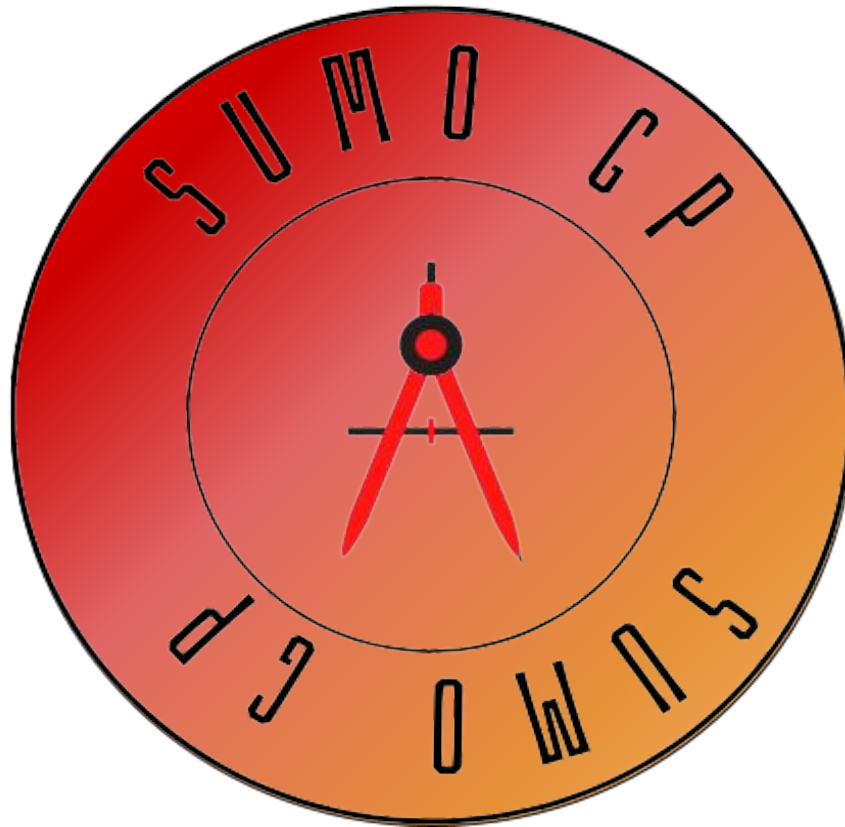


# APP WEB de venta de materiales de oficina



---

Guillermo Mendoza Castro  
Pau Motos Piquer

**Director:** Josep Vaño Chic.  
**Realizado a:** INK SPOT S.L.L  
**Nombre del ciclo:** CFGS segundo curso  
Desarrollo de Aplicaciones Web.  
**Nombre del módulo:** MP12-Proyecto de  
desarrollo de Aplicaciones Web.  
**Instituto:** Ins Daniel Blanxart i Pedrals.  
**Curso:** 2022-2023.  
**Fecha de entrega:** 19 de mayo de 2023.

## RESUMEN DEL PROYECTO

Este proyecto consiste en realizar una página web para la empresa INK SPOT S.L.L , que nos pidió que lo hiciéramos para poder expandirse y vender productos de oficina a través de su web por el alrededor de barcelona.

Nosotros decidimos hacer una página que tuviera un control del stock con una base de datos, y para ello la plataforma se ha implementado con el framework **laravel** de php al back-end, y para el front-end utilizamos **HTML5** i **Bootstrap**.

El diseño de esta aplicación no incluye responsive para dispositivos móviles de momento,pero para tablets y ordenadores si lo admite.El principal objetivo en este apartado para la aplicación es que el usuario puede navegar por diferentes apartados de la página con una buena y cómoda estructura de información.

Dicho esto , esta aplicación incluye dos áreas, una que es la visualización y otra la modificación de datos que se aplica en el área de visualización. Importante saber que el área de modificación está restringida en función de los permisos de acceso que tenga el usuario.

## RESUM DEL PROJECTE

Aquest projecte consisteix a fer una pàgina web per a l'empresa INK SPOT S.L.L, que ens va demanar que ho féssim per poder expandir-se i vendre productes d'oficina a través de la seva web al voltant de Barcelona.

Nosaltres vam decidir fer una pàgina que tingués un control de l'estoc amb una base de dades, i per això la plataforma s'ha implementat amb el framework laravel de php al back-end, i per al front-end utilitzem HTML5 i Bootstrap.

El disseny d'aquesta aplicació no inclou responsive per a dispositius mòbils de moment, però per a tauletes i ordinadors si ho admet. informació.

Dit això, aquesta aplicació inclou dues àrees, una que és la visualització i una altra la modificació de dades que s'aplica a l'àrea de visualització. És important saber que l'àrea de modificació està restringida en funció dels permisos d'accés que tingui l'usuari.

## PROJECT SUMMARY

This project consists of making a web page for the company INK SPOT S.L.L, which asked us to do it in order to expand and sell office products through its website around Barcelona.

We decided to make a page that had a control of the stock with a database, and for this the platform has been implemented with the laravel php framework in the back-end, and for the front-end we use HTML5 and Bootstrap.

The design of this application does not include responsiveness for mobile devices at the moment, but for tablets and computers it does. The main objective in this section for the application is that the user can navigate through different sections of the page with a good and comfortable structure of information.

That being said, this application includes two areas, one that is the visualization and another the data modification that is applied in the visualization area. It is important to know that the modification area is restricted based on the access permissions that the user has.

# ÍNDICE

<b>RESUMEN DEL PROYECTO.....</b>	<b>2</b>
<b>RESUM DEL PROJECTE.....</b>	<b>2</b>
<b>PROJECT SUMMARY.....</b>	<b>3</b>
<b>1. Especificación del sistema.....</b>	<b>7</b>
1.1 Descripción del proyecto.....	7
1.2 Situación inicial.....	7
1.3 Definición de objetivos y motivación del problema.....	7
1.3.1 Objetivos generales.....	7
1.3.2 Objetivos específicos.....	8
1.4 Enfoque y metodología.....	8
1.5 Análisis de riesgos.....	9
<b>2. Planificación del proyecto.....</b>	<b>10</b>
2.1 Distribución del tiempo.....	10
2.2 Fechas de entrega.....	10
2.3 Diagrama de Gantt.....	11
<b>3. Análisis del proyecto.....</b>	<b>12</b>
3.1 Requisitos.....	12
3.1.1 Requisitos funcionales.....	12
3.1.2 Requisitos no funcionales.....	12
3.1.3 Requisitos tecnicos.....	13
3.2 Casos de uso.....	13
3.2.1 Diagrama de casos de uso.....	13
3.2.2 Descripción de los casos de uso.....	14
<b>4. Diseño.....</b>	<b>19</b>
4.1 Arquitectura.....	19
4.2 Diagrama de clases.....	20
4.3 Diagrama de secuencias.....	21
4.4 Diagrama Entidad - Relación.....	23
4.5 Prototipo.....	24
<b>5. Implementacion.....</b>	<b>28</b>
5.1 Instalación.....	28
5.1.1 Instalación XAMPP.....	28
5.1.2 Instalación Laravel.....	28
5.1.3 Instalación Composer.....	28
5.1.4 Últimos pasos para iniciar la página.....	29
5.2 Laravel.....	30
5.2.1 Introducción Laravel.....	30
5.2.2 Base de datos.....	30
5.2.3 Patrón.....	31
5.2.3.1 Modelos.....	31
5.2.3.2 Vista.....	32

5.2.3.3 Controlador.....	32
5.2.4 Plugins.....	33
5.2.5 Enrutamiento.....	34
5.3 Implementaciones propias.....	35
<b>6. Pruebas.....</b>	<b>35</b>
<b>7. Conclusiones.....</b>	<b>36</b>
<b>8. Bibliografia y webgrafia.....</b>	<b>36</b>
<b>9. Anexos.....</b>	<b>36</b>
9.1 Manual de instalación.....	36

## ÍNDICE FIGURAS Y DIAGRAMAS

Figura 1: Fases del desarrollo del proyecto.....	8
Figura 2: Diagrama de casos de uso.....	13
Figura 3: Arquitectura.....	19
Figura 4: Diagrama de clases.....	20
Figura 5: Diagrama de secuencias “no administradores”.....	21
Figura 6: Diagrama de secuencias “administradores”.....	22
Figura 7: Diagrama entidad relación.....	23
Figura 8: Prototipo “Home”.....	24
Figura 9: Prototipo “About us”.....	25
Figura 10: Prototipo “Productos”.....	25
Figura 11: Prototipo “Carrito”.....	26
Figura 12: Prototipo “Login”.....	26
Figura 13: Prototipo “Register”.....	27
Figura 14: Prototipo “Register”.....	28
Figura 15: Instalación Xampp.....	29
Figura 16: Iniciar servidor.....	29
Figura 17: .env.....	30
Figura 18: edición de .env.....	30
Figura 19: edición de .env “db_host”.....	30
Figura 20: Comanda para crear un modelo con su archivo de migración.....	31
Figura 21: Directorio de migraciones.....	31
Figura 22: Directorio de modelos.....	31
Figura 23: Directorios de vistas.....	32
Figura 24: Directorios de controladores.....	32
Figura 25: Request Forgery.....	33
Figura 26: Views auth.....	33
Figura 27: Controllers auth.....	33
Figura 28: Las rutas aplicadas en nuestro proyecto laravel.....	34
Figura 29: Ruta de ejemplo.....	34
Figura 30: Verificación de administrador.....	35
Figura 31: pruebas.....	35

# **1. Especificación del sistema**

## **1.1 Descripción del proyecto**

INK SPOT es una innovadora empresa donde surgió el siguiente ambicioso proyecto del que se quiere modernizar haciendo que la empresa esté en la red, haciendo que cualquier persona en el mundo pueda comprar en la tienda sin necesidad de estar presencialmente. Por eso esta empresa contactó con nosotros para hacerlo.

