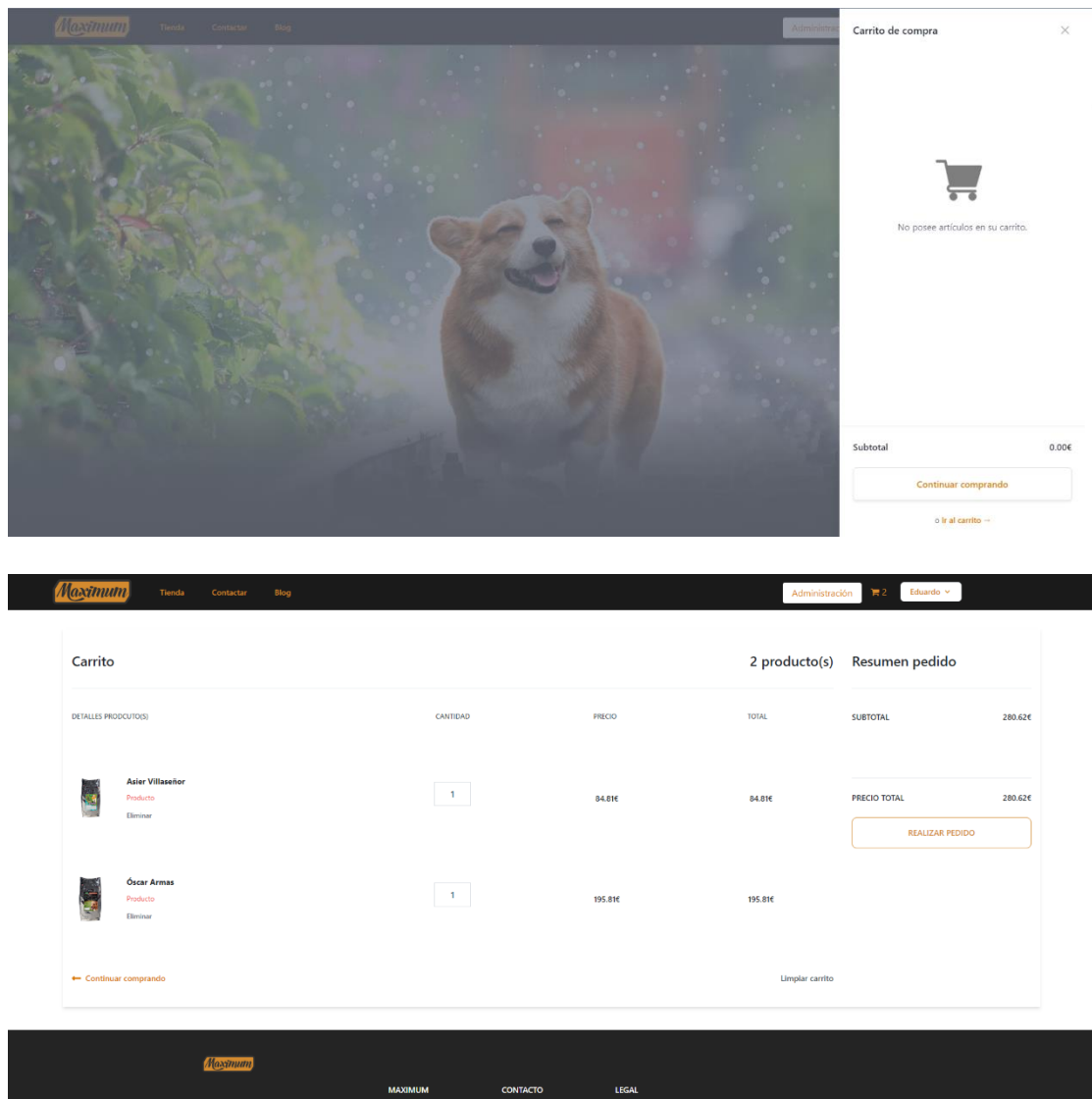
- **Login/Registro:** en estas pantallas el usuario puede loguearse a través de su cuenta o registrarse en caso de no poseer una.

