



# MEMORIA DEL PROYECTO

## TUSEGUNDAROPA.COM

**Autor:** Aitor Darío Serna Gamayo

**Tutor:** Felicidad García

**CSGS:** Desarrollo de Aplicaciones Web



**GENERALITAT VALENCIANA**  
CONSELLERIA DE CULTURA, EDUCACIÓ I ESPORT



**Unión Europea**  
Fondo Social Europeo  
El FSE invierte en tu futuro

**I. E. S. MARE NOSTRUM**

Beato Fco. Castelló Aleu s/n • ☎

965936520

03008 ALICANTE

[www.iesmarenostrum.com](http://www.iesmarenostrum.com)

[correo@iesmarenostrum.com](mailto:correo@iesmarenostrum.com)



## Índice

<b>1. INTRODUCCIÓN .....</b>	<b>2</b>
1.1 Breve descripción del proyecto .....	2
1.2 Exposición de los objetivos que se persiguen .....	3
<b>2. ANÁLISIS .....</b>	<b>4</b>
2.1 Identificación y necesidades del sector productivo .....	4
2.2 Análisis de la competencia, tanto a nivel local como en internet .....	6
2.3 Actividad de la empresa y su ubicación en el sector de desarrollo de software .....	6
2.4 Viabilidad técnica y oportunidad de proyecto .....	7
2.5 Características del proyecto: pliego de condiciones. ....	9
2.6 Tecnologías a emplear .....	10
<b>3. DISEÑO .....</b>	<b>12</b>
3.1 Estructura general del proyecto. Diagrama de casos de uso .....	12
3.2 Elaboración de un guion de trabajo para el desarrollo del proyecto. ....	16
3.3 Fases del proyecto. Tareas y plazos de ejecución. ....	17
3.4 Recursos materiales y personales .....	22
3.5 Estimación de gastos .....	24
3.6 Estimación de ingresos y precios de venta .....	25
3.7 Viabilidad económica.....	26
3.8 Necesidades de financiación.....	31
3.9 Definición y elaboración de la documentación del diseño. Control de calidad.....	32
3.10 Revisión normativa aplicable .....	38
<b>4. IMPLEMENTACIÓN .....</b>	<b>39</b>
4.1 Desarrollo del modelo de datos.....	39
4.2 Desarrollo de la programación del entorno de servidor .....	44
4.3 Desarrollo de la programación del entorno de cliente .....	48
4.4 Panel de PowerBI.....	50
4.5 Despliegue en entorno cloud.....	51
<b>5. VALIDACIÓN .....</b>	<b>53</b>
5.1 Definición del procedimiento de evaluación, seguimiento y control del proyecto. Indicadores de calidad.....	53
5.2 Definición de procedimientos para la participación de los usuarios en la evaluación del proyecto. Documentos específicos.....	56
5.3 Registro de resultados. ....	59
<b>6. CONCLUSIONES.....</b>	<b>62</b>
<b>7. BIBLIOGRAFIA Y REFERENCIAS .....</b>	<b>62</b>

## 1. INTRODUCCIÓN

### 1.1 Breve descripción del proyecto

La idea de la creación de este proyecto de venta de segunda mano online nace de la situación vivida en la comarca de la Vega Baja del Segura en la última década. Durante años, muchísimas familias han vivido tradicionalmente de la venta de ropa en los mercadillos ambulantes o en tiendas físicas. Pero debido a la desindustrialización del arco mediterráneo, la crisis económica, la creación de grandes centros comerciales, posteriormente la pandemia y el aumento de la venta online, muchas familias han visto como sus negocios dejaban de ser rentables, llevando a muchas a la ruina y la desesperación por perder el sustento.

De lo expuesto anteriormente nace la idea de crear “TuSegundaRopa.com” una aplicación web enfocada a facilitar a los negocios físicos de venta de ropa de segunda mano una transición ágil y rápida al mercado online, para así poder aprovechar la gran bolsa de potenciales clientes que existen en internet. Nuestra aplicación pretende ser un portal atractivo al usuario final, adaptando la vista de la web para cada uno de nuestros clientes con su propia marca, sencillo de comprender y además con servicios añadidos a nuestro software que consistirán en mantenimiento del catálogo de artículos en la web y análisis de los datos de los clientes para poder aplicar mejoras en su negocio en base a sus propios datos de su clientela real.

Otra de las virtudes que tendrá nuestro proyecto y no menos importante para nosotros, es que es un software que va enfocado a un modelo de negocio (la venta de ropa de segunda mano) que promueve la protección del medioambiente a través de la reutilización de prendas textiles. Además, reduce la generación de residuos y contribuye a la lucha contra el cambio climático. El reciclaje textil durante el año pasado representó un ahorro de 51.500 toneladas de dióxido de carbono a la atmósfera. Por cada kilo de ropa recuperado se dejan de emitir 3,169 kg de CO<sub>2</sub>.

Según un estudio realizado por la empresa ThredUp, este mercado crecerá casi dos veces más que el mercado de venta de ropa de “primera mano” y estima que las ventas online de segunda mano aumentarán durante este año casi en un 69%.

Por lo tanto, creemos firmemente que este proyecto de software para comercio online de ropa de segunda mano es una oportunidad de transformación y de avance para un negocio tradicional, el cual va a ofrecer al pequeño comerciante un nuevo mundo de oportunidades a través de Internet.



## 1.2 Exposición de los objetivos que se persiguen

Los objetivos que se persiguen con este proyecto son:

1. Crear una aplicación web de calidad con las funcionalidades necesarias que satisfaga las necesidades de los negocios que confíen en nosotros, la aplicación debe tener las funcionalidades básicas de:
  - El usuario debe poder registrarse y visualizar los productos, comprar productos y publicar un anuncio de un producto deseado que no esté disponible en la web.
  - Una sección con productos con un precio más elevado y que el exceso de precio vaya a una fundación benéfica a elegir por la empresa
  - Panel de administración para gestionar el stock.
2. Ser capaces de aportar valor añadido a la aplicación con los siguientes servicios extra:
  - Mantenimiento de artículos en stock en la web.
  - Análisis de datos para mejorar el negocio del cliente con panel de PowerBI.
  - Corrección de posible fallos o vulnerabilidades futuras a cargo nuestra empresa.
3. Analizar y mostrar la viabilidad económica y técnica del proyecto.
4. Promover a través de nuestro software un modelo de negocio sostenible como es el de la venta de ropa de segunda mano.



## 2. ANÁLISIS

### 2.1 Identificación y necesidades del sector productivo

Actualmente el crecimiento de las ventas online está en clara tendencia al alza ya que este tipo de compras se ha extendido por su facilidad al usuario tanto para comprar como para vender. Nuestra idea de software encaja perfectamente en el marco de negocio que se vive actualmente ya que creemos que al estar en una época en la cual el comercio online está completamente en expansión, podremos aprovechar y establecernos en nuestro propio hueco dentro del sector, pudiendo proporcionar a través de nuestra aplicación web soluciones a pequeños negocios para poder abrirse al nuevo mercado mundial online.

Como podemos apreciar en el siguiente gráfico de un estudio publicado en la web, las ventas online en España solo han hecho que aumentar desde el año 2016.

**EVOLUCIÓN TRIMESTRAL DEL VOLUMEN DE NEGOCIO DEL COMERCIO ELECTRÓNICO Y VARIACIÓN INTERANUAL (millones de euros y porcentaje)**



Fuente: CNMC

Además, respecto al mercado concreto de la ropa de segunda mano hay datos también muy positivos que nos indican esta nueva realidad al alza, en el informe de Thredup (*"2021 Resale Report"*) también se habla de que las ventas aumentarán en los próximos 5 años, llegando a alcanzar los 77.000 millones de dólares. En este otro estudio publicado recientemente por eMarketer, empresa de investigación de mercado, que pronostica un aumento del 132,4% en las ventas promedio anual de ropa de segunda mano online por comprador para el 2025, alcanzando los 788,38 dólares. En 2021 las ventas de este sector fueron de 339,18 dólares.

Fuente [El mercado de ropa de segunda mano comienza a encontrar su hueco \(ecommerce-news.es\)](https://ecommerce-news.es/)



Como podemos observar dentro del siguiente gráfico, la rama de actividad que más volumen de negocio genera es la ropa, en nuestro caso, nuestra herramienta está enfocada a la venta de ropa de segunda mano e iría incluida en esta rama.

**LAS DIEZ RAMAS DE ACTIVIDAD CON MAYOR PORCENTAJE DE VOLUMEN DE NEGOCIO DEL COMERCIO ELECTRÓNICO (I-21, porcentaje)**



Fuente: CNMC

Por lo tanto y como opinión personal, la realización de este proyecto tiene aparentemente, con los datos en la mano, un futuro prometedor ya que hoy en día todo el mundo tiene en casa un dispositivo desde el cual puede acceder a miles de tiendas y puede gastar dinero sin moverse del salón de su hogar, lo cual es una gran ventaja para los comercios ya que pueden llegar a millones de hogares si plantean bien su modelo de negocio.



## 2.2 Análisis de la competencia, tanto a nivel local como en internet

Vamos a analizar la competencia que habría para nuestro modelo de negocio tanto a nivel local como en internet.

A nivel local, dado que vivimos en una zona muy cerca de Alicante, Elche y Murcia, hemos encontrado bastantes tiendas físicas de ropa de segunda mano y almacenes al por mayor. Esto lejos de verlo como una amenaza lo vemos como una oportunidad de crear futuros clientes para que puedan implantar nuestra aplicación web y puedan lanzarse a los brazos del e-commerce, en caso de que ya tuvieran una plataforma o web para vender sus productos intentaríamos convencerlos explicándoles las bondades de nuestra aplicación y los servicios extra que podríamos ofrecerles.

En cuanto a la competencia en internet, encontramos bastantes webs que se dedican a la compra-venta de ropa de segunda mano.

Casi todas las webs lo que hacen es ofrecer un portal para que la gente pueda vender su ropa usada y a su vez que cualquier persona pueda comprarla, quedándose la empresa que proporciona el portal una comisión por cada venta que se realiza en su plataforma. El porcentaje oscilaría dependiendo del precio de la prenda. Esto se aleja de lo que nuestra idea de negocio pretende ya que pretendemos crear tiendas para empresas individuales y no portales de compra-venta.

Los portales de compra-venta más famosos son Vinted, Ropasion, Micolet y Wallapop.

Por lo tanto, no nos fijaremos demasiado en la competencia y todos nuestros esfuerzos irán a que cuatro puntos clave se cumplan:

1. Precios competitivos de nuestros servicios a las empresas cliente.
2. Portal web cuidado y que resulte una experiencia agradable al usuario navegar por él.
3. Servicios post implantación de calidad.
4. Atención a los clientes personalizada como prioridad.

## 2.3 Actividad de la empresa y su ubicación en el sector de desarrollo de software

El negocio podría constituirse en principio de varias maneras, pero dado que el proyecto está comenzando y aun no tendríamos una cartera de clientes asentada, planteamos la creación de nuestra empresa de software en el modelo de autónomo con un único dueño, siendo el dueño uno de los desarrolladores, por las siguientes razones:

- Los tramites y costes de constitución son más baratos que una SL.
- Los costes de gestión son más bajos ya que la contabilidad es más fácil de llevar. Además, desde el 1 de enero de 2019, aplicándose la tarifa plana, durante el primer año la cuota de autónomos se queda en 60 euros mensuales los 12 primeros meses y, durante el segundo año disfrutas de bonificaciones del 50 y el 30%.
- No necesita tener una aportación de capital inicial obligatorio
- Se tributa por el IRPF, que es un impuesto progresivo. (Si los beneficios aumentan mucho ya plantearíamos el paso a SL ya que el IRPF que tuviéramos que pagar podría superar al tipo fijo que pagan las SL del 25%)

Esta podría ser una propuesta de logotipo de la empresa la cual nos daría una imagen más corporativa y una seña de identidad en el mercado:



Dentro del sector del desarrollo de software quedaríamos dentro del espectro de empresas que desarrollan aplicaciones web para comercio electrónico, como ya hemos explicado anteriormente un modelo de negocio en expansión y que nos podría permitir en un futuro avanzar hacia otro tipo de tiendas ya que está en aclaro crecimiento.

#### 2.4 Viabilidad técnica y oportunidad de proyecto

La viabilidad técnica de este proyecto no debería suponer un problema ya que hoy en día tenemos a nuestra disposición multitud de herramientas para poder desarrollar software sin tener que desembolsar un gasto excesivo. Para la creación del proyecto, aparte de la parte de hardware totalmente necesaria, vamos a utilizar distintas herramientas de desarrollo tales como editores de texto, software de control de versiones, navegadores, gestor de base de datos, frameworks para el front-end como el back-end y además de alojamiento para los distintos proyectos de los clientes. Todo esto lo detallaremos más adelante y tendremos en cuenta el gasto que va a suponer en la parte de viabilidad económica.

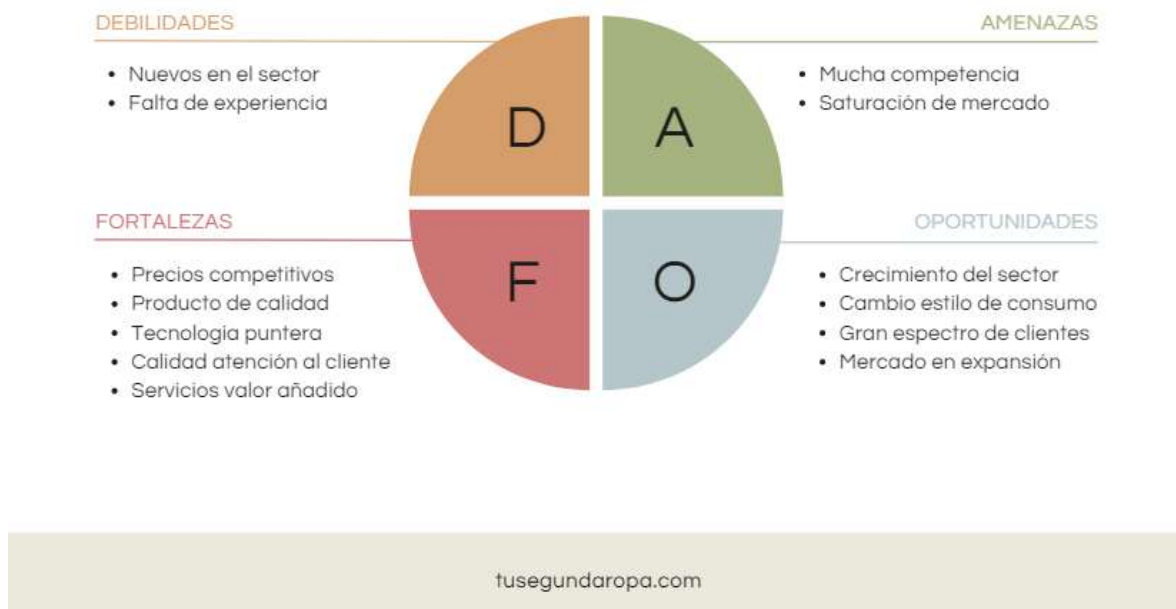
Por lo tanto, no suponiendo un problema la parte de hardware y software, durante estos dos años hemos adquirido los conocimientos necesarios para poder desarrollar la aplicación web de este proyecto y estamos convencidos de que se puede llevar a buen puerto.





Respecto a la parte de oportunidad de proyecto, ya hemos explicado anteriormente el por qué supone una oportunidad la creación e implantación a nivel de modelo de negocio, pero vamos a tratar de mostrar con un análisis DAFO cuales creemos que podrían ser de manera objetiva las debilidades, amenazas, fortalezas y oportunidades de nuestro proyecto.

## ANALISIS DAFO



Una vez realizado nuestro análisis DAFO, deberíamos intentar aplicar el concepto CAME, como contestación a nuestro DAFO, cuyas siglas nos dicen:

- **CORREGIR** nuestras debilidades
- **AFRONTAR** las amenazas del entorno
- **MANTENER** las fortalezas de las que ya disponemos
- **EXPLOTAR** las oportunidades que nos ofrece nuestro entorno

## 2.5 Características del proyecto: pliego de condiciones.

Todo nuestro proyecto se basa en ofrecer a empresas una aplicación web para que puedan desarrollar su actividad de negocio, en este caso venta de ropa de segunda mano, a través de internet de una manera fácil y sin quebraderos de cabeza. Por lo tanto, en nuestro proyecto va a estar incluida tanto la creación del software como la posterior implantación en Azure.

Para nosotros la infraestructura de hosting pensada será un servicio en Azure PaaS (Platform as a Service) en el cual tendremos contratado un "Service Plan" dentro del cual, podremos alojar todas las "web apps" de nuestros clientes y a su vez, tener enlazadas estas "web apps" a una instancia de BBDD SQL en la cual dentro estén cada una de las BBDD de cada una de las web apps. Esto nos va a permitir centrarnos en el desarrollo y evolución del software y olvidarnos del mantenimiento de la infraestructura, sistemas operativos etc. Otra de las ventajas que nos ofrece trabajar con Paas es su gran flexibilidad ya que podemos comenzar con unos planes más económicos y a medida que nuestra cartera de clientes aumente podemos escalar nuestros recursos tanto el "Service plan" como la capacidad de la instancia de BBDD.



Esta infraestructura para el cliente será transparente ya que para nuestros clientes (empresas que nos paguen por desplegarles nuestro software) el servicio será de tipo SaaS (Software as a Service), lo que significa que ellos no van a tener que preocuparse de infraestructura ni del código o el software desplegado en la plataforma, interactuarán con el software a través de un cliente (navegador) como si de un servicio más se tratase.








Respecto a los tipos de usuarios que usarán nuestro software, podemos clasificarlos de la siguiente manera:







- **Usuarios No registrados:** Usuarios que podrán navegar por la aplicación web, ver los artículos, acceder a la información detallada del artículo, buscar un artículo con el buscador y por supuesto podría registrarse si así lo desea.
- **Usuarios registrados:** Usuarios registrados en nuestra plataforma que accederán mediante login los cuales, además de hacer todas las acciones de los usuarios no registrados, podrán comprar artículos con cualquiera de los métodos de pago incluidos, comprar con donación, valorar el sitio web y ver su histórico de compras.
- **Usuarios administradores:** Este tipo de usuario estará disponible solo para la empresa que tenga el servicio contratado con nosotros desde una pestaña con un login específico, con este usuario podrá entrar al panel de administración de artículos y gestionar todos los artículos, además podrá ver el listado de clientes registrados y pedidos.  
Adicionalmente, siempre y cuando tenga contratado el servicio extra de análisis de datos, podrá ver también un panel del análisis generado con los datos del registro de sus usuarios creado con **¿PowerBi?** en el cual podrá ver artículos más vendidos/menos, espectro de clientes que más compran en su web, tipo de artículo más demandado etc.