Esta propuesta trata de poder realizar las compras que se harían presencialmente desde cualquier lugar, haciendo que sea mucho más sencillo y ágil comprar los materiales de oficina. Este proyecto tendrá funciones donde se podrán compartir productos, realizar compras masivas y tener información de cada producto al instante. También es una gran ayuda para los trabajadores en su zona de trabajo.

## **1.2 Situación inicial**

Nuestra situación antes de iniciar el desarrollo del proyecto era que no teníamos mucho conocimiento de cómo implementar las ideas de nuestros clientes de forma que se pueda reflejar en la página web, por eso decidimos buscar información de un framework cuyo objetivo sea tener una comodidad de organizar y almacenar los registros que desean tener nuestros clientes de INK SPOT S.L.L

## **1.3 Definición de objetivos y motivación del problema**

El objetivo principal que tiene este proyecto es cubrir las posibles necesidades que tendrán en la empresa. Estos nos darán una gran facilidad a la hora de aplicar nuevos productos y características en INK SPOT S.L.L.

La empresa es una nueva empresa que se encarga de vender productos de oficina. Está ubicada en Barcelona donde abren una tienda y quieren innovar haciéndola digital para poder expandirse y poder vender productos fuera de esta (En el sentido de poder vender y enviar los productos vía online por toda Barcelona).

Por este motivo es que contactaron con nosotros para poder expandirse. Nosotros facilitaremos que puedan vender estos productos mediante la página web que vamos a preparar. Es por esto que han contactado con el equipo de SUMO GP.

### **1.3.1 Objetivos generales**

El objetivo general de este proyecto es complacer las necesidades del cliente como poder comprar productos o como descargar las facturas de ciertas compras. Como empresa también podrá controlar los movimientos de los clientes.

### 1.3.2 Objetivos específicos

Creación de registros dentro de la BBDD(PHP+SQL→Laravel) como los productos, usuarios, facturas, categorías y marcas.

Eliminación de registros dentro de la BBDD(PHP+SQL→Laravel) como los productos, usuarios, facturas, categorías y marcas.

Modificación de registros dentro de la BBDD(PHP+SQL→Laravel) como los productos, usuarios, categorías y marcas.

También poder registrarse e iniciar sesión en la página como cliente y administrador (Con Framework Laravel).

Descargar las facturas registradas dentro de la BBDD(PHP+SQL→Laravel).

## 1.4 Enfoque y metodología

Nos basaremos en un método de desarrollo lineal y secuencial que es lo que cumple el modelo en cascada.



Hemos escogido este modelo porque nos ha parecido muy simple a la hora de implementarlo. Como se puede observar en la figura 1 todas las tareas están vinculadas por una dependencia, es decir que para poder avanzar en la siguiente tarea antes se debe finalizar la anterior.

**Figura 1:** Fases del desarrollo del proyecto

Antes de pasar a la siguiente tarea se realizará una comprobación de la anterior para poder identificar cualquier error que pueda desviar el desarrollo del proyecto.



Este proceso dispondrá de unos requerimientos del sistema como del propio software compatible con el cliente de INK SPOT S.L.L, seguido de un análisis a profundidad de los requisitos, un diseño adecuado al ambiente del cliente, un software desarrollado por nosotros, unas pruebas en general del proceso por la posibilidad de identificar instancias en las que se pueda descubrir errores y por último las operaciones.

## 1.5 Análisis de riesgos

Riesgo 1	
Id. Riesgo:	Ampliación/Petición de nuevas funcionalidades en el proyecto final
Descripción	El cliente quiere más funcionalidades de las acordadas en la reunión inicial
Impacto en	Todo el proyecto
Probabilidad	Baja
Decisión sobre gestión	Realizar reuniones constantes con el cliente con la intención también de peticiones de nuevas funcionalidades.

Riesgo 2	
Id. Riesgo:	Falta de tiempo por implementación de funcionalidades
Descripción	Las funcionalidades requeridas no se adecua al tiempo estimado
Impacto en	Todo el proyecto
Probabilidad	Alta
Decisión sobre gestión	Tener un buen plan de trabajo en el que tengamos claro los objetivos principales y funcionalidades obligatorias.

Riesgo 3	
Id. Riesgo:	Falta de experiencia en los campos requeridos
Descripción	Al no haber utilizado muchas de las tecnologías existe la posibilidad de no poder alcanzar lo acordado en el plan de trabajo
Impacto en	Parte del proyecto
Probabilidad	Media
Decisión sobre gestión	Realizar investigaciones extracurriculares para poder completar todos los objetivos.

## **2. Planificación del proyecto**

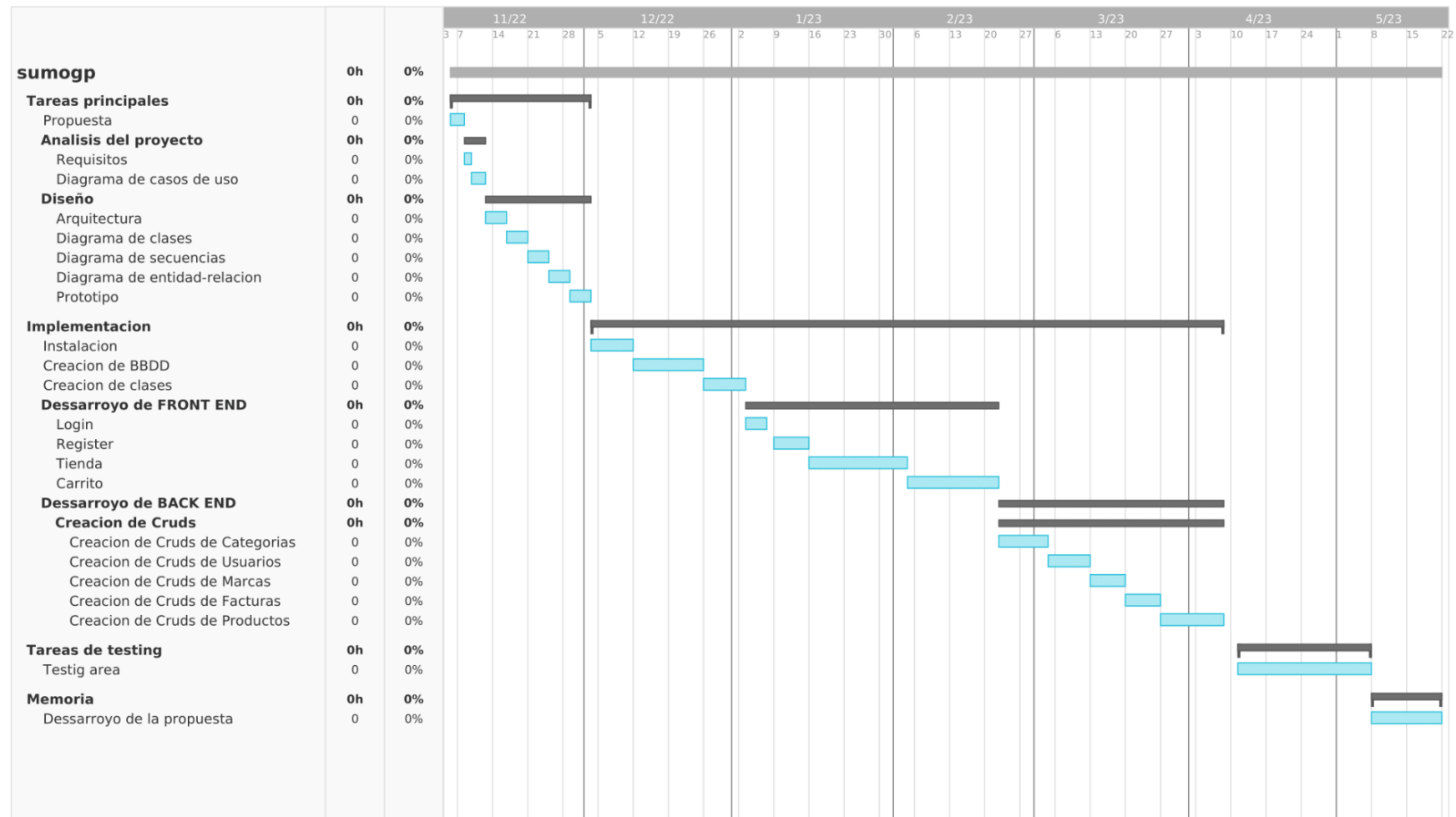
### **2.1 Distribución del tiempo**

ENTREGA	DATA INICIAL	FECHA FINAL
Planificación del proyecto	07/11/2022	15/11/2022
Análisis del proyecto	15/11/2022	23/11/2022
Diseño del proyecto	23/11/2022	02/12/2022
Implementación	02/12/2022	15/05/2023
Memoria	15/05/2023	19/05/2023
Defensa	19/05/2023	23/05/2023 i 24/05/2023

### **2.2 Fechas de entrega**

ENTREGA	FECHA
Planificación del proyecto	15/11/2022
Análisis del proyecto	23/11/2022
Diseño del proyecto	02/12/2022
Implementación	15/05/2023
Memoria	19/05/2023
Defensa	23/05/2023 i 24/05/2023

## 2.3 Diagrama de Gantt



## 3. Análisis del proyecto

### 3.1 Requisitos

#### 3.1.1 Requisitos funcionales

Estos requerimientos nos ofrecen la funcionalidad de ciertas actividades que tendrá el sistema. También decir que estos mismos tienen el deber de controlar o gestionar el comportamiento particular del sistema cuando estas condiciones se cumplen y ejecuten.

