Prototipo de ecommerce web APP

Autor: Manuel Sánchez Manzanares
Tutor: Jordi Ustrell Garrigos.
Profesor: Ferran Adell Español

Grado de Multimedia
Ingeniería Web

8 de diciembre de 2021

Copyright

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Prototipo web de comercio electronico*
Nombre del autor:	*Manuel Sánchez Manzanares*
Nombre del colaborador/a docente :	Jordi Ustrell Garrigos
Nombre del PRA:	Ferran Adell Español
Fecha de entrega (mm/aaaa):	*10/2021*
Titulación o programa:	*Grado de Multimedia*
Área del Trabajo Final:	*Ingeniería Web*
Idioma del trabajo:	*Español*
Palabras clave	*Memoria, TFG, ecommerce*

Resumen del Trabajo (máximo 250 palabras):

Este TFG desarrolla un prototipo semi funcional de un *website* de comercio electrónico. Este es el primer paso de cara a satisfacer el requerimiento de un cliente real que regenta una tienda física y desea vender sus productos y servicios a través de internet

El desarrollo se orienta a la puesta en práctica de competencias de *full stack development*, con un *frontend* basado en Next.js/React, una API REST basada en Node.js/Expres.js, *testing* basado en Jest, Supertest y Puppertee y el despliegue en un servidor Linux mediante una instancia EC2 en Amazon AWS.

El resultado final del prototipo difiere de la versión final de la aplicación en el uso de una plataforma de pago real y de la conexión a base de datos real de un proveedor externo de *dropshipping*

Abstract (in English, 250 words or less):

This TFG develops a semi-functional ecommerce website prototype. This is the first step to satisfy the requirement of a real client who runs a physical store and wants to sell their products and services through the internet.

The development seeks the implementation of full stack development skills, with a frontend based on Next.js / React, a REST API based on Node.js / Expres.js, testing based on Jest, Supertest and Puppertee and the deployment on a Linux server using an EC2 instance on Amazon AWS.

The result of the prototype differs from the final version of the application in the use of a real payment platform and the connection to a real database of an external dropshipping provider

Notaciones y Convenciones

Titulo 1

Nombre del capítulo. Tipografía Arial, negrita, tamaño 20.

Titulo 2

Apartado dentro del capítulo. Tipografía Arial, negrita, tamaño 13.

Titulo 3

Subapartado dentro del capítulo. Tipografía Arial, negrita, tamaño 13.

Xenismo

Extranjerismo que conserva su grafía original. Tipografía Arial, cursiva, tamaño 10.

Texto

Textos generales de la memoria. Tipografía Arial, tamaño 10, con interlineado de 1,15 y texto justificado.

Índice

Figuras y tablas

Índice de figuras

Índice de tablas

1. Introducción

1.1. Prefacio

A medida que superaba las asignaturas de la titulación, mis intereses se reorientaban hacía la programación y, en particular, a las disciplinas y tecnologías que conforman el vasto mundo del desarrollo web. Ahora, una vez sentadas las bases, es el momento de abordar un proyecto integral que combine las distintas herramientas que empiezo a dominar con otras nuevas que aprenderé durante este proceso.

Este proyecto busca desarrollar un prototipo de aplicación web que responde la necesidad de disponer de un *website* de una persona de mi entorno, que jugará el rol de cliente a lo largo de este trabajo. Actualmente, el cliente regenta una tienda física y su deseo es disponer de una aplicación web, que le permita vender sus productos y servicios en internet. Además, quiere aprovechar la oportunidad para trabajar en paralelo con proveedores de *dropshipping*.

Finalmente, dado que mi objetivo principal es crecer como desarrollador, este proyecto se lleva cabo haciendo uso de algunas tecnologías que desconozco. Esto me permitirá medir, además de mis conocimientos actuales, mi capacidad de aprender sobre la marcha, algo que considero fundamental en un desarrollador

1.2. Descripción/Definición

Este proyecto nace para dar respuesta a la necesidad de un cliente. Este, actualmente, dispone de una página web que solo proporciona información sobre los servicios que ofrece en su tienda física, pero no está orientada a la venta online. El cliente, para cubrir este vacío, desea remodelar su *website* actual para que su stock sea accesible desde internet. Además, algunos de sus proveedores ofrecen la posibilidad de *dropshipping*.

Al margen de todas las ramificaciones que se derivan de un *website* de comercio electrónico (y que abordaremos más adelante), encontramos dos casos fundamentales: la venta de productos mediante dropshipping y la venta de stock propio. Si bien esto último requiere una API REST que permita operaciones CRUD, el servicio de *dropshipping* provee un catálogo, en formato XML o CVS, que contiene los campos con la información de cada producto:

El auge del comercio electrónico ha derivado en una extensa oferta de productos que cubren ambas modalidades de venta online. En la actualidad, es posible comprar una aplicación web “prefabricada” y orientada al *dropshipping*. Además, las empresas que ofrecen estos productos se encargan de facilitar a sus clientes algunos proveedores. Básicamente, el sistema se reduce a elegir lo que quieres vender, seleccionar una plantilla para la interfaz de la web y pagar para tener operativo un negocio rápidamente. Por su parte, la tendencia de los negocios físicos a mostrarse en internet ha sido imparable en los últimos años. No obstante (especialmente en los pequeños negocios), aun es fácil encontrar *websites* que, por diversos motivos, no son funcionales: tecnologías arcaicas, mala implementación, webs estáticas, dificultad para sincronizar el stock físico con el catálogo de la web y un largo etcétera

La temática abordada es tan relevante hoy en día que, si bien este prototipo se enfoca a un caso particular, no tendría sentido desarrollarlo sin tener en cuenta factores como la escalabilidad y la reusabilidad. El desarrollo del *frontend*, al llevarse a cabo sobre REACT, permitirá un esquema de componentes lo suficientemente flexible como para permitir, sin dificultad, la reformulación de la interfaz para adaptarla a futuros clientes. En el caso de *backend*, se creará una estructura que permita trabajar con stock propio, mediante *dropshipping* y, como es el caso que nos ocupa, con una fusión entre ambos conceptos.

El prototipo de aplicación web difiere de la versión final en dos elementos fundamentales.

- Pago: en lugar de contar con un servicio real de pasarela de pago, en el prototipo este proceso es simulado.
- Datos del proveedor: en lugar de descargar un archivo XML/CVS desde un *endpoint* de un proveedor real, en el prototipo se hará uso de un archivo XML/CVS estático para simular la importación de un catálogo externo de productos.