## 2.6 Tecnologías a emplear





En este proyecto de software vamos a utilizar muchas tecnologías las cuales vamos a listar a continuación separadas por desarrollo e implantación de la aplicación (Hosting) en un entorno cloud como es Azure.

### Desarrollo:

Nombre	Descripción	Icono
Visual Studio	Entorno de desarrollo completo	
ASP.NET Core	Framework Back-End con MVC	
Entity Framework	Herramienta ORM (Object Relational Mapping) mediante la cual podremos trabajar con bases de datos relacionales desde las clases de nuestro código	
SQL	Lenguaje de bases de datos relacionales	
HTML 5	Lenguaje de marcas para estructurar la web	

<b>CSS 3</b>	Lenguaje para hojas de estilos en cascada	
<b>Bootstrap</b>	Framework de estilos	
<b>C#</b>	Lenguaje de programación multiparadigma desarrollado y estandarizado por la empresa Microsoft como parte de su plataforma .NET	
<b>Git</b>	Software control de versiones	
<b>GitHub</b>	Plataforma de desarrollo colaborativo	
<b>PowerBI</b>	Herramienta de análisis de datos	

**Implantación en entorno cloud:**

Nombre	Descripción	Icono
<b>Azure Cloud</b>	Servicio de computación en la nube	
<b>Azure Service Plan</b>	Plan de servicios de azure que nos proporciona distintos tipos de servicios para implementar nuestras aplicaciones web	
<b>Web App service</b>	Servicio para instalar una aplicación web dentro de un Service plan de Azure	
<b>Azure SQL Database</b>	Servicio para implementar una BBDD relacional	

### 3. DISEÑO

#### 3.1 Estructura general del proyecto. Diagrama de casos de uso

Para poder explicar la estructura del proyecto en base a los posibles actores intervinientes, vamos a detallar dichos actores y además vamos a mostrar el diagrama de casos de uso para dichos actores.

##### **ACTORES:**

Los distintos actores podrán realizar dentro de nuestra aplicación las siguientes acciones y su identificador de caso de uso para su posterior ficha de detalle:

##### **Usuarios no registrados:**

- Ver artículos - [CU001](#)
- Ver información detallada del artículo - [CU002](#)
- Buscar un artículo - [CU003](#)
- Añadir artículos al carrito - [CU004](#)
- Registrarse - [CU005](#)

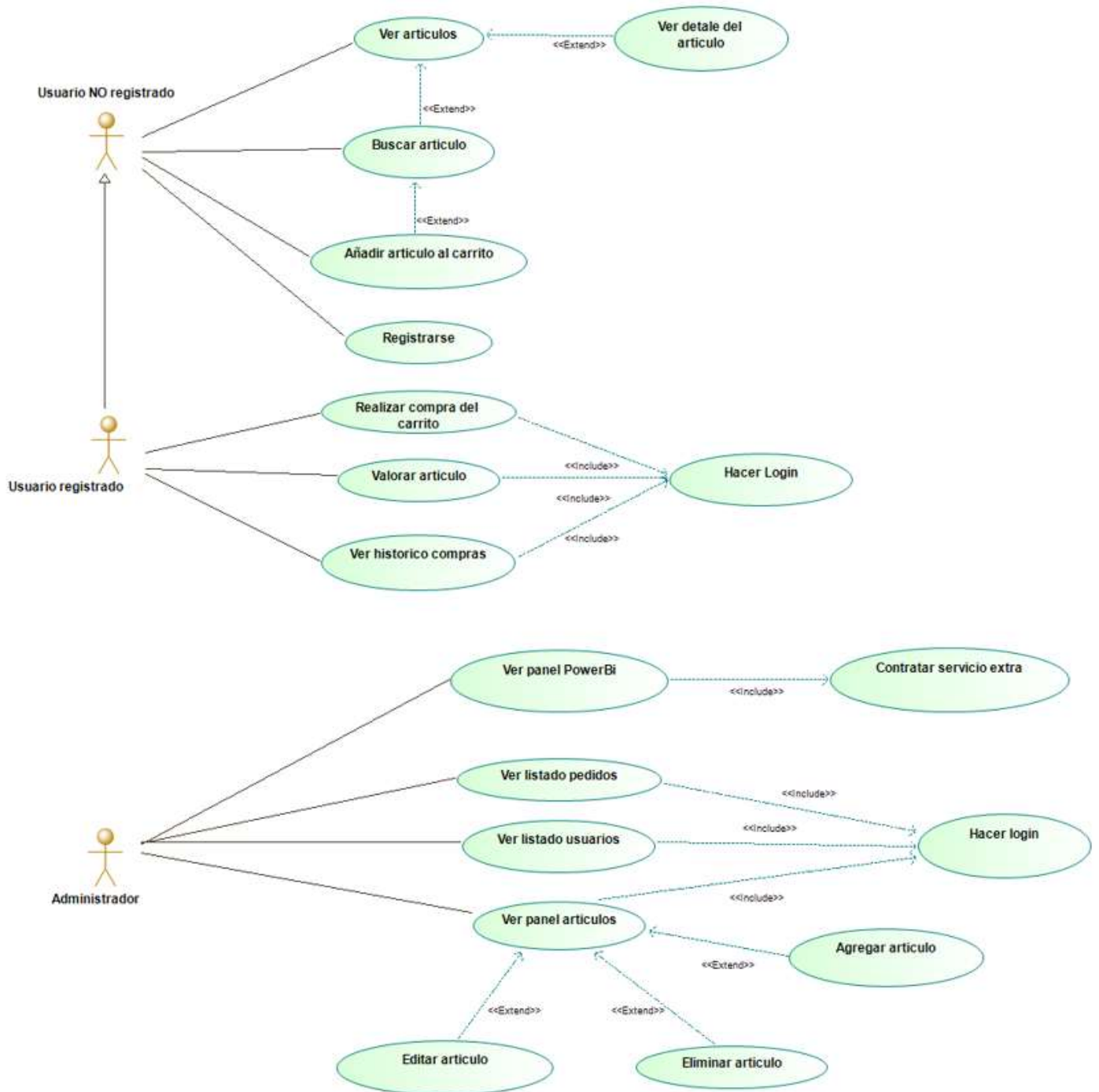
##### **Usuarios Registrados:**

- Hacer login - [CU006](#)
- Ver artículos - [CU007](#)
- Ver detalles de los artículos - [CU008](#)
- Realizar compra de los artículos del carrito - [CU009](#)
- Valorar artículo - [CU010](#)
- Ver histórico de compras propias - [CU011](#)

##### **Usuarios Administradores:**

- Hacer Login como Administrador - [CU006](#)
- Ver panel de artículos - [CU012](#)
- Agregar artículo - [CU013](#)
- Eliminar artículo - [CU014](#)
- Editar artículo - [CU015](#)
- Ver listado de usuarios registrados - [CU016](#)
- Ver listado de pedidos - [CU017](#)
- Ver panel de PowerBi con análisis (Solo si tiene el servicio extra contratado) - [CU018](#)

## CASOS DE USO:





A continuación, vamos a detallar cuatro casos de uso de nuestro diagrama para explicar sus funcionalidades y condiciones para poder realizarlos.

<b>Nombre:</b> Hacer Login <b>ID:</b> CU006
<b>Descripción:</b> El usuario podrá loguearse en el sistema con su usuario y contraseña, tras ser validados, el usuario podrá realizar las acciones que el sistema le permita.
<b>Actores:</b> Administrador, Cliente
<b>Precondiciones:</b> Estar registrado como administrador o cliente
<b>Curso normal:</b> <ol style="list-style-type: none"> <li>1. El usuario introduce su número de usuario</li> <li>2. El usuario introduce su contraseña</li> <li>3. El sistema valida los datos</li> <li>4. El usuario entra en el sistema y puede realizar las acciones para las cuales tenga permisos.</li> </ol>
<b>Postcondiciones:</b> Ninguna
<b>Alternativa 1:</b> <ol style="list-style-type: none"> <li>2.1 En caso de no recordar la contraseña, podrá recuperarla.</li> </ol>
<b>Alternativa 2:</b> <ol style="list-style-type: none"> <li>3.1 Si los datos no son correctos, el usuario podrá volver a intentarlo.</li> <li>3.2 El usuario puede registrarse (sino lo está) accediendo al formulario de registro</li> </ol>

<b>Nombre:</b> Agregar artículo <b>ID:</b> CU013
<b>Descripción:</b> El usuario podrá dar de alta nuevos productos dentro del sistema para la posterior comercialización de estos.
<b>Actores:</b> Administrador
<b>Precondiciones:</b> Estar registrado como administrador y estar logueado en el sistema, además que el producto no exista ya en el sistema
<b>Curso normal:</b> <ol style="list-style-type: none"> <li>1. El usuario en el panel de administrador selecciona la opción de añadir producto</li> <li>2. Introduce un producto nuevo</li> <li>3. Confirma el nuevo producto</li> <li>4. El nuevo producto podrá ser visualizado en la aplicación web por el resto de usuarios</li> <li>5. <b>Include</b> Hacer login</li> </ol>
<b>Postcondiciones:</b> Ninguna
<b>Alternativa 1:</b> <ol style="list-style-type: none"> <li>4.1 Si el producto ya existe en la aplicación, el sistema lanzará un aviso y el producto no podrá ser añadido y volverá a tener la opción de agregar un nuevo producto.</li> </ol>

**Nombre:** Buscar artículo  
**ID:** CU003

**Descripción:** El usuario podrá buscar productos a través del buscador de la aplicación web ya sea por nombre o categoría, una vez dentro del producto tendrá la posibilidad de añadirlo al carrito

**Actores:** Usuario no registrado

**Precondiciones:** Estar dentro de la aplicación web

**Curso normal:**

1. El usuario seleccionará el buscador de la aplicación web
2. Introducirá el nombre del producto que quiere buscar
3. Se le mostrarán los productos que coinciden con su búsqueda
4. **extend** Podrá entrar a ver detalladamente el producto que seleccione
5. **extend** Añadir producto al carrito

**Postcondiciones:** Ninguna

**Alternativa 1:**

- 3.1 Si no existe ninguna coincidencia con la búsqueda, el sistema mostrará un mensaje Indicándolo.
- 3.2 El usuario podrá volver a realizar una nueva búsqueda o editar la que había introducido en el input del buscador

**Alternativa 2:**

- 5.1 En caso de no haber stock, el sistema mostrará un mensaje indicándolo
- 5.2 El usuario podrá realizar una nueva búsqueda

**Nombre:** Valorar artículo  
**ID:** CU010

**Descripción:** El usuario registrado podrá hacer una valoración de un producto en concreto del sitio web y que su valoración se publique en la vista de detalle del producto

**Actores:** Usuario registrado

**Precondiciones:** Estar registrado como usuario y estar logueado en el sistema

**Curso normal:**

1. El usuario selecciona ver el detalle de un producto
2. Hace click en el botón de valorar producto
3. realiza la valoración del producto
4. Hace click en el botón de publicar valoración
5. La valoración del producto se guarda en la BBDD y se publica en la web
6. **Include** Hacer login

**Postcondiciones:** Ninguna

### 3.2 Elaboración de un guion de trabajo para el desarrollo del proyecto.

Para abordar el desarrollo de nuestra aplicación de manera ordenada y metódica, debemos tener un plan preestablecido con los pasos definidos que deberemos ir dando para la creación e implantación de nuestra aplicación:

#### **ANALISIS**

1. Conceptualización de nuestra aplicación
2. Elección de tecnologías para crear e implantar el proyecto
3. Adquisición del hardware y software necesario junto con sus licencias

#### **DISEÑO**

4. Creación de los casos de uso
5. Creación y diseño del modelo de datos relacional

#### **CODIFICACIÓN**

6. Desarrollo de la aplicación (Codificación)
  - Sprint 1 – Creación funcionalidades y vistas usuario no registrado
  - Sprint 2 – Creación funcionalidades y vistas usuario registrado
  - Sprint 3 – Creación funcionalidades y vistas usuario administrador

#### **PRUEBAS**

7. Pruebas Unitarias (Durante todo el proceso de codificación)
8. Pruebas de funcionalidad usuario (Incluidas dentro de cada sprint)
9. Pruebas de estrés infraestructura (Al finalizar todos los sprint)

#### **IMPLANTACIÓN**

10. Implantación inicial de la aplicación en Azure (entorno Cloud)

#### **DOCUMENTACIÓN**

11. Como tarea continua durante todo el guion realizaremos una actividad clave y crucial de cara a futuro, documentaremos todo lo que se haga en las distintas fases para que en un futuro sea más fácil, en caso de fallos, rastrear el origen de los mismos

#### **MANTENIMIENTO**

12. Mantenimiento y posibles mejoras en aplicación y tipo de despliegues.

### 3.3 Fases del proyecto. Tareas y plazos de ejecución.

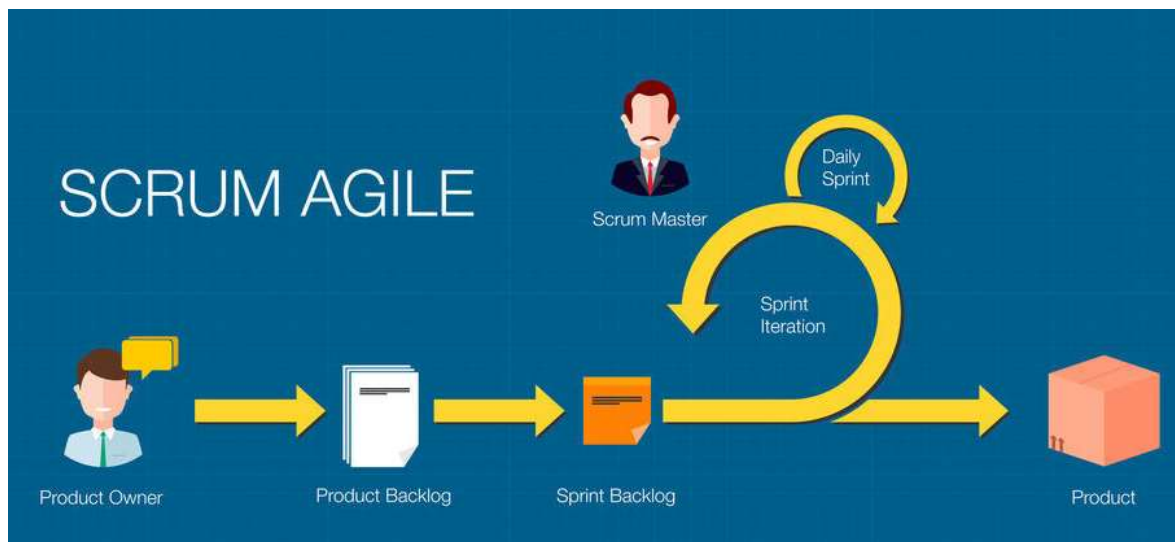
#### **Metodología de desarrollo**

Para desarrollar todo el proyecto y poder crear nuestro producto utilizaremos utilizando una metodología ágil, concretamente SCRUM.

Pero ¿Qué es una metodología ágil?

El desarrollo ágil de software se refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, estas metodologías son imprescindibles en un mundo en el que nos exponemos a cambios recurrentemente. Siempre hay que tener en cuenta como programadores que lo que es la última tendencia hoy puede que no exista mañana y por esto existe la metodología ágil donde los requisitos y soluciones evolucionan mediante la colaboración de grupos auto organizados y multidisciplinares.

SCRUM hace uso de un concepto llamado "Sprint" que básicamente son ciclos de trabajo o desarrollo con una duración sugerida entre 2 a 3 semanas. Al finalizar cada "Sprint" se obtiene un resultado completo y el cual puede ser aprobado por el cliente o un usuario. De tal forma que se puede obtener su retroalimentación y si es necesario, ajustar o mejorar nuestro modulo del producto desarrollado.



Una vez tenemos elegida nuestra metodología de desarrollo, vamos a detallar las actividades que haremos dentro de cada uno de los puntos de nuestro guion de trabajo en este proyecto.

### **Análisis**

En esta fase realizaremos toda la conceptualización inicial de nuestra aplicación estudiando sus posibilidades a nivel de negocio, viabilidad, análisis DAFO y además definiremos que funcionalidades debería tener para satisfacer las necesidades de nuestros potenciales clientes. Además, elegiremos las tecnologías a utilizar para su desarrollo junto con el software necesario. También en esta fase definiremos donde desplegaremos o implantaremos nuestra aplicación para que nuestros futuros clientes puedan usarla y explotarla junto a nuestros servicios adicionales.

### **Diseño**

En la fase de diseño, crearemos los diagramas necesarios de casos de uso que nos permitan tener una visión más clara de las funcionalidades de nuestra aplicación y, además, definiremos y crearemos nuestro diseño relacional de BBDD en el cual se basará la consistencia de los datos que manejará nuestra aplicación.

### **Codificación**

En la fase de codificación pasaremos a desarrollar nuestra aplicación con las tecnologías previamente definidas y basándonos en el diseño de funcionalidades previamente creado.

Dado que nos basaremos en una metodología ágil como SCRUM, en esta fase haremos sprints para desarrollar cada uno de los bloques de funcionalidades de cada uno de los actores que interactuarán con nuestra aplicación. Además, realizaremos durante todo el proceso pruebas unitarias y dentro de cada sprint realizaremos pruebas de usuario y la implantación de esa funcionalidad.

#### **Sprint 1 – Creación funcionalidades y vistas usuario no registrado**

- Ver artículos
- Ver detalle articulo
- Buscar articulo
- Añadir articulo al carrito
- Registrarse

#### **Sprint 2 – Creación funcionalidades y vistas usuario registrado**

- Hacer login
- Realizar compra del carrito
- Valorar web
- Ver histórico de compras

#### **Sprint 3 – Creación funcionalidades y vistas usuario administrador**

- Ver panel de PowerBI
- Ver listado pedidos
- Ver listado usuarios
- Ver panel artículos
- Editar articulo
- Agregar articulo
- Eliminar articulo

### Pruebas

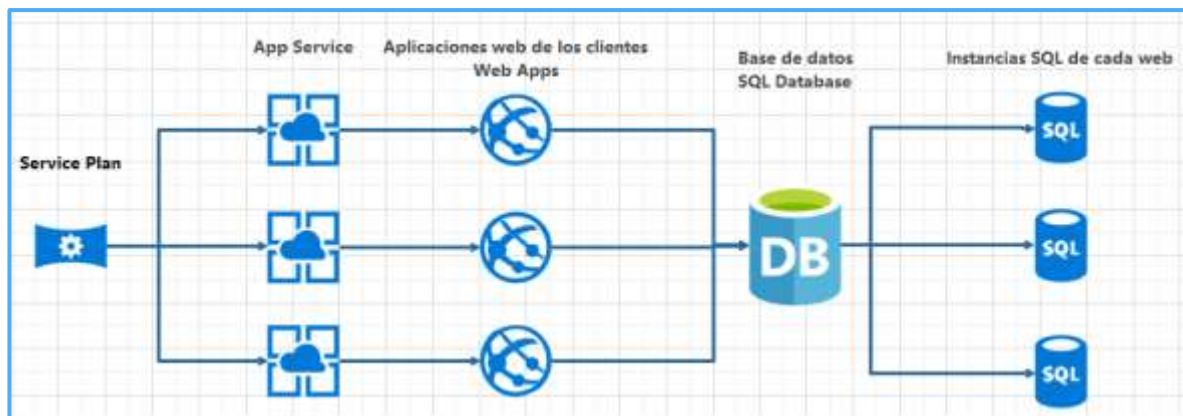
En esta fase realizaremos las pruebas necesarias para verificar la estabilidad de nuestra aplicación. Estará incluida dentro de la fase de codificación.