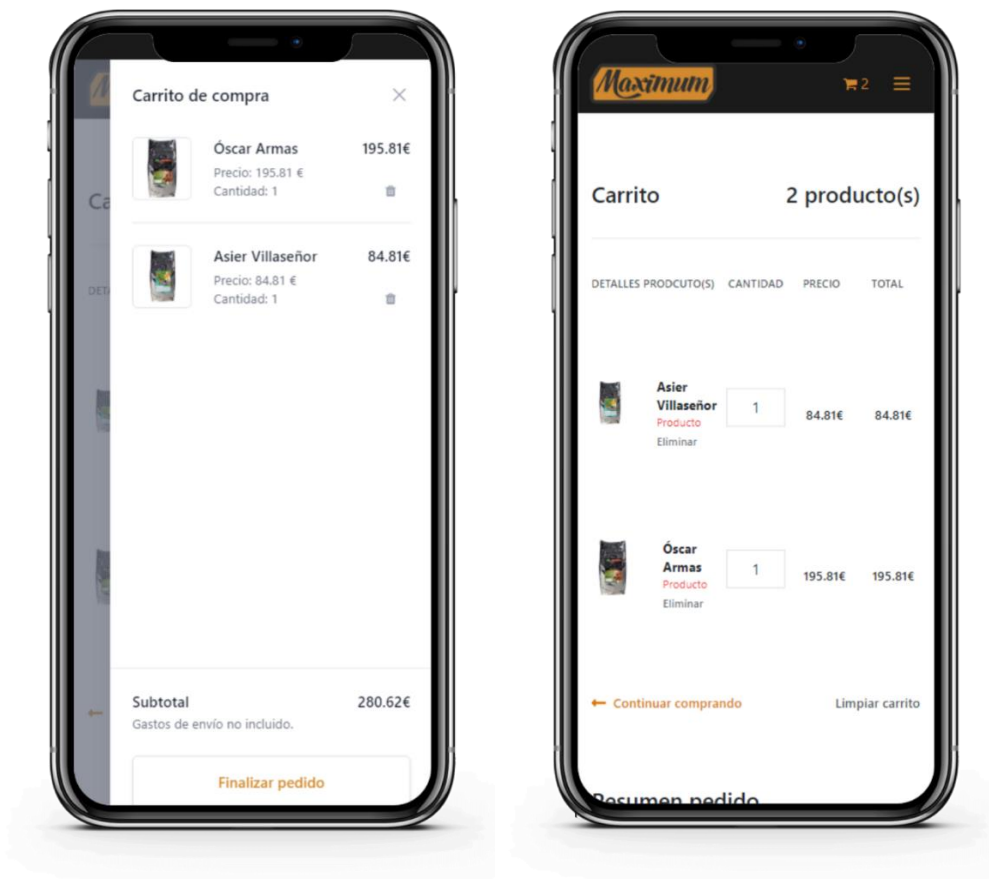
The screenshot shows the login page of the Maximum website. At the top, there is a dark navigation bar with the Maximum logo on the left and links for 'Tienda', 'Contactar', 'Blog', 'Inicio Sesión', and 'Registrarse' on the right. The main content area has a light gray background. In the center, there is a white box containing the login form. The form is titled 'Iniciar sesión' with a sub-link '¿Todavía no tienes cuenta? ¡Regístrate!' below it. The form includes two input fields: 'Correo electrónico' and 'Contraseña'. Below the password field is a checkbox labeled 'Recordarme'. At the bottom of the form, there is a link '¿Olvidaste tu contraseña?' and an orange 'Ingresar' button.

The screenshot shows the registration page of the Maximum website. It features the same dark navigation bar as the login page. The main content area has a light gray background. In the center, there is a white box containing the registration form. The form is titled 'Crear cuenta' with a sub-link '¿Ya tienes cuenta? ¡Inicia sesión!' below it. The form includes four input fields: 'Nombre', 'Correo electrónico', 'Contraseña', and 'Confirmar contraseña'. At the bottom of the form, there is a link '¿Tienes cuenta?' and an orange 'Register' button.

