



*ugr* | Universidad  
de **Granada**

TRABAJO FIN DE GRADO  
GRADO EN INGENIERÍA INFORMATICA

# Herramienta de gestión de negocios de cita previa y venta de productos

---

Web Service de gestión de negocios de cita previa y venta de  
productos junto con aplicación web/móvil consumidora del  
mismo

**Antonio Jiménez Rodríguez**

---

**Autor**

Antonio Jiménez Rodríguez

**Director**

Javier Martínez Baena



**ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE  
TELECOMUNICACIÓN**

---

Granada, Junio de 2022

**Herramienta de gestión de negocios de cita previa y venta de productos:**

**Web Service de gestión de negocios de cita previa y venta de productos junto con aplicación web/móvil consumidora del mismo**

Antonio Jiménez Rodríguez

**Palabras clave:** desarrollo web, aplicación móvil, APIRest, código abierto

**Resumen**

En el presente Trabajo de Fin de Grado se expondrá el proceso de creación de un Web Service donde será posible gestionar múltiples negocios de cita previa a la vez, con una administración para su monitorización y que pueda utilizarse como puente hacia la Base de Datos. También se desarrollará una aplicación web/móvil de un negocio real donde se podrá ver el acoplamiento Front-End y Back-End.

El desarrollo del proyecto constará de distintas fases: análisis del problema, análisis de requisitos, diseño del modelo de datos, diseño de interfaz de usuario y finalmente la implementación de todo ello.

El diseño de la interfaz tiene gran importancia ya que deberá ser intuitiva a la vez que llamativa para el usuario. Si la zona de tienda es cómoda de usar mejorará la experiencia y aumentará las probabilidades de que se efectúen compras. Así mismo el apartado para las reservas de citas deberá ser sencillo, con el objetivo de facilitar la reserva de citas quitándole la necesidad al usuario de llamar por teléfono para saber que días hay disponibles. Esto repercute directamente en los trabajadores del negocio ya que no tendrán la necesidad de atender a esos clientes y podrán seguir centrados en su trabajo.

Por otra parte, el diseño del modelo de datos y la administración de la misma también tiene un papel importante. En esta deberá haber roles de usuarios donde no todas las personas con permisos puedan acceder a los mismos datos. Además, con un buen control de errores será posible llevar a cabo una monitorización de todo lo que pasa en las distintas webs y controlar/solucionar posibles fallos.

Antonio Jiménez Rodríguez

**Business management tool for appointments and product sales:  
Appointment and product sales business management web  
service together with web/mobile application consuming it**

Antonio Jiménez Rodríguez

**Keywords:** *web development, app, APIRest , open source*

**Abstract**

In this Final Degree Project, we will present the process of creating a Web Service where it will be possible to manage multiple business appointments at the same time, with an administration for its monitoring and that can be used as a bridge to the database. We will also develop a web/mobile application of a real business where it will be possible to see the Front-End and Back-End coupling.

The development of the project will consist of different phases, problem analysis, requirements analysis, data model design, user interface design and finally the implementation of all these studies.

The design of the interface is of great importance since it must be intuitive and eye-catching for the user. The store area should be comfortable to use to enhance the experience and increase the likelihood of purchases are more likely to be made. Likewise, the section for booking appointments should be simple, in order to facilitate appointment booking by removing the need to phone to find out what days are available. This has a direct impact on the business' employees, as they will no longer have to attend to these clients and will be able to continue focusing on their work.

On the other hand, the design of the data model and its administration also plays an important role. In this, there should be user roles where not all the people who access it can access the same data. On the other hand, with a good error control it will be possible to carry out a monitoring of everything that happens on the different websites and control/solve possible failures.

Antonio Jiménez Rodríguez

---

D. **Javier Martínez Baena**, Profesor del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada.

**Informo:**

Que el presente trabajo, titulado ***Herramienta de gestión de negocios de cita previa y venta de productos***, ha sido realizado bajo mi supervisión por **Antonio Jiménez Rodríguez**, y autorizo la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a Junio de 2022.

**El director:**

(Javier Martínez Baena)

Antonio Jiménez Rodríguez

Antonio Jiménez Rodríguez

## Agradecimientos

A mis compañeros de trabajo por hacerme crecer como profesional y persona.

A mi familia y amigos por apoyarme siempre en esta etapa y creer en mi.

A mi tutor, D. Javier Martínez Baena por su ayuda, predisposición y compresión en el camino.

Antonio Jiménez Rodríguez

# Índice general

<b>1. Introducción</b>	<b>17</b>
1.1. Descripción del problema . . . . .	17
1.2. Estado del arte . . . . .	18
1.2.1. Aplicaciones actuales . . . . .	18
1.2.2. Crítica al estado del arte . . . . .	18
1.2.3. Propuesta . . . . .	19
<b>2. Análisis del Sistema</b>	<b>21</b>
2.1. Casuísticas . . . . .	21
2.1.1. Acceso al público . . . . .	21
2.1.2. Reserva de citas . . . . .	22
2.1.3. Compra de productos . . . . .	22
2.2. Tipos de usuarios . . . . .	22
<b>3. Planificación</b>	<b>25</b>
3.1. Metodología utilizada . . . . .	25
3.2. Temporización . . . . .	25
3.2.1. Sprints definidos . . . . .	26
3.3. Presupuesto . . . . .	27
3.3.1. Factura total . . . . .	28
<b>4. Análisis del problema</b>	<b>29</b>
4.1. Implicados . . . . .	29
4.1.1. Tipos de usuario . . . . .	29
4.1.2. Resumen de implicados . . . . .	30
4.1.3. Perfiles de los implicados . . . . .	30
4.2. Especificación de requisitos . . . . .	31
4.2.1. Requisitos funcionales . . . . .	31
4.2.2. Requisitos de información . . . . .	34
4.2.3. Restricciones semánticas . . . . .	34