- **Pruebas unitarias** de nuestro código para comprobar todas las posibles casuísticas de datos entrantes o salientes, se realizarán durante de toda la fase de codificación.
- **Pruebas de usuario** para probar la funcionalidad de nuestro producto desde el punto de vista del usuario final. Se realizarán cada vez que finalice un sprint.
- **Pruebas de rendimiento** en la infraestructura para saber si nuestro host y la BBDD soportarán un flujo de interacciones altas por parte de los usuarios. Se realizarán una vez hayan terminado los 3 sprints y se pueda probar todo junto.

### Implantación en Azure

En esta fase desplegaremos nuestro software en un entorno cloud, concretamente en Azure, permitiéndonos proporcionar al usuario final un tiempo de disponibilidad de nuestro software nunca inferior a un 95%. Además, el tipo de arquitectura planteado nos permitirá poder escalar nuestra infraestructura a medida que nuestro negocio vaya creciendo ya que una de las grandes ventajas de la nube es su gran flexibilidad para escalar tanto horizontalmente o verticalmente los recursos contratados.

Este sería un ejemplo de la arquitectura que pretendemos desplegar en el entorno cloud y que nos permitirá tener todos los recursos de nuestros clientes centralizados en un único alojamiento, permitiéndonos esto una mejor gestión y fácil escalabilidad. Explicaremos esta infraestructura al detalle en el punto 4.4.





### **Mantenimiento**

En esta última fase, daremos mantenimiento y soporte a nuestra aplicación desplegando actualizaciones continuas para aplicar mejoras o corregir posibles bugs. Adicionalmente si los clientes lo tienen contratado daremos soporte de control de stock y les proporcionaremos un panel de PowerBi con el análisis de sus datos refrescado dinámicamente.

Como futura mejora no incluida en este proyecto, nos gustaría en un futuro incorporar los despliegues automatizados de nuestras versiones (releases) a través de la herramienta de automatización DevOps en Azure.

El término DevOps, que es una combinación de los términos ingleses development (desarrollo) y operations (operaciones), designa la unión de personas, procesos y tecnología para ofrecer valor a los clientes de forma constante.

“¿Qué significa DevOps para los equipos? DevOps permite que los roles que antes estaban aislados (desarrollo, operaciones de TI, ingeniería de la calidad y seguridad) se coordinen y colaboren para producir productos mejores y más confiables. Al adoptar una cultura de DevOps junto con prácticas y herramientas de DevOps, los equipos adquieren la capacidad de responder mejor a las necesidades de los clientes, aumentar la confianza en las aplicaciones que crean y alcanzar los objetivos empresariales en menos tiempo”

Fuente: <https://azure.microsoft.com/es-es/overview/what-is-devops/>



### **Plazos de ejecución**

Los plazos de ejecución del proyecto irán desde el 1 de marzo hasta el 30 de junio incluido, para ello hemos realizado un diagrama de Gantt en el cual se detallan las tareas realizadas por los 2 desarrolladores durante este periodo añadiendo además la tarea de documentación que se realizará al mismo tiempo que se van realizando las distintas tareas. La parte del mantenimiento se realizará a partir de cuando finalice la fase de implantación, a partir de ese momento entraremos en la fase en la cual se podrán añadir mejoras y corregir bugs durante tiempo indefinido ya que los clientes nos pagarán una cuota mensual donde irá este servicio incluido.

### Diagrama de Gantt

En este diagrama podemos ver cómo se va a desarrollar el proyecto, las fechas de inicio y fin de cada una de las tareas planteadas, así como las duraciones totales de cada uno de los pasos que se irán dando en el proceso de la creación de nuestro software.

Se puede observar en el diagrama lo que dura cada una de las partes genéricas y dentro de cada parte cada una de las subtareas. A destacar como dentro de la parte de “CODIFICACIÓN” se realizarán los 3 sprint y además al mismo tiempo se van realizando las pruebas unitarias, pruebas de usuario cada vez que finaliza un sprint, implementación de cada funcionalidad tras las pruebas de usuario después de cada sprint y las pruebas de rendimiento cuando finalicen los 3 sprint.

ID	Nombre de Tarea	Fecha de inicio	Fecha de finalización	Duración Total	01/03/2022	01/04/2022		
					01/03/2022	01/04/2022	01/05/2022	01/06/2022
1	ANÁLISIS	01/03/2022	21/03/2022	15.0 d.				
2	Conceptualización	01/03/2022	14/03/2022	10.0 d.				
3	Elección tecnologías	15/03/2022	21/03/2022	5.0 d.				
4	Adquisición software y hardware	07/03/2022	16/03/2022	8.0 d.				
5	DISEÑO	22/03/2022	30/03/2022	7.0 d.				
6	Creación casos de uso	22/03/2022	30/03/2022	7.0 d.				
7	Modelo relacional	22/03/2022	30/03/2022	7.0 d.				
8	CODIFICACIÓN	01/04/2022	07/06/2022	48.0 d.				
9	SPRINT 1	01/04/2022	20/04/2022	14.0 d.				
10	Funcionalidades y vistas	01/04/2022	14/04/2022	10.0 d.				
11	Pruebas usuario	19/04/2022	18/04/2022	2.0 d.				
12	Implementación	19/04/2022	20/04/2022	2.0 d.				
13	SPRINT 2	22/04/2022	11/05/2022	14.0 d.				
14	Funcionalidades y vistas	22/04/2022	05/05/2022	10.0 d.				
15	Pruebas usuario	06/05/2022	09/05/2022	2.0 d.				
16	Implementación	10/05/2022	11/05/2022	2.0 d.				
17	SPRINT 3	13/05/2022	01/06/2022	14.0 d.				
18	Funcionalidades y vistas	13/05/2022	26/05/2022	10.0 d.				
19	Pruebas usuario	27/05/2022	30/05/2022	2.0 d.				
20	Implementación	31/05/2022	01/06/2022	2.0 d.				
21	Pruebas unitarias	01/04/2022	01/06/2022	44.0 d.				
22	Pruebas rendimiento	02/06/2022	07/06/2022	4.0 d.				
23	IMPLANTACIÓN EN AZURE	08/06/2022	29/06/2022	16.0 d.				
24	Despliegue en entorno cloud	08/06/2022	29/06/2022	16.0 d.				
25	DOCUMENTACIÓN	01/03/2022	29/06/2022	87.0 d.				
26	Documentación del desarrollo	01/03/2022	29/06/2022	87.0 d.				
27	Documentación para el cliente	01/03/2022	29/06/2022	87.0 d.				

### 3.4 Recursos materiales y personales

Para poder llevar a cabo este proyecto vamos a necesitar de unos recursos tanto personales como materiales. Para ellos vamos a detallar cuales van a ser estos recursos y que papel van a desempeñar dentro del proyecto.

#### **Recursos personales**

En cuanto a las personas que van a intervenir en el proyecto, tal y como hemos explicado anteriormente, serán 2 desarrolladores a tiempo completo. El resto de tareas relacionadas con la gestión administrativa o marketing lo externalizaremos.

- Desarrollador dueño autónomo
- Desarrollador contratado

Por lo tanto, entre los 2 desarrolladores realizarán todas las tareas planteadas en este proyecto (Análisis, diseño, codificación, pruebas, implantación y documentación) y todo lo relacionado con el marketing digital y la gestión administrativa se realizará de manera externa pagando una cuota a distintas entidades. La cuestión administrativa la derivaremos a una gestoría y la cuestión de márketing la derivaremos a una empresa especializada.

Puesto	Titulación	Tipo de alta en Seguridad Social
Desarrollador dueño	Grado superior DAW	Autónomo
Desarrollador empleado	Grado superior DAW	Contrato a tiempo completo

#### **Recursos materiales**

Para poder realizar el proyecto además necesitaremos realizar un desembolso inicial en material y licencias de software.

En cuanto a material fijo necesitaremos comprar inicialmente:

- 2 portátiles para los desarrolladores (uno para cada uno)



- 2 pantallas de 24" para cada uno de los desarrolladores



- Licencias de software necesarias
- Material de oficina básico (libretas, bolígrafos etc)

### Lugar de trabajo

Respecto al lugar en el cual trabajaremos, lo haremos en modalidad "Coworking". El coworking nos permite tener un espacio de trabajo en una oficina compartida pagando una mensualidad no muy alta, esto nos va a dar una gran flexibilidad en cuanto a los gastos de los primeros meses de nuestro negocio y nos permitirá relacionarnos con otros pequeños emprendedores con los cuales compartiremos espacio de trabajo.

Además, esta modalidad nos permite tener entre otras cosas lo siguiente:

- Fibra de alta velocidad
- Sala de reuniones
- Aire acondicionado
- Recepción de correspondencia y paquetería
- Mobiliario de oficina
- Servicio de limpieza diario

Nuestro proyecto concretamente lo realizaremos en la ciudad de Elche en unas oficinas de coworking en la que el alquiler que se nos ofrece tiene un precio asequible al mes completo de 150€. Permittiéndonos tener un espacio de trabajo con todas las comodidades a un precio bastante asequible para comenzar.



### 3.5 Estimación de gastos

A la hora de montar cualquier tipo de negocio es muy importante tener en cuenta, de manera aproximada, que gastos iniciales vas a tener el primer año. De esta forma empezaremos a darle forma a la viabilidad económica de nuestro proyecto.

Con todo lo expuesto anteriormente, nuestros gastos obligatorios durante el primer año serían aproximadamente los siguientes:

#### **GASTOS APROXIMADOS PRIMER AÑO**

Concepto	Inversión	Gastos	6 meses	12 meses
Autónomo bonificado		60 €	360 €	720 €
Módulos trimestrales		500 €	1.000 €	2.000 €
Amortización préstamo		466 €	2.796 €	5.592 €
Gestoría		40 €	240 €	480 €
Sueldo desarrollador		1.200 €	7.200 €	14.400 €
Publicidad		50 €	300 €	600 €
Portátiles desarrollo (x2)	2.017,88 €			2.018 €
Pantallas 24" (x2)	252 €			252 €
Licencias software		50 €	300 €	600 €
Coworking		150 €	900 €	1.800 €
Tarifa Móvil		9 €	54 €	108 €
<b>Total</b>	<b>2269,88</b>	<b>2.525 €</b>	<b>13.150 €</b>	<b>28.570 €</b>

En primer lugar, tendríamos la cuota de autónomo, esta cuota de autónomo es una cuota bonificada durante el primer año de 60€ mensuales, tras este año la cuota aumentaría. Además, pagaríamos el IVA a través de la modalidad de módulos trimestrales.

Para arrancar nuestro negocio, además de poner un capital inicial propio que detallaremos más adelante, pediremos un préstamo ICO el cual iríamos devolviendo durante los siguientes 5 años.

También podemos reflejar en la tabla los gastos de gestoría, coworking, sueldo de un desarrollador, publicidad y material necesario. Además, incluiríamos las licencias de Visual Studio Enterprise y una licencia de PowerBi individual para crear un modelo de panel de análisis de datos (40€ + 10€).

Con todo lo expuesto en la tabla creemos que podríamos arrancar el proyecto de una manera coherente ya que no son unos gastos desorbitados para 1 año completo de un negocio. Está claro que tendremos algunos gastos más como el alojamiento en entorno cloud de todos los proyectos de nuestros clientes, pero estos gastos irán sufragados con el precio que cobraremos a nuestros clientes por implantarles y personalizarles nuestro software en su negocio.

### 3.6 Estimación de ingresos y precios de venta

En el punto anterior hicimos una estimación aproximada de los gastos que podríamos tener durante los primeros 12 meses de vida de nuestro negocio. En este punto vamos a establecer el precio de nuestro software y los servicios asociados mediante los cuales debemos hacer rentable nuestro proyecto. Debemos intentar que al menos durante estos primeros 12 meses no perdamos dinero.

Este proyecto está basado en vender una aplicación web que permita a los clientes vender ropa de segunda mano online, para hacer rentable esta idea hemos considerado que cada cliente que quiera contar con nuestro software deberá hacer un pago inicial de 700€ ya que recordemos que debemos personalizar las vistas de la aplicación web para cada cliente que contrate nuestros servicios.

Tras la implantación de la aplicación web, cada cliente pagará 200€/mes en los cuales irá incluido alojamiento web, alojamiento de BBDD y dominio. Adicionalmente venderemos un primer servicio opcional de mantenimiento de stock en la tienda de cada cliente, para que se despreocupen de esta tarea, por el cual cobraremos 150€/mes y también tendrán los clientes disponible un segundo servicio opcional de análisis de datos, a través un dashboard de PowerBi integrado en una pestaña del panel de administrador mediante el cual podrán consultar datos importantes relativos a sus propios clientes (edad mayoritaria de compradores, tipo de producto más comprado, media de compra por cliente etc..), este servicio adicional costaría 150€/mes.

Por lo tanto, si hacemos una tabla estimativa para ver que ingresos tendríamos a los 6 meses con 5 clientes con todos los servicios contratados y que ingresos tendríamos a los 12 meses con 8 clientes con todos los servicios contratados, veríamos que a los 6 meses ya tendríamos beneficios sobre los gastos y que a los 12 meses los beneficios ya serian bastante visibles en comparación con los gastos.

#### **ESTIMACIÓN INGRESOS POR SOFTWARE**

Servicios	Precio servicio	Periodicidad	6 MESES (5 CLIENTES)	12 MESES (8 CLIENTES)
Implantación inicial	700 €	Único	3.500 €	5.600€
Cuota	200 €	Mensual	6.000 €	19.200 €
Mantenimiento stock	150 €	Mensual	4.500 €	14.400 €
Dashboard PowerBI	150 €	Mensual	4.500 €	14.400 €
<b>Beneficio total</b>			<b>18.500 €</b>	<b>53.600 €</b>

Obviamente esto solo es una estimación sobre un supuesto optimista en cuanto a la contratación de todos nuestros servicios por parte de 8 clientes, pero se puede observar que el negocio podría ser rentable con una cantidad de clientes contenida. Si ninguno de estos clientes contratara por ejemplo el dashboard de PowerBI, aun seguiríamos teniendo beneficios. Si ninguno de los 8 clientes contrata ninguno de los servicios opcionales, ahí si podríamos estar en problemas en cuanto a beneficios por lo que una de nuestras principales misiones será vender esos dos servicios opcionales a nuestros clientes para poder obtener mayor beneficio y hacer nuestro negocio y producto más rentable.



### 3.7 Viabilidad económica.

Antes de emprender un proyecto, sea del sector que sea, es necesario tener en consideración si el proyecto va a ser viable económicamente. Podemos tener una idea espectacular, pero si no es viable económicamente difícilmente podrá llevarse a cabo.

Para esto existe el plan de viabilidad, mediante el cual vamos a ver con cifras en base a una hipotética evolución del negocio si el proyecto puede ser rentable o no.

En este plan vamos a presentar un plan de tesorería en el cual vamos a poder ver como sería la evolución durante el primer año de vida de nuestro proyecto basándonos en una idea estimativa de clientes.

Presentaremos también una cuenta de resultados mediante la cual podremos conocer el resultado económico del proyecto a partir de la diferencia entre los ingresos/beneficios y los gastos/perdidas.

A continuación, veremos el balance de previsión el cual reflejará la situación del patrimonio (conjunto de bienes, derechos y obligaciones) de la empresa en un momento determinado.

Por último, mostraremos el umbral de rentabilidad el cual hace referencia a la cantidad que tenemos que ingresar por ventas para comenzar a obtener beneficios.

#### Plan de tesorería

En este plan de tesorería del primer año nos vamos a basar en los datos de gastos expuestos anteriormente y en los precios de los servicios descritos también en puntos anteriores, además hemos estimado una situación en la cual arrancaremos el primer mes sin clientes, durante los primeros 6 meses tendremos 6 clientes (capturaríamos los primeros 6 clientes entre abril y junio antes de finalizar el desarrollo de nuestra aplicación) y al finalizar el año tendremos 8 clientes (8 webs implantadas).