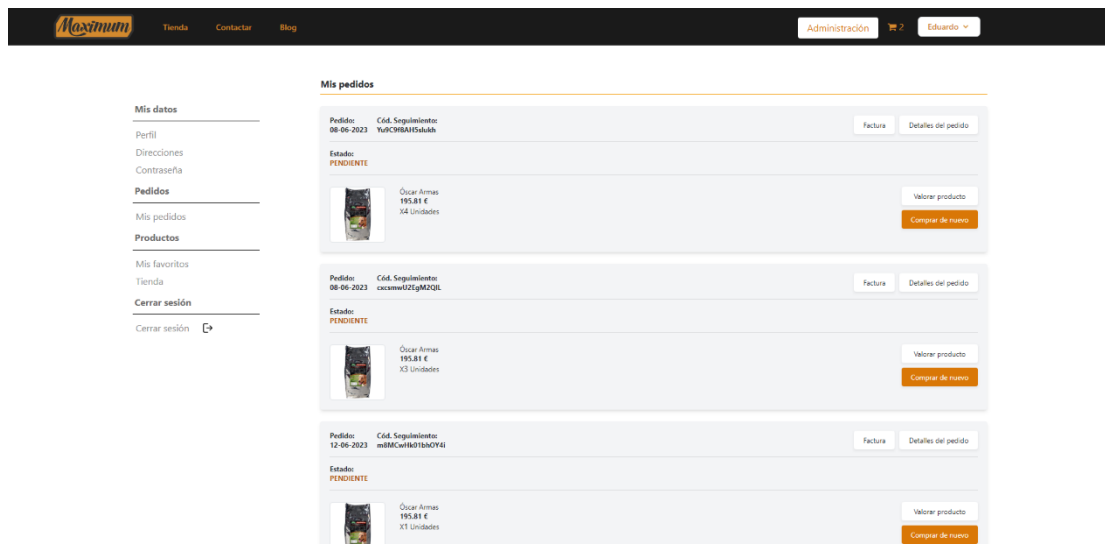


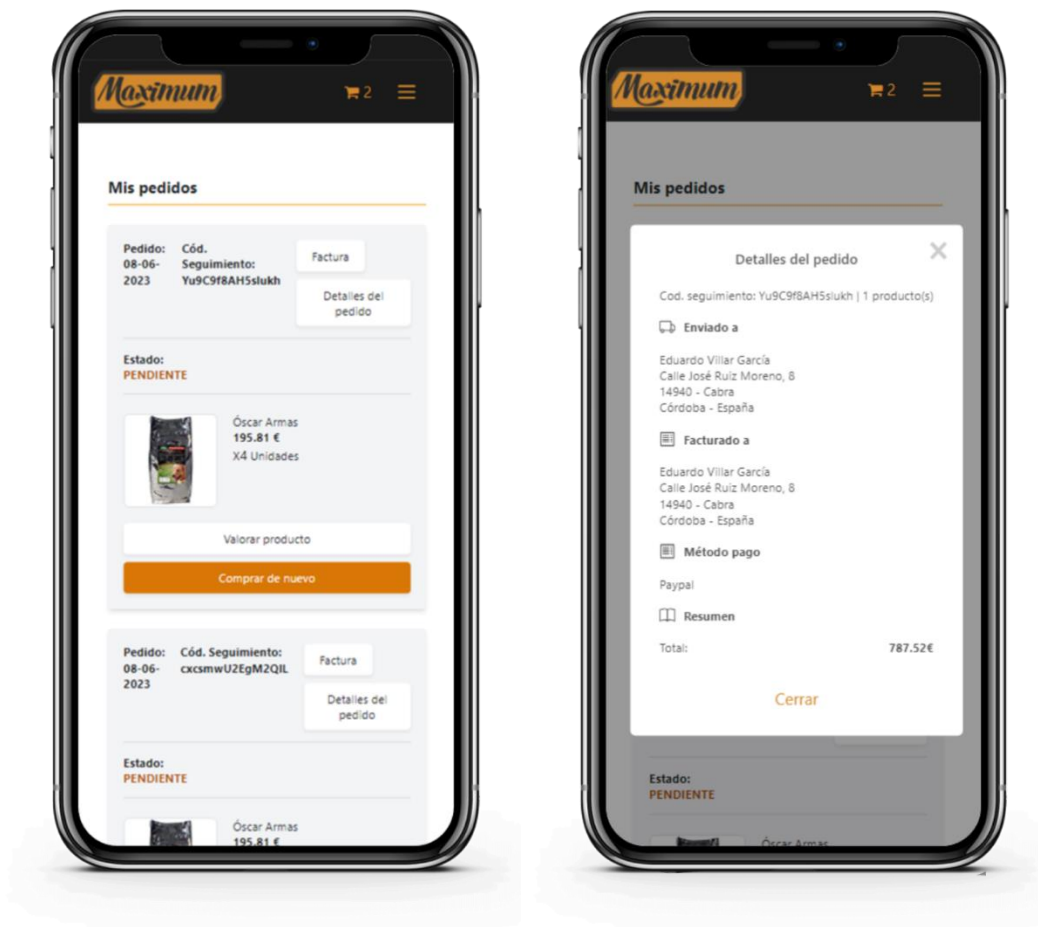
- **Carrito:** esta vista se divide en 2, el *dropcart* el cual se despliega a través del icono del navbar y que es visual desde cualquier vista de la web, y la vista unitaria del carrito.