4.2.4. Requisitos No Funcionales . . . . .	36
4.3. Modelo de casos de uso . . . . .	37
4.3.1. Diagramas de casos de uso . . . . .	37
4.3.2. Descripción de los casos de uso . . . . .	39
4.4. Diagrama de secuencia . . . . .	50
<b>5. Diseño</b>	<b>59</b>
5.1. Base de datos . . . . .	59
5.1.1. Paso a tablas y fusión . . . . .	60
5.1.2. Normalización . . . . .	61
5.2. Arquitectura del Sistema . . . . .	62
5.3. Interfaz de Usuario . . . . .	63
5.3.1. Página de Inicio . . . . .	63
5.3.2. Página de Tienda . . . . .	65
5.3.3. Página de Reserva de Citas . . . . .	66
5.3.4. Página de Perfil de Usuario . . . . .	68
<b>6. Implementación</b>	<b>73</b>
6.1. Tecnologías . . . . .	73
6.2. Librerías y Bundles . . . . .	74
6.2.1. EasyAdminBundle . . . . .	74
6.2.2. LexikJWTAuthenticationBundle . . . . .	74
6.2.3. Symfony mailer . . . . .	74
6.2.4. Stripe . . . . .	75
6.2.5. Axios . . . . .	76
6.2.6. Reporte de errores con Telegram . . . . .	77
6.3. Estructuración de código en Symfony . . . . .	79
6.4. Estructuración de código en ReactJS . . . . .	81
6.5. Uso y funcionamiento . . . . .	81
6.5.1. Página de Home . . . . .	81
6.5.2. Sección de Introducción . . . . .	82
6.5.3. Sección de Red Social . . . . .	82
6.5.4. Sección de Servicios . . . . .	83
6.5.5. Sección de Información del negocio . . . . .	84
6.5.6. Página de Productos . . . . .	84
6.5.7. Página de Citas . . . . .	86
6.5.8. Página de Perfil . . . . .	87
<b>7. Conclusiones y trabajos futuros</b>	<b>91</b>
7.1. Temporización final . . . . .	91
7.2. Objetivos alcanzados . . . . .	91
7.3. Aprendizaje . . . . .	92
7.4. Desarrollo futuro . . . . .	93

# Índice de figuras

3.1. Previsión de sprint a desarrollar . . . . .	27
4.1. Diagrama de caso de uso - Gestión de usuarios . . . . .	38
4.2. Diagrama de caso de uso - Gestión de citas . . . . .	38
4.3. Diagrama de caso de uso - Gestión de tienda . . . . .	39
4.4. DS: Alta de cliente, trabajador o superadministrador. . . . .	51
4.5. DS: Baja de cliente, trabajador o superadministrador. . . . .	51
4.6. DS: Consultar datos de cliente, trabajador o superadministrador.	52
4.7. DS: Modificar datos de cliente, trabajador o superadministrador.	53
4.8. DS: Consultar citas por cliente, trabajador o superadministrador.	53
4.9. DS: Reserva de cita por cliente o trabajador. . . . .	54
4.10. DS: Cancelación de cita por cliente o trabajador. . . . .	54
4.11. DS: Alta de producto para venta online por trabajador o superadministrador. . . . .	55
4.12. DS: Baja de producto para venta online por trabajador o superadministrador. . . . .	55
4.13. DS: Realizar pedido de productos por cliente. . . . .	56
4.14. DS: Cancelar pedido por cliente. . . . .	57
4.15. DS: Consultar pedido de productos por cliente. . . . .	58
5.1. Diagrama Entidad-Relación . . . . .	60
5.2. Paso a tablas y fusión . . . . .	61
5.3. Modelo Vista Controlador . . . . .	63
5.4. Página de Inicio de Web . . . . .	64
5.5. Página de Tienda de Web . . . . .	65
5.6. Página de Tienda de Web con Carrito de Compra . . . . .	66
5.7. Página de Reserva de Citas de Web . . . . .	67
5.8. Página de Reserva de Citas (selección de día) . . . . .	68
5.9. Página de Perfil de Usuario (datos personales) . . . . .	69
5.10. Página de Perfil de Usuario (citas) . . . . .	70

5.11. Página de Perfil de Usuario (pedidos) . . . . .	71
5.12. Página de Perfil de Usuario (datos de pedido) . . . . .	72
6.1. Confirmación de pago en ReactJS con Stripe . . . . .	76
6.2. Historial de pagos en la plataforma Stripe . . . . .	76
6.3. Constructor de TelegramService: inicialización de parámetros . . . . .	77
6.4. Métodos públicos de TelegramService: enviar mensajes y AppErrors . . . . .	78
6.5. Método privado de TelegramService: formatear mensaje de error . . . . .	79
6.6. Funcionalidad de AppController: creación de respuesta JSON para API . . . . .	80
6.7. Configuración de la sección de Introducción en el Back-End . . . . .	82
6.8. Renderización de la configuración en la Web . . . . .	82
6.9. Configuración de la sección de Red Social en el Back-End . . . . .	82
6.10. Renderización de la configuración en la Web . . . . .	83
6.11. Configuración de la sección de Servicios ofrecidos en el Back-End . . . . .	83
6.12. Renderización de la configuración en la Web . . . . .	83
6.13. Configuración de la sección de Información del negocio en el Back-End . . . . .	84
6.14. Configuración de los turnos del negocio en el Back-End . . . . .	84
6.15. Renderización de la configuración en la Web . . . . .	84
6.16. Configuración de las categorías de productos del negocio en el Back-End . . . . .	85
6.17. Configuración de los productos vendidos por el negocio en el Back-End . . . . .	85
6.18. Renderización de la configuración en la Web . . . . .	85
6.19. Carrito de compra en proceso de pago . . . . .	86
6.20. Página de citas: vista inicial . . . . .	86
6.21. Página de citas: día seleccionado . . . . .	87
6.22. Página de citas: usuario con cita pendiente . . . . .	87
6.23. Página de Perfil: datos personales . . . . .	88
6.24. Página de Perfil: direcciones postales . . . . .	88
6.25. Página de Perfil: historial de citas . . . . .	89
6.26. Página de Perfil: historial de pedidos . . . . .	89

Antonio Jiménez Rodríguez

# Índice de tablas

4.1. Resumen de implicados . . . . .	30
4.2. Perfil de implicado: Superadministrador . . . . .	30
4.3. Perfil de implicado: Trabajador . . . . .	31
4.4. Perfil de implicado: Cliente . . . . .	31

Antonio Jiménez Rodríguez

Capítulo

1

# Introducción

Este proyecto es software libre, y está liberado con la licencia [6].

Debido a la situación tan extraordinaria que hemos vivido en estos últimos años con la Covid-19 muchas de nuestras actividades y negocios han sufrido cambios bastante notables. Por ejemplo el teletrabajo en las empresas privadas, aparentemente ha llegado para quedarse y dar así al trabajador una mayor libertad de movimiento con el objetivo de aumentar su comodidad y productividad.