Entradas	Mar	Abr	May	Jun	Jul	Ago	Sep	Oct	Nov	Dic	Ene	Feb	Total
Aportación Inicial	10.000 €												10.000 €
Subvenciones	2.500 €												2.500 €
Préstamo ICO	25.000 €												25.000 €
Pago inicial cliente		1.400 €	1.400 €	1.400 €			1.400 €						5.600 €
Cuotas					1.200 €	1.200 €	1.200 €	1.600 €	1.600 €	1.600 €	1.600 €	1.600 €	11.600 €
Servicio PowerBI					900 €	900 €	900 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	8.700 €
Servicio Stock					900 €	900 €	900 €	900 €	900 €	900 €	900 €	900 €	7.200 €
<b>TOTAL ENTRADAS</b>	<b>37.500 €</b>	<b>1.400 €</b>	<b>1.400 €</b>	<b>1.400 €</b>	<b>3.000 €</b>	<b>3.000 €</b>	<b>4.400 €</b>	<b>3.700 €</b>	<b>3.700 €</b>	<b>3.700 €</b>	<b>3.700 €</b>	<b>3.700 €</b>	<b>70.600 €</b>
<b>TOTAL VENTAS</b>													<b>33.100 €</b>
<b>Salidas</b>													
Devolución préstamo	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	465,51 €	5.586,12 €
Coworking	150 €	150 €	150 €	150 €	150 €	150 €	150 €	150 €	150 €	150 €	150 €	150 €	1.800 €
Compra ordenadores	2.017,88 €												2.017,88 €
Alojamiento Cloud		300 €	300 €	300 €	300 €	300 €	300 €	300 €	300 €	300 €	300 €	300 €	3.300,00 €
Compra pantallas	252 €												252 €
Autonomo	60 €	60 €	60 €	60 €	60 €	60 €	60 €	60 €	60 €	60 €	60 €	60 €	720 €
Modulos autonomo			500 €			500 €			500 €			500 €	2.000 €
Salario desarrollador	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	1.200 €	14.400 €
Gestoría	40 €	40 €	40 €	40 €	40 €	40 €	40 €	40 €	40 €	40 €	40 €	40 €	480 €
Publicidad	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	600 €
Móvil	9 €	9 €	9 €	9 €	9 €	9 €	9 €	9 €	9 €	9 €	9 €	9 €	108 €
Licencias	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	50 €	600 €
<b>TOTAL SALIDAS</b>	<b>4.294,39 €</b>	<b>2.324,51 €</b>	<b>2.824,51 €</b>	<b>2.324,51 €</b>	<b>2.324,51 €</b>	<b>2.824,51 €</b>	<b>2.324,51 €</b>	<b>2.324,51 €</b>	<b>2.824,51 €</b>	<b>2.324,51 €</b>	<b>2.324,51 €</b>	<b>2.824,51 €</b>	<b>31.864,00 €</b>
<b>Entradas menos salidas</b>	<b>33.206 €</b>	<b>-924,51 €</b>	<b>-1.424,51 €</b>	<b>-924,51 €</b>	<b>675,49 €</b>	<b>175,49 €</b>	<b>2.075,49 €</b>	<b>1.375,49 €</b>	<b>875,49 €</b>	<b>1.375,49 €</b>	<b>1.375,49 €</b>	<b>875,49 €</b>	<b>38.736,00 €</b>
<b>Saldo en el banco</b>	<b>33.206 €</b>	<b>32.281,10 €</b>	<b>30.856,59 €</b>	<b>29.932,08 €</b>	<b>30.607,57 €</b>	<b>30.783,06 €</b>	<b>32.858,55 €</b>	<b>34.234,04 €</b>	<b>35.109,53 €</b>	<b>36.485,02 €</b>	<b>37.860,51 €</b>	<b>38.736,00 €</b>	<b>38.736,00 €</b>

Tal y como podemos observar en la tabla anterior, iniciaremos el primer mes sin ingresos por nuestro software, pero lo compensaremos con las aportaciones iniciales personales, subvención y el total de un préstamo ICO. A partir del segundo mes continuaremos desarrollando el proyecto de software, pero al mismo tiempo iremos captando clientes cobrando la cantidad inicial obligatoria para poder personalizar su futura web.

En julio empezaríamos a ingresar ya dinero de nuestros clientes tanto de cuotas mensuales como de servicios extra contratados, la idea es finalizar el año con 8 clientes en total con 6 de ellos con cuota + los 2 servicios extra y 2 de ellos solamente con un servicio extra.

La finalidad del primer año de vida de nuestro negocio es no perder dinero, por lo que podemos observar en la tabla un dato muy importante ya que el “total de ventas” es ligeramente superior al “total de salidas”, por lo que podríamos decir en base a esto que el negocio al menos no produciría pérdidas en el primer año de vida.

Hay otro dato que nos indicaría que no hemos perdido dinero y es que el saldo en el banco sería ligeramente superior al total de entradas del mes inicial marzo, por lo que no habríamos ganado dinero el primer año, pero tampoco habríamos perdido el dinero que entraría el primer mes procedente de aportaciones, subvención y préstamo.

Para ver más objetivamente esto vamos a mostrar una tabla con la cuenta de resultados.

### **Cuenta de resultados**

Como podemos observar en la siguiente tabla, el resultado antes de impuestos es positivo por lo que este dato fortalece la idea de que nuestro proyecto será rentable ya que no habremos perdido dinero en este primer año de vida de nuestro negocio.

Obviamente el objetivo del siguiente año y sucesivos sería aumentar la cartera de clientes e intentar que todos los clientes aprecien el valor añadido de nuestros 2 servicios extra para que así los ingresos por las ventas aumenten de manera significativa y de manera progresiva.

PRIMER AÑO			
INGRESOS DE EXPLOTACIÓN		GASTOS DE EXPLOTACIÓN	
Ventas totales	33.100 €	Coworking	1.800 €
		Ordenadores	2.017,88 €
		Alojamiento Cloud	3.300 €
		Pantallas	252 €
		Autonomo	720 €
		Modulos autonomo	2.000 €
		Salario desarrollador	14.400 €
		Gestoria	480 €
		Publicidad	600 €
		Movil	108 €
		Licencias	600 €
<b>TOTAL</b>	<b>33.100 €</b>		<b>26.277,88 €</b>
INGRESOS FINANCIEROS		GASTOS FINANCIEROS	
		Intereses prestamo	539,08 €
<b>RESULTADO DE EXPLOTACIÓN</b>			<b>6.822,12 €</b>
<b>RESULTADO FINANCIERO</b>			<b>-539,08 €</b>
<b>RESULTADO ANTES DE IMPUESTOS</b>			<b>6.283,04 €</b>

**Balance de previsión**

Ahora vamos a mostrar la tabla con el balance, el balance de previsión representa la situación del patrimonio de la empresa en un día concreto (en nuestro caso el último día del año completo). El patrimonio hace referencia al conjunto de bienes, derechos y deudas que tiene una empresa.

El patrimonio se divide en tres partes: Activo, Pasivo y Neto. Hay que tener en cuenta que el Activo debe ser igual al Pasivo + Neto, de esta manera nuestro negocio estará en equilibrio.

PRIMER AÑO			
ACTIVO NO CORRIENTE		NETO PATRIMONIAL	
Inmovilizado intangible		Resultados del ejercicio	6.283,04 €
Licencias	600 €	Subvención	2.500 €
Amortización licencias	-600 €	Aportación inicial	10.000 €
Inmovilizado material			
Portátiles	2.017,88 €		
Amortización portátiles	-2.017,88 €		
Pantallas	252 €		
Amortización pantallas	-252 €		
ACTIVO CORRIENTE		PASIVO NO CORRIENTE	
Existencias		Deuda a largo plazo	19.952,96 €
Realizable			
Disponible		PASIVO CORRIENTE	
Cuenta del banco	38.736,00 €	Deuda corto plazo	0 €
<b>TOTAL ACTIVO</b>	<b>38.736,00 €</b>	<b>TOTAL NETO Y PASIVO</b>	<b>38.736,00 €</b>

**Umbral de rentabilidad**

*“Cualquier empresa necesita obtener beneficios en algún momento, beneficios que no se logran hasta que el volumen de negocio supera los costes. Sin embargo, el volumen de negocios generado tras la creación de la empresa no suele ser suficientemente alto. A menudo, en los primeros meses/año, los costes superan la cifra de negocios, lo cual significa que la empresa se encuentra inicialmente en la zona de pérdidas. Si la facturación va incrementándose con el tiempo, el negocio estará cada vez más cerca de obtener beneficios, hasta alcanzar el umbral de la zona de ganancias, conocido como break even point (BEP) o umbral de rentabilidad. Es aquí donde los costes y el volumen de negocio se encuentran al mismo nivel.*

*El umbral de rentabilidad también se denomina punto de equilibrio o punto muerto, ya que separa la zona de pérdidas de la zona de ganancias. Cuando la empresa está en el umbral significa que ya no genera ni pérdidas ni ganancias.*

*Si se desea calcular el umbral de rentabilidad de un producto concreto, se tiene en cuenta el número de unidades (análisis de un solo producto). Por el contrario, el umbral de rentabilidad de varios productos o de una empresa entera se establece según el volumen de ventas que debe alcanzarse en conjunto (análisis de varios productos).”*

Fuente: <https://www.ionos.es/startupguide/gestion/umbral-de-rentabilidad/>



Para calcular el umbral de rentabilidad es necesario disponer de dos cifras: el volumen de negocio y los costes de la empresa. Se alcanza el umbral de rentabilidad cuando volumen de negocio y costes se igualan. Se calcularía con la siguiente formula:

**CF = Costes fijos; P = Precio producto; CV = Coste variable por unidad;**

$$\text{Umbral de rentabilidad} = CF / (P - CV)$$

Según los datos expuestos en los puntos anteriores, tenemos los siguientes datos del primer año:

- **Costes totales anuales = 31864,00€**
- **Ventas totales anuales = 33100,00€**
- **Costes fijos mensuales = 31864 / 12 = 2655,33€**

Dado que nuestro producto no se pudo cuantificar fácilmente vamos a sacar un coste medio para el cliente de la siguiente manera:

- Dividimos la cuota inicial que deben pagar de 700€ entre 12 meses, esto nos dará el precio mensual que pagaría el cliente: **700/12 = 58,33€/coste pago inicial al mes**
- Teniendo en cuenta que nuestros clientes pagarían una cuota mensual de 200€ y que además pueden contratar alguno de nuestros servicios extra por 150€/mes cada uno, vamos a sacar el cálculo poniéndonos en la situación de que cada cliente contrata un servicio extra + la cuota mensual: **200 + 150 = 350€**

Por lo tanto, en base a esto, podemos establecer que **de media nuestro producto cuesta al cliente 408,33€**, cifra resultante de sumar **350 + 58,33 = 408,33€**.

Ya que nos es muy difícil sacar el coste de nuestros servicios y el margen de beneficio en cada servicio y cuota, cogeremos el precio por unidad de nuestro producto la media que hemos sacado anteriormente, ósea el valor **408,33€**.

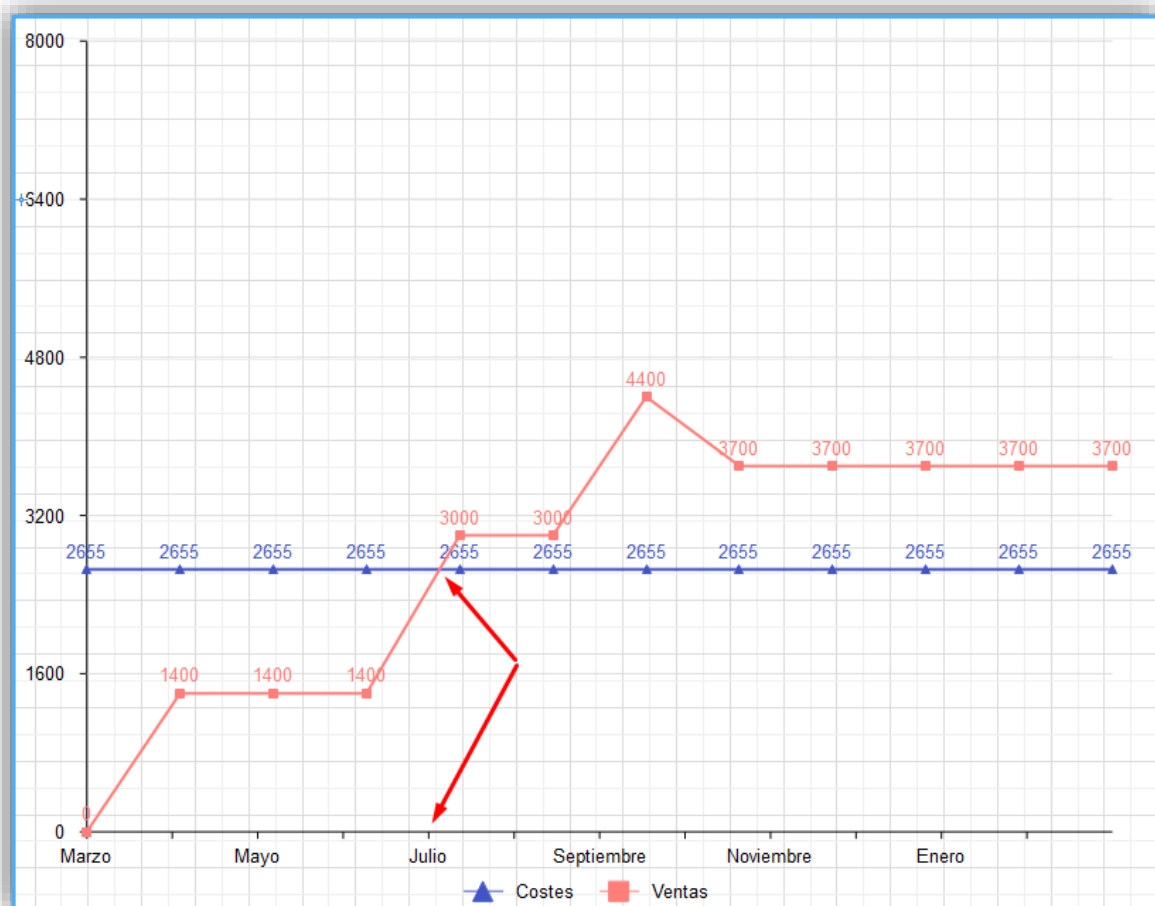
Por lo tanto:

**Umbral de rentabilidad = Costes fijos mensuales / Precio medio servicio al mes.**

- **Umbral rentabilidad =  $2655,33/408,33 = 6,50$  clientes**

Esto quiere decir que debemos vender al mes a un poco más de 6 clientes nuestro producto (parte proporcional pago inicial + cuota mensual + servicio extra) para que el negocio no pierda dinero y comience a ser rentable. Esto se conseguiría en cuanto alguno de los 6 hipotéticos clientes contratase los 2 servicios extra en vez de solo uno.

Analizando nuestros datos podemos observar que a partir de Julio tendríamos 6 clientes con servicios contratados, por lo que dejaríamos de tener perdidas y nuestro negocio comenzaría a ser rentable.



### 3.8 Necesidades de financiación.

Para poder arrancar nuestro proyecto vamos a recurrir a diferentes vías de ingresos, dado que este proyecto se va a ser arrancado por una única persona en la modalidad de autónomo, el cual será dueño único del negocio, se hará una aportación inicial de 10000€ para darse de alta en la seguridad social y poder sufragar los primeros gastos. También se solicitaría una ayuda de 2500€ que concede la cámara de comercio para nuevos emprendedores a fondo perdido y por último recurriríamos a un préstamo ICO con un interés no muy alto de 25000€.

Por lo tanto, las entradas de capital iniciales serían las siguientes:

Fuente económica	Importe inicial
Capital aportado por dueño	10.000€
Subvención a fondo perdido	2.500€
Préstamo ICO	25.000€
<b>Ingresos iniciales totales</b>	<b>37.500€</b>

En cuanto al préstamo ICO, se solicitaría dicha cantidad para devolverla en los próximos 5 años a un interés de un 4,54% TAE el cual no tendría ninguna comisión de apertura extra.

Los intereses totales a pagar de esta operación serían 2930,44€, pagando una cuota mensual del préstamo de 465,51€. A la finalización del pago del préstamo pagaremos un total de 27.930,44€.

Cuota mensual desde

**465,51**

euros al mes

Desde: 4,45% T.I.N. (4,54% T.A.E.)<sup>2</sup>

Comisión de apertura de 0,00% realizando el proceso de solicitud 100% online.

Importe total del crédito: 25.000,00 €.

Coste total del crédito/intereses: 2.930,44 €.

Última cuota: 465,35 €.

Total a pagar **27.930,44 €**

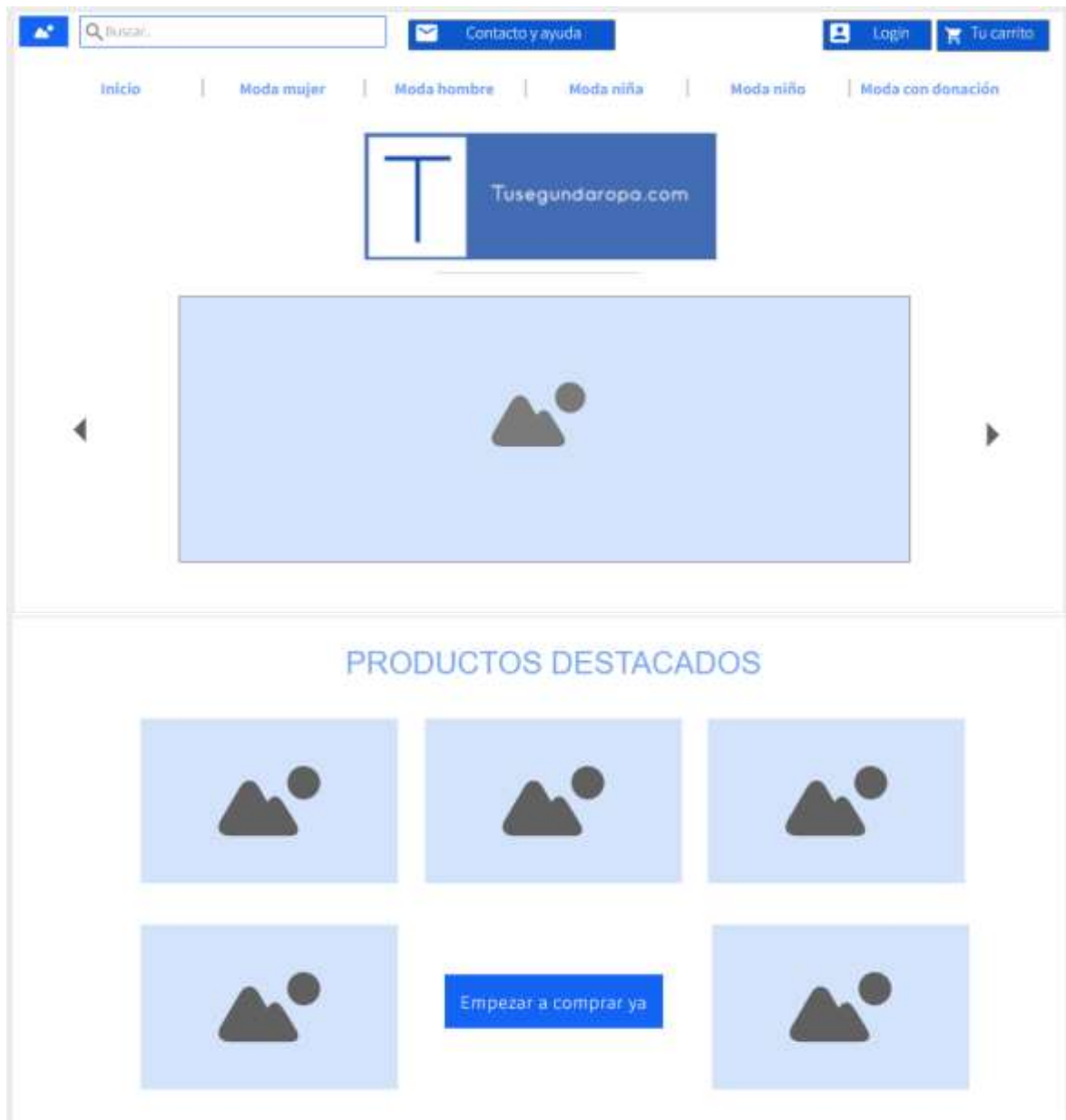
Con este capital inicial tendríamos cubiertos todos los gastos de un año por lo que podríamos trabajar con una relativa tranquilidad en cuanto a liquidez, en los sucesivos años el aumento de ingresos por ventas debería aumentar nuestro capital en el banco de forma considerable no debiendo tener problemas para pagar el préstamo solicitado.