1.3. Objetivos generales

1.3.1. Objetivos principales

Objetivos de la aplicación:

- Simular el proceso de venta online de componentes informáticos
- Trabajar indistintamente con stock propio o con proveedores de dropshipping
- Automatizar la realización de pedidos del administrador al proveedor
- Ofrecer un canal de comunicación asíncrona entre administradores y usuarios
- Permitir operaciones CRUD sobre los productos de stock propio
- Permitir la búsqueda y filtrado de producto
- Ofrecer un sistema robusto de autenticación
- Ofrecer una zona privada para administradores y usuarios
- Ofrecer un diseño responsive para los dispositivos más comunes

Objetivos del cliente:

- Introducirse en el comercio electrónico
- Posicionarse en el comercio electrónico
- Automatizar la venta de productos

Objetivos personales del autor del TF:

- Aprender a manejar nuevas tecnologías
- Ganar soltura en las tecnologías ya conocidas
- Ganar experiencia de cara al manejo de presupuestos
- Ganar experiencia en la gestión de proyectos

1.3.2. Objetivos secundarios

Objetivos adicionales.

- Generar una documentación clara y completa
- Conseguir cierto nivel de automatización en el despliegue

1.4. Metodología y proceso de trabajo

Para poder desarrollar una metodología sólida, que permita un proceso de trabajo fluido, se requiere contextualizar las particularidades del proyecto, que son las siguientes:

- Este trabajo contiene plazos inmutables de cara a la entrega de las distintas PECs.
- El prototipo de aplicación web responde al encargo de una cliente real.
- El cliente tiene claros muchos aspectos relacionados con el diseño de la interfaz
- El cliente desea un feedback continuo que permita un desarrollo flexible
- El desarrollo es llevado a cabo por una sola persona
- El desarrollador está aprendiendo algunas de las tecnologías que usa sobre la marcha, lo cual impide hacer estimaciones de tiempo con exactitud.

Teniendo en cuenta lo anterior, se ha optado por un enfoque basado en la metodología *agile*. Este, al ser un marco de trabajo con métodos iterativos, ofrece varias ventajas. Al dividir el trabajo en ciclos o etapas, permite introducir el testeo, el despliegue y la revisión como herramientas fundamentales en cada uno de estos ciclos de producción.

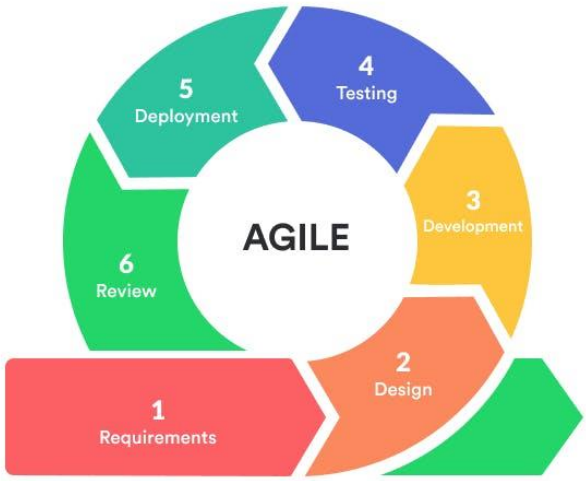

Figura 1: Diagrama de la metodología *agile*

El despliegue es automatizado por medio de la herramienta *GitHub Actions*. Por su parte, el proceso de testeo sigue la metodología *Test-driven development* o TDD, que se basa en guiar el diseño mediante test en lugar de reducir el *testing* a superar los test primero. Por tanto, cada ciclo asegura un entregable, testeado y desplegado en un entorno de preproducción, que permite al cliente constatar la evolución del proyecto mediante resultados tangibles y funcionales.

Teniendo en cuenta la estructura de plazos de las PECs, el desarrollo de este proyecto se ha dividido en cinco fases

- Fase 1: Inicio, donde se fusionan las clásicas fases de requisitos y análisis.
- Fase 2: Diseño, donde se define la estructura de la aplicación, la arquitectura de la información y el diseño de la interfaz.
- Fase 4: Desarrollo de la aplicación
- Fase 5: Cierre, con la entrega del proyecto en la última PEC y su defensa.

Dentro de la metodología *agile*, haremos uso de su variante Kanban para organizar el flujo de trabajo. Así, el desarrollo de la aplicación queda dividido en entregables/ciclos que responden las siguientes funcionalidades de orden superior:

- Walking skeleton
- Configuración del servidor
- Automatizar despliegue
- API
- Registro, login y logout
- Formularios
- Zonas privadas
- Importación de productos externos
- Filtrado de productos
- Subir productos propios
- Comunicaciones por mail
- Simulación proceso de compra
- Integración en RRSS
- Responsive

Esto nos permite, en un primer momento, abstraernos de abundantes subtareas y dependencias, pues estás son susceptibles de ser modificadas a medida que se avanza en el desarrollo de la aplicación. El método Kanban se basa identificar las tareas con tarjetas que se disponen sobre un tablero. Para ello, usaremos la versión gratuita de la herramienta Trello. La disposición inicial del tablero se basa en una primera estimación de las necesidades de organización.

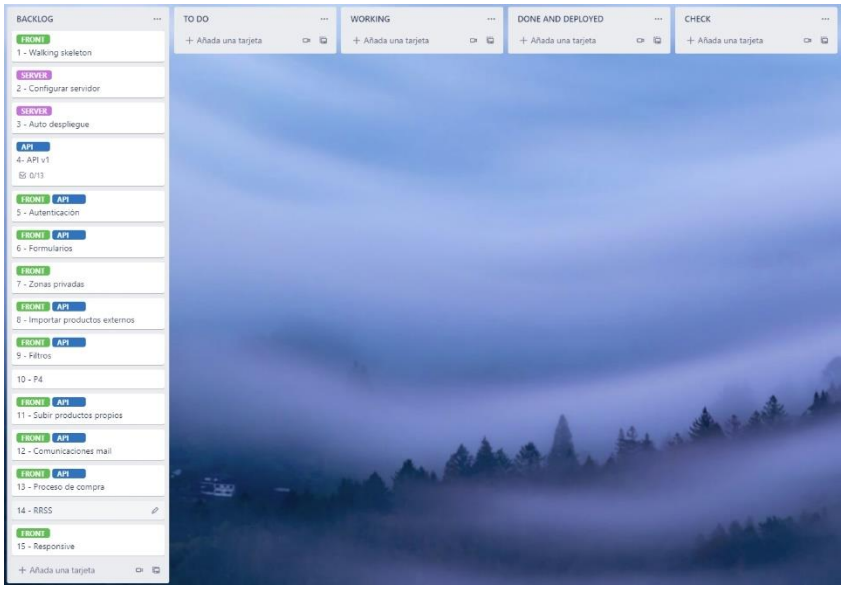

Figura 2: Tablero de Trello usado en este trabajo

Las tarjetas contendrán una *checklist* con la previsión inicial de subtareas y dependencias, que serán susceptibles de ser modificadas en cualquier momento si el contexto lo requiere.