Otros de los sectores afectados han sido los negocios de citas. Estos se vieron obligados a trabajar únicamente con cita previa debido a la prohibición de poder haber varias personas esperando en una sala de espera o en la propia sala del negocio.

Ante este problema surge la necesidad de crear un aplicativo que gestione de una manera sencilla todo lo relacionado con la reserva de citas y que ayude a los usuarios de estos negocios en el proceso.

Para ello se propone el desarrollo de un sistema gestor de negocios de cita previa con la posibilidad abrirles la puesta también a la venta online. Todo esto gestionado de una manera centralizada en la que los dueños de los negocios podrán configurar a su manera su entorno y gestionar toda la información registrada.

## 1.1. Descripción del problema

Dado un caso real, donde a una barbería normalmente acudía la gente sin previo aviso para esperar a poder ser atendida, tuvo que adaptarse a las nuevas normativas y habilitar un teléfono para la reserva de citas.

Se plantea a partir de esta situación, la creación de una aplicación web/móvil que permita la gestión de las citas automáticamente, sin necesidad de que el trabajador se distraiga de su labor. Facilitando también a los usuarios la consulta de horarios y disponibilidad del mismo, permitiendo elegir la fecha deseada con unas simples acciones, además de tener un registro de las diferentes reservas.

## 1.2. Estado del arte

El software libre y sus licencias [6] ha permitido llevar a cabo una expansión del aprendizaje de la informática sin precedentes.

### 1.2.1. Aplicaciones actuales

En la actualidad existen diversas páginas para la reserva de citas en distintos negocios de cita previa:

- **miCita**: Aplicación móvil desarrollada con el propósito de gestionar el apartado de citas de un negocio o empresa. En ella se rellenarán los datos del propio negocio con un asistente de configuración donde se completará información como los servicios ofrecidos y horarios. Los clientes podrán reservar su cita y observar el calendario en tiempo real desde la aplicación. Para acceder a estas funcionalidades como cliente se deberá abonar una cuota mensual de **19,90€/mes (IVA no incluido)**.[10]
- **Reservio**: Aplicación web para la gestión de reservas o citas en un negocio. Con un calendario para las reservas se podrá acceder como trabajador o como cliente permitiendo la reserva online. Además tendrá la posibilidad de gestionar los perfiles de los clientes y estos podrán ser recompensados con pases de fidelidad. Esta aplicación también incluye asistencia de negocio con estadísticas y recordatorios vía SMS.[16]
- **Shopify**: Shopify es una plataforma online para montar webs de venta online. Con Shopify se puede montar webs con diseños predefinidos y despreocuparse de la gestión de marketing, transacciones online y envíos. Para acceder a este contenido y poder montar un e-commerce Shopify ofrece cuotas mensuales que parten de **29\$/mes**[19]

### 1.2.2. Crítica al estado del arte

Las aplicaciones mencionadas en el apartado anterior son los subconjuntos necesarios para la creación de este proyecto. Las dos primeras incluyen la gestión de citas, y usuario (en el caso de *Reservio*). La tercera web constituye la parte de la venta online, donde se gestiona todo lo relacionado con pedidos, pagos, envíos, etc.:

- **miCita**: un punto negativo al usar una aplicación como esta es que no tienes una interfaz propia que te diferencie de otras tiendas que usen este mismo medio.
- **Reservio**: se trata de una aplicación de gestión de negocio de citas gratuita, cosa positiva a valorar, que permite la reserva de citas por partes de los clientes. Sin embargo con este aplicativo no se podría cubrir la venta online de un negocio, ya que no da soporte para ello.

- **Shopify:** al contrario que la anterior, Shopify se trata de una herramienta de creación de webs para venta de productos online. Al ser solo la construcción de la tienda online no tendríamos la posibilidad de tener unificada en una misma aplicación nuestra gestión de citas y ventas online.

### 1.2.3. Propuesta

Teniendo en cuenta la situación actual sobre el estado del arte en lo referente a aplicaciones para gestión de negocios de cita previa, se propone crear un sistema genérico que permita la administración de estos. La principal finalidad del proyecto es conseguir un sistema sólido y unificado, en el que se puedan integrar todos los negocios que se deseen (de características similares) de una manera sencilla.

No obstante al hacer una separación entre Front-End y Back-End será posible hacer única cada web si se desea aunque internamente estén montadas con la misma estructura de datos.

Además de la gestión de citas se incluirán las funcionalidades necesarias para la implementación de una tienda online en el que cada negocio podrá vender sus productos sin necesidad de atarse únicamente a sus clientes locales.

Para la parte del Back-End se creará una vista de administración con roles de usuario:

- Cada negocio deberá tener al menos un usuario **administrador** que podrá acceder a los datos almacenados referentes únicamente a su negocio. Estos usuarios serán los trabajadores del negocio o el propio dueño.
- Existirá un **superadministrador** que acceda al total de los datos del aplicativo para su gestión y control de errores.

En este tipo de administración se podrán crear funcionalidades al gusto para hacer correcciones o cualquier tipo de necesidad, siendo también posible la accesibilidad por roles de usuario. Los clientes solo podrán acceder a sus datos desde los diferentes sitios webs creados.

Antonio Jiménez Rodríguez

Capítulo

# 2

## Análisis del Sistema

Este sistema a implementar pretende resolver el posible problema de un negocio de cita previa a saltar al terreno online.

Se desarrollará un sistema donde un cliente del mismo solo tenga que completar la información requerida en la base de datos del aplicativo. Mediante un despliegue de un nuevo entorno, que será realizado por los programadores, se obtendrá una nueva página web totalmente funcional en la que dicho negocio podrá tener a sus clientes identificados y ofrecerles la posibilidad de la reserva de citas o de compra de producto de una manera más cómoda desde casa.

### 2.1. Casuísticas

Al ser un sistema que intenta abarcar varios frentes bastante amplios existen muchas casuísticas que se podrían dar en el aplicativo.