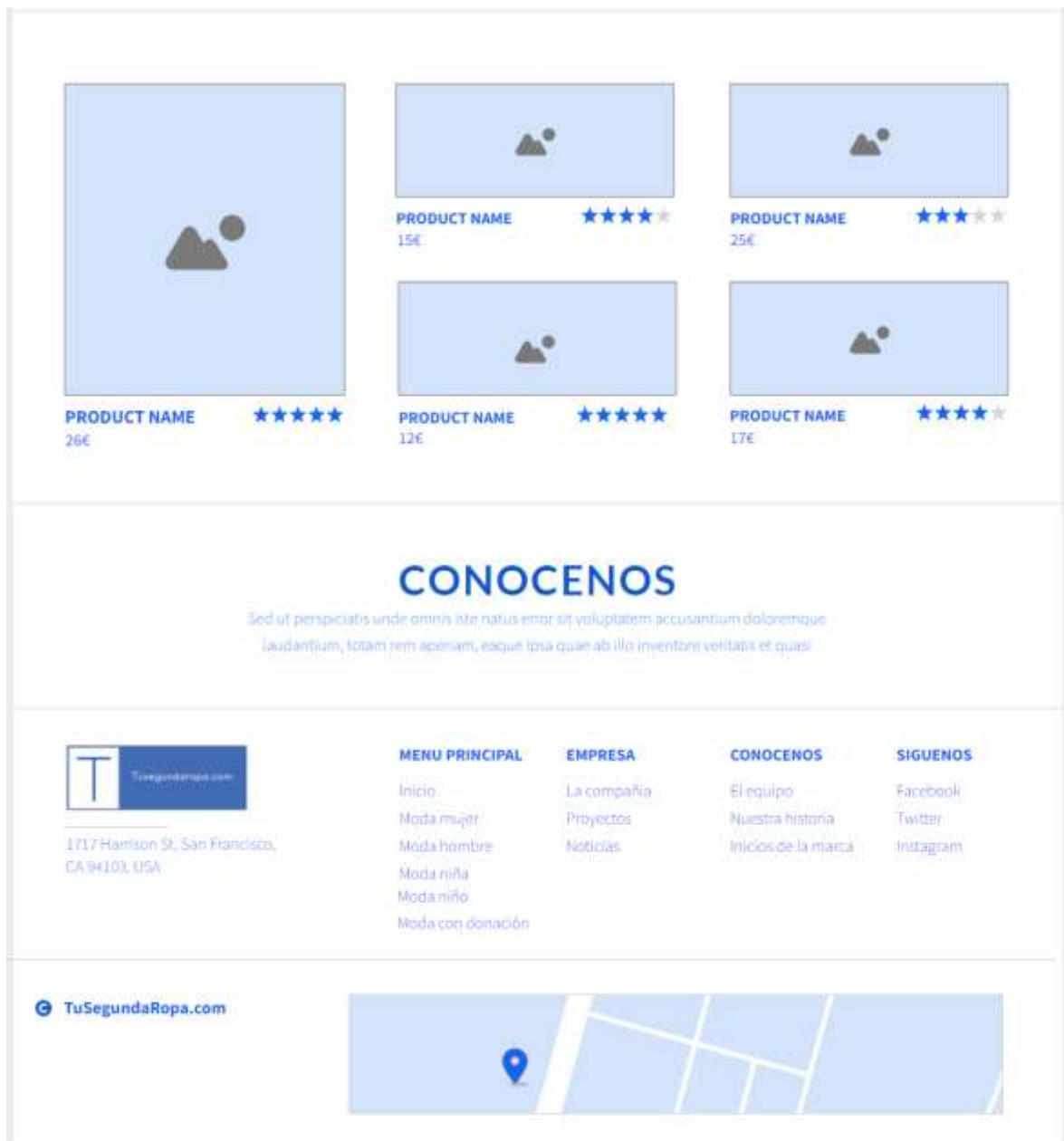
### 3.9 Definición y elaboración de la documentación del diseño. Control de calidad.

En este punto vamos a tratar de mostrar como sería el diseño de nuestra interfaz web, teniendo en cuenta que cada interfaz web iría personalizada para cada uno de nuestros clientes. Esto sería una idea aproximada de lo que podrían ser las distintas partes de nuestras webs en las cuales se podrán observar las funcionalidades creadas. Vamos a tratar de que sea una interfaz sencilla y atractiva para el usuario. La barra de navegación de la parte superior será la misma para todas las páginas.

#### **Página principal**




Como podemos observar, se trata de una interfaz minimalista y nada estridente en la cual el usuario desde el primer momento está viendo productos de la tienda del cliente lo que puede incitar a registrarse y comprar. Además, en la parte superior podremos contactar con la empresa vía email, registrarnos o hacer login y ver nuestro carrito de compra. Dependiendo del tipo de usuario que haga login, verá el panel de administrador o no. En la parte de abajo en el footer, pondremos enlaces a todas las secciones de la página, información de la empresa y enlaces a las redes sociales, además si tenemos un negocio físico podremos poner la ubicación de nuestra tienda física en un mini mapa.



### Detalle de artículo

En la página de detalle de un artículo podremos ver una imagen del mismo, las valoraciones, el precio, seleccionar la talla, comprar, comprar con donación, valorar el artículo y una sección debajo con productos parecidos al que estamos viendo en ese momento y valoraciones de usuarios del producto que estamos observando en el ese momento.



## Nombre del producto

★★★★★ 5 valoraciones

27€

Selecciona talla ▼

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation


Valorar producto

★★★★★


Comprar

Comprar con donación


### Productos similares



**PRODUCT NAME**  
★★★★★  
15€




**PRODUCT NAME**  
★★★★★  
12€




**PRODUCT NAME**  
★★★★★  
27€

### Opiniones



**JAVIER SANCHEZ** ★★★★★  
agosto 14, 2022

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in



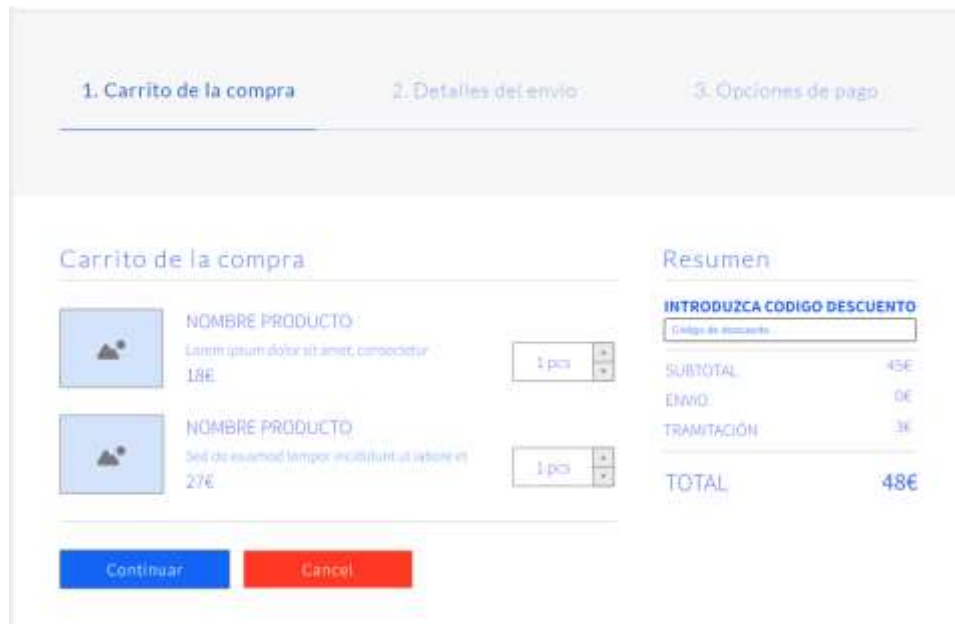
**PEDRO RUIZ** ★★★★★  
octubre 22, 2022

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in

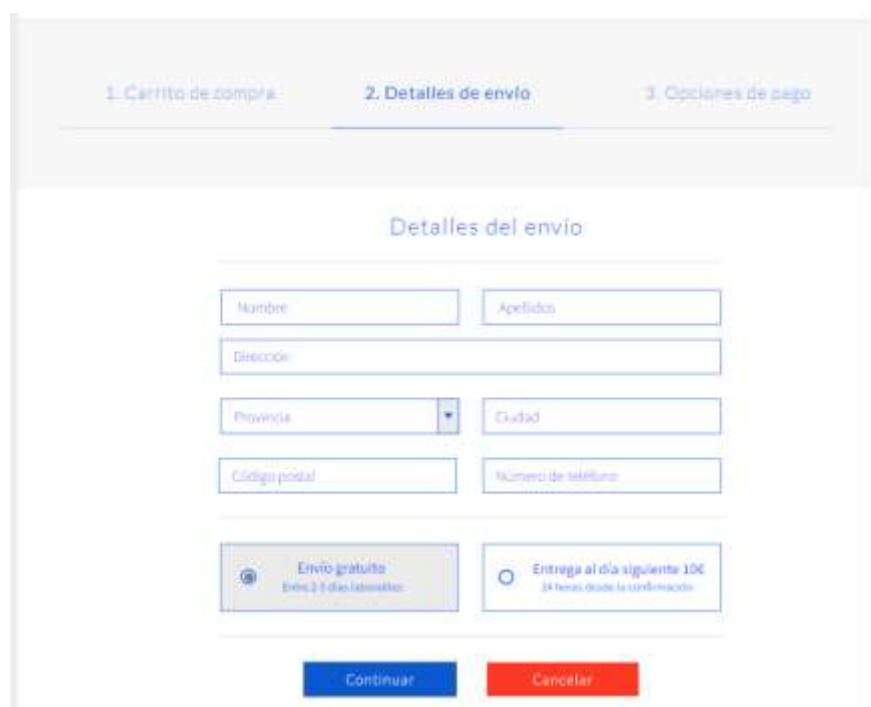
### Carrito de la compra

La parte del carrito de la compra es una parte de la web muy importante ya que es donde el usuario realiza el proceso de confirmación de su compra, introduce sus datos de envío y realiza el pago. Por lo tanto, esta parte la dividiremos en 3 secciones:

- Carrito de la compra: revisaremos si los artículos son correctos, elegiremos la talla adecuada, revisaremos el importe total y confirmaremos dándole al botón de “continuar”.



- Detalles del envío: en esta parte introduciremos los datos donde queremos que se nos envíe el producto ya que podría darse el caso que cada vez que hagamos un pedido lo queramos enviar a un sitio distinto.



- Opciones de pago: en esta sección elegiremos el método de pago ya sea por tarjeta de crédito o PayPal. Una vez elegido el método e introducido los datos, procederemos a pagar.

1. Carrito de compra

2. Detalles de envío

3. Opciones de pago

Método de pago

Tarjeta de crédito

0000 0000 0000 0000

MM / YY

CVV

Nombre usuario tarjeta

Paypal

PayPal

Pagar ahora

Cancelar

Resumen

NOMBRE PRODUCTO18€

NOMBRE PRODUCTO27€

SUBTOTAL45€

ENVÍOFREE

IMPUESTOS1€

TOTAL48€

**Panel administrador**

Este podría ser un boceto de como sería el panel de administración para mantener el stock, ver usuarios, pedidos y el panel de PowerBI en caso de tener el servicio contratado.

Panel de administrador

Listado de productos

Agregar nuevo producto

Listado de usuarios

Listado de pedidos

Panel PowerBI

Nombre Producto	Precio	Descripción	Acciones
Producto 1	12 €	simply dummy text of the printing and typesetting industry. Lorem ipsum has been	<div>EditarBorrar</div>
Producto 2	25 €	is simply dummy text of the printing and typesetting industry. Lorem ipsum has been	<div>EditarBorrar</div>
Producto 3	18 €	is simply dummy text of the printing and typesetting industry. Lorem ipsum has been	<div>EditarBorrar</div>
Producto 4	20 €	is simply dummy text of the printing and typesetting industry. Lorem ipsum has been	<div>EditarBorrar</div>

En cuanto al control de calidad de nuestro software, lo llevaremos a cabo mediante distintos tipos de pruebas a lo largo de todo el proceso de codificación.

Tal y como hemos expuesto en puntos anteriores, realizaremos los siguientes tipos de prueba:

1. Pruebas Unitarias (Durante todo el proceso de codificación).
2. Pruebas de funcionalidad usuario (Incluidas dentro de cada sprint).
3. Pruebas de estrés infraestructura (Al finalizar todos los sprint).

Para estos 3 tipos de pruebas nos basaremos en 3 puntos indispensables para nosotros:

- El acercamiento a cero defectos.
- El cumplimiento de los requisitos intrínsecos y expresos en todas y cada una de las funcionalidades.
- La satisfacción del cliente como objetivo prioritario.

También como punto de calidad de nuestro producto, se irá creando durante todo el proceso una documentación clara y detallada que ayude a entender a nuestros clientes el porqué de todas y cada una de las funcionalidades de nuestro software y además nos ayude en n futuro a encontrar posibles soluciones a fallos o bugs que puedan surgir.



### 3.10 Revisión normativa aplicable

En cuanto a normativas tendremos que tener varias en cuenta a la hora de desarrollar nuestro proyecto. La principal será **la nueva ley de protección de datos**.

Un usuario puede dejar 2 tipos de datos personales:

- Los de menor rango como el correo electrónico o el nombre, lo cuales tiene poco riesgo.
- Los de mayor rango como la información médica, la religión, las cuentas bancarias, entre otros; que si tienen mayor riesgo en cuanto a su conocimiento.

En cuanto a términos generales las normas que regulan la nueva ley de protección de datos son:

- El **RGPD** (Reglamento General de Protección de Datos).
- La **LOPD** (Ley Orgánica de Protección de Datos).
- El **RGDD** (Reglamento europeo de protección de datos).
- La **Nueva Ley de Protección de Datos y Garantía de Derechos Digitales**.
- La **Ley 41/2002**, que habla de que los datos personales están constituidos por historia clínica y todos los aspectos relacionados con temas de salud.
- El Real Decreto, a partir de la **Ley 5/2018** donde se adapta el Derecho de protección de datos de España a las normas que establece la Unión Europea para la protección de los mismos.

Por lo que, como se puede observar no solo un ente gubernamental o legal está involucrado en la nueva ley de protección de datos, sino que es un gran conjunto que hace un consenso para establecer todas las normativas.

Necesitamos garantizar obligatoriamente al usuario una serie de derechos en nuestras webs:

- Limitación del tratamiento de la información
- Derecho de acceso
- Derecho de rectificación
- Derecho de portabilidad
- Derecho de supresión
- Derecho de oposición

Nuestro software incluirá obligatoriamente 4 documentos que podrán ser leídos por los usuarios:

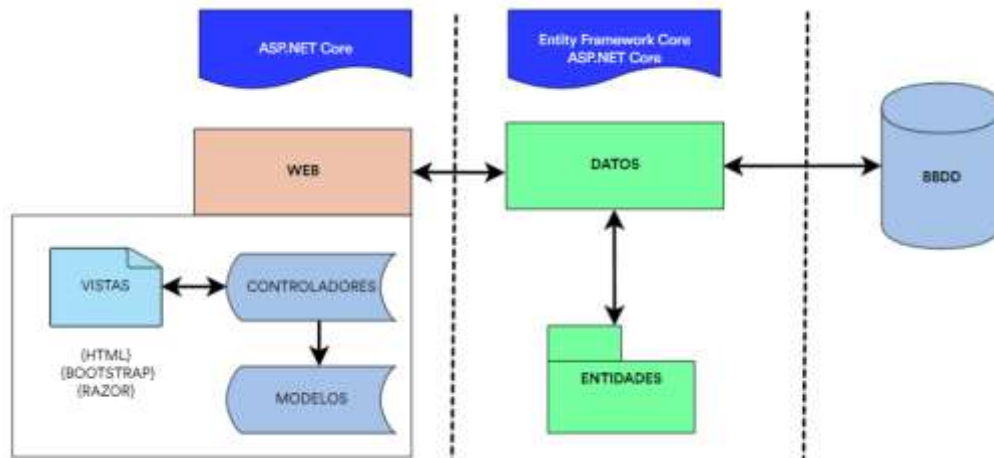
- **Aviso legal:** Recoge los datos empresariales del titular de la web.
- **Política de privacidad:** informa del medio de obtención de los datos y el tratamiento que se les va a dar a esos datos.
- **Política de cookies:** debemos avisar siempre al usuario de las cookies activas y que pueda elegir si permitir las o no.
- **Condiciones de uso y venta:** Este documento expone las condiciones de uso y venta que la web pone a disposición de los usuarios.



## 4. IMPLEMENTACIÓN

La implementación de nuestra aplicación va a seguir el siguiente esquema:

- La parte del cliente estará desarrollada con HTML, CSS, BOOTSTRAP y Razor.
- La parte de servidor estará desarrollada con ASP.NET Core y el ORM Entity Framework Core de ASP.NET basándonos en el modelo de arquitectura MVC
- Modelo de datos en BBDD relacional SQL



### 4.1 Desarrollo del modelo de datos

Cualquier aplicación que se precie necesita un buen diseño de modelo de datos. El modelado de datos es un instrumento que nos ayuda a representar la realidad. El proceso de diseño de una base de datos consiste en representar un determinado “universo de discurso (UoD)” mediante los objetos que proporciona el modelo de datos que estemos utilizando y aplicando las reglas que establezca dicho modelo. Este proceso de diseño se hace en varias etapas:

1. Diseño conceptual; suele ser realizado con el modelo E/RE y da como resultado un diagrama o esquema E/RE.
2. Diseño lógico; se elabora con el modelo relacional y da como resultado un diagrama, esquema o grafo relacional.
3. Diseño lógico específico; adaptando el diseño lógico a las características particulares del sistema gestor de bases de datos sobre el que se hará el diseño físico. Dará como resultado un diagrama, esquema o grafo relacional específico.
4. Análisis relacional; aplicando la teoría de la Normalización y comprobando si el diseño satisface una serie de restricciones agrupadas en lo que se denomina formas normales. Se obtendrá como resultado un esquema o diagrama normalizado.
5. Diseño físico; que se lleva a cabo cuando se implementa la base de datos en un sistema gestor concreto. Se obtiene como resultado el propio diseño de la base de datos (la propia base de datos).