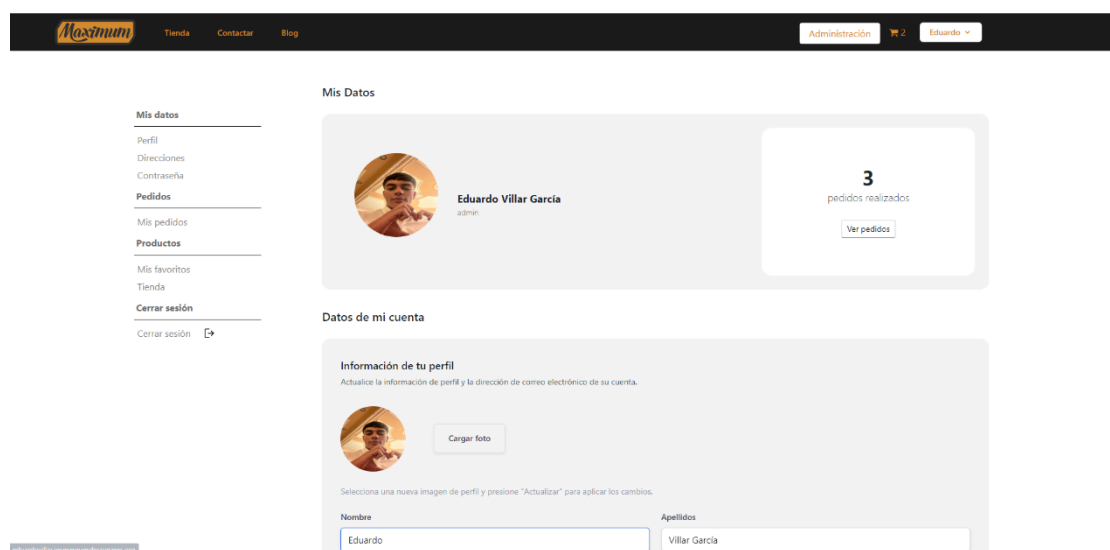


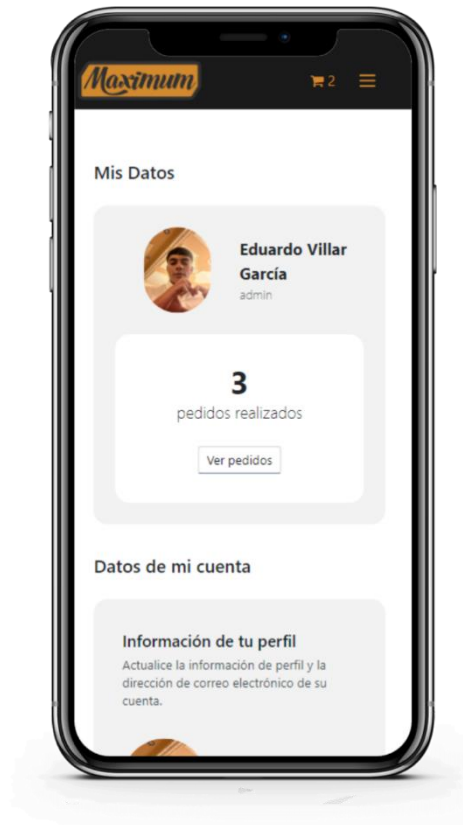
- **Mis pedidos:** en este apartado veremos una recapitulación de los pedidos que hemos realizado:



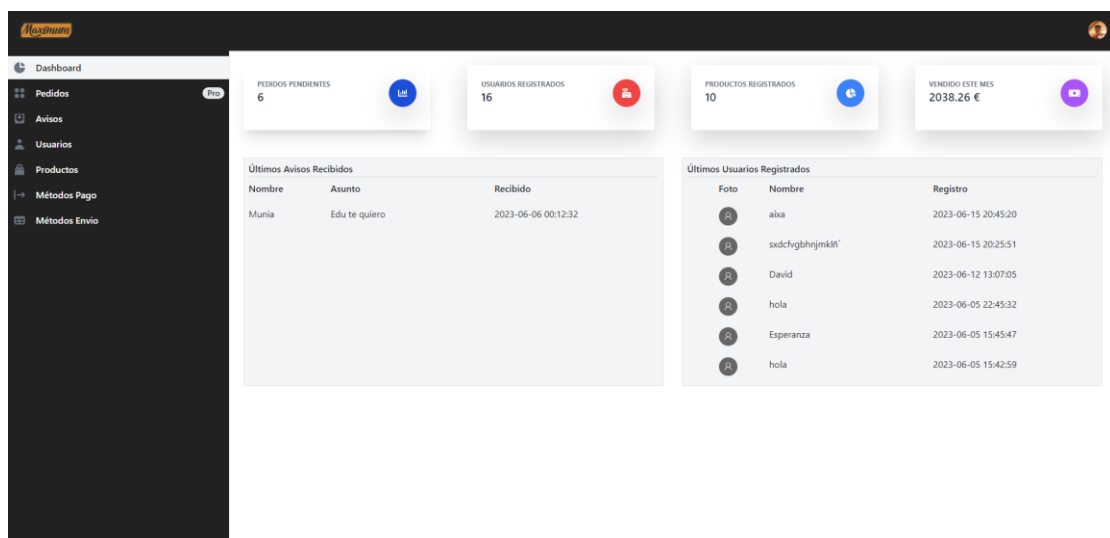


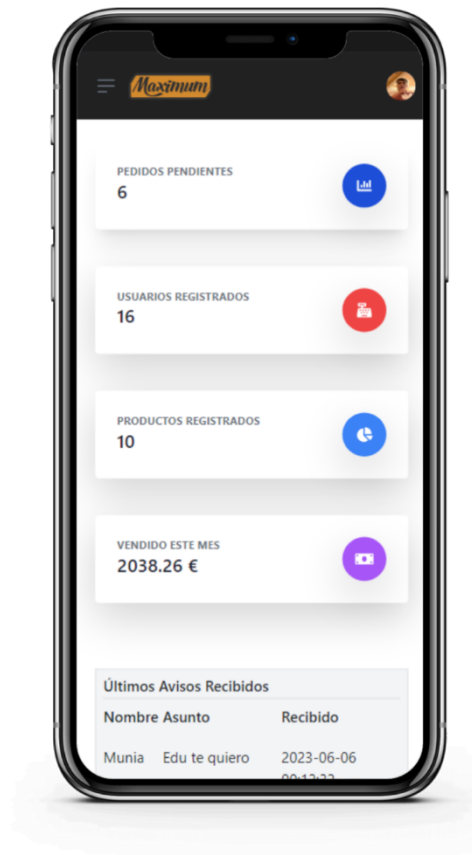
- **Perfil:** En este apartado mostramos toda la información del usuario como la posibilidad de modificar dicha información.





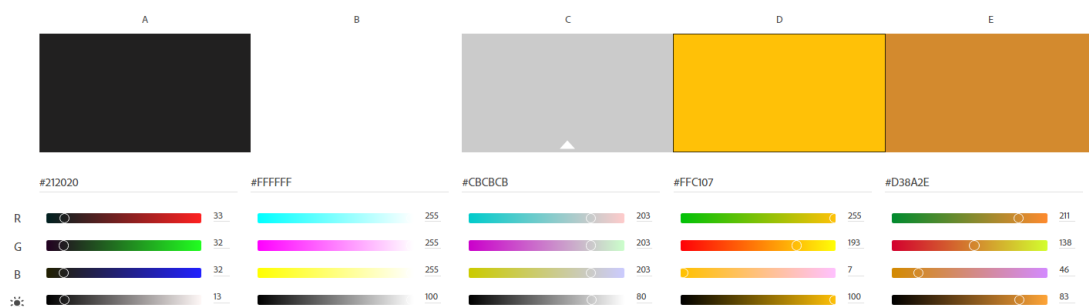
- **Administración:** En este apartado se muestra las funcionalidades del administrador sobre las distintas entidades, tenemos una navegación destinada a cada una de las entidades:






3.2.3 Diseño de la interfaz web/app.


La paleta de colores seleccionada ha sido elegida para dar una imagen de calidad, de un producto *premium* queriendo transmitir la calidad de los materiales y el resultado final que se ha conseguido con ello en los productos.