A continuación mostraremos algunos ejemplos que cumplen como requisitos funcionales en la página web de INKSPOT S.L.L

- El proceso de realización de cualquier registro, tanto sea por el admin de la página como el cliente (login y/o registro).
- La consulta a los directorios (CRUD) para la visualización de los registros de cada personal
- La edición y la visualización del perfil, ya sea por el cliente o por el personal.
- La acción de aplicar un filtro para visualizar materiales específicos que indique el usuario.
- La consulta en el listado de la compra (carrito + CRUD) del cliente.

\*El **CRUD** se basa en la creación, leer, actualizar y borrar registros, que se utilizará para referirse a las funciones básicas en nuestra base de datos.\*

#### 3.1.2 Requisitos no funcionales

Los requerimientos no funcionales se asocian a las propiedades y características del software, no con las principales funcionalidades y funciones.

- Seguridad de la página
  - ◆ La página evita ataques sql injection para evitar algún ataque contra la información de los clientes registrados.
  - ◆ Se distinguen dos páginas a nivel de vista; vista usuario y vista administrador. Ambos teniendo unas funcionalidades totalmente distintas y privilegios distinguidos.
- Rendimiento de la página
  - ◆ Implementación de técnicas para optimizar la página para que la BBDD realice la mayoría del trabajo.

### 3.1.3 Requisitos tecnicos

- El lenguaje CSS para editar el estilo de los elementos.
- El lenguaje HTML5 para la implementación de las páginas.
- El framework Laravel.
- El gestor de bases de datos MySQL.
- El lenguaje PHP para realizar las consultas de la base de datos.
- El lenguaje de programación JavaScript para implementar código jQuery

## 3.2 Casos de uso

### 3.2.1 Diagrama de casos de uso

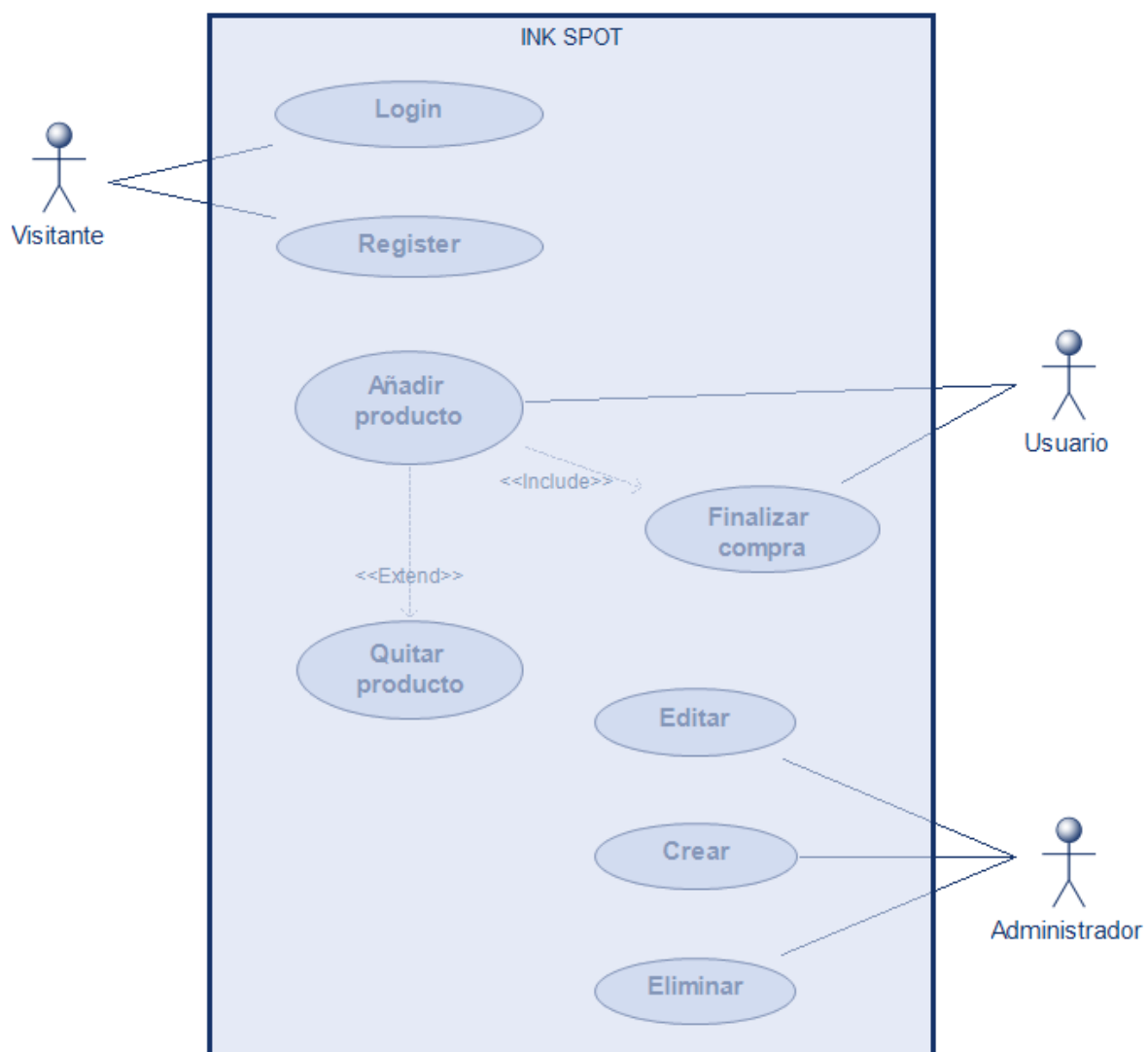


Figura 2: Diagrama de casos de uso

### 3.2.2 Descripción de los casos de uso

<b>Descripción</b>	Registrar usuario
<b>Actores</b>	Cliente con usuario registrado y servidor
<b>Precondiciones</b>	Estar no logueado, usuario, nombre, apellidos, correo electrónico, contraseña y aceptación de términos y condiciones.
<b>Flujo básico</b>	1- Inicias la pagina 2- Accedes al apartado de registro 3- Introduces los datos personales que te pide (usuario, nombre, apellidos, correo electrónico, contraseña y aceptación de términos y condiciones). 4- Servidor verifica si el formato de cada campo es correcto y crea un nuevo registro. 5- Accedes a la página ya registrado y con el acceso de loguearte por las próximas conexiones.
<b>Flujo alternativo</b>	1- El servidor detecta algún dato con un formato no compatible con su campo. 2- No accedes a la página y sale un mensaje de error
<b>Postcondiciones</b>	El usuario visualizara la página HOME ya registrado y con el acceso de loguearse por las próximas conexiones.

<b>Descripción</b>	Acceso
<b>Actores</b>	Visitante
<b>Precondiciones</b>	Usuario y contraseña
<b>Flujo básico</b>	1- Inicias la página 2- Accedes al apartado de login 3- Introduces usuario y contraseña 4- Servidor encuentra usuario en la BBDD 5- Accedes a la página ya logueado
<b>Flujo alternativo</b>	1- El servidor no encuentra al usuario en la BBDD 2- No accedes a la página y sale un mensaje de error
<b>Postcondiciones</b>	El usuario visualizara la página HOME ya logueado.

<b>Descripción</b>	Contacto
<b>Actores</b>	Visitante, usuario
<b>Precondiciones</b>	Tener mail
<b>Flujo básico</b>	1- El usuario entra en la página contacto 2- El usuario introduce sus datos 3- Introduce el mensaje 4- Envía el mensaje
<b>Flujo alternativo</b>	1- El mail introducido no es correcto 2- Problemas con la conexión
<b>Postcondiciones</b>	El mensaje se envía correctamente

<b>Descripción</b>	Añadir productos al carrito
<b>Actores</b>	Usuario
<b>Precondiciones</b>	Estar logueado
<b>Flujo básico</b>	1- Seleccionar un producto 2- Seleccionar cantidad 3- Pulsas el botón de añadir al carrito 4- El servidor guarda en la BBDD el/los producto/s seleccionado/s
<b>Flujo alternativo</b>	1- No se pudo añadir al carrito por falta de stock 2- Error de conexión
<b>Postcondiciones</b>	El material se ha añadido correctamente al carrito

<b>Descripción</b>	Quitar productos del carrito
<b>Actores</b>	Usuario
<b>Precondiciones</b>	Estar logueado y tener materiales dentro del carrito
<b>Flujo básico</b>	1- Acceder al carrito 2- Seleccionar el/los material/es a suprimir con las cantidades específicas. 3- Vaciar el carrito o sólo los materiales seleccionados clicando en la opción de suprimir y vaciar el carrito. 4- El servidor recibe la acción y suprime el/ registro/s en la BBDD.
<b>Flujo alternativo</b>	1- Error de conexión.
<b>Postcondiciones</b>	El material se ha suprimido correctamente del carrito

<b>Descripción</b>	Finalizar compra
<b>Actores</b>	Usuario
<b>Precondiciones</b>	Tener algún producto en el carrito.
<b>Flujo básico</b>	1- Entrar en tu carrito 2- Pulsar el botón de finalizar la compra 3- Es buida el carrito 4- El servidor guarda el pedido en la BBDD
<b>Flujo alternativo</b>	1- No hay productos en el carrito 2- Valores negativos 3- Error de conexión
<b>Postcondiciones</b>	El pedido se ha enviado correctamente



<b>Descripción</b>	Actualizar usuario/marca/categoría/producto.
<b>Actores</b>	Administrador
<b>Precondiciones</b>	Estar logueado
<b>Flujo básico</b>	1- Clicas sobre tu usuario y entras en la pagina de este 2- Escoges la opción correspondiente 3- Guardas los cambios 4- El servidor actualiza los cambios en la BBDD
<b>Flujo alternativo</b>	1- Los campos actualizados no son correctos 2- Problemas con la conexión
<b>Postcondiciones</b>	Recibirás un mensaje de confirmación en tu pantalla

<b>Descripción</b>	Eliminar usuario/marca/categoría/producto.
<b>Actores</b>	Administrador
<b>Precondiciones</b>	Estar logueado y motivo de ejecutar una eliminación de un usuario (tanto sea por el admin como el propio usuario)
<b>Flujo básico</b>	1- Clicas sobre tu usuario y accedes al apartado de modificar usuario. 2- Escoges la opción de eliminar usuario. 3- Se mostrará un mensaje dando a saber que eres consciente de que eliminarás al usuario permanentemente. 4- El servidor suprimirá el registro en la BBDD.
<b>Flujo alternativo</b>	1- Problemas con la conexión.
<b>Postcondiciones</b>	Se mostrara la página en el apartado HOME sin ningún usuario logueado.