#### 2.1.1. Acceso al público

Está claro que el objetivo principal es llegar a una mayor cantidad de personas y dar el negocio a conocer por la red. Sin embargo, no se puede proporcionar una web en la que todos los sitios sean accesibles por cualquier persona. Para esto se va a implementar un sistema de usuarios por negocio. Un usuario sin identificarse tendrá acceso a la página de presentación del negocio y a la página de productos de la tienda. Por el contrario, sin autenticación no se le dará acceso a la reserva de citas ni a la realización de pedidos online. Con esto también conseguimos quitar una carga al servidor de Back-End, ya que por ejemplo, para un usuario nuevo que aún no se ha registrado en el negocio no se harán peticiones desde las webs al servidor para la obtención de todos estos datos.

### 2.1.1.1. Ataques al sistema

Debido al acceso público se pueden dar casos de ataques al sistema. En internet se realizan miles y miles de ataques ciberneticos al segundo y nuestro sistema no tiene por qué ser una excepción. Para ello se tendrá especial atención en el trato de datos de inserción en el sistema y se validarán los usuarios registrados con el fin de intentar filtrar lo máximo posible a los usuarios reales de los que no lo son.

### 2.1.2. Reserva de citas

Otro de los bloques principales es la reserva de cita previa. Aquí se debe conseguir no romper la organización del propio negocio, no dándole la posibilidad a un cliente que reserve a una hora y minuto exacto a la que él desee ser atendido, sino que los clientes deberán seleccionar las horas que vienen definidas por el propio negocio. Parece obvio, pero una mala gestión de esta parte podría causar un caos en el sistema y en el propio negocio a la hora de atender a estos clientes.

Para ello se realizarán todas las comprobaciones y validaciones necesarias antes (al mostrar las citas disponibles a los clientes) y después (una vez el cliente haya enviado su reserva) para no cometer ningún error ni colapso.

### 2.1.2.1. Cancelación de citas

Este frente puede ser problema para el negocio, ya que cualquiera podría registrarse con sus datos, realizar una reserva de una cita online y cancelarla cinco minutos antes de tener que ser atendido. Para evitar esto se deberá implementar funcionalidades para que un usuario pueda cancelar una cita dentro de un límite establecido. Si aun así el cliente no asiste a la cita por un motivo no justificado esta deberá poder ser marcada para una posible penalización.

### 2.1.3. Compra de productos

El último gran bloque de este sistema es la compra de productos online ofertados por cada negocio. Desde el sistema y las diferentes webs se podrá acceder a un catálogo de productos gestionado por cada negocio en la que los usuarios podrán crear un carrito de la compra y procesar su pedido. Además tendrá acceso al estado y seguimiento de este. Los pagos deberán gestionarse por alguna herramienta externa que de soporte para ello y se adaptará la información almacenada en base de datos para los datos proporcionados por la herramienta elegida.

## 2.2. Tipos de usuarios

Para este sistema se definirán tres tipos de usuarios los cuales podrán interactuar con él de diferente forma debido a los permisos de accesibilidad.

#### **2.2.0.1. Superadministrador**

Este usuario será el que controle todo el aplicativo completo teniendo acceso al total de los datos almacenados en la base de datos. Podrá ver, modificar o eliminar datos si así lo ve oportuno, ya sea por errores producidos o por peticiones de los jefes de los negocios. Además será el encargado de revisar las notificaciones de errores que se registren y deberá mantener el orden en el funcionamiento de cada negocio.

#### **2.2.0.2. Trabajadores**

Ellos tendrán acceso completo a los datos almacenados a nivel de negocio. Podrán consultar las citas o pedidos realizados por los siguientes usuarios y notificar a un superadministrador en caso de que se haya encontrado alguna anomalía. Podrán reservar citas para clientes ya registrados o en su defecto reservar citas con el número del cliente que lo deseé. Si por ejemplo, una persona mayor que no se maneje bien con el mundo de internet desea una cita en el negocio, el trabajador podría reservar una cita con el número de teléfono del cliente sin necesidad de que este esté previamente registrado.

#### **2.2.0.3. Clientes**

El cliente es el usuario encargado de que el sistema tenga vida. Estos podrán registrarse en los distintos negocios (serán distintas cuentas por negocio), reservar citas previas y realizar pedidos de productos. Debido a que los clientes serán los encargados de alimentar la base de datos deberemos tener gran cuidado a la hora de tratar la información proporcionada para así evitar posibles ataques o errores humanos. Esto lo podemos conseguir sanitizando la información y validando los datos en tiempo real mostrándole al usuario los posibles errores cometidos.

Antonio Jiménez Rodríguez

Capítulo **3**

## Planificación

### 3.1. Metodología utilizada

Para el desarrollo de este proyecto se ha elegido como metodología el **Desarrollo Ágil** [14]. En mi opinión, este tipo de enfoque se acerca más al que desarrollan las empresas en la actualidad, donde se definen sprints en los que los requisitos y soluciones van evolucionando según la necesidad del proyecto. No obstante cada iteración deberá tener su planificación, análisis de requisitos, diseño, etc.

### 3.2. Temporización

La temporización general de este proyecto va a estar dividida en dos: construcción del Web Service para la gestión de los datos del aplicativo y la creación de un entorno Front-End para los usuarios. A su vez estos dos objetivos principales estarán divididos en diferentes hitos para realizar un desarrollo progresivo.

#### ■ Web Service:

- **Gestión de citas previas:** En este primer hito se completará toda la funcionalidad referente a la gestión de cita previa. Contendrá todo lo necesario para que desde la aplicación Webs se puedan realizar operaciones como reserva de citas por parte de usuarios; obtención de histórico de citas ya sea del negocio, usuarios o trabajadores; cancelación de citas; así como todas las comprobaciones y controles para la gestión de estas citas. De la mano de este hito se completará también la gestión de inicio de sesión de usuario y la encriptación de datos privados en la base de datos del aplicativo.
- **Gestión de venta de productos online:** Una vez finalizadas las dos etapas anteriores pasaremos a desarrollar otro de los puntos principa-

les del problema, la gestión de venta online de productos. En este hito se deberá implementar la gestión de los pedidos que lleguen desde la aplicación Front-End de los usuarios de las Webs.

- **Administración Back-End:** Por último se va a añadir para el Back-End desarrollado una vista de administración para poder acceder a todos los datos registrados por cada negocio. Además se podrán añadir funcionalidades para los trabajadores o un panel de estadísticas donde ver información relevante.