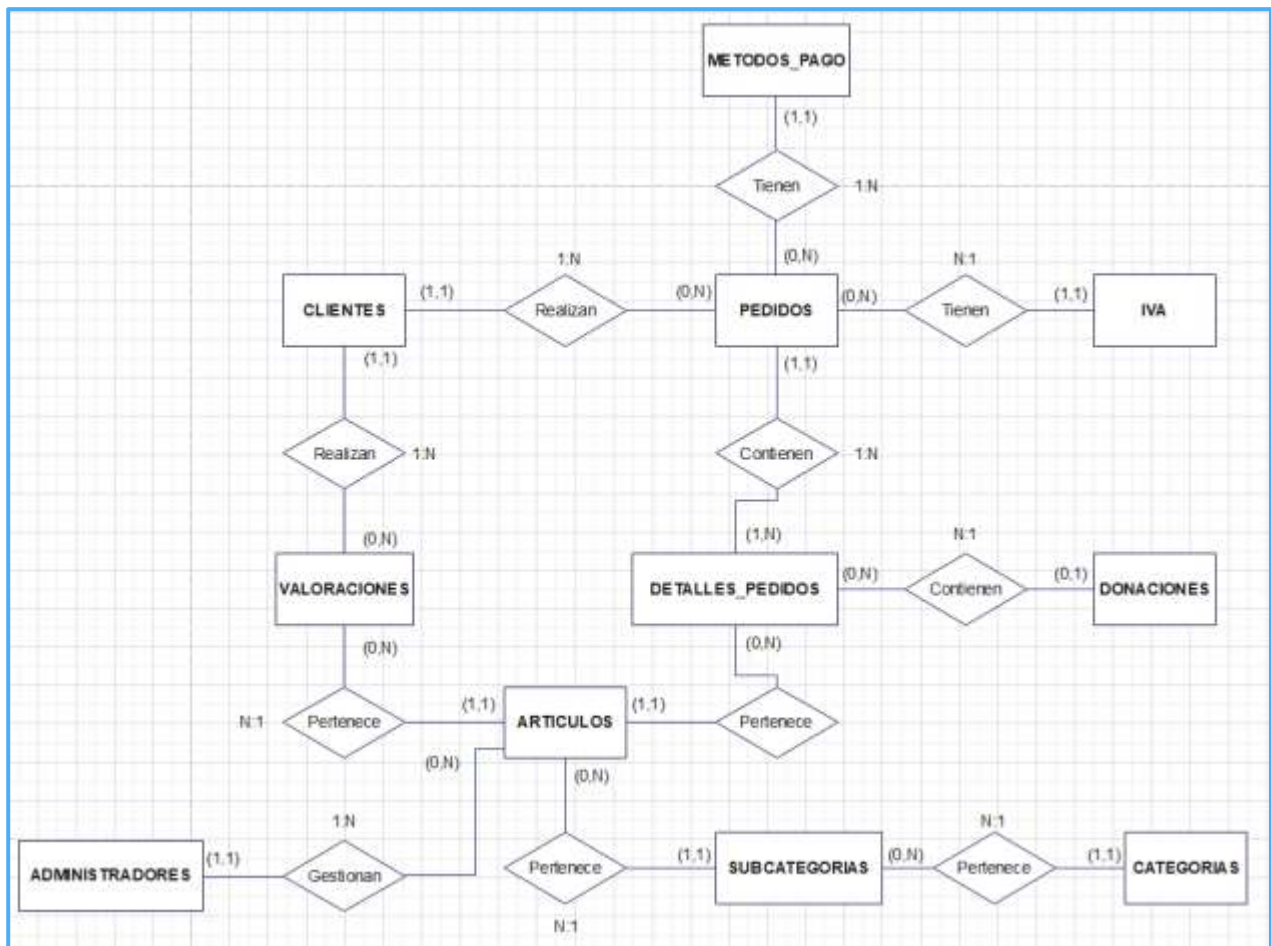
Nosotros nos vamos a centrar en los puntos 1, 2, 4 y 5:

### Modelo Entidad/Relación

El diseño de un esquema relacional es una parte fundamental dentro del desarrollo del modelo de datos ya que nos permite plasmar la realidad de las posibles interacciones de nuestra aplicación con los datos de manera visual. También nos permite detectar posibles consecuencias de un mal diseño e intentar corregirlas antes de crear la BBDD. Es importantísimo tener en cuenta que un mal diseño de una BBDD relacional puede traer a futuro una gran cantidad de errores de difícil solución, por lo tanto, el objetivo es adelantarnos y crear una estructura de tablas relacionadas entre sí lo más robustas posible para que los datos de nuestra aplicación, esto último se consigue intentando aplicar las conocidas “Formas normales”.

Una vez tengamos creado el esquema relacional, se les debería aplicar la teoría de normalización intentando que nuestra BBDD llegue a la forma normal más alta, normalmente si un diseño alcanza las primeras 3 formas normales, se considera un buen diseño.

Este sería nuestro diagrama E/R para nuestra BBDD:



**Diseño lógico**

El diseño lógico de una base de datos relacional es un proceso más riguroso y estricto que el diseño conceptual. Partiendo de nuestro esquema Entidad/Relación crearemos el esquema de diseño lógico en el cual se verán representadas las Primary Key de cada tabla y las restricciones con las Foreign Keys que contenga cada tabla y que son las que nos indicarán las relaciones directas entre tablas. En este diseño podremos apreciar además usando las reglas de normalización (3 primeras formas normales) si nuestro diseño pudiera tener algún tipo de inconsistencia de cara a futuro con los datos.

---

**CLIENTES** (id, nombre, apellidos, email, dirección, localidad, codigoPostal, teléfono, password)

PK: id

---

**PEDIDOS** (id, fecha, precioTotal, idCliente, idIva, idMetodoPago)

PK: id

FK: idCliente → CLIENTES

FK: idIva → IVAS

FK: idMetodosPago → METODOSPAGO

---

**METODOSPAGO** (id, nombre)

PK: id

---

**IVA** (id, tipo, descripcion)

PK: id

---

**DETALLES\_PEDIDOS** (id, cantidad, fecha, precioTotal, idArticulo, idPedido, idDonacion)

PK: id

FK: idArticulo → ARTICULOS

FK: idPedido → PEDIDOS

FK: idDonacion → DONACIONES

---

---

**DONACIONES** (id, nombre, cantidad, organizacion)

PK: id

---

**ARTICULOS** (id, nombre, talla, descripcion, color, precio, stock, imagen, idSubcategoria, idAdministrador)

PK: id

FK: idSubcategoria → SUBCATEGORIAS

FK: idAdministrador → ADMINISTRADORES

---

**SUBCATEGORIAS** (id, nombre, descripcion, idCategoria)

PK: id

FK: idCategoria → CATEGORIAS

---

**CATEGORIAS** (id, nombre, descripcion)

PK: id

---

**ADMINISTRADORES** (id, nombre, apellidos, email, dirección, localidad, codigoPostal, teléfono, password)

PK: id

---



### Análisis relacional de normalización

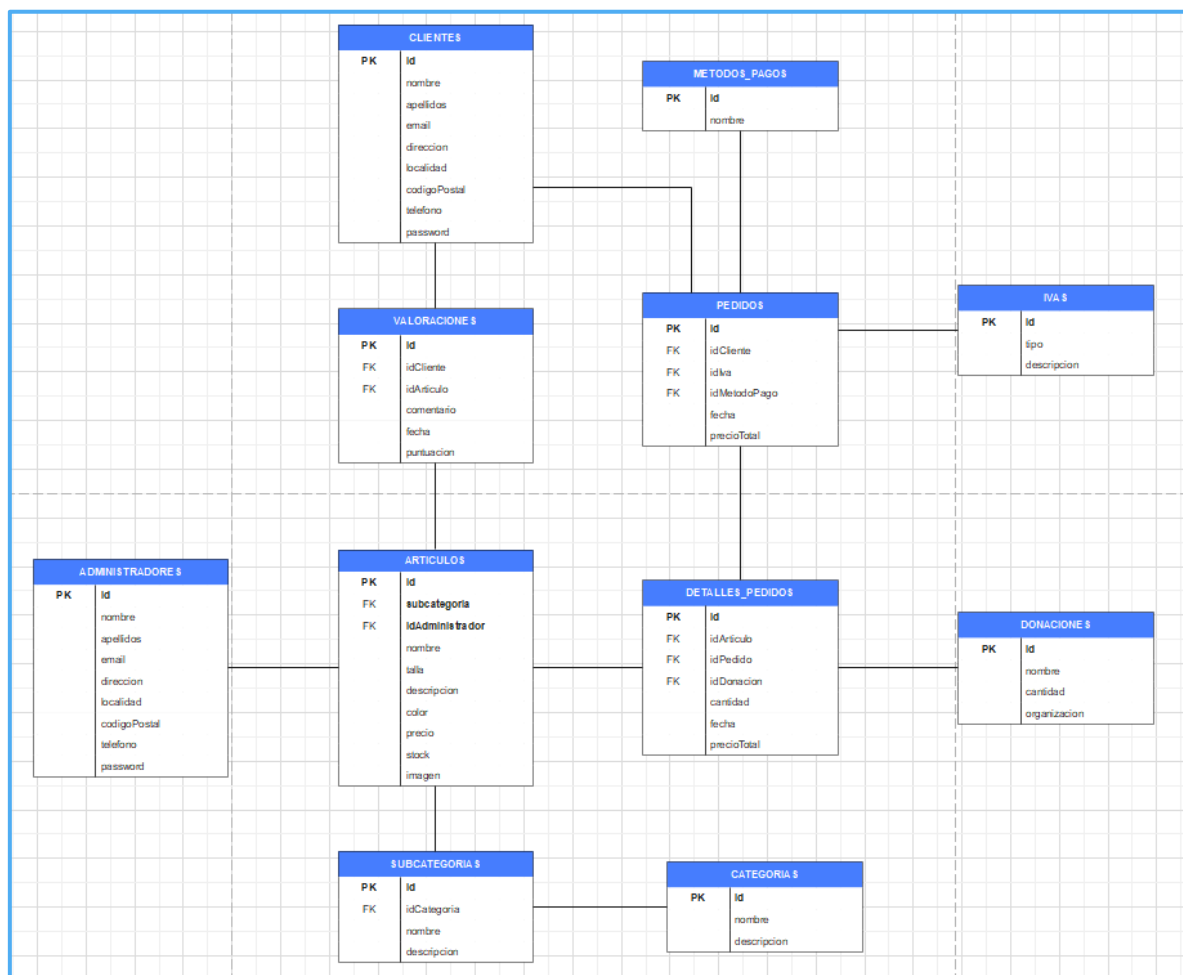
Ahora deberíamos realizar una normalización de nuestra BBDD para que al menos se cumplan las 3 primeras formas normales (existen un total de 6) y nuestra base de datos tenga una estructura consistente y robusta que evite errores futuros de inconsistencia de datos.

Vamos a explicar rápidamente a continuación en qué consisten estas 3 formas normales:

1. La **primera forma normal** se alcanza en un esquema de base de datos al establecer la restricción de clave primaria y asegurando que no habrá atributos multivaluados (sólo un valor por cada atributo de cada tupla).
2. La **segunda forma normal** podrá asegurarse si el esquema está en primera forma normal y las claves primarias de las tablas son simples (no están formadas por más de un atributo), ya que no podría haber dependencias funcionales de manera completa.
3. Para llegar a **tercera forma normal**, además de cumplir lo anterior, debe asegurarse que los atributos que no sean claves (primaria, alternativa o ajena) solamente deberán almacenar información sobre la clave primaria y no sobre atributos que no sean clave.

Por lo tanto, una vez tenemos nuestra BBDD al menos en la 3ª forma normal, podríamos continuar con el desarrollo de nuestro software y comenzar a codificar.

### Diseño físico

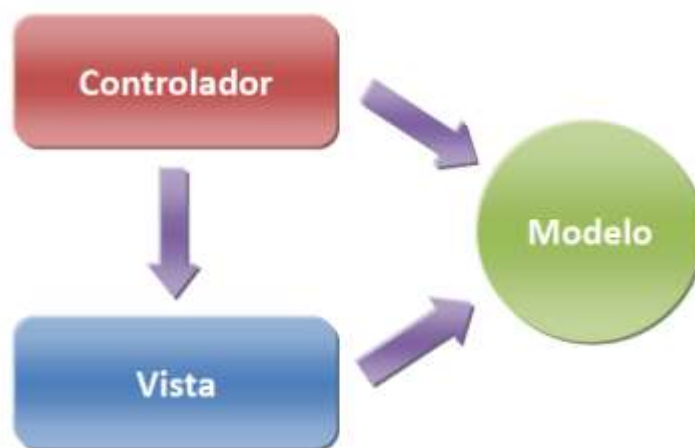


## 4.2 Desarrollo de la programación del entorno de servidor

Para el desarrollo de nuestra aplicación trabajaremos con el entorno de desarrollo Visual Studio Enterprise 2019.

Usaremos el lenguaje de programación **C#** y la tecnología de Microsoft **ASP.NET Core**, la cual utiliza el modelo de arquitectura de software **MVC** (Modelo-Vista-Controlador).

Este modelo de arquitectura nos permitirá separar la lógica de negocio de la interfaz de usuario y de la capa de datos de una manera eficiente. Además, nos permitirá manejar la complejidad de la aplicación dividiéndola en partes, evitando duplicación de código, simplificando las pruebas a realizar y facilitando enormemente el mantenimiento del código en un futuro al tener los componentes de las funcionalidades separados.



### Modelo

Representa la información que maneja la aplicación en un momento determinado y además gestiona los accesos a la BBDD para realizar las acciones de manipulación solicitadas desde el controlador. Puede estar constituido por un conjunto de clases junto con sus métodos y propiedades.

### Controlador

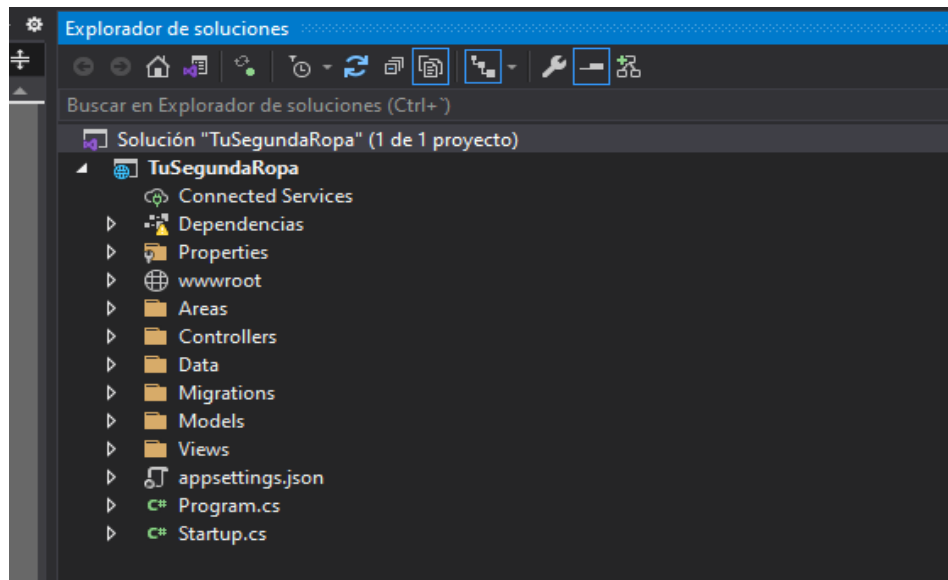
Se encarga de responder a las acciones solicitadas por el usuario, traduciendo estas acciones en alteraciones sobre el modelo o la vista. Puedes estar constituido por una clase y sus métodos. Es habitual que exista un controlador por cada una de las funcionalidades de la aplicación y es el encargado de seleccionar la vista a desplegar.

### Vista

Es la encargada de representar la interfaz de usuario a partir de los datos proporcionados por el modelo. El controlador es el encargado de solicitar la vista correspondiente a desplegar.

Importante resaltar que ni la vista ni el controlador se pueden conectar directamente a las clases del modelo, se conectan a este a través de una interfaz publica expuesta por este.

La estructura de nuestro proyecto en ASP.NET Core sería la siguiente:



### Entity Framework Core

Junto con **ASP.NET Core**, utilizaremos un ORM de Microsoft llamado **Entity Framework Core**. Esta herramienta nos va a permitir crear la BBDD a partir de las clases de datos de nuestro código utilizando una práctica de desarrollo llamada **"Code First"**. Además, esta herramienta nos permitirá realizar directamente operaciones de manipulación de datos sobre el modelo sin necesidad de realizar costosas consultas en SQL a la BBDD. Para crear las clases de datos nos basaremos en el esquema E/R creado anteriormente y en el esquema lógico para asignar las restricciones.

Dentro del modelo tendríamos las **Clases de datos**, que representan las entidades (tablas) de la BBDD, cada clase de datos tiene sus propios métodos (GET, POST, DELETE, PUSH) y sus atributos. De modo que cada instancia de un objeto de una clase de datos corresponde con una fila de una tabla de la BBDD.

Este sería un ejemplo de **clase de datos** para la tabla "Subcategorías"

```
public class Subcategoria
{
    public int Id { get; set; }

    [Display(Name = "Nombre")]
    [Required(ErrorMessage = "El nombre de la subcategoria es un campo requerido.")]
    public string Nombre { get; set; }

    [Display(Name = "Descripcion")]
    [Required(ErrorMessage = "La descripción es un campo requerido.")]
    public string Descripcion { get; set; }

    // clave ajena
    public int CategoriaId { get; set; }

    public Categoria Categoria { get; set; }
}
```



Por otro lado, tendríamos la **Clase del contexto de datos**, que se encargará de actualizar las instancias (filas) de las clases de datos en la BBDD de la aplicación. Digamos que son las clases que hacen de intermediario entre las clases de datos y las tablas reales de la BBDD.

Este sería el ejemplo para la **clase de contexto de datos**:

```
public class TuSegundaRopaContexto : DbContext
{
    public TuSegundaRopa(DbContextOptions<TuSegundaRopaContexto> options): base(options)
    {
    }
    public DbSet<Cliente> Clientes { get; set; }
    public DbSet<Pago> Pagos { get; set; }
    public DbSet<Pedido> Pedidos { get; set; }
    public DbSet<Iva> Ivas { get; set; }
    public DbSet<Detalle> Detalles { get; set; }
    public DbSet<Donacion> Donaciones { get; set; }
    public DbSet<Articulo> Articulos { get; set; }
    public DbSet<Subcategoria> Subcategorias { get; set; }
    public DbSet<Categoria> Categorias { get; set; }
    public DbSet<Administrador> Administradores { get; set; }
}
```

Una vez creadas estas clases tendríamos que realizar una **operación de migración** para poder crear la BBDD a partir de nuestras clases en el código. En todo momento, debemos asegurarnos de que existía una correspondencia entre la especificación del modelo de la aplicación Web y la estructura del esquema de la base de datos.