<b>Descripción</b>	Añadir usuario/marca/categoría/producto.
<b>Actores</b>	Administrador
<b>Precondiciones</b>	Estar logueado
<b>Flujo básico</b>	1- Entres a l'opció de añadir usuario/marca/categoría/producto. 2- Aplicas los valores correspondientes 3- Guardas los cambios 4- El servidor guarda los cambios en la BBDD
<b>Flujo alternativo</b>	1- Los datos introducidos no son correctos 2- Problemas de conexión
<b>Postcondiciones</b>	Recibir un mensaje de confirmación

<b>Descripción</b>	Administrar personal
<b>Actores</b>	Administrador
<b>Precondiciones</b>	Ser usuario administrador
<b>Flujo básico</b>	1- Clicas en el apartado de personal 2- Selecciones a la persona que quieres actualizar 3- Guardas los cambios 4- El servidor actualiza los datos en la BBDD
<b>Flujo alternativo</b>	1- Datos introducidos incorrectos 2- Problemas de conexión
<b>Postcondiciones</b>	Recibir un mensaje de confirmación.

## 4. Diseño

### 4.1 Arquitectura

La arquitectura que utilizamos es MVC (Modelo Vista Controlador) puesto que trata de un modelo muy maduro y ya ha demostrado su validez en todo tipo de aplicaciones, y sobre todo a cantidad de lenguajes.

El modelo contiene una representación de los datos que domina el sistema, su lógica de negocio y sus mecanismos de persistencia.

La vista que cuenta la información que se envía al cliente y los mecanismos de interacción con éstos

El controlador actúa como intermediario entre modelo y vista.

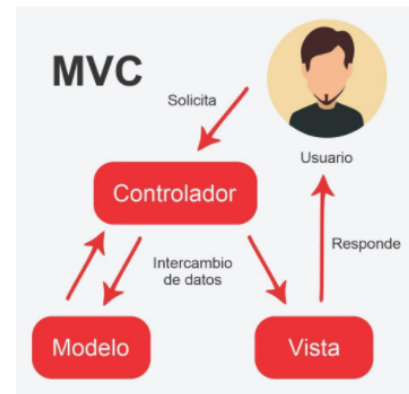


Figura 3: Arquitectura

Por eso utilizaremos el framework Laravel, ya que éste tiene esta arquitectura implementada.

## 4.2 Diagrama de clases

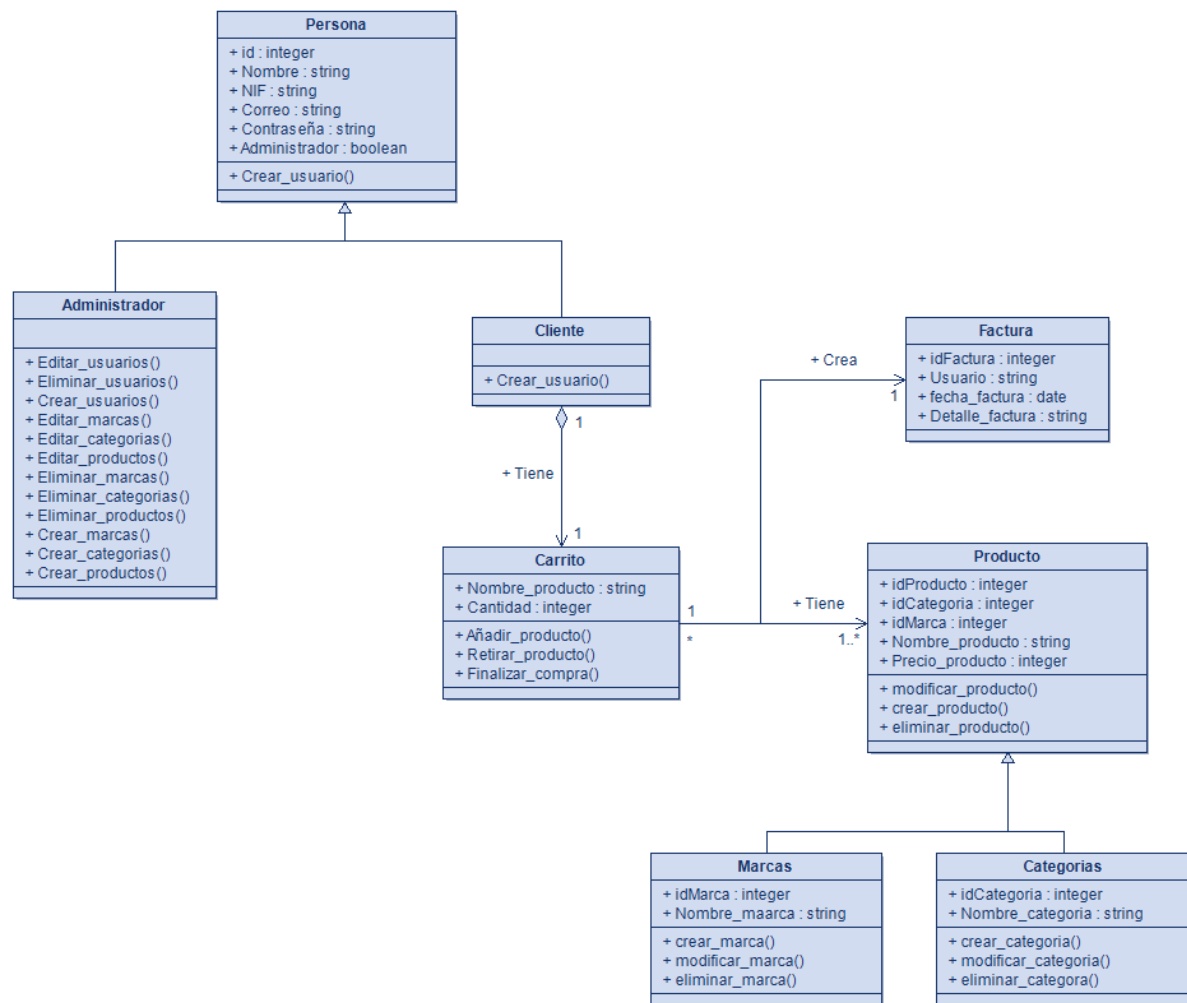


Figura 4: Diagrama de clases

## 4.3 Diagrama de secuencias

### Escenarios para los 'no administradores' :

- Usuario no registrado completará el formulario para poder registrarse (Sistema → BBDD).
- Usuario ya registrado se identificará para poder loguearse (Sistema → BBDD).
- Usuario ya registrado consultará el catálogo (incorporará búsquedas con filtros) para poder realizar la compra de algún producto y que los datos que se solicite al usuario se incorporen a la base de datos (Sistema → BBDD).
- El Usuario ya registrado consultara las facturas de su perfil de cuenta y que los datos modificados se actualizarán en la base de datos (Sistema → BBDD).