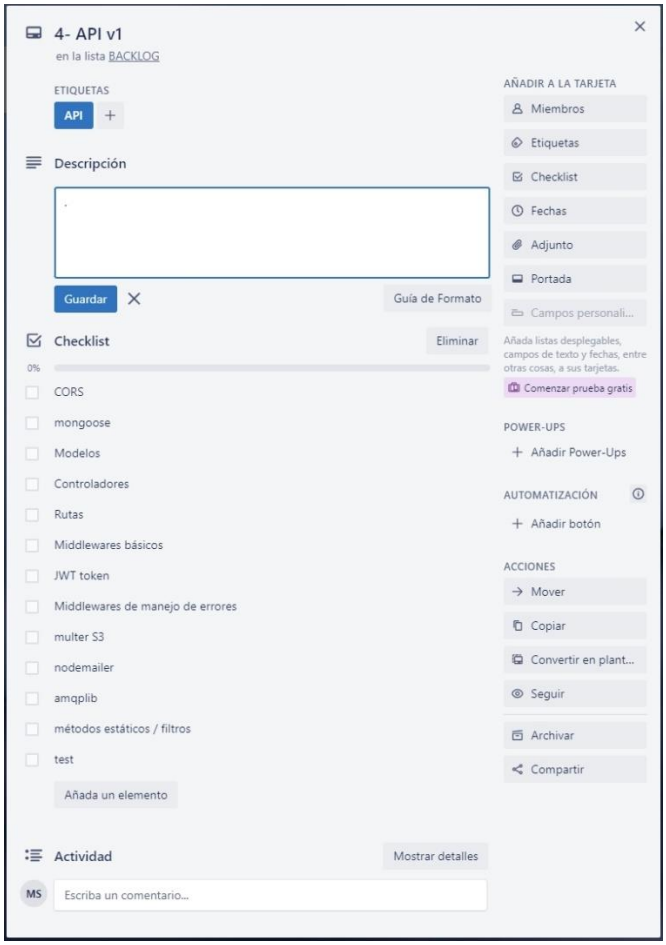

Figura 3: Tarjeta de Trello usada en este trabajo

1.5. Planificación

Teniendo en cuenta la estructura de plazos de las PECs, el desarrollo de este proyecto se ha dividido en cinco fases

- Fase 1: Inicio, donde se fusionan las clásicas fases de requisitos y análisis.
- Fase 2: Diseño, donde se define la estructura de la aplicación, la arquitectura de la información y el diseño de la interfaz.
- Fase 4: Desarrollo de una primera versión de la aplicación. Este queda dividido en los siguientes entregables/ciclos que responden las siguientes funcionalidades de orden superior:
    - Walking skeleton
    - Configuración del servidor
    - Automatizar despliegue
    - API
    - Registro, login y logout
    - Formularios
    - Zonas privadas
    - Importación de productos externos
    - Filtrado de productos
    - Subir productos propios
    - Comunicaciones por mail
    - Simulación proceso de compra
    - Integración en RRSS
    - Responsive
- Fase 5: Cierre, con la entrega del TFG en la última PEC y su defensa.

A continuación, se muestra el diagrama de Gantt con una estimación de las fechas clave. Al trabajar con algunas tecnologías desconocidas, se optado por una planificación conservadora en cuanto a los tiempos de desarrollo. En cualquier caso, el diagrama de Gantt ofrece una visión general que complementa el tablero de Trello y facilita y contextualiza la consecución de los entregables.

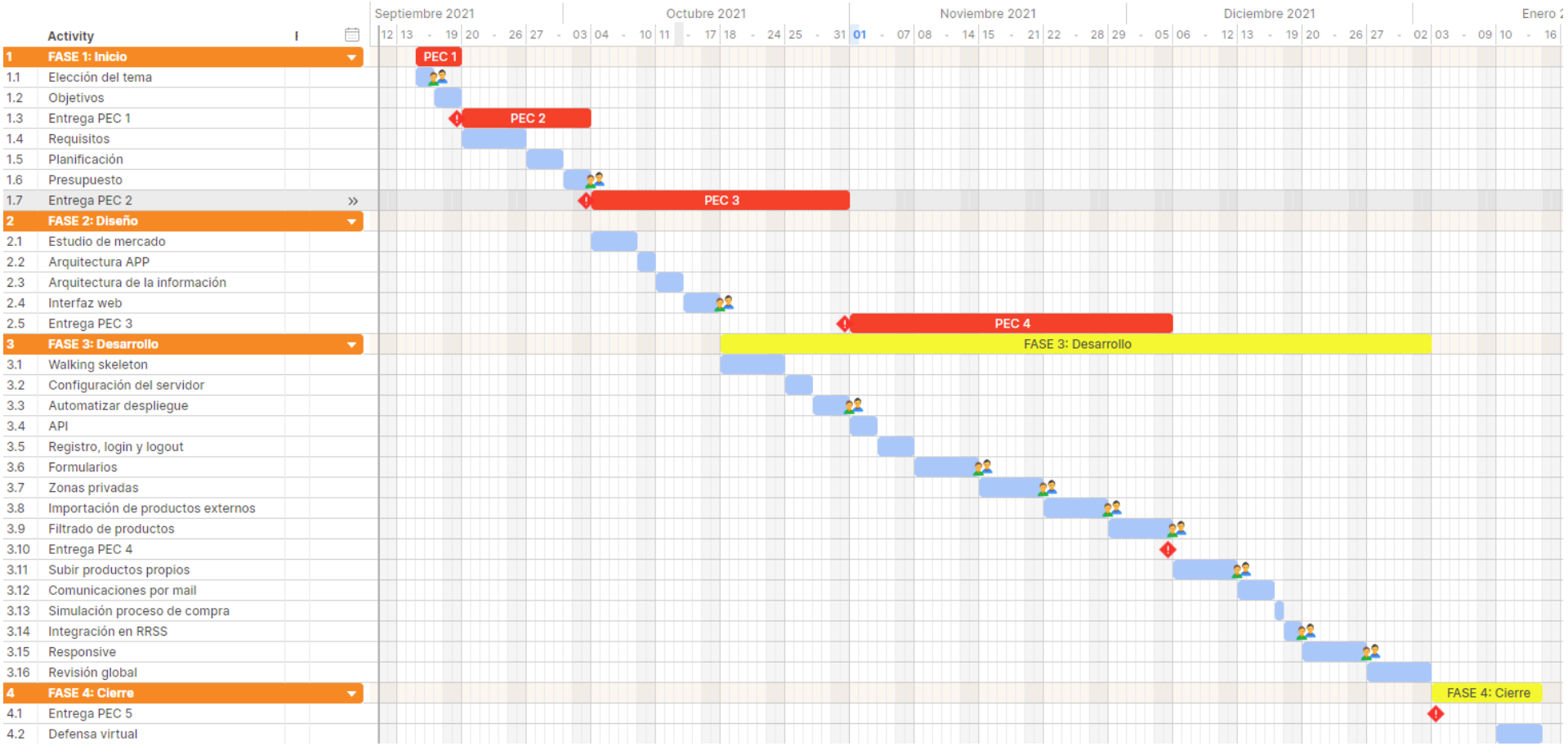

Figura 4: Diagrama de Gantt usado en este trabajo

1.7. Presupuesto

Dado que el proyecto se basa en un prototipo, este presupuesto es una estimación meramente informativa de los costes que supondría la ejecución de una versión completamente funcional de la aplicación. Al tener poca experiencia como desarrollador y estar usando tecnologías que me son desconocidas, la estimación de horas necesarias para completar una tarea no es demasiado exacta. Como la planificación del apartado anterior es conservadora en los tiempos, la aproximación se ha llevado a cabo basándome en el diagrama de Gantt y computando una media de 2 horas de trabajo por día para la mayoría de las tareas.