■ **Aplicación Web/Móvil:**

- **Esqueleto y Home:** El primer hito correspondiente al desarrollo Front-End del aplicativo será la construcción del esqueleto Web (header, barra de navegación, footer, etc.) Junto con la Home con información y presentación del negocio.
- **Interfaz para la reserva de citas:** Una vez cumplido el primer hito pasaremos al desarrollo de lo referente a las citas previas. Se creará una vista donde ver las citas disponibles del negocio y poder reservarlas como usuario autenticado, además de una vista individual donde ver un histórico de citas reservadas y próximas citas.
- **Interfaz para las compras online de productos:** Por último finalizaremos este trabajo con el desarrollo de la interfaz para la tienda online. Se creará una vista donde se muestren los productos del negocio con los filtros correspondientes para los productos. Se podrán añadir productos al carrito de compra y realizar el pago de los productos deseados para realizar los pedidos.

### 3.2.1. Sprints definidos

Además de la organización para el desarrollo del sistema, también se debe organizar el tiempo a invertir para el análisis previo necesario y el diseño de base de datos o interfaces de la aplicación. Una vez tengamos todos los puntos estudiados y definidos podremos comenzar con el desarrollo en programación. Podemos ver la organización prevista en la siguiente imagen:

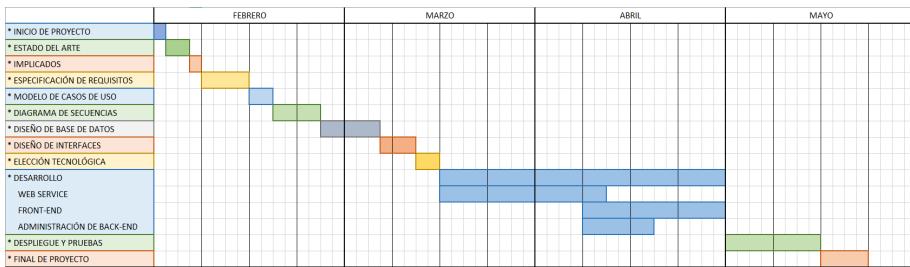


Figura 3.1: Previsión de sprint a desarrollar

Como se indica, el tiempo transcurrido desde que se comienza este proyecto hasta que se pasa a la parte de desarrollo web es de aproximadamente mes y medio. Esto es debido a la cantidad de información y casuísticas que debemos analizar. Además el análisis previo es muy importante a la hora del diseño de la base de datos, ya que a partir de este obtendremos toda la información necesaria de almacenar y las relaciones que tendremos entre nuestras entidades.

La programación ocupará también alrededor de un mes y medio de trabajo. Con todo el análisis y diseño ya realizado no debería llevar un tiempo muy prolongado su implementación, aunque siempre podemos encontrarnos problemas de integración con herramientas externas que utilicemos.

Por último tendremos dos semanas para el despliegue en producción y pruebas para comprobar que todo funciona perfectamente.

### 3.3. Presupuesto

Suponiendo que el proyecto se realizara bajo una contratación para su desarrollo, vamos a proceder a hacer un desglose del presupuesto que sería necesario.

En primer lugar, haremos un cálculo del salario mensual de un programador junior. Observando los diferentes portales de ofertas de trabajo como pueden ser InfoJobs o LinkedIn vemos que los salarios netos anuales rondan entre los 16.000 y 24.000 euros brutos. Supondremos un sueldo medio de entre el rango dado, es decir, 20.000€, que al mes supondrán un gasto de 1.666 € aproximadamente.

En segundo lugar, calcularemos el gasto añadido de las licencias de software requeridas. Las únicas licencias que supondrán un gasto extra en el proyecto serán las de los IDEs utilizados, debido a que para los repositorios de los proyectos haremos uso de GNU General Public License v3.0. Al usar varios IDEs para el desarrollo de este proyecto como es PHPStorm (para la parte de Back-End), WebStorm (para Front-End) y DataGrip (para el acceso a base de datos) usaremos el paquete en la que se proporcionan todas las herramientas de JetBrains con un precio de 301,29€, el primer año. Por la duración del proyecto (4 meses)

realizaremos una regla de tres para calcular cual sería el importe a pagar en estos meses de desarrollo, siendo 100,43€.

Por último añadiremos al presupuesto el hardware utilizado. En este caso tendremos un ordenador portátil Intel® Core™ de 7.<sup>a</sup> generación y una tarjeta gráfica independiente NVIDIA GTX 1050 por 700€. Supondremos que se realizará una financiación de este equipo en los meses de trabajo por lo que supondrá un gasto mensual de 175€.

### 3.3.1. Factura total

Realizaremos finalmente el cálculo total (aproximado) de los costes por mes y total del proyecto con la duración que se estimó en los puntos anteriores:

Tipo de Gasto	Mensual	Total (4 meses)
Salario	1.666€	6.664€
Software	25,11€	100,43€
Hardware	175€	700€
<b>Gasto total</b>	<b>1.866,11€</b>	<b>7.464,44€</b>

Capítulo

# 4

## Análisis del problema

Una vez llegados a este punto vamos a proceder a realizar el análisis y requerimientos [4] del problema para poder implementar una solución que cubra las necesidades mínimas y que se adapte a las situaciones y circunstancias requeridas para el desarrollo del mismo.

### 4.1. Implicados

En esta sección vamos a centrarnos en los implicados del problema a resolver, ya que estos serán los usuarios finales de la aplicación. Definiremos a continuación los distintos tipos de usuario.

#### 4.1.1. Tipos de usuario

Los usuarios potenciales de la aplicación son:

- Superadministrador. Su función es gestionar la totalidad del aplicativo, teniendo acceso completo a los datos almacenados por cada uno de los negocios. Podrá monitorizar el correcto funcionamiento de la aplicación y realizar correcciones o registrar incidencias si es necesario.
- Trabajador. Su función es gestionar los datos de un negocio. En este caso el trabajador tendrá acceso a los datos del mismo. Si detecta posibles errores también podrá registrar incidencias.
- Usuarios. Podrán consultar y reservar citas, así como realizar pedidos en la tienda online. Además tendrán acceso a todos sus datos: histórico de citas e histórico de pedidos.

#### 4.1.2. Resumen de implicados

Nombre	Descripción	Tipo	Responsabilidad
Superadministrador	Administrador del sistema	Usuario producto	Gestionar el sistema, monitorear sistema, gestionar incidencias.
Trabajador	Trabajador del negocio	Usuario producto	Gestionar el negocio, modificar datos del negocio (citas, productos,...), registrar incidencias.
Usuario	Cliente que usa el sistema de manera convencional.	Usuario producto	Reservar/modificar/cancelar citas, realizar pedidos, gestionar perfil de usuario.