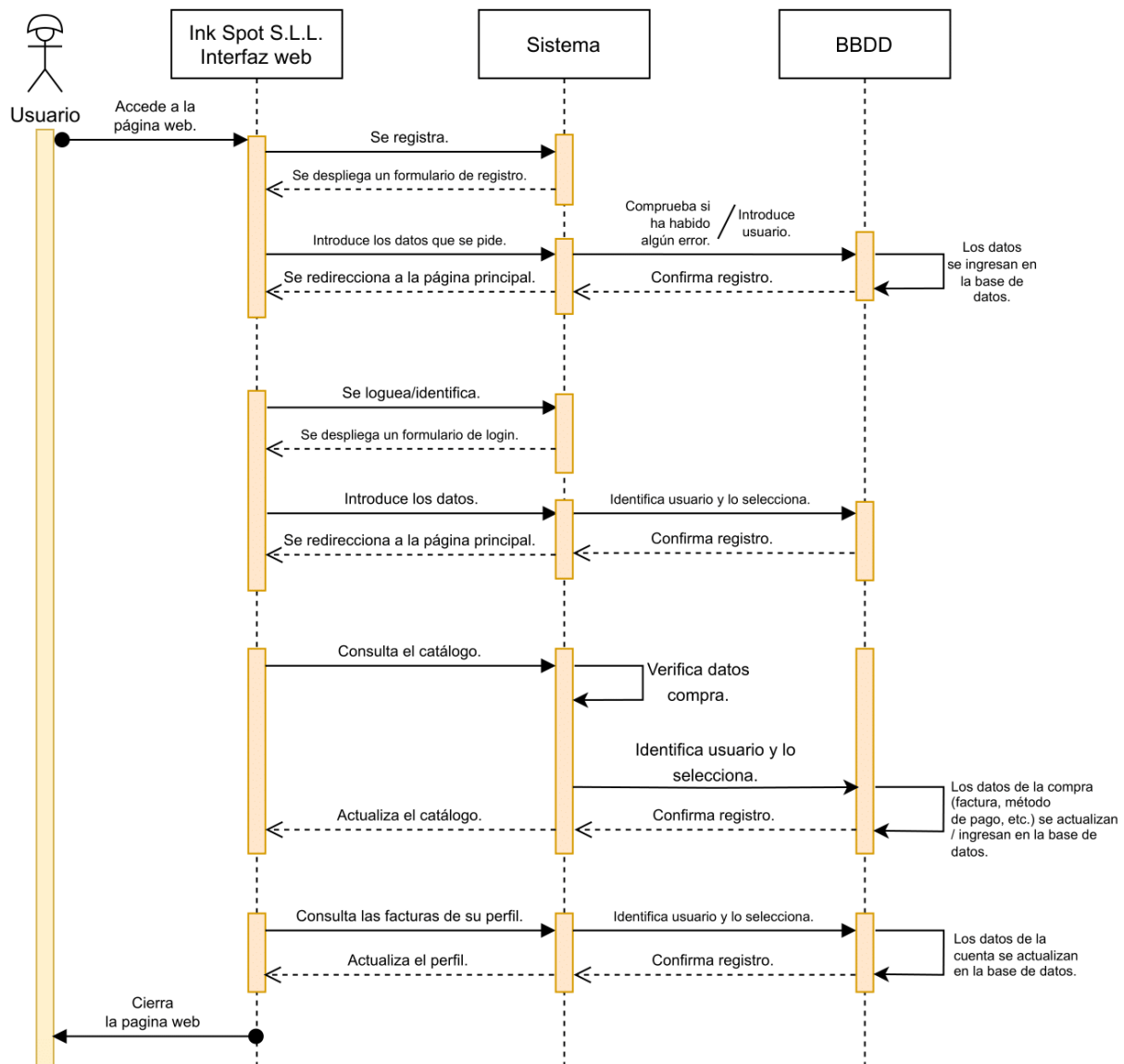
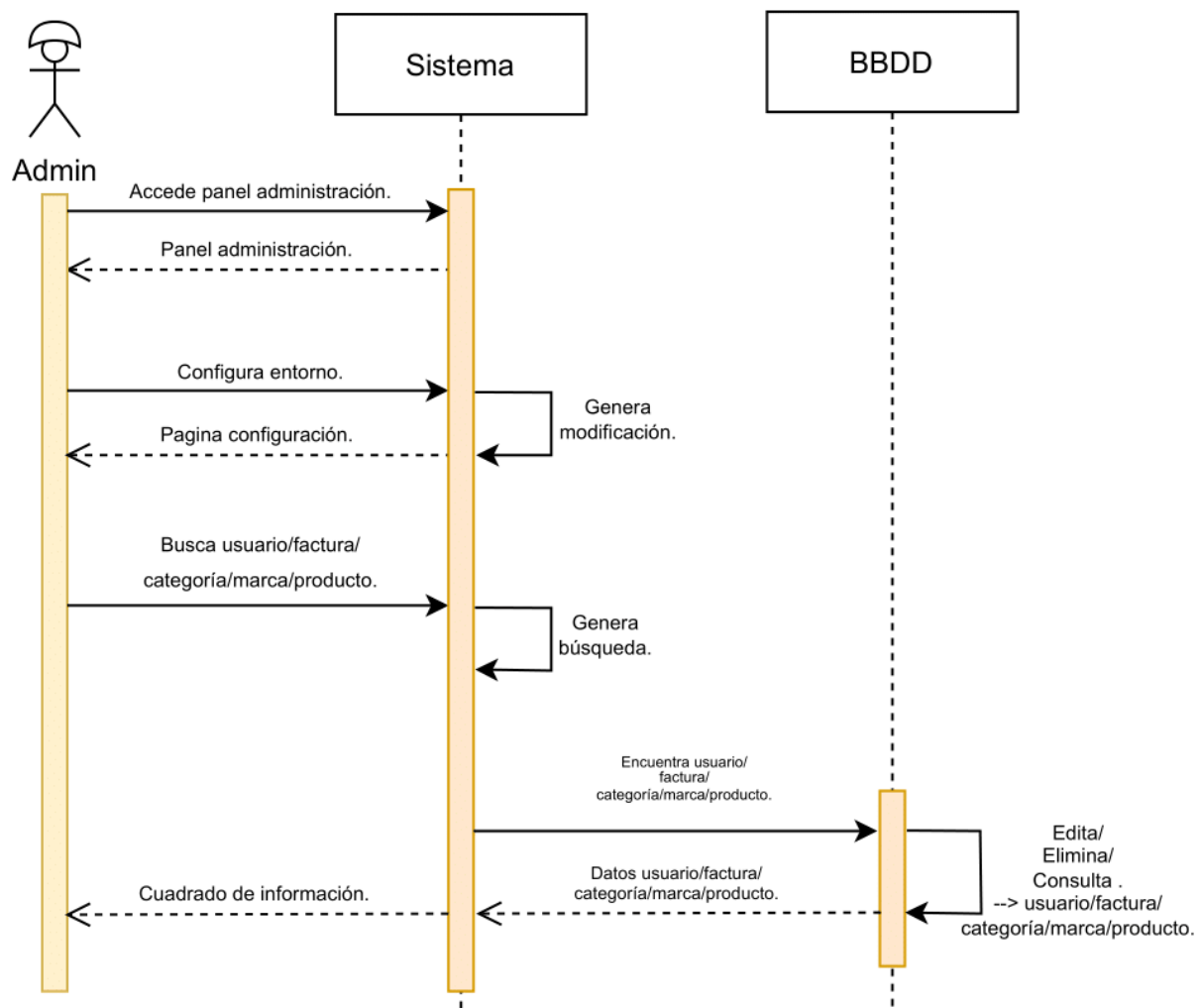


Figura 5: Diagrama de secuencias “no administradores”

**Escenarios para los ‘administradores’ :**

- Admin mitjançant el panel d'administració accedeix al sistema.
- Admin amb la verificació que s'ha fet en el panel té permís de configurar l'entorn de la web.
- Admin té dret de poder cercar usuario/marca/categoría/producto.per consultar-los, modificar-los o eliminar-los.

**Figura 6: Diagrama de secuencias “administradores”**

## 4.4 Diagrama Entidad - Relación

Con el Diagrama ER mostramos cómo es la lógica del programa.

- Un usuario tiene un carrito y un carrito solo es de un usuario.
- Un carrito puede no tener ningún producto o tener muchos, al igual que el producto puede no estar en ningún carrito como en muchos.
- Un usuario puede no tener ninguna factura o tener muchas, pero una factura solo puede tener un usuario.

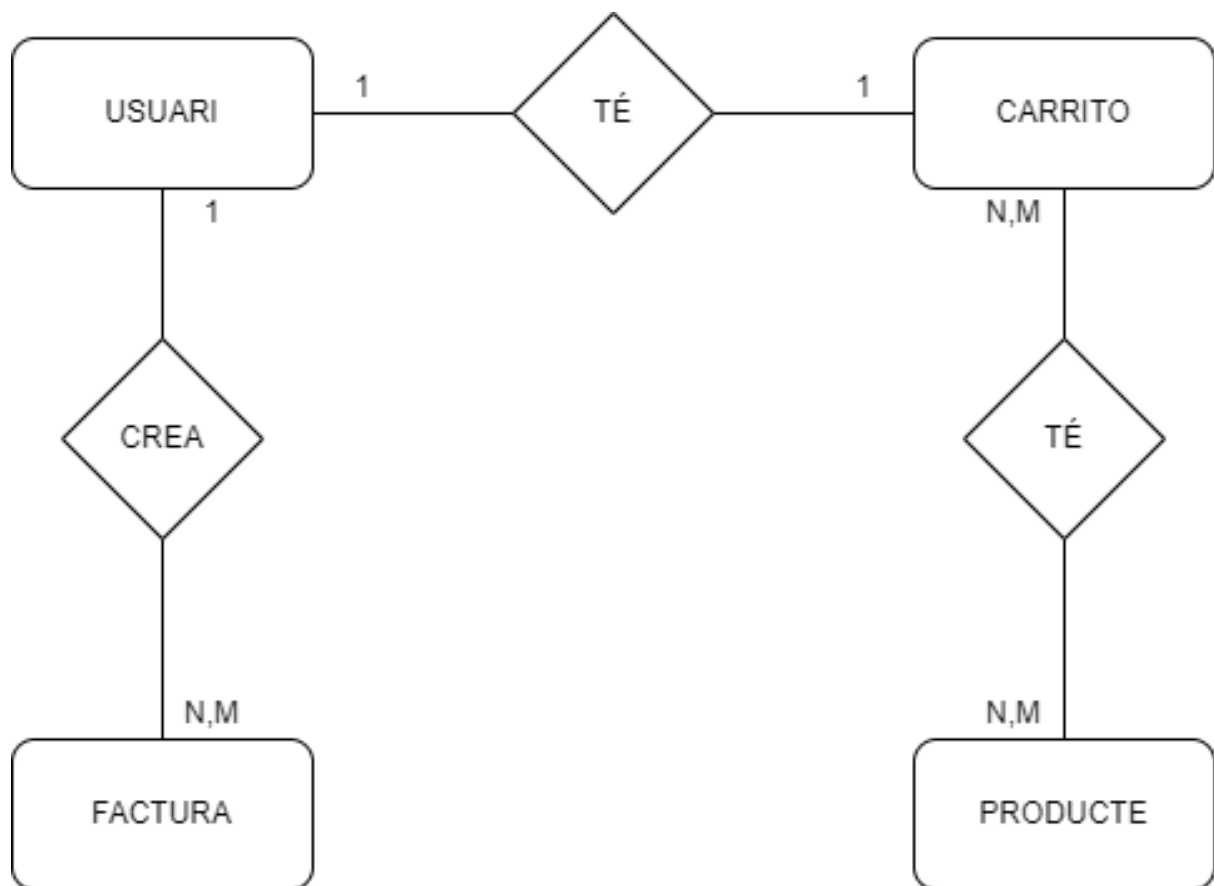


Figura 7: Diagrama entidad relación

## 4.5 Prototipo

### Home

La página principal, también denominada HOME, es la página donde se muestra la información más relevante como las ofertas o los productos más vendidos. En la parte superior podemos ver un menú para poder seleccionar la página que quieres visitar, podrás entrar con tu usuario a través de los botones de “LOGIN” y “REGISTER”. La página comienza con un scroll con fotos que ocupa parte de la pantalla. Bajo este scroll va un texto de información con fotos relacionadas con ésta.