Durante el proyecto podemos ver diferentes iconos en nuestra aplicación los cuáles nos facilitaran de manera visual las diferentes acciones que ofrecen, para ello hemos hecho uso de imágenes vectoriales (.svg) los cuáles a través de la herramienta figma conseguimos un sencillo modo de obtener las etiquetas <svg> de estos.


También hacemos uso de iconos procedentes de la librería Bootstrap.
Aquí algunos ejemplos de los diferentes iconos que puedes encontrar:


 [Tienda](#) [Contactar](#) [Blog](#) [0](#) [Iniciar Sesión](#) [Registrarse](#)


 **UBICACIÓN**
Crtra, Calle Fuente Alhama, km 0.1
14800 Priego de Córdoba, Córdoba


 **LLÁMANOS**
Teléfono: 957 70 05 43

 **HORARIO**
Mañanas: Lun - Sáb ... 9.00 - 14.00
Tardes: Lun - Vie ... 16.00 - 20.00

PEDIDOS PENDIENTES
4 

USUARIOS REGISTRADOS
14 

PRODUCTOS REGISTRADOS
10 

VENDIDO ESTE MES
1779.52 € 

4. Implementación

4.1 Implementación de la BD.

Para la implementación de la base de datos, nos hará falta tener un sistema de BD ya instalado previamente, como he explicado anteriormente haciendo uso de *XAMPP* disponemos de *PhpMyAdmin* en caso de instalación en local y en caso de instalación en servidor, hacemos uso de un servidor MySQL. Para configurar en nuestro proyecto la conexión a las distintas bases de datos, anteriormente creadas, debemos configurar el fichero `.env`, para no modificar el fichero `.env` de nuestro servidor ya que poseerán configuraciones distintas, añadiremos el fichero al `.gitignore` de nuestro repositorio, evitando así que detecte cambios y lo suba constantemente rompiendo en nuestro hosting la conexión.

4.2 Descripción de la estructura de ficheros y carpetas.

Laravel usa un sistema de ficheros muy simple, dividiendo en carpetas todos estos de una manera muy intuitiva y fácil de usar.

Nombre	Último commit message	Último commit date
..		
app	cambios finales??	2 hours ago
bootstrap	spliting	2 months ago
config	pedido completado	3 weeks ago
database	actualizacion backend 50/50	last week
public	cambios	3 days ago
resources	final??	1 hour ago
routes	detalles	3 hours ago
storage	spliting	2 months ago
tests	spliting	2 months ago
└─ .editorconfig	spliting	2 months ago
└─ .env.example	spliting	2 months ago
└─ .gitattributes	spliting	2 months ago
└─ .gitignore	modificado	last month
└─ README.md	spliting	2 months ago
└─ artisan	spliting	2 months ago
└─ composer.json	generacionfacturas y detalles pedido completado	3 weeks ago
└─ composer.lock	generacionfacturas y detalles pedido completado	3 weeks ago
└─ package-lock.json	spliting	2 months ago
└─ package.json	spliting	2 months ago
└─ phpunit.xml	spliting	2 months ago
└─ postcss.config.js	spliting	2 months ago
└─ tailwind.config.js	spliting	2 months ago
└─ vite.config.js	spliting	2 months ago

En la carpeta ‘app’ encontramos todas las carpetas correspondientes a los Controladores y Modelos de nuestra aplicación.

La carpeta ‘database’ almacena todas las migraciones, seeder y factories de la aplicación, siendo estos muy fáciles de localizar.

También destaca la carpeta ‘resources’ donde vienen todas las vistas, css y js de nuestra aplicación.

En la carpeta ‘routes’ encontramos definidos distintos ficheros donde se especifican las diferentes rutas de nuestra aplicación.

Para la gestión de imágenes y documentos en nuestra aplicación haremos uso de las carpetas ‘public’ y ‘storage’, en *public* añadiremos las imágenes y contenido que queremos compartir de manera pública en nuestra web, como logo

corporativo, mientras que en la carpeta *storage* añadiremos las imágenes con un trato más privativo como las imágenes personales de los usuarios, las cuales ninguna persona debe tener acceso.