Tabla 4.1: Resumen de implicados

#### 4.1.3. Perfiles de los implicados

##### Administrador

Representante	Nombre Apellidos
Descripción	Superadministrador
Tipo	Encargado del sistema completo, con responsabilidades de gestión sobre el mismo.
Responsabilidades	Gestionar el sistema correctamente: supervisar los errores que puedan surgir en el aplicativo y controlar el funcionamiento.
Criterios de éxito	Que el sistema funcione adecuadamente con los parámetros óptimos y no exista información incorrecta.
Implicación	Total, pues es el encargado principal del sistema.

Tabla 4.2: Perfil de implicado: Superadministrador

##### Trabajador

<b>Representante</b>	Nombre Apellidos
<b>Descripción</b>	Administrador de negocio
<b>Tipo</b>	Encargado del negocio al que pertenece, con responsabilidades de gestión sobre el mismo.
<b>Responsabilidades</b>	Gestionar su negocio: supervisar y controlar los datos referentes a su negocio para que no aparezcan incongruencias.
<b>Criterios de éxito</b>	Que no se detecten errores en los datos almacenados.
<b>Implicación</b>	Total a nivel de negocio.

Tabla 4.3: Perfil de implicado: Trabajador

#### Cliente

<b>Representante</b>	Nombre Apellidos
<b>Descripción</b>	Cliente de un negocio
<b>Tipo</b>	Hace uso del aplicativo Front-End sin tener conocimiento inicial de uso.
<b>Responsabilidades</b>	Crear su cuenta de usuario. Reservar citas. Realizar pedidos. Gestionar su perfil de usuario.
<b>Criterios de éxito</b>	Poder realizar sus responsabilidades sin errores y de forma sencilla.
<b>Implicación</b>	Total a nivel de negocio. Sin clientes no hay negocio.

Tabla 4.4: Perfil de implicado: Cliente

## 4.2. Especificación de requisitos

### 4.2.1. Requisitos funcionales

Desarrollaremos aquí los requisitos funcionales para la definición de funcionalidades del sistema de software.

**RF-1. Gestión de superadministradores.** Se permitirá el alta/baja con el rol de superadministrador del sistema. Podrá acceder a todos los datos almacenados en este.

**RF-1.1. Alta de superadministrador.** Se dará de alta a cada superadministrador con sus datos personales.

**RF-1.2. Baja de superadministrador.** Se dará de baja al superadministrador y la información relativa a él será eliminada. Siempre

deberá haber un usuario de este rol para las revisiones técnicas del aplicativo.

**RF-1.3. Consultar datos del sistema.** Se podrán consultar todos los datos almacenados en el sistema.

**RF-1.4. Modificar datos del sistema.** Se podrán modificar todos los datos almacenados en el sistema.

**RF-2. Gestión de trabajadores.** Se permitirá el alta/baja con el rol de trabajador de negocio, así como consultar y modificar sus datos. También podrá acceder a los datos de los clientes dados de alta en el mismo negocio.

**RF-2.1. Alta de trabajador.** Se dará de alta a cada trabajador con sus datos personales.

**RF-2.2. Baja de trabajador.** Se dará de baja al trabajador y la información relativa a él será eliminada.

**RF-2.3. Consultar datos de trabajador.** Se podrá consultar los datos relativos a un determinado trabajador (datos personales, citas asignadas, etc.).

**RF-2.4. Modificar datos de trabajador.** Se podrán modificar los datos personales de un trabajador.

**RF-2.5. Consultar citas de trabajador.** Se podrá consultar las citas previas asignadas al trabajador.

**RF-3. Gestión de clientes.** Se permitirá el alta/baja con el rol de usuario del negocio, así como consultar y modificar sus datos.

**RF-3.1. Alta de cliente.** Un cliente potencial se podrá dar de alta en un negocio determinado.

**RF-3.2. Baja de cliente.** Un cliente se podrá dar de baja de un negocio determinado.

**RF-3.3. Consultar datos de cliente.** Un cliente podrá consultar sus datos de un respectivo negocio.

**RF-3.4. Modificar datos de cliente.** Un cliente podrá modificar sus datos personales con los que se ha registrado en un negocio.

**RF-3.5. Consultar citas de trabajador.** Un cliente podrá consultar su historial de citas previas reservadas.

**RF-4. Gestión de citas previas.** Se permitirá realizar reservas o cancelaciones de citas previas así como acceder a un histórico de estas según el tipo de usuario.

- RF-4.1. Reserva de cita previa por cliente.** Una cita previa podrá ser reservada por un cliente siempre que la fecha sea válida y haya trabajadores disponibles en el negocio.
- RF-4.2. Cancelación de cita previa por cliente.** Una cita previa podrá ser cancelada por un cliente siempre que esté pendiente.
- RF-4.3. Reserva de cita previa por trabajador.** Una cita previa podrá ser reservada por un trabajador siempre que la fecha sea válida, el trabajador pueda atenderla y el cliente no tenga una cita ya pendiente.
- RF-4.4. Cancelación de cita previa por cliente.** Una cita previa podrá ser cancelada por un trabajador siempre que la cita a cancelar esté asignada a él y esté en estado pendiente.

**RF-5. Gestión de productos.** Se permitirá realizar gestiones sobre los productos de la tienda para su venta online.

- RF-5.1. Alta de producto.** Un producto podrá ser dado de alta por un trabajador de un negocio o por un superadministrador.
- RF-5.2. Baja de producto.** Un producto podrá ser dado de baja por un trabajador de un negocio o por un superadministrador.
- RF-5.3. Modificación de información de producto.** La información de un producto podrá ser modificado por un trabajador de un negocio o por un superadministrador.

**RF-6. Gestión de pedidos.** Se permitirá realizar pedidos online de productos ofertados por el propio negocio así como acceder al historial y estado de los pedidos.

- RF-6.1. Realización de pedido.** Un pedido podrá ser realizado por un usuario del sistema de productos disponibles en la tienda web ofrecidos por el negocio.
- RF-6.2. Cancelación de pedido.** Un pedido podrá ser cancelado por un usuario del sistema siempre y cuando pertenezca al propio usuario mismo y siga en un estado pendiente.
- RF-6.3. Seguimiento de pedido.** Un pedido podrá ser seguido por un usuario del sistema siempre y cuando pertenezca al propio usuario mismo. Podrá ver los estados por los que ha ido pasando en el tiempo (pendiente, en preparación, enviado, entregado, etc.), además de poder ver un resumen del mismo.