Figura 8: Prototipo “Home”



## About us

Esta página da información de los trabajadores de la empresa para poder conocerlos. No es necesario estar logueado para poder visitarla, pero en el ejemplo que tenemos aquí sí que lo está.

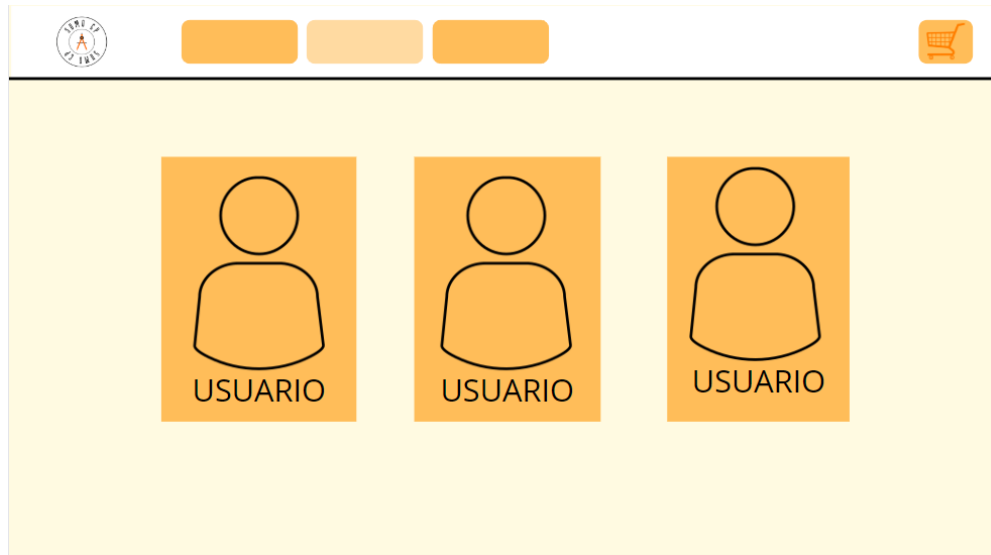


Figura 9: Prototipo “About us”

## Productos

En la siguiente página tenemos los artículos con sus respectivos precios y con su información. Tampoco hay que estar alojado para poder visitarla, pero sí estarlo para poder añadir productos al carrito.

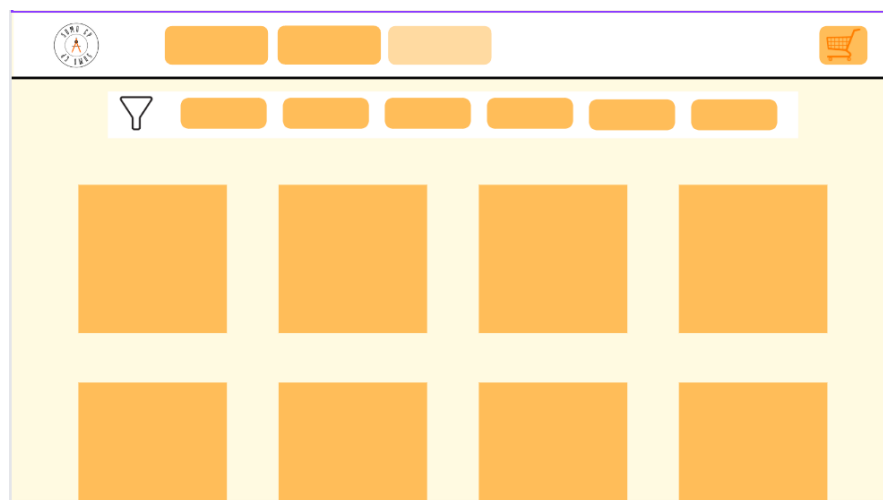


Figura 10: Prototipo “Productos”

## CARRITO

Aquí podemos ver cómo quedaría el carrito una vez que ya han introducido productos.

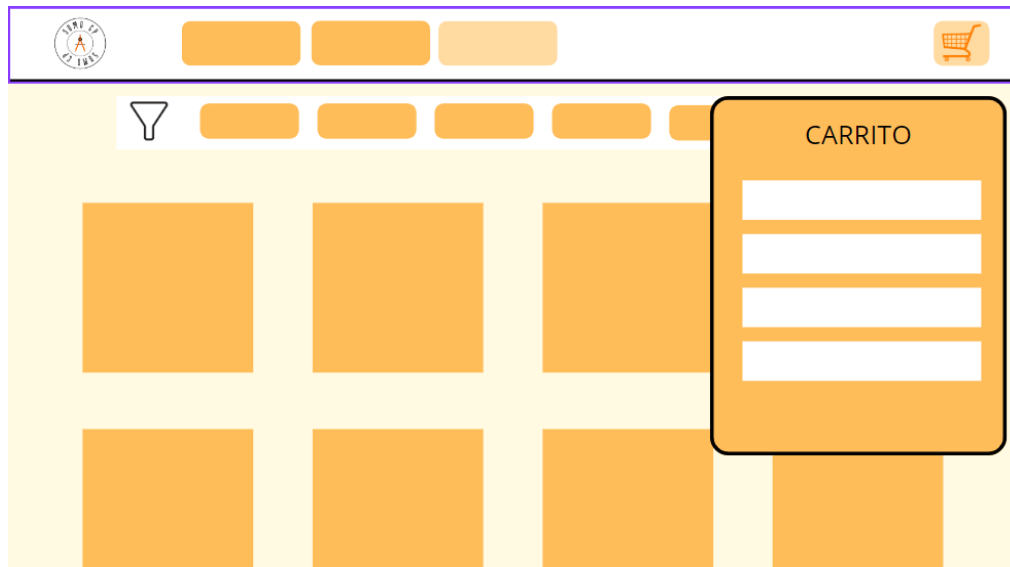


Figura 11: Prototipo “Carrito”

## LOGIN

Aquí podemos ver cómo sería la página de “LOGIN” con su respectivo formulario. También tendrá texto de información para poder ayudar al visitante.



Figura 12: Prototipo “Login”

## REGISTER

Aquí podemos ver cómo sería la página de “REGISTER” con su respectivo formulario. También tendrá texto de información para poder ayudar al visitante.



Figura 13: Prototipo “Register”

## FOOTER

Y por último tenemos el pie de página (FOOTER) que es donde saldrá información directa de las redes sociales, teléfono, novedades de la página y acceso a las condiciones generales de la empresa y la accesibilidad.

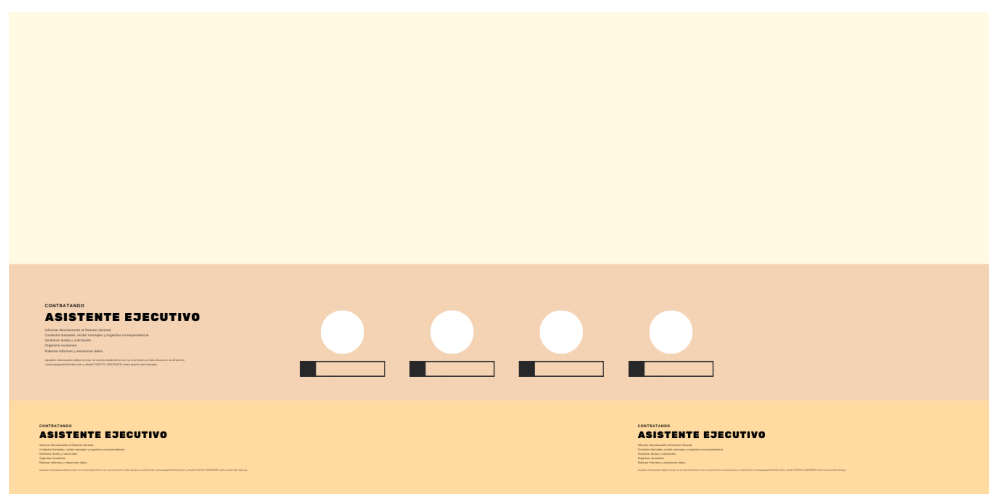


Figura 14: Prototipo "Register"

## 5. Implementacion

### 5.1 Instalación

#### 5.1.1 Instalación XAMPP

Hemos utilizado Xampp (7.4.10) para tener instalado apache y SQL ya que son requisitos para el funcionamiento de nuestro proyecto laravel en windows. Hemos utilizado PHPmyAdmin + sql para las funciones (El phpmyadmin ya viene incorporado en el xampp).

#### 5.1.2 Instalación Laravel

Primero de todo hemos utilizado Laravel 8 (8.83.27). La página web está hecha a HTML y css con la librería BOOSTRAP + La propia librería que incorpora laravel. También tenemos nuestro propio archivo css que es con lo que esta diseñado la gran parte de la web.