Concepto	Horas	Precio hora	Total
Equipo humano			**3.345€**
Análisis			**375€**
Estudio de requisitos	10	15€	150€
Estudio de mercado	15	15€	225€
Diseño			**600€**
Arquitectura APP	5	15€	75€
Arquitectura de la información	10	15€	150€
Interfaz web	25	15€	375
Desarrollo			**2.250€**
Implementación	150	15€	2.250€
Despliegue			**120€**
Configuración del servidor	8	15€	120€
Equipo técnico			**26,20€**
Instancia EC2 en AWS	160	0.07€	11,20€
Dominio			15€
Total			**3.371,2€**

Mesa 1: Presupuesto

1.8. Estructura del resto del documento

El segundo capítulo de la memoria esta se dedica a un análisis de mercado que permita definir el target de la aplicación y contextualice el estado actual de la competencia.

En el tercer capítulo se concreta la propuesta que ofrece este trabajo como factor diferenciador de cara a su posicionamiento en el mercado.

El cuarto capitulo aborda el diseño del porfolio y de la aplicación, tanto en su vertiente técnica como visual.

En el quinto capitulo se documenta la aplicación desde el punto de vista de su instalación y uso.

Por último, en el sexto capitulo se trazan las conclusiones extraídas a lo largo del desarrollo de este trabajo.

2. Análisis de mercado

A la hora de estructurar el análisis de mercado, nos será útil definir el escenario en el que se desarrolla este TFG. Como se ha comentado anteriormente, este trabajo se aborda el encargo de un cliente para desarrollar un *website* que permita a su negocio local introducirse en el comercio electrónico. En concreto, el cliente desea vender productos relacionados con la informática.

Exponer dos perspectivas, una general y otra especifica, nos ayudará a contextualizar el caso que nos ocupa.

2.1. Público objetivo (i.e. *target audience*) y perfiles de usuario

En términos generales, el público objetivo de una web venta de productos informáticos se divide en dos segmentos:

- Usuario avanzado:
    - Conocimientos de hardware medio-avanzado.
    - Puede ejercer el rol de comprador particular o profesional.
    - Conoce webs especializadas de venta de componentes, tanto nacionales como extranjeras.
    - Potencialmente un usuario habitual
- Usuario promedio:
    - Conocimientos de hardware básicos o nulos.
    - Es un comprador particular
    - Puede conocer webs generalistas de venta de productos informaticos.
    - Puede depender de los buscadores para encontrar una web de venta de componentes
    - Puede que nunca haya comprado componentes informáticos
    - Puede que nunca haya comprado online
    - Puede ser un usuario ocasional o habitual

A partir de esta delimitación, mostramos ejemplos de personas que cubren ambos segmentos

Usuario avanzado	Usuario avanzado	Usuario promedio
Juana Rivas	**Ramón Paz**	**Beatriz García**
		
Fuente: Pexels (Christina Morillo)	Fuente: Pexels (Andrea Piacquadio)	Fuente: Pexels (Hanna Nelson)
"Valoro la velocidad y la organización al navegar por una web"	*"Por el escaso tiempo libre del que dispongo, solo me planteo la compra online"*	*"Con ayuda, espero conseguir el equipo que necesito"*
Edad 32	**Edad** 54	**Edad** 25
Profesión Dir.ª compras	**Profesión** Desarrollador	**Profesión** Estudiante
Localización Barcelona	**Localización** Valencia	**Localización** Vigo
Juana trabaja para una pequeña empresa de servicios audiovisuales. Ocupa un puesto como directora de compras, de manera que dedica un tiempo importante a la gestión de proveedores, generalmente a través de internet.	Ramón es un desarrollador freelance. Le gusta estar al tanto de las novedades en el sector y es muy exigente con el rendimiento de los componentes que utiliza. Acostumbra a repasar los productos que ofrecen sus tiendas online de referencia.	Beatriz está terminando el Grado en Multimedia. Trabaja con portátil que lastra su productividad. Sabe que necesita un ordenador más potente para ofrecer resultados profesionales, pero tiene conocimientos básicos y le gustaría recibir asesoramiento.
Necesidades	**Necesidades**	**Necesidades**
• Velocidad de navegación • Filtrado de productos eficiente • Categorías estructuradas	• Lista de deseos • Fichas de productos completa • Interfaz amigable	• Asesoramiento / canal de comunicación con la web • Valoraciones de usuario • Interfaz amigable

Mesa 2: Perfiles de usuario

2.2. Competencia/Antecedentes (o marco teórico)

Amazon

Amazon[1] dispone del mayor stock de productos relacionados con la informática. Sin embargo, al ser una web dedicada a la venta de todo tipo de artículos, ofrece una interfaz excesivamente cargada y la búsqueda por categorías resulta confusa y, en ocasiones, poco eficiente. Además, el manejo de un stock tan grande provoca que sea habitual encontrar descripciones de productos incompletas o erróneas, al igual que ocurre con los datos de las fichas de producto. Por otra parte, la navegación es fluida en todos los sentidos y el filtrado de productos mediante la barra de búsqueda es eficiente. Además de las opiniones de los compradores, ofrece un sistema de preguntas y respuestas en la página de cada producto.

PCComponentes

En el segmento de la informática, PPComponentes[2] es el único competidor de Amazon en España. Su interfaz es austera pero funcional. Los productos están descritos correctamente y las fichas de productos suelen ser completas, con toda la información que ofrecen los propios fabricantes. Sin embargo, el sistema de navegación perjudica la experiencia de usuario: cuando filtramos por categorías, si elegimos algún producto y luego volvemos atrás, el filtrado no persiste y los productos previamente filtrados han desaparecido. El filtro mediante la barra de búsqueda, por su parte, funciona correctamente. El sistema de valoraciones permite, además, responder a las opiniones de otros usuarios. Además, ofrece la opción de contacto/soporte durante la navegación.