Las operaciones de migración pueden ser de dos tipos:

- Migración inicial, nos permite crear la BBDD a partir de las clases de datos creadas en nuestro código.
- Migración de datos, nos permiten actualizar la BBDD cada vez que se realiza un cambio en nuestro modelo de datos.

Las tareas de migración se realizan con ayuda de 2 comandos, el primero compara el esquema de la base de datos con la especificación del modelo y genera un script en la carpeta /Migrations con las instrucciones necesarias para actualizar el esquema de la BBDD. El segundo comando actualiza la BBDD con el script de migración más reciente que exista en la carpeta /Migrations:

- PM> **Add-Migration** <nombre\_migración> -context <nombre\_del\_contexto>
- PM> **Update-Database** -context <nombre\_del\_contexto>

### ASP.NET Core Identity

Para la creación de usuarios y asignación de permisos utilizaremos una tecnología de .NET llamada **ASP.NET Core Identity** la cual nos permitirá utilizar roles mediante los cuales daremos permisos de accesos a diferentes partes de nuestra aplicación dependiendo del tipo de usuario ya sea Administrador o Cliente. Para configurar esto se utilizará la clase de nuestro proyecto Startup.cs.



Ejemplo de **autorización para un usuario** tipo Administrador

```
namespace TuSegundaRopa.Controllers
{
    [Authorize(Roles = "Administrador")]
    public class ArticulosController : Controller
    {
        private readonly TuSegundaRopaContexto _context;
        public ArticulosController(TuSegundaRopaContexto context)
        {
            _context = context;
        }
        // GET: Articulos
        public async Task<IActionResult> Index()
        {
            return View(await _context.Articulos.ToListAsync());
        }
    }
}
```

### 4.3 Desarrollo de la programación del entorno de cliente

Para la parte de desarrollo del entorno cliente trabajaremos con diferentes tecnologías tales como HTML y CSS dentro del propio proyecto creado con ASP.NET Core. La idea es que la parte del cliente que se mostrará en el navegador a los usuarios sea una interfaz sencilla, moderna y la vez que le haga tener al usuario final una experiencia agradable mientras visita el sitio web.

Al estar realizando el proyecto con ASP.NET Core y el MVC (Modelo-Vista-Controlador), las paginas de nuestra aplicación que se mostrarán en el navegador serán creadas de manera dinámica con el motor generador de vistas Razor.

Razor es en realidad un lenguaje de marcado que nos permite ejecutar código de servidor dentro del código HTML en las vistas que lanzarán nuestros controladores. Por lo que, además del código de marcado tradicional HTML, podremos ejecutar código del entorno de servidor como es **C#** en todas nuestras vistas para generar contenido dinámico.

Para que Razor sepa distinguir entre lo que es HTML tradicional y código C#, utiliza el carácter **@** al inicio de una instrucción.

Un ejemplo de cómo incrustar código de servidor con Razor para mostrar una lista de usuarios en el HTML de manera dinámica podría ser el siguiente:

```
<ul>
    @for (var i = 0; i < usuarios.Length; i++)
    {
        var usuario = usuarios[i];
        <li>Nombre: @usuario.Name</li>
    }
</ul>
```

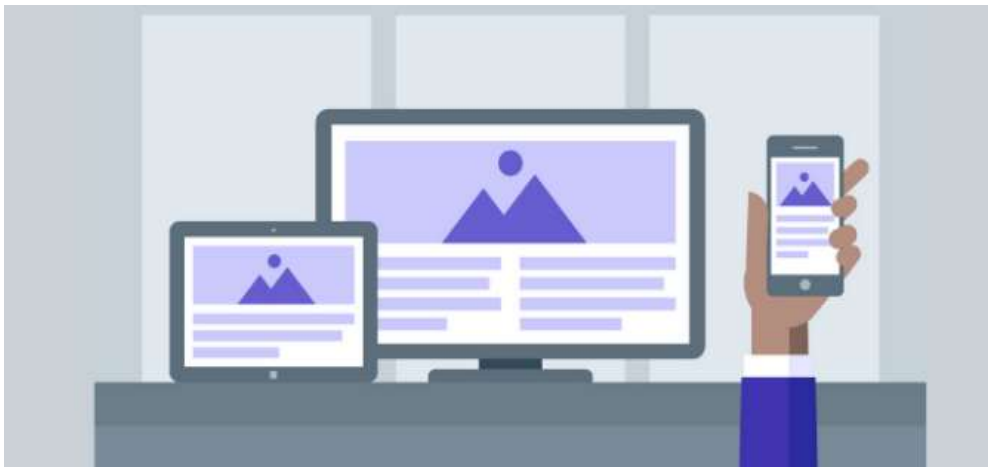
Los ficheros que contienen código de marcado de Razor suelen tener la extensión **.cshtml**. En nuestro proyecto tendremos todas las vistas correspondientes a los métodos de nuestros controladores y además tendremos un fichero compartido de vista general llamado **\_Layout.cshtml**. En este fichero tendremos la cabecera y el pie de pagina para no tener que repetir esta parte de código en todas las vistas generadas. Dentro de este fichero habrá un container que incluirá un contenedor "main" en el cual se encontrará el método **@RenderBody()** en el cual se generará de manera dinámica el código del resto de vistas.

```
<div class="container">
    <main role="main" class="pb-3">
        @RenderBody()
    </main>
</div>
```

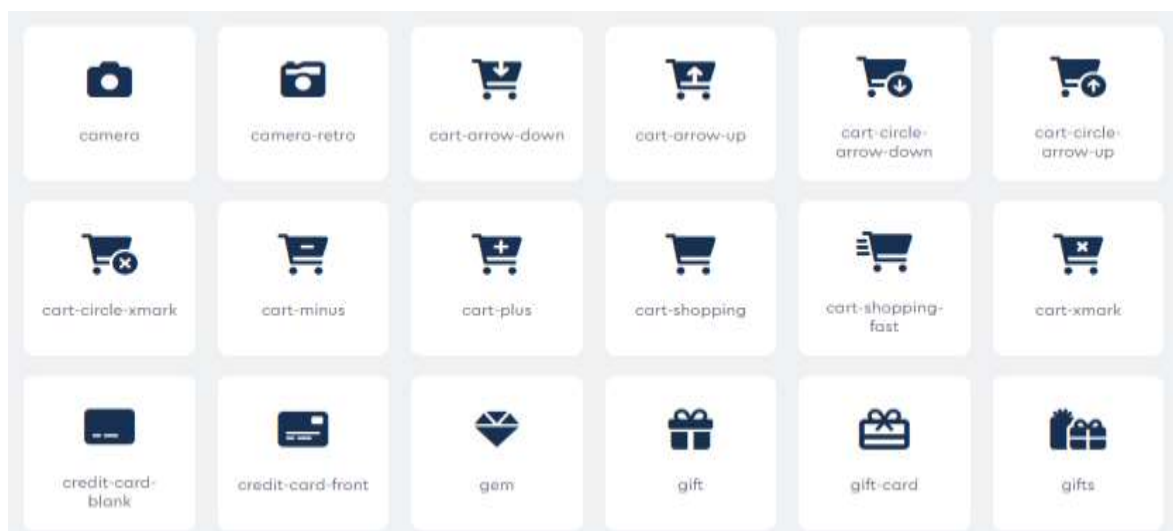
Para darle una apariencia agradable a las páginas de nuestra aplicación utilizaremos, además, la tecnología de hojas de estilo en cascada CSS y el framework Bootstrap. Con estas tecnologías conseguiremos que la experiencia del usuario sea mas agradable ya que podremos, entre otras cosas, asignar colores, tamaños de fuentes y crear efectos atractivos para el usuario final.



Además, con Bootstrap conseguiremos de una manera eficiente y sencilla que nuestra aplicación sea “Responsive”, es decir que se adapte perfectamente a cualquier tipo de dispositivo ya sea un móvil, una tableta o una pantalla de ordenador.



Por último, para los iconos que necesitemos incluir dentro de nuestra aplicación web utilizaremos los de una web llamada “Font Awesome”, en la cual podremos encontrar una gran variedad de iconos para cualquier cosa que se nos ocurra.



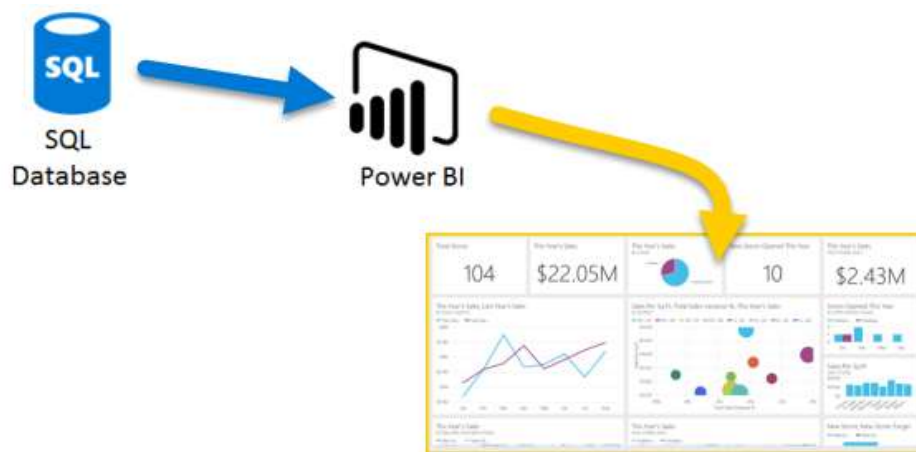
#### 4.4 Panel de PowerBI

Uno de los servicios extra que ofreceremos a nuestros clientes es el análisis de datos mediante la herramienta PowerBI, cada cliente que contrate este servicio tendrá una cuenta individual en la herramienta de análisis de datos y podrá acceder los informes siempre que lo necesite.



PowerBI es una solución de análisis empresarial basado en la nube, que permite unir diferentes fuentes de datos, analizarlos y presentar un análisis de estos a través de informes y paneles en tiempo real. Una de las ventajas de esta herramienta es que cualquier administrador de la empresa podrá acceder al análisis de datos desde fuera del portal web a través de las herramientas PowerBI Desktop, PowerBI Service y PowerBI Mobile.

Para obtener las gráficas y análisis de los datos en tiempo real, podremos conectar nuestra BBDD en Azure directamente con PowerBI de una manera fácil y eficiente ya que son ambas tecnologías de Microsoft.



PowerBI nos permite la extracción de datos desde múltiples orígenes y con muchos lenguajes tales como R, Python, SQL etc. Nosotros conectaremos con nuestra Azure SQL Database y extraeremos los datos mediante consultas SQL, con estos datos montaremos las graficas y paneles para que los clientes tengan una visión de los datos que genera su negocio atractiva y real.

Entre muchas posibilidades, podríamos mostrar los siguientes análisis de datos:

- Método de pago más usado.
- Tipo de artículo más comprado.
- Localidad desde donde se compra más en el portal.
- Donaciones por cantidad u organización.
- Pedidos organizados por importe máximo y mínimo.
- Categoría o subcategoría más demandada.

Con estos datos tendremos la oportunidad de aconsejar y orientar a nuestros clientes hacia una mejora de sus negocios añadiendo con esto un valor extra a nuestro software.

#### 4.5 Despliegue en entorno cloud

Nuestra aplicación será desplegada completamente en un entorno cloud como es Azure. Con esto conseguiremos garantizar a nuestros clientes la disponibilidad al 95% de sus aplicaciones web. Nunca bajaría de ese porcentaje ya que Azure nos lo garantiza al contratar sus servicios y este es uno de los valores añadidos del denominado “Cloud computing”.



Una de las grandes ventajas de desplegar nuestro software en un entorno cloud es la facilidad que nos ofrece de cara a escalar nuestros recursos a medida que nuestro negocio vaya creciendo. Esto quiere decir que si nuestro número de clientes aumenta, podremos ir aumentando los recursos de nuestra infraestructura de manera sencilla ya sea CPU, RAM, espacio en disco o número de instancias de máquinas virtuales necesarias. Otra de las grandes ventajas es la seguridad y privacidad que nos ofrece Azure junto con sus precios comeditos, que, si bien pueden no parecer económicos a primera vista, si son rentables ya que no nos preocuparíamos del mantenimiento de toda la infraestructura que hay por detrás (roturas, actualizaciones, sustituciones, catástrofes).

Para hacer todo esto posible, contrataremos un **App Service Plan**. Cuando contratas un **App Service Plan** lo que estas contratando es una cantidad determinada de recursos de procesamiento o computación para tu aplicación. Estos recursos de proceso son análogos a una granja de servidores de un hospedaje web convencional.

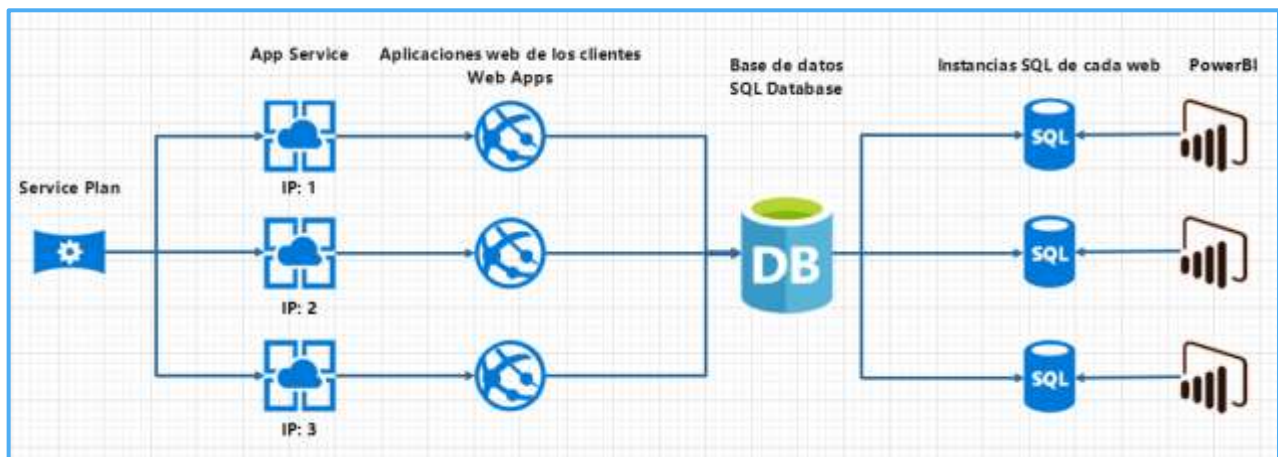
Existen varias opciones de **App Service Plan**, nosotros comenzaríamos con la **versión Estándar** ya que nos permitirá tener los recursos dedicados a nuestras APP Service, escalado automático y una cantidad ilimitada de aplicaciones web:

	Gratis Prueba gratis	Compartida Entorno de desarrollo y pruebas	Basic Entorno dedicado a desarrollo y pruebas	Estándar Ejecutar cargas de trabajo de producción	Premium Rendimiento y escala mejorados	Aislado Alto rendimiento, mayor seguridad y aislamiento
Aplicaciones web, móviles o de API	10	100	Ilimitado	Ilimitado	Ilimitado	Ilimitado
Espacio en disco	1 GB	1 GB	10 GB	50 GB	250 GB	1 TB
N.º máximo de instancias	—	—	Hasta 3	Hasta 10	Hasta 30*	Hasta 100
Dominio personalizado	—	Admitido	Admitido	Admitido	Admitido	Admitido
Escalado automático	—	—	—	Admitido	Admitido	Admitido
Conectividad híbrida	—	—	Admitido	Admitido	Admitido	Admitido
Conectividad de red virtual	—	—	—	Admitido	Admitido	Admitido
Puntos de conexión privados	—	—	—	—	Admitido	Admitido
Tipo de proceso	Shared	Shared	Dedicated	Dedicated	Dedicated	Isolated

Dentro de este **App Service Plan**, tendríamos un **App Service** para cada uno de nuestros clientes para de esta forma poder separar de manera lógica a todos ellos. Con esto conseguiremos que cada aplicación implantada tenga una IP distinta, la cual la asigna Azure de manera automática cuando se crea el propio **APP Service**. Dentro de cada App Service se pueden tener aplicaciones web ilimitadas. En nuestro caso, solamente ubicaremos una web por cada App Service para lograr un alto nivel de aislamiento entre las distintas webs.

A su vez, todas las aplicaciones web de nuestros clientes conectarán con una Azure SQL Database o lo que es lo mismo una instancia de BBDD en Azure y a su vez dentro de esta instancia tendremos separadas todas y cada una de las BBDD de nuestros clientes.

Este sería el esquema de la infraestructura cloud de lo explicado anteriormente:



Para nosotros es un servicio PaaS (Platform as a Service) ya que no nos preocuparíamos de las máquinas virtuales ni de sus sistemas operativos y para los usuarios finales sería de tipo SaaS (Software as a Service) ya que ellos entrarían a través de un navegador a consumir las aplicaciones sin importarles toda la infraestructura y configuración que hay por detrás implantada.

### ¿Cuál es el truco de todo esto?

La clave en este tipo de infraestructura es que todas las App Service dentro de un Service Plan comparten los recursos contratados, por lo que habría que tener en cuenta que a medida que nuestro negocio vaya creciendo, deberá crecer nuestro Service Plan para que nuestras App Service de cada uno de los clientes puedan seguir funcionando con holgura. Lo mismo habrá que aplicar a la Azure SQL Database.

Por lo tanto, podemos resumir todo en 2 grandes ventajas: **Alta disponibilidad y Escalabilidad.**

Creemos firmemente que este tipo de infraestructura en la nube nos permitirá tener un control óptimo sobre nuestro negocio a nivel de costes y además nos permitirá ofrecer a nuestros clientes una infraestructura para sus aplicaciones web de calidad, en un entorno respaldado por Microsoft.



## 5. VALIDACIÓN

### 5.1 Definición del procedimiento de evaluación, seguimiento y control del proyecto. Indicadores de calidad.

Una parte muy importante en la creación de software es asegurarnos que cumple con unos estándares y unos mínimos de calidad para que los futuros usuarios tengan una buena experiencia con nuestra aplicación.

Para conseguir esto realizaremos distintas pruebas durante el proceso de desarrollo:

- **Pruebas unitarias:** Estas pruebas se realizarán durante todo el proceso de desarrollo del software para confirmar que las funcionalidades de la aplicación quedan bien implementadas.
- **Pruebas de usuario:** Pruebas en las cuales comprobaremos que cada una de las funcionalidades que debe realizar un usuario en la aplicación funcionan correctamente, se realizarán a la finalización de cada sprint en la fase de desarrollo.
- **Pruebas de estrés de infraestructura:** Cuando nuestra aplicación este desplegada en nuestra infraestructura, realizaremos unas pruebas de rendimiento para verificar que la aplicación responde como debe.