#### 5.1.3 Instalación Composer

Con el siguiente enlace descargamos el instalador de composer → <https://getcomposer.org/download/> (La versión que instalamos de composer es la 2.5.4)  
Instalación composer para Windows:

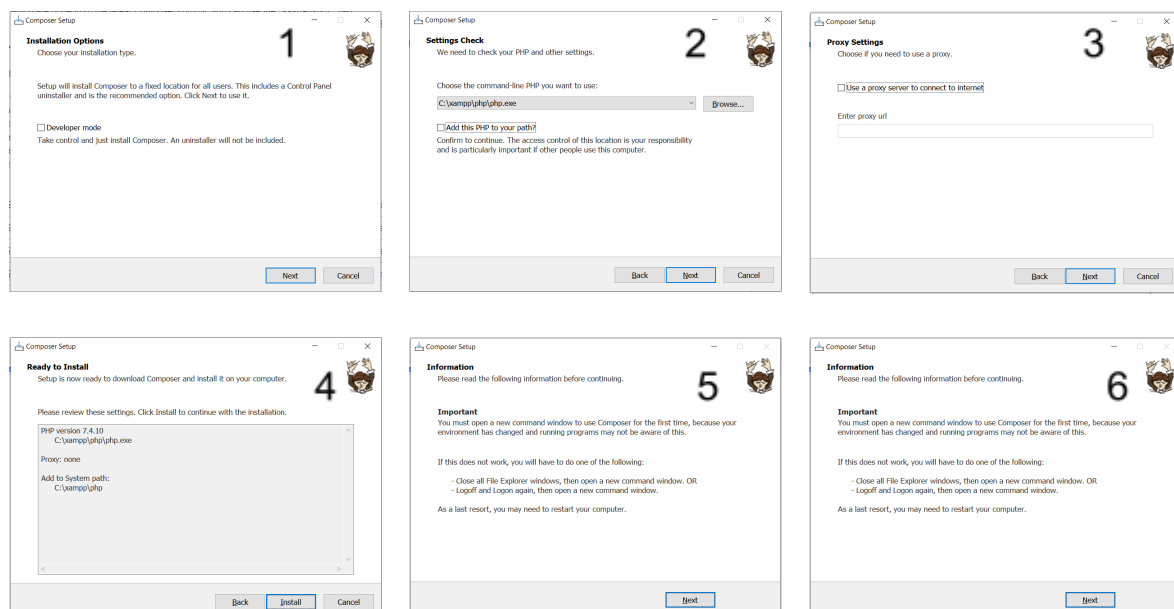


Figura 15: Instalación Xampp

### 5.1.4 Últimos pasos para iniciar la página

También instalamos node js en nuestro sistema para poder implementar node modules en nuestra carpeta laravel del proyecto (La versión de node que tenemos es la 18.15.0)

Una vez tengamos todo instalado y configurado ya podremos instalar nuestro proyecto laravel, para ello crearemos nuestro proyecto laravel via composer con estos comandos(**Debemos de estar en la carpeta htdocs de xampp**)

→ **composer global require laravel/installer** (Inicializa laravel en el htdocs)

→ **laravel new nombre\_proyecto** (Crea la carpeta de nuestro proyecto laravel en htdocs)

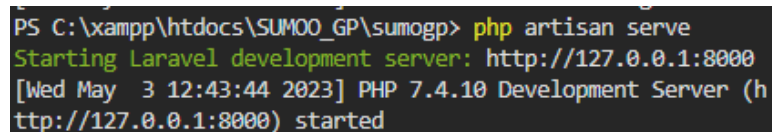
Una vez tengamos la carpeta de nuestro proyecto creada, deberemos introducir esta comanda:

→ **composer install**

y nos creara un par de archivos(carpeta vendor,etc) y debemos de crear una key en el archivo .env (la comanda para crear dicha llave ess → **php artisan key:generate**) + indicar el nombre de nuestra BBDD + la dirección + nombre y password.

En este punto ya podremos modificarlo y funcionarlo, para arrancar nuestro proyecto debemos de entrar a la carpeta de dicho proyecto en la terminal (cd .\sumogp\ ) y aplicar este comando → **php artisan serve**

y ya estara el funcionamiento,para visualizarlo en el navegador habrá que poner la dirección que se nos muestra en la terminal después de hacer la comanda →



```
PS C:\xampp\htdocs\SUMOO_GP\sumogp> php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Wed May 3 12:43:44 2023] PHP 7.4.10 Development Server (http://127.0.0.1:8000) started
```

Figura 16: Iniciar servidor

## 5.2 Laravel

### 5.2.1 Introducción Laravel

Nosotros hemos utilizado “*Laravel*” ya que nos parece una opción que facilita de cara al programador y es sencillo de entender y en nuestro caso de aprender a utilizarlo.

### 5.2.2 Base de datos

Laravel está diseñado para trabajar con bases de datos y proporciona una amplia variedad de herramientas para interactuar con la base de datos y realizar operaciones de CRUD (Crear, Leer, Actualizar, Eliminar) de manera eficiente. Laravel también proporciona una serie de herramientas para migraciones de base de datos, lo que permite a los desarrolladores mantener y actualizar la estructura de la base de datos de manera programática en lugar de hacerlo manualmente en la base de datos. Laravel utiliza el patrón **ORM (Mapeo Objeto-Relacional)** para interactuar con la base de datos, también llamado **Eloquent**.

Pero antes de hacer funcionar la conexión hay que configurar ciertas cosas

Se debe de crear el archivo `.env` que será donde indique la dirección y puerto de la BBDD



Figura 17: `.env`



Figura 18: edición de `.env`

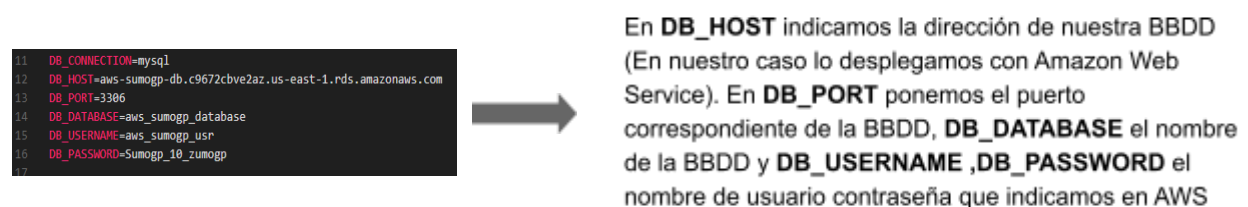


Figura 19: edición de `.env` “db\_host”

### 5.2.3 Patrón

Laravel es un framework de PHP que utiliza el patrón de arquitectura de software Modelo-Vista-Controlador (MVC) como patrón de implementación.

Este patrón divide una aplicación en tres partes principales:

**Modelo:** representa la capa de acceso a datos de la aplicación.

**Vista:** representa la capa de presentación de la aplicación.

**Controlador:** representa la capa que maneja la lógica de la aplicación.

#### 5.2.3.1 Modelos

```
PS C:\xampp\htdocs\SUMOO_GP\sumogp> php artisan make:model Facturas --m
```

Figura 20: Comanda para crear un modelo con su archivo de migración

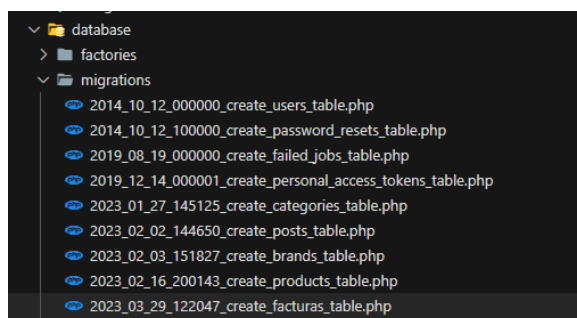


Figura 21: Directorio de migraciones

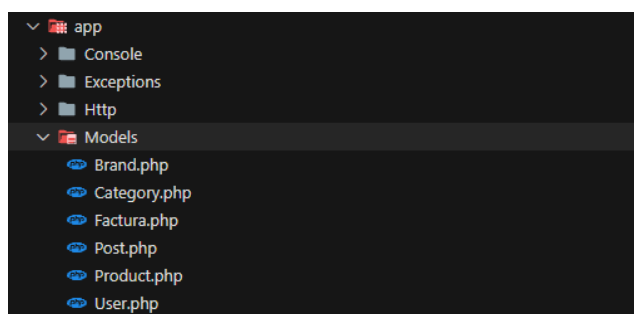
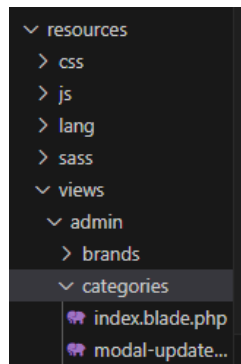


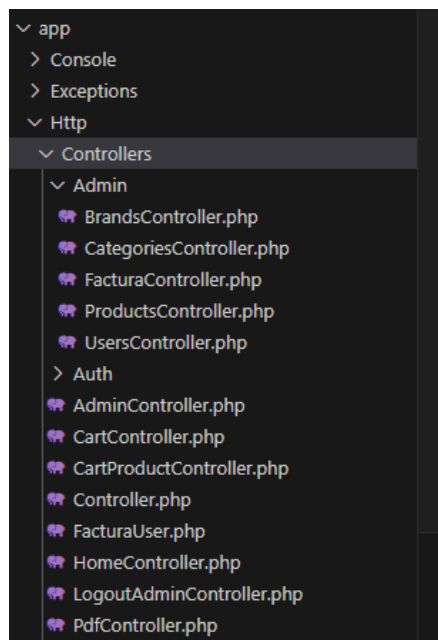
Figura 22: Directorio de modelos

### 5.2.3.2 Vista



**Figura 23:** Directorios de vistas

### 5.2.3.3 Controlador




**Figura 24:** Directorios de controladores



## 5.2.4 Plugins

Laravel ofrece por defecto plugins como “**Auth**” que lo hemos utilizado para hacer el “*Login*” ya que proporciona un sistema de autenticación preconstruido y fácil de implementar en tu aplicación. Este plugin incluye rutas, controladores y vistas predefinidas para la autenticación, lo que significa que puedes agregar fácilmente funcionalidad de inicio de sesión, registro, y cierre de sesión a tu aplicación sin tener que escribir todo el código desde cero.