#### 4.2.2. Requisitos de información

Requisitos de información, describen la información que debe almacenar y gestionar el sistema para dar soporte a los procesos de negocio.

##### RI-1. Administrador

**Contenido:** email, nombre, apellidos, contraseña, número de teléfono, dirección y fecha de creación.

**Requisitos asociados:** RF-1

##### RI-2. Trabajador

**Contenido:** email, nombre, apellidos, contraseña, número de teléfono, dirección y fecha de creación.

**Requisitos asociados:** RF-2.1, RF-2.2, RF-2.3, RF-2.4

##### RI-3. Cliente

**Contenido:** email, nombre, apellidos, contraseña, número de teléfono, dirección y fecha de creación.

**Requisitos asociados:** RF-3.1, RF-3.2, RF-3.3, RF-3.4

##### RI-4. Citas

**Contenido:** trabajador, cliente, fecha de reserva.

**Requisitos asociados:** RF-4, RF-2.5, RF-3.5

##### RI-5. Producto

**Contenido:** nombre, descripción, categoría, stock, precio, fecha de creación.

**Requisitos asociados:** RF-5 RF-6.1

##### RI-6. Pedido

**Contenido:** cliente, dirección de envío, fecha de pedido, productos, estado, importe total.

**Requisitos asociados:** RF-6

#### 4.2.3. Restricciones semánticas

Restricciones aplicadas a los requisitos funcionales y a los requisitos de información para su correcto uso y almacenamiento.

**RS-1. Alta superadministrador.** En el formulario de alta de superadministrador deberá comprobarse:

- RS-1.1.** **Email:** debe tener un formato válido.
- RS-1.2.** **Nombre:** no puede ser nulo o estar vacío.
- RS-1.3.** **Apellidos:** no puede ser nulo o estar vacío.
- RS-1.4.** **Número de teléfono:** debe ser un número válido.
- RS-1.5.** **Contraseña:** alfanumérica de más de 8 caracteres.
- RS-1.6.** **Dirección:** puede ser nulo hasta el momento de la realización de un pedido.

**RS-2. Alta de trabajador.** En el formulario de alta de trabajador deberá comprobarse:

- RS-2.1.** **Email:** debe tener un formato válido.
- RS-2.2.** **Nombre:** no puede ser nulo o estar vacío.
- RS-2.3.** **Apellidos:** no puede ser nulo o estar vacío.
- RS-2.4.** **Número de teléfono:** debe ser un número válido.
- RS-2.5.** **Contraseña:** alfanumérica de más de 8 caracteres.
- RS-2.6.** **Dirección:** puede ser nulo hasta el momento de la realización de un pedido.

**RS-3. Alta de cliente.** En el formulario de alta de cliente deberá comprobarse:

- RS-3.1.** **Email:** debe tener un formato válido.
- RS-3.2.** **Nombre:** no puede ser nulo o estar vacío.
- RS-3.3.** **Apellidos:** no puede ser nulo o estar vacío.
- RS-3.4.** **Número de teléfono:** debe ser un número válido.
- RS-3.5.** **Contraseña:** alfanumérica de más de 8 caracteres.
- RS-3.6.** **Dirección:** puede ser nulo hasta el momento de la realización de un pedido.

**RS-4. Reserva de cita previa por cliente.** Habrá varias restricciones para realizar la reserva de citas

- RS-4.1.** **Fecha válida:** la fecha está en horario de trabajo.
- RS-4.2.** **Trabajador disponible:** hay un trabajador disponible para atender la cita a la fecha indicada.
- RS-4.3.** **No existen citas pendientes:** el cliente no tiene ya una cita previa pendiente.

**RS-5. Reserva de cita previa por trabajador.** Habrá varias restricciones para realizar la reserva de citas

**RS-5.1. Fecha válida:** la fecha está en horario de trabajo.

**RS-5.2. Trabajador disponible:** el trabajador debe estar disponible para atender la cita.

**RS-5.3. Cliente:** deberá indicarse un cliente existente para la cita.

**RS-5.4. No existen citas pendientes:** el cliente no tiene ya una cita previa pendiente.

**RS-6. Alta de producto.** En el formulario de alta de producto deberá comprobarse:

**RS-6.1. Nombre:** no puede ser nulo o estar vacío.

**RS-6.2. Descripción:** no puede ser nulo o estar vacío.

**RS-6.3. Categoría:** no puede ser nulo o estar vacío.

**RS-6.4. Stock:** debe de ser un entero mayor o igual a cero.

**RS-6.5. Precio:** debe ser un número con decimales mayor o igual a cero.

**RS-7. Realización de pedido.** En el formulario de alta de producto deberá comprobarse:

**RS-7.1. Dirección de envío:** no puede ser nulo o estar vacío.

**RS-7.2. Productos:** no puede ser nulo o estar vacío.

#### 4.2.4. Requisitos No Funcionales

Estos requisitos especifican criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos ya definidos en los requisitos funcionales.

##### Usabilidad

**RNF-1. Alertas.** Se mostrarán alertas tras las operaciones realizadas por el usuario para informarle del resultado de la misma.

##### Fiabilidad

**RNF-3. Encriptación.** Se encriptarán todos los datos sensibles relativos a los usuarios.

**RNF-4. Administración.** Sólo los superadministradores y administradores podrán acceder a funciones de gestión del sistema como puede ser configuración de negocio, cambio en estado de pedidos, etc.

### Rendimiento

**RNF-7. Concurrencia.** Podrán trabajar varios administradores de manera simultánea y hacer uso del sistema varios usuarios al mismo tiempo.

**RNF-8. Disponibilidad.** El sistema se encontrará disponible 24 horas al día.

### Soporte

**RNF-10. Mantenimiento.** El sistema deberá mantenerse en un desarrollo continuo para nuevas implementaciones y mantenimiento o actualización de versiones.

### Restricciones de implementación

**RNF-13.** La implementación se realizará con el patrón Modelo-Vista-Controlador, donde el modelo y el controlador utilizarán el lenguaje PHP bajo el framework de Symfony (siendo MySQL el motor de base de datos del modelo) y JSX bajo el framework de ReactJS para la vista de usuario.