5. Despliegue de la aplicación.

5.1 Instrucciones para el despliegue de la aplicación.

Durante mi desarrollo decidí desplegar la aplicación en un hosting que me fue ofrecido por el instituto y en el cual tuve ciertos problemas al desplegarlo. El servidor se encuentra creado en un sistema operativo Ubuntu Server (v. 20.04), tuve que realizar la actualización de las versiones de MySQL y Apache, posteriormente instalar *composer* y actualizarlo a su versión 2.

Una vez hecho esto, cloné mi repositorio en `/var/www/html/` donde conseguí bajarme el proyecto.

Una vez bajado fue necesario ejecutar varias veces desde la raíz del repositorio clonado el comando `'composer install'` para la instalación correcta de las distintas librerías y dependencias del proyecto, gracias al fichero `composer.json`.

También fue necesario la configuración del VirtualHost, indicando la carpeta raíz del proyecto, ya que por un fallo mío, de mi repositorio cuelga una carpeta llamada *maximum* la cual es la carpeta madre del proyecto.

Más tarde tuve que configurar el fichero de conexión a base de datos `.env`, teniendo problemas con este ya que tuve que crear un nuevo usuario con todos los permisos en una nueva base de datos dedicada para mi proyecto, ya que no tenía correcto acceso al usuario root.

Con todos los problemas que tuve, a día de hoy la web sigue operativa y puede ser consultada por cualquiera en:

<http://eduardovillar.iesmarquesdecomares.org/>

6. Conclusiones.

Una vez expuesto mi proyecto parte por parte, dando la mayor explicación posible y después de muchas horas invertidas durante estos meses, es momento de indicar que la versión recibida al final de este proceso, no me tiene contento, he tenido varias dificultades en ciertos aspectos que me han hecho limitarme la cantidad de funciones finales que quería añadir, no consiguiendo el resultado esperado.

Aún así considero que esas dificultades y vueltas a muchos aspectos me han hecho por una parte mejorar en estos y conocer gustos que desconocía.

Creo que he dado un paso adelante en el desarrollo de *PHP*, especialmente en *Laravel*, cierto lenguaje y *framework* que me ha generado un interés superior por ellos que cualquier otro *framework* o lenguaje aprendido.

También he avanzado en el uso de ciertas herramientas como *Livewire* descubriendo nuevas posibilidades que desconocía.

He descubierto el gusto por el diseño de webs, invirtiendo muchas horas durante estos meses con mi compañero José Antonio, compañero de clase y prácticas, el cuál me ha ayudado mucho a mejorar tanto con herramientas como *Photoshop* y simplemente gusto y distintas ideas de diseño que antes no había contemplado.

Haciendo estas dos cosas, crearme un gran interés futuro por convertirme en *Full Stack Developer*, el cuál voy a intentar seguir progresando por llegar a conseguirlo.

6.1 Futuras ampliaciones.

Como he nombrado anteriormente, considero que muchos de los aspectos que quería implementar no he tenido el tiempo suficiente para hacerlo, pero aun así seguiré trabajando en ello para en un futuro obtener una versión completa de esta aplicación ya que el objetivo real de esta es el despliegue en un futuro para su uso en el mundo comercial web.

A continuación nombraré algunas de las ampliaciones principales que tengo para un futuro:

- Implementación de pasarelas de pago.
- Gestión de precios dependiendo del tipo de usuario que es, ya que aunque actualmente poseo el campo `precio_empresa` en los productos, no hago uso de este campo.
- Un sistema de seguimiento de los pedidos de manera más completa.
- Implementación de un sistema de favoritos permitiendo al usuario el añadir ciertos productos a favoritos favoreciendo el acceso rápido a estos.
- Implementación de sistemas de correo, añadiendo actualización de stock, actualizaciones de los pedidos, ofertas, noticias del blog, etc.

- Implementación de filtros más específicos por edad/sabor/marca en los productos.
- Implementación de librería *Canvajs* para generar gráficos en administración sobre total de ventas por meses, etc.ç
- Mejora de los controladores a *Livewire*.

7. Bibliografia.

- Adobe - <https://www.adobe.com/>
- Figma - <https://www.figma.com/>
- Laravel - <https://laravel.com/docs/10.x/readme>
- Visual Studio Code - <https://code.visualstudio.com/>
- ChatGPT - <https://chat.openai.com/>
- W3Schools - <https://www.w3schools.com/>
- Livewire - <https://laravel-livewire.com/>
- Rix - <https://hashnode.com/rix>
- Photopea - <https://www.photopea.com/>