Coolmod

La interfaz de Coolmod[3] es poco práctica, pues está cargada de imágenes de gran tamaño que evocan a la publicidad más molesta. Estás imágenes se repiten en todos los lugares, desde la home a las fichas de producto, lo cual perjudica la experiencia de usuario. El filtrado por categorías ofrece filtros poco funcionales y los menús, por su parte, sufren un molesto *lag* a la hora de desplegarse. No dispone de sistema de valoraciones y las páginas con los detalles de un producto, además de contener imágenes que no aportan nada, no siempre muestran la información que se busca.

[1] Enlace a la página de Amazon

[2] Enlace a la página de PCComponentes

[3] Enlace a la página de Coolmod

	Amazon	PCComponentes	Coolmod
Interfaz amigable	−	✓	✗
Filtro por búsqueda	✓	✓	✓
Filtro por categorías	−	✓	✗
Ficha/detalle de producto	−	✓	✗
Valoraciones sobre producto	✓	✓	✗
Preguntas sobre producto	✓	✗	✗
Sistema de navegación	✓	−	−
Soporte/asesoramiento accesible	✗	✓	✗
Lista de deseos	✓	✓	✓

Mesa 3: Comparativa competencia

2.3. Análisis DAFO

	Factores internos	Factores externos
Positivos	**Fortalezas**	**Oportunidades**
	Personalización Orientación SEO Orientación UX Diseño responsive Modularidad	Alta demanda Aumento de la conciencia de compra en pequeños comercios Website respaldado por una tienda física ya asentada
Negativos	**Debilidades**	**Amenazas**
	Rendimiento en consultas BD Rendimiento del servidor Deuda técnica Plazos de desarrollo	Sector dominado por empresas afianzadas Dependencia del servidor del proveedor de *dropshipping*

Mesa 4: Análisis DAFO

3. Propuesta

En base al escenario planteado en el apartado anterior, la solución que se ofrece es un prototipo de aplicación web en el que se redefine el *website* actual del cliente. Así, este pasará de ser una página estática a ser una web app que, además de incluir los contenidos actuales en cuanto a los servicios de la empresa, también ofrecerá un servicio de venta online. De este modo, el cliente pasará a competir en un terreno donde ya existen empresas que, además de una larga trayectoria, tienen un gran volumen de negocio.

De este modo, el reto de este proyecto es dotar a un pequeño comercio de un nuevo *website* que le permita competir de, igual a igual, frente a estas empresas. La clave es combinar el carácter de la marca del cliente con una interfaz consistente. Con esto, se busca que la interfaz del *website* esté a la altura de los grandes del sector, dejando atrás a otros competidores menores que basan sus webs en plantillas y estructuras propias de los CMS. Los objetivos funcionales se detallan en el siguiente apartado.

3.1. Definición de objetivos/especificaciones del producto

Objetivos principales del *website*

- Venta de productos informáticos
- Informar sobre los servicios que ofrece la empresa

Funcionalidades del prototipo

- Simular el proceso de venta online de componentes informáticos
- Trabajar indistintamente con stock propio o con proveedores de *dropshipping*
- Automatizar la realización de pedidos del administrador al proveedor
- Ofrecer un canal de comunicación asíncrona entre administradores y usuarios
- Permitir operaciones CRUD sobre los productos de stock propio
- Permitir la búsqueda y filtrado de producto
- Ofrecer un sistema robusto de autenticación
- Ofrecer una zona privada para administradores y usuarios
- Ofrecer un diseño responsive para los dispositivos más comunes

4. Diseño

4.1. Arquitectura general de la aplicación/sistema/servicio

La aplicación se basa en la arquitectura *serverless web app*. Este modelo permite concentrar los recursos en el desarrollo del código de la aplicación, evitando el proceso que supone la configuracion y mantenimiento de un servidor personalizado. A continuación, se exponen las tecnologías que forman la estructura de la aplicación.

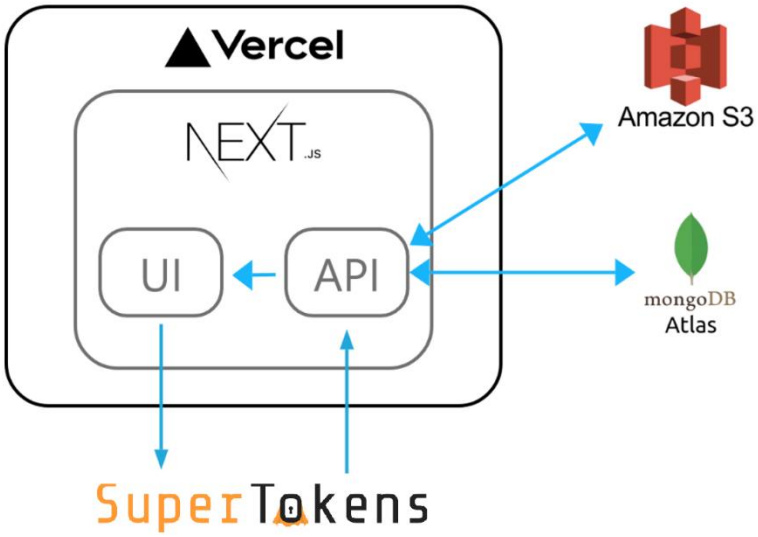

Figura 5: Diagrama de la arquitectura de la aplicación

ReactJS

ReactJS[1] es una librería Javascript de código abierto sobre la que está construido NextJS. Está diseñada para crear interfaces de usuario

Vercel

Vercel [2] es una plataforma *cloud* de código abierto para sitios estáticos y funciones sin servidor. Permite el despliegue de páginas y servicios web sin necesidad de configurar un servidor

[1] Enlace a la página de React https://reactjs.org/

[1] Enlace a la página de Vercel https://vercel.com/

NextJS:

NextJS[1] es un framework de React que puede ejecutar código de servidor, permitiendo el renderizado previo y evitando que sea el navegador el encargado de renderizarlo todo desde cero. Existen dos enfoques para conseguirlos

- Renderizado del lado del servidor (SSR):en cada solicitud, la página se renderiza previamente en el servidor
- Static Generation (SSG), el renderizado se realiza en el momento de la compilación.

Combinar ambas funcionalidades nos permite crear webs híbridas. Además, podemos implementar rutas API que ejecuten código de servidor y permitan operaciones CRUD sobre una base de datos, eliminando la necesidad de un *backend* externo.

SuperTokens

SuperTokens[2] es una librería para gestionar la autenticación en aplicaciones

MongoDB

MongoDB[3] es un sistema de bases de datos NoSQL

MongoDB Atlas

MongoDB Atlas[4] es un servicio *cloud* que permite gestionar una base de datos mongoDB. De este modo, podemos usar una base de datos sin necesidad de instalar mongoDB en un servidor.