Laravel también incluye características de seguridad adicionales, como protección contra ataques de SQL Inyección y protección contra **CSRF (Cross-Site Request Forgery)**, lo que hace que sea fácil para los desarrolladores implementar una solución de autenticación segura en su aplicación.



```
</div>
<form action="{{ route('admin.products.store') }}" method="POST" enctype="multipart/form-data">
  {{ csrf_field() }}
  <div class="modal-body">
    <div class="form-group">
```

Figura 25: Request Forgery

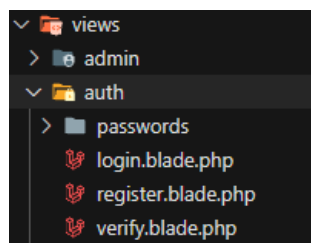


Figura 26: Views auth

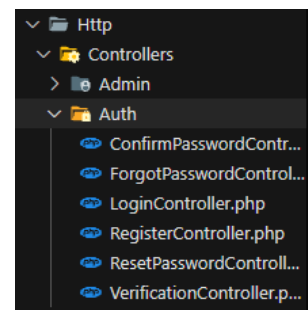


Figura 27: Controllers auth

## 5.2.5 Enrutamiento

Laravel utiliza un sistema de enrutamiento simple pero seguro que le permite definir rutas utilizando una sintaxis clara y concisa. Cuando un usuario realiza una solicitud a la aplicación, el sistema de enrutamiento de Laravel determina qué controlador y qué método deben manejar la solicitud. Esto hará que nuestra aplicación sea organizada, escalable y fácil de mantener, ya que cada ruta está asociada a un controlador y un método específicos que manejan esa solicitud en particular.

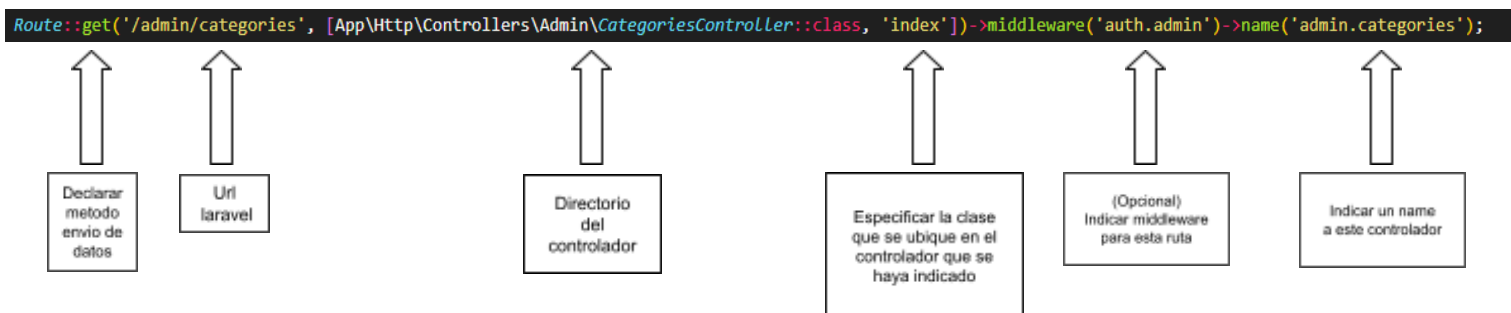
Ubicación del archivo para declarar rutas en laravel

```

85 // web.php
86 //Middleware Auth Admin(Verificación login administrador y función para autorizar la ruta de acceso)
87 Route::get('/admin', [AdminController::class, 'index'])
88     ->middleware('auth.admin')
89     ->name('admin.index');
90
91
92 //Función para mostrar el home de panel admin, solo se ejecutara esta ruta en la función index de la ruta anterior teniendo el middleware Auth Admin y no se podra ir directamente a esta
93 Route::get('/home', function(){
94     return view('admin.home_admin');
95 })->middleware('auth.admin');
96
97
98 //Panel ADMIN + CRUD Categorías//
99 Route::get('/admin/categories', [App\Http\Controllers\Admin\CategoriesController::class, 'index'])->middleware('auth.admin')->name('admin.categories');
100 Route::post('/admin/categories/store', [App\Http\Controllers\Admin\CategoriesController::class, 'store'])->middleware('auth.admin')->name('admin.categories.store');
101 Route::post('/admin/categories/{categoryId}/update', [App\Http\Controllers\Admin\CategoriesController::class, 'update'])->middleware('auth.admin')->name('admin.categories.update');
102 Route::delete('/admin/categories/{categoryId}/delete', [App\Http\Controllers\Admin\CategoriesController::class, 'delete'])->middleware('auth.admin')->name('admin.categories.delete');
103
104
105
106
107 //Panel ADMIN + CRUD Productos//
108 Route::get('/admin/products', [App\Http\Controllers\Admin\ProductsController::class, 'index'])->middleware('auth.admin')->name('admin.products');
109 Route::post('/admin/products/store', [App\Http\Controllers\Admin\ProductsController::class, 'store'])->middleware('auth.admin')->name('admin.products.store');
110 Route::post('/admin/products/{productId}/update', [App\Http\Controllers\Admin\ProductsController::class, 'update'])->middleware('auth.admin')->name('admin.products.update');
111 Route::delete('/admin/products/{productId}/delete', [App\Http\Controllers\Admin\ProductsController::class, 'delete'])->middleware('auth.admin')->name('admin.products.delete');
112
113
114
115
116 //Panel ADMIN + CRUD Marcas//
117 Route::get('/admin/brands', [App\Http\Controllers\Admin\BrandsController::class, 'index'])->middleware('auth.admin')->name('admin.brands');
118 Route::post('/admin/brands/store', [App\Http\Controllers\Admin\BrandsController::class, 'store'])->middleware('auth.admin')->name('admin.brands.store');
119 Route::post('/admin/brands/{brandId}/update', [App\Http\Controllers\Admin\BrandsController::class, 'update'])->middleware('auth.admin')->name('admin.brands.update');
120 Route::delete('/admin/brands/{brandId}/delete', [App\Http\Controllers\Admin\BrandsController::class, 'delete'])->middleware('auth.admin')->name('admin.brands.delete');
121
122
123
124
125 //Panel ADMIN + CRUD Users//
126 Route::get('/admin/users', [App\Http\Controllers\Admin\UsersController::class, 'index'])->middleware('auth.admin')->name('admin.users');
127 Route::post('/admin/users/store', [App\Http\Controllers\Admin\UsersController::class, 'store'])->middleware('auth.admin')->name('admin.users.store');
128 Route::post('/admin/users/{userId}/update', [App\Http\Controllers\Admin\UsersController::class, 'update'])->middleware('auth.admin')->name('admin.users.update');
129 Route::delete('/admin/users/{userId}/delete', [App\Http\Controllers\Admin\UsersController::class, 'delete'])->middleware('auth.admin')->name('admin.users.delete');
  
```

Figura 28: Las rutas aplicadas en nuestro proyecto laravel

Figura 29: Ruta de ejemplo



## 5.3 Implementaciones propias

Igualmente hemos hecho implementaciones propias como es el caso de los dos tipos de usuarios como son el usuario normal y el administrador. El usuario normal puede comprar en la tienda y mirar sus facturas mientras que el administrador puede acceder al back end donde puede ver la información, editarla y además puede visitar la tienda y hacer compras para hacer testeos.

```

87 //Middleware Auth Admin(Verificacion login adminstrador) + funcion para introducir la route de abajo
88 Route::get('/admin', [AdminController::class, 'index'])
89     ->middleware('auth.admin')
90     ->name('admin.index');
91
92

```

Figura 30: Verificación de administrador

## 6. Pruebas

Entregamos la aplicación web a nuestros clientes para que hicieran pruebas y hasta la fecha no hemos recibido queja alguna.

5	Ink	Spot S.L.L	inkspots186@gmail.com		<a href="#">Editar</a>
					<a href="#">Eliminar</a>
6	Anass	Benzian	aben22@insdanielblanxart.cat		<a href="#">Editar</a>
					<a href="#">Eliminar</a>
ID	Nombre	Apellido	Email	Role	Acciones

Mostrando página 1 de 1

Anterior **1** Siguiente

5	6	Anass	2023-05-19 03:55:47	<a href="#">Descargar</a>
				<a href="#">Eliminar</a>
6	6	Anass	2023-05-19 03:56:57	<a href="#">Descargar</a>
				<a href="#">Eliminar</a>

Figura 31: pruebas

## **7. Conclusiones**

Este proyecto nos ha servido para aprender a utilizar el framework laravel y a perfeccionar nuestro dominio con bootstrap. También nos ha servido para mejorar el desarrollo de nuestro conocimiento sobre las bases de datos en entorno php.

También hemos aprendido a utilizar AWS (Amazon Web Service) para desplegar la página web.

## **8. Bibliografía y webgrafia**

[Youdevs](#) → Hemos sacado información de laravel en este canal para poder desarrollar nuestro proyecto.

## **9. Anexos**

### **9.1 Manual de instalación**

[Manual de instalación de App Web de venta de materiales de oficina](#)