Además, realizaremos varios test para asegurarnos que la accesibilidad a nuestra aplicación web es la adecuada para cualquier tipo de usuario, de esta forma conseguiremos que cualquier persona pueda disfrutar de la aplicación, aunque tenga algún tipo de minusvalía.

Seguiremos las pautas de la **WCGA 2.0** <https://www.w3.org/TR/WCAG20/> para asegurarnos que cumplimos con estos 4 principios fundamentales:

- Perceptibilidad.
- Operatividad.
- Comprensibilidad.
- Robustez.



Para ello realizaremos los siguientes test:

- **Test HTML:** Utilizando la pagina web <https://validator.w3.org/> realizaremos un test para validar que nuestro HTML tiene una correcta estructuración y que no va a resultar un problema en caso de que un usuario con discapacidad tenga que usar algún tipo de herramienta especial para hacer uso de la web. Con esto también aseguramos que cualquier usuario normal pueda acceder de manera intuitiva a las diferentes secciones de la aplicación.
- **Test CSS:** Mediante la pagina <https://jigsaw.w3.org/css-validator/> comprobaremos si nuestra aplicación web cumple con las normas y recomendaciones necesarias, en caso de que no las cumpla realizaremos las modificaciones necesarias para asegurarnos que sigue el estándar que nos indica la WCGA 2.0

Además, con la página [WAVE Web Accessibility Evaluation Tool \(webaim.org\)](http://webaim.org) realizaremos un examen exhaustivo de todo lo anterior y además nos ayudará a verificar si existe algún enlace mal, si algún icono o imagen no tienen texto alternativo etc.

Una vez tenemos esto realizaremos un análisis de accesibilidad para personas con los problemas con discapacidad:

- **Personas daltónicas:** Debemos asegurarnos de no representar mediante colores ninguna de las funcionalidades de nuestra web, lo ideal o aconsejable sería que todo esté descrito mediante texto para que los usuarios daltónicos no se vean en el problema de no saber qué hace un botón al no poder distinguir los colores bien. En caso de que tengan colores, además deberán incluir texto indicando que funcionalidad tiene esa parte de la web.
- **Personas sordas:** Los videos de la aplicación web (en caso de que existan) deberían incluir subtítulos detallados de todo lo que ocurre en el video, los subtítulos se pueden incluir directamente en el portal de YouTube ya que hace unos años se añadió la opción en el portal. Otra opción podría ser realizar una transcripción de texto de todo lo que ocurre en el video para que el usuario pueda leerlo como si de una historia se tratase al mismo tiempo que visualiza el video.
- **Personas ciegas:** Debemos asegurarnos de que todo lo que hay incluido en la web tiene descripciones de texto para que cuando un usuario ciego utilice una herramienta de lectura de nuestra web, la herramienta le pueda describir con el mayor detalle posible todo lo que aparece en la página de nuestra aplicación. Una de las herramientas más comunes y útiles para estas personas ciegas es **Screen Reader**.



En cuanto a indicadores de calidad mediante **normas ISO**, trataremos de aplicar y cumplir el estándar **ISO/IEC 25000** el cual tiene las siguientes ventajas para nosotros como empresa de cara a nuestros clientes.

- Alinea los objetivos del software con las necesidades reales que se le demandan.
- Certificar que el software aumenta la satisfacción del cliente y mejora la imagen de la empresa.
- Cumplir los requisitos contractuales y demostrar a los clientes que la calidad del software es primordial.
- El proceso de evaluaciones periódicas ayuda a supervisar continuamente el rendimiento y la mejora.
- Demostramos nuestro compromiso con la calidad del software a todos y cada uno de nuestros clientes.

Y, por último, trataremos de registrarnos en algún sistema de aseguramiento de calidad contenidos dentro de la certificación **ISO 9000**. La norma ISO 9000 describe en términos generales los elementos de aseguramiento de la calidad que se aplican a cualquier negocio, sin importar los productos o servicios ofrecidos.



Para conseguir este certificado deberemos pasar una auditoria de calidad realizada por un grupo de expertos. Una vez obtenida la certificación deberemos pasar auditorias cada 6 meses para asegurar que seguimos cumpliendo los parámetros y exigencias de la norma ISO 9000.

Estar en posesión de este tipo de certificaciones dotará a nuestra empresa de una mejor reputación y empaque de cara a obtener futuros clientes.

## 5.2 Definición de procedimientos para la participación de los usuarios en la evaluación del proyecto. Documentos específicos.

Una vez tenemos creado nuestro software, una parte fundamental es saber la opinión de los usuarios finales y si la usabilidad de nuestra aplicación cumple con los requisitos del cliente y los requisitos establecidos, por nosotros como empresa, para cumplir con los estándares planteados.

La usabilidad de una aplicación nos indica la facilidad con la que un usuario final la va a poder utilizar, debe abarcar tanto la sencillez que tenga un usuario para cumplir una acción dentro de la herramienta como que la propia experiencia del usuario sea agradable e intuitiva.

Deberíamos poder seguir los 10 principios de la usabilidad que marca el experto **Jakob Nielsen**:

1. **Visibilidad del estado del sistema:** El usuario debe saber donde está en la aplicación y que está pasando en todo momento.
2. **Relación entre el sistema y el mundo real:** la información debe estar en un orden lógico y en un lenguaje convencional.
3. **Control y libertad del usuario:** Los usuarios podrán rehacer o deshacer sus pasos al navegar.
4. **Estándares y consistencia:** No deben existir varias maneras de nombrar una acción dentro de la aplicación para no llevar a confusión al usuario
5. **Prevención de errores:** Establecemos varias formas de evitar errores en la plataforma mediante mensajes de confirmación o aviso.
6. **Reconocimiento en vez de recuerdo:** El usuario no debe memorizar las acciones que puede hacer, en todo momento tendrá de manera visible que puede o no realizar.
7. **Flexibilidad y eficiencia de uso:** Los distintos tipos de usuario de la aplicación tendrán satisfechas sus necesidades en cuanto a acciones que puedan realizar.
8. **Diseño minimalista y estético:** Todo lo superfluo será eliminado de las vistas, dejando únicamente lo relevante en cuanto a las necesidades del propietario de la web y sus clientes.
9. **Asistencia a usuarios para reconocer, diagnosticar y corregir errores:** Los errores se mostrarán a los usuarios de manera comprensible para que los puedan entender fácilmente.
10. **Ayuda y documentación:** Añadiremos una lista de preguntas frecuentes para que el usuario en caso de duda con alguna acción pueda ir a consultar.

Para realizar este tipo de pruebas de usabilidad contactaremos con un grupo de 5 personas aleatorias y que estén dispuestas a seguir unas pequeñas instrucciones para probar las funcionalidades de uno de ellos actores de nuestra aplicación. Seguiremos el **método de pruebas de usabilidad moderadas** ya que este método nos va a permitir estar en contacto con los usuarios que realicen las pruebas para aclararles dudas o que ellos nos expliquen las suyas.

Para cada uno de los actores intervinientes en nuestra aplicación, seleccionaremos un grupo de 5 voluntarios y definiremos que funcionalidades debe probar cada grupo:

**Usuario no registrado:**

- Visualizar todas y cada una de las páginas de la aplicación que se le permite a un usuario no registrado.
- Verificar que todas las imágenes son correctas y corresponden con el producto indicado.
- Ver los artículos añadidos en la aplicación de cada una de las secciones
- Buscar artículos en la aplicación que existen y otros que no
- Ver el detalle de los artículos
- Añadir artículos al carrito de compra
- Verificar que no puede realizar una compra
- Verificar que no puede valorar un artículo
- Verificar que no puede acceder a través de la barra de navegación a secciones que no tiene permitidas
- Realizar un registro y comprobar que se realiza correctamente.
- Enviar mail de contacto para comprobar que se abre la aplicación de correo al darle al enlace.

**Usuario Registrado:**

- Realizar login con una cuenta de usuario ya registrado y verificar que se accede a la aplicación con ese usuario
- Realizar un login con una cuenta inexistente para probar que no podría loguearse en la aplicación.
- Probar las funcionalidades de un usuario no registrado para comprobar que las puede realizar estando logueado
- Realizar una compra normal y otra con donación para verificar que todo el proceso funciona correctamente
- Valorar productos de la web para verificar esta funcionalidad
- Ver el histórico de sus compras, en caso de no tener compras verificar que el histórico muestra un mensaje indicando que no ha realizado compras en el portal
- Verificar que no puede acceder a las funcionalidades de un usuario administrador

**Usuario administrador:**

- Realizar login como administrador
- Ver panel de PowerBI enlazado a la web
- Ver el listado de pedidos realizados en la web
- Ver el listado de usuarios registrados en la web
- Verificar que puede acceder al panel de artículos
- Realizar la acción de agregar artículos en la aplicación
- Realizar la acción de editar artículos en el panel de artículos
- Realizar la acción de eliminar artículos en el panel de artículos
- Verificar que cuando añade, edita o elimina un artículo, la acción se ve reflejada en la web

Una vez tengamos los resultados de las pruebas realizadas por los 3 grupos de usuarios para cada uno de los actores de nuestra aplicación, analizaremos todas las opiniones aportadas y además usaremos unas métricas para medir la usabilidad de los usuarios.

Aplicaremos estas 3 métricas de usabilidad:

1. **Eficacia:** Facilidad con la que un usuario ha podido realizar las acciones (tareas en el primer intento, errores encontrados, etc)
2. **Eficiencia:** Tiempo en poder realizar cada acción.
3. **Satisfacción:** Opinión completamente personal de los usuarios respecto a la experiencia.

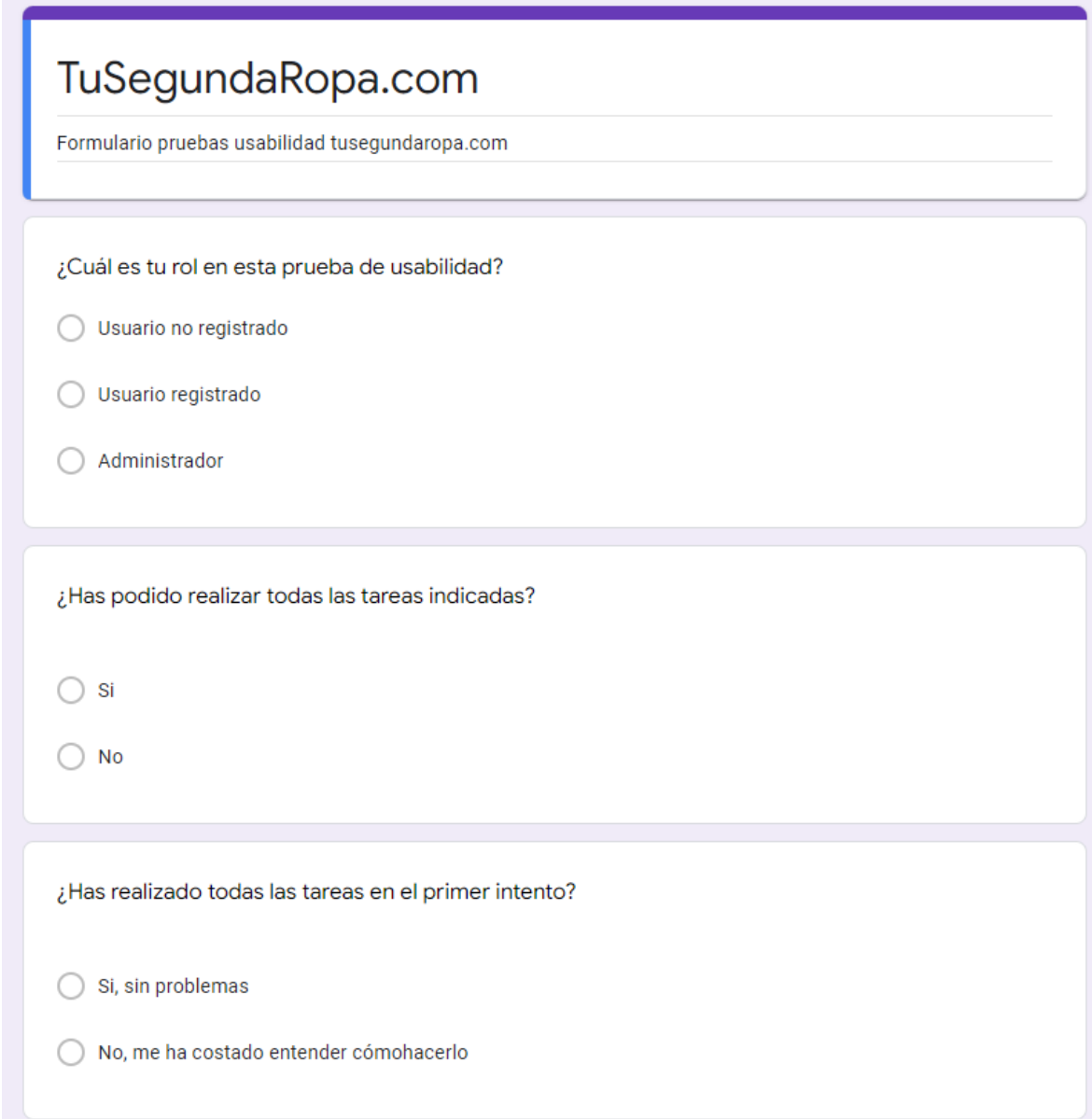
Para poder aplicar todo lo anterior, será necesario recoger toda la información de estos usuarios en un formulario el cual tendrán que rellenar completamente y enviar por correo, indicando en el asunto a que grupo pertenecen. Este formulario lo detallaremos en el siguiente punto.



### 5.3 Registro de resultados.

Con el fin de poder analizar y tener un registro de las pruebas de usabilidad realizadas, deberemos proporcionar a nuestros usuarios voluntarios un formulario que deberán rellenar y enviar obligatoriamente. Este formulario será creado en la plataforma Google Forms ya que nos permitirá posteriormente realizar un análisis con graficas de los resultados obtenidos.

Este formulario incluirá los siguientes puntos o preguntas:



The image shows a Google Forms interface for a usability test. The title is 'TuSegundaRopa.com' and the subtitle is 'Formulario pruebas usabilidad tusegundaropa.com'. There are three questions, each with radio button options.

**TuSegundaRopa.com**  
Formulario pruebas usabilidad tusegundaropa.com

¿Cuál es tu rol en esta prueba de usabilidad?

- ☐ Usuario no registrado
- ☐ Usuario registrado
- ☐ Administrador

¿Has podido realizar todas las tareas indicadas?

- ☐ Si
- ☐ No

¿Has realizado todas las tareas en el primer intento?

- ☐ Si, sin problemas
- ☐ No, me ha costado entender cómo hacerlo



¿El tiempo en realizar cada acción te parece apropiado?

- ☐ Si, se tarda poco en realizarlas
- ☐ No, es todo confuso

¿Te ha resultado intuitiva la aplicación?

- ☐ Si, mucho
- ☐ Algo intuitiva
- ☐ No es intuitiva

¿Te ha parecido que todo sigue un orden lógico dentro de la web?

- ☐ Si
- ☐ No
- ☐ Un poco lioso todo

¿Has encontrado errores en la web o cosas a mejorar?

- ☐ Si
- ☐ No

¿Utilizarías la web para realizar compras de ropa de segunda mano?

- ☐ Sí
- ☐ No
- ☐ Tal vez

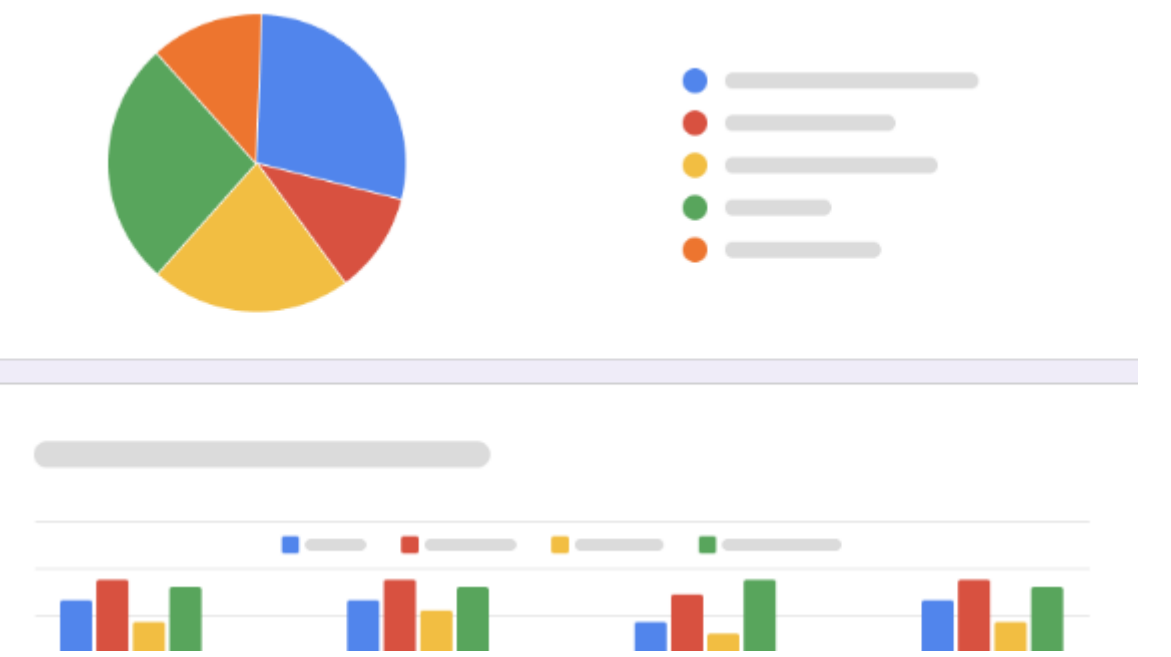
Comenta que fallos ves en nuestra aplicación

Texto de respuesta larga

Danos tu opinión personal sobre nuestra aplicación web, tu opinión será tomada en cuenta e intentaremos corregir lo que no te haya gustado del portal

Texto de respuesta larga

Quando tengamos todos los resultados de nuestros voluntarios, realizaremos un estudio de las respuestas y de las pruebas realizadas para ver si cumplimos con la exigencia de nuestro proyecto, solo en caso de que una amplia mayoría de respuestas sea positiva, aprobaremos el lanzamiento en producción nuestra aplicación web.



## 6. CONCLUSIONES

## 7. BIBLIOGRAFIA Y REFERENCIAS