AWS S3 (Amazon Web Services – Simple Storage Service)

AWS S3[5] es un servicio *cloud* de almacenamiento ofrecido por Amazon que nos permitirá almacenar archivos y obtener sus URL´s de acceso, que serán almacenadas en nuestra base de datos.

[1] Enlace a la página de NextJS https://nextjs.org/

[2] Enlace a la página de SuperTokens https://supertokens.io/

[3] Enlace a la página de MongoDB https://www.mongodb.com/

[4] Enlace a la página de MongoDB Atlas https://www.mongodb.com/atlas/database

[5] Enlace a la página de AWS S3 https://aws.amazon.com/es/s3/

4.2. Arquitectura de la información y diagramas de navegación

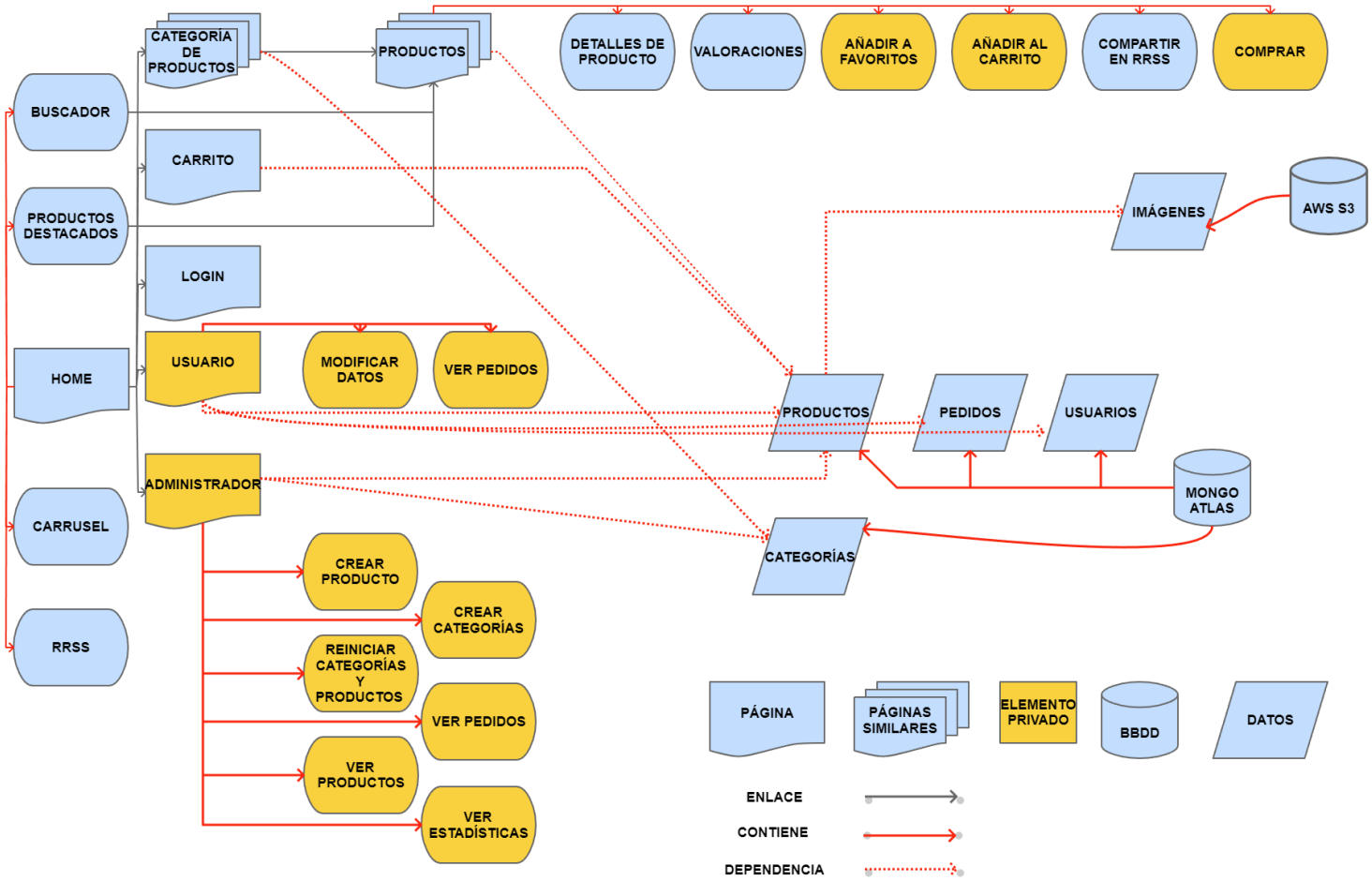

Figura 6: Navegación por árbol de contenidos

4.3. Diseño gráfico e interfaces

4.3.1. Estilos

Figura 7: Logotipo

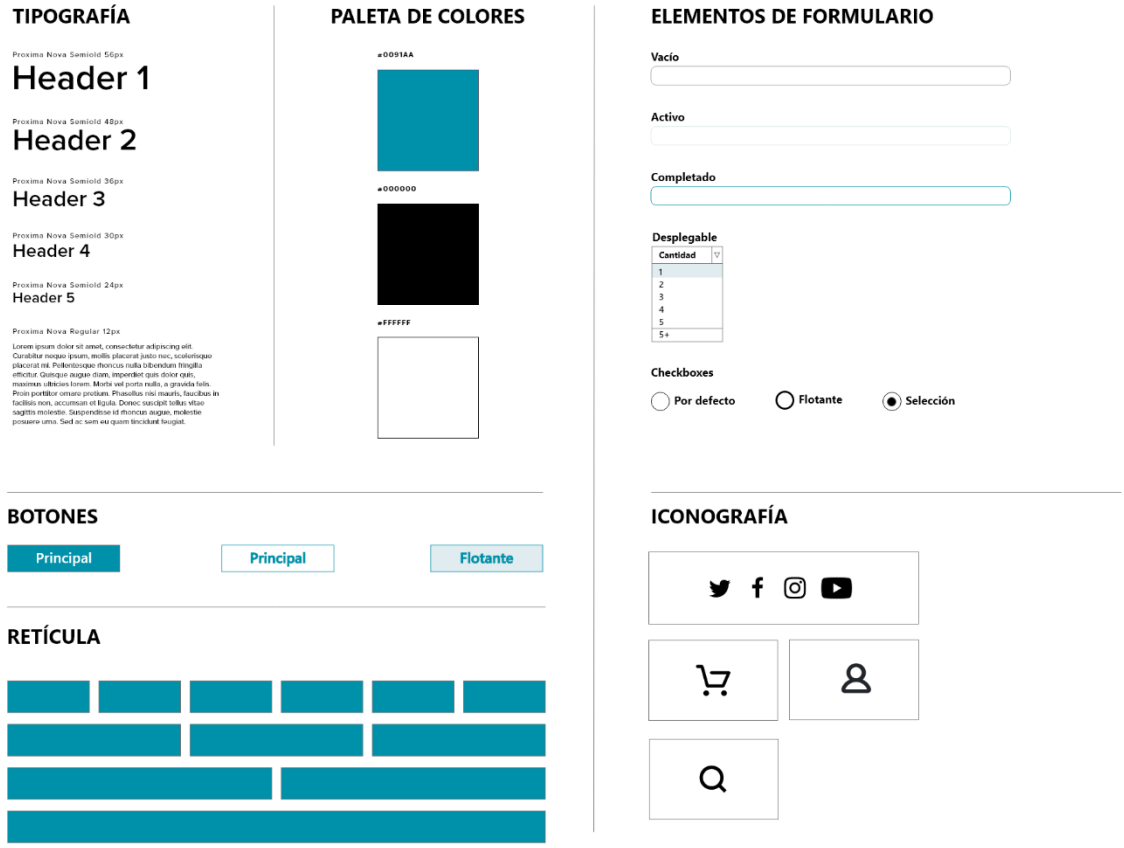

Figura 8: Guía de estilo

5.1.1. Prototipos Hi-Fi

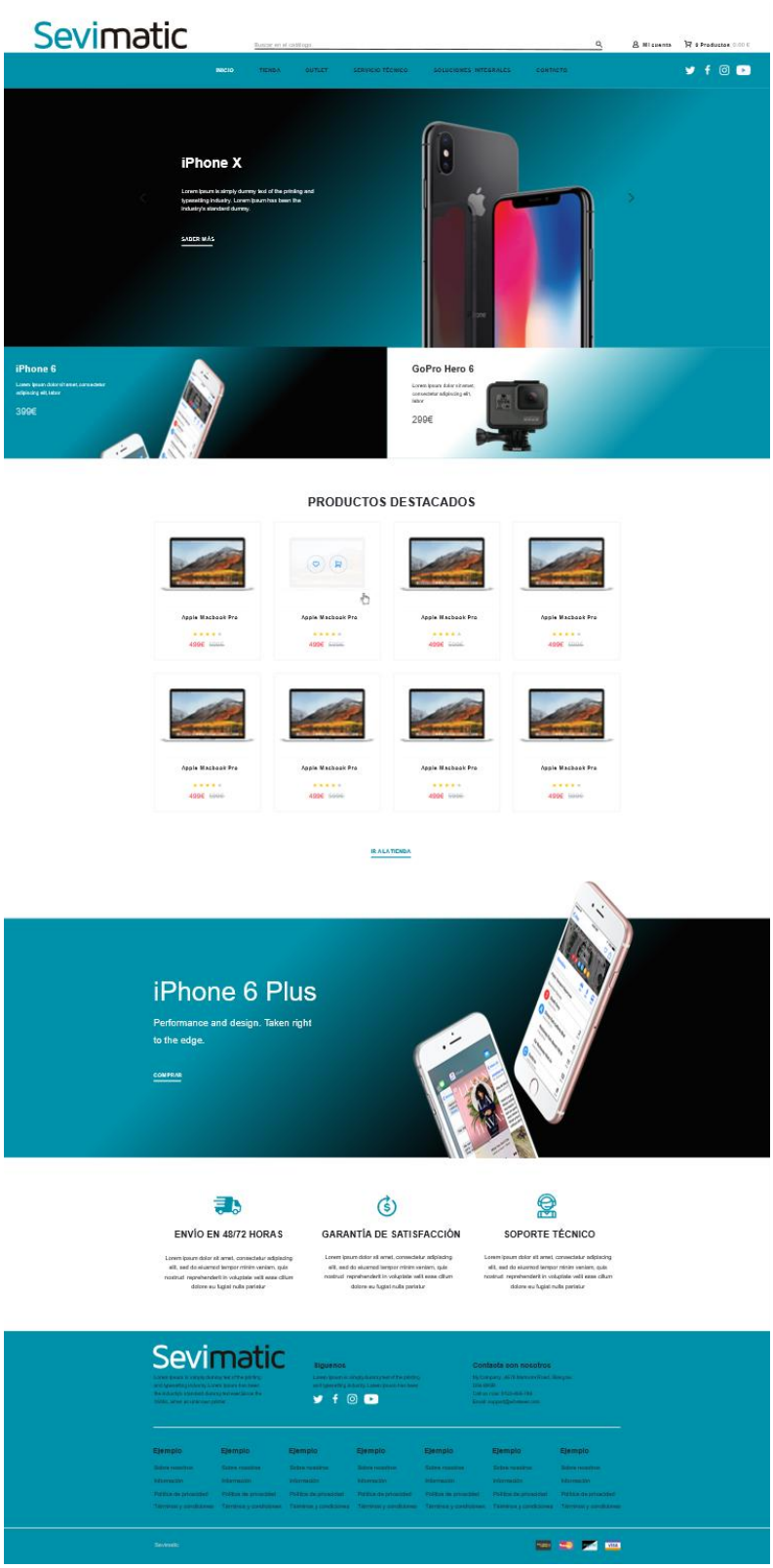

Figura 9: Mock-up Home

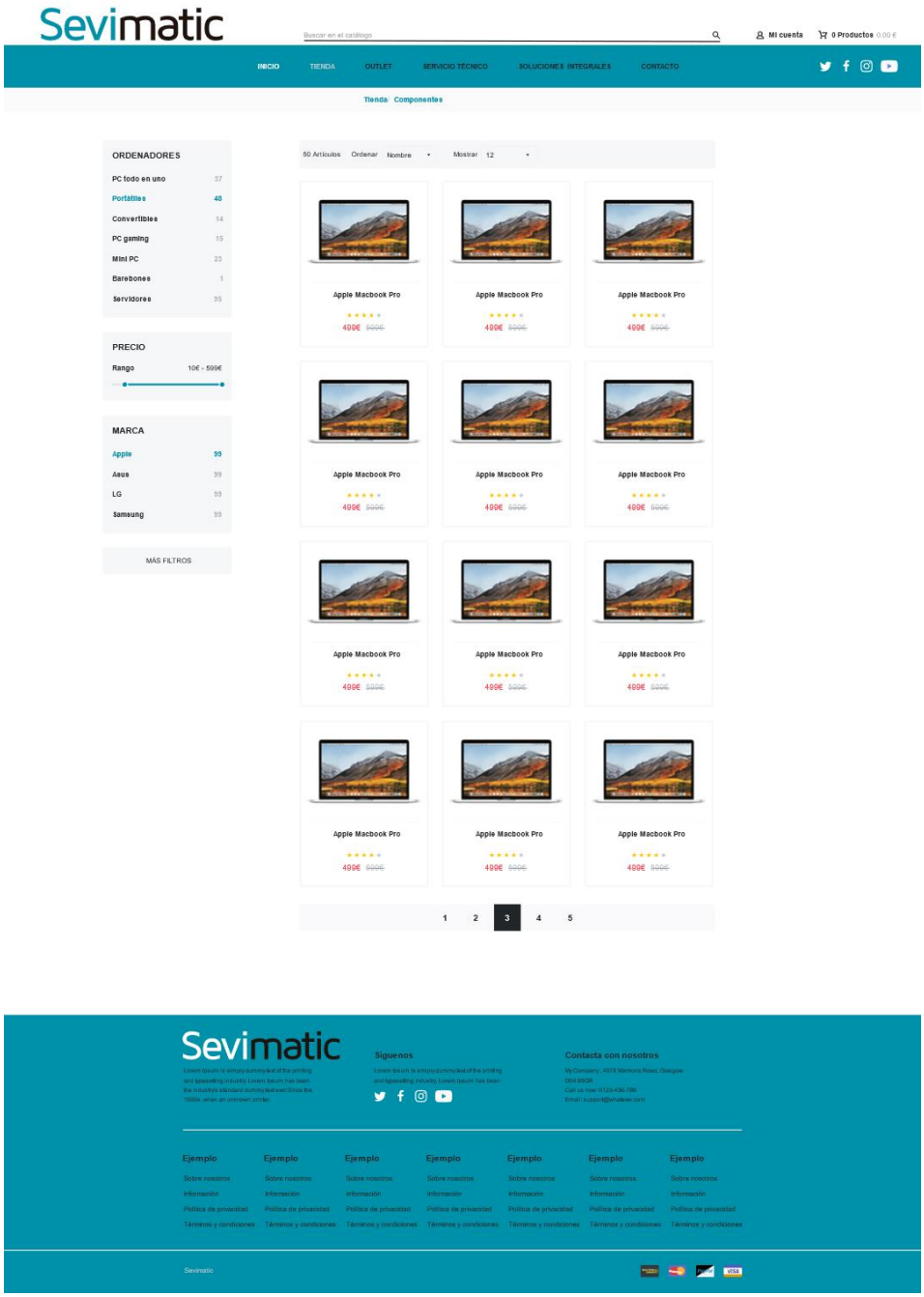

Figura 10: Mock-up Productos

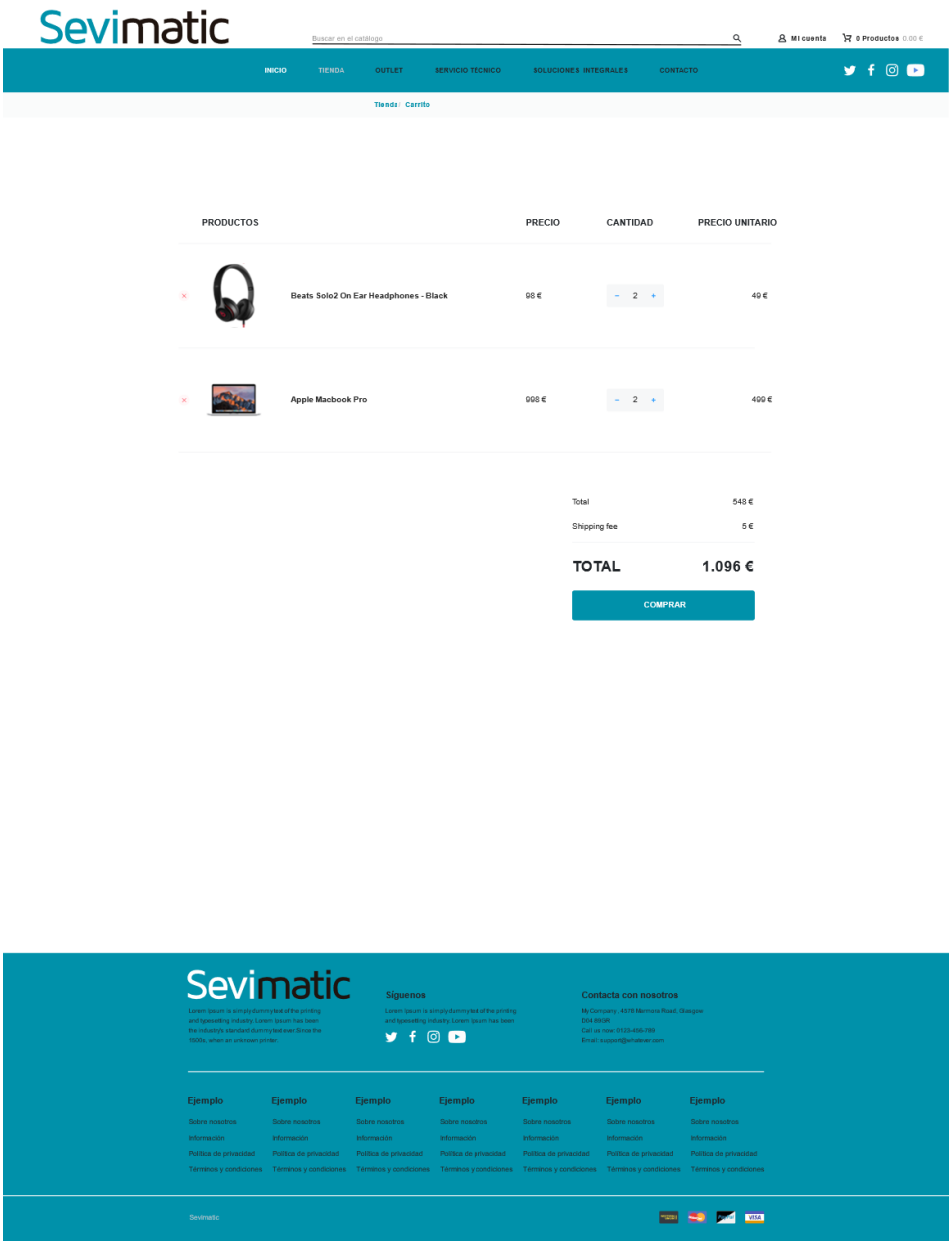

Figura 11: Mock-up Carrito

Bibliografía

WIRED website: http://www.wired.com/epicenter/2012/04/ff_andreessen/, consultado 12/12/2012

STACKOVERFLOW website: https://stackoverflow.com/, consultado 07/01/2022

REACTJS website: https://reactjs.org/ /, consultado 07/01/2022

MUI website: https://mui.com/, consultado 07/01/2022

SUPERTOKENS website: https://supertokens.io/, consultado 07/01/2022

MONGOOSE website: https://mongoosejs.com/, consultado 07/01/2022

MONGODB website: https://www.mongodb.com/es/, consultado 07/01/2022

AWS website: https://aws.amazon.com/es /, consultado 07/01/2022

VERCEL website: https://vercel.com/dashboard/, consultado 07/01/2022