## 4.3. Modelo de casos de uso

Los modelos de caso de uso proporcionan información detallada sobre los comportamientos del sistema o la aplicación de software que se está desarrollando. Vamos a exponer en los siguientes puntos los diagramas necesarios para la implementación de las funcionalidades del sistema.

### 4.3.1. Diagramas de casos de uso

Un diagrama de casos de uso es una forma de diagrama de comportamiento UML mejorado. Con estos representaremos los procesos del sistema de negocio así como los procesos de programación orientada a objetos.

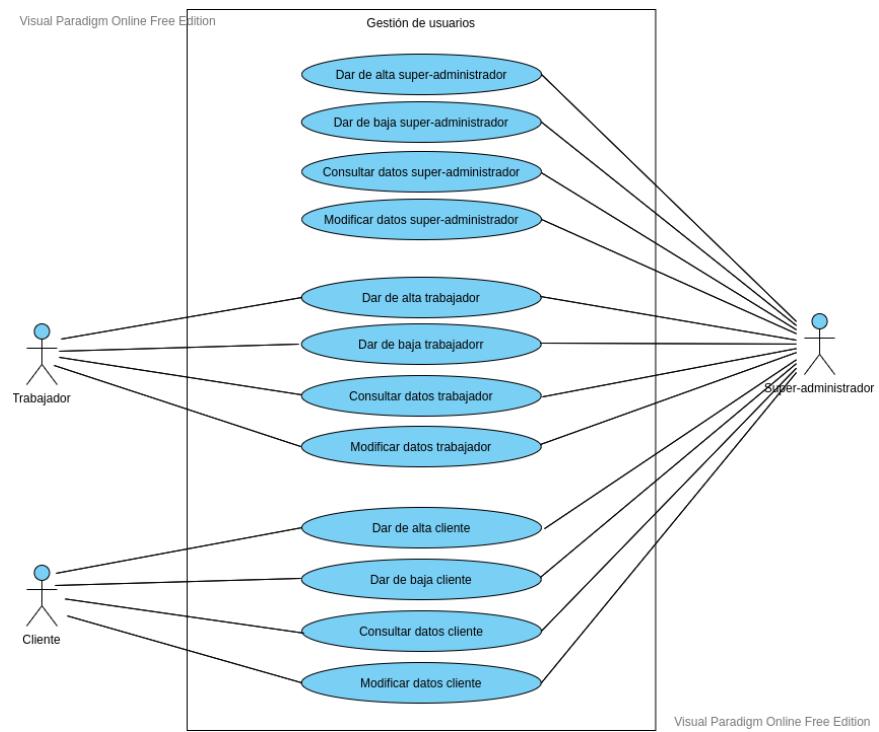


Figura 4.1: Diagrama de caso de uso - Gestión de usuarios

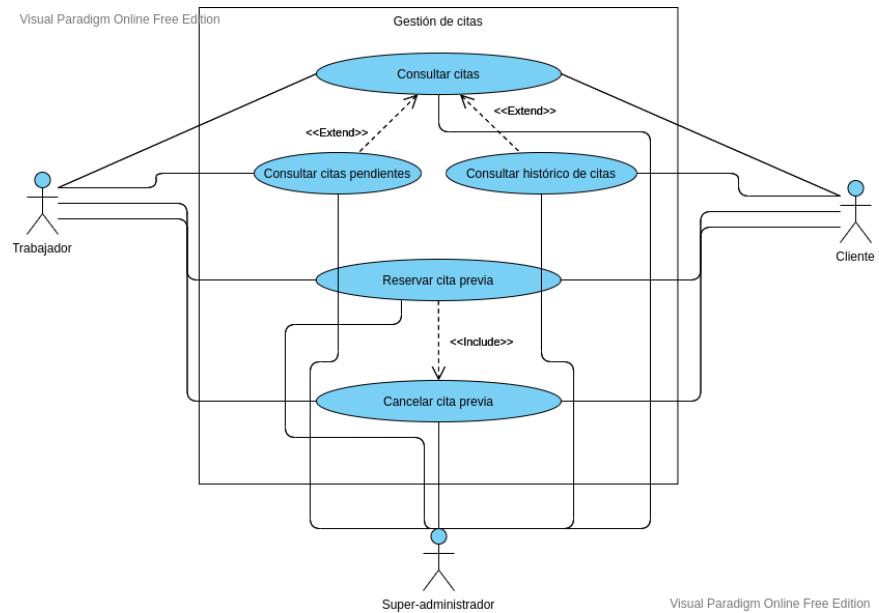


Figura 4.2: Diagrama de caso de uso - Gestión de citas

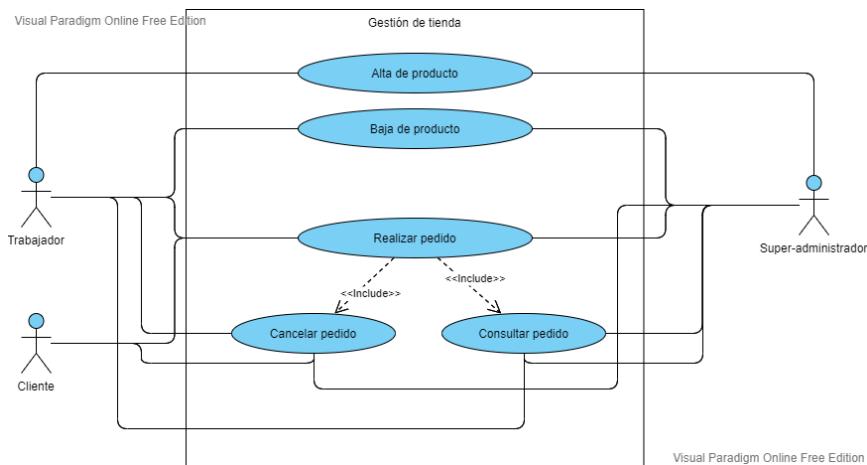


Figura 4.3: Diagrama de caso de uso - Gestión de tienda

#### 4.3.2. Descripción de los casos de uso

En las siguientes tablas se detallan los casos de uso definidos en el apartado anterior. Podemos ver información sobre los actores que interactúan en ella, el tipo, los requisitos funcionales incluidos, las precondiciones y postcondiciones relativas a este y el propósito que cumple, además de un breve resumen que describe a un más alto nivel lo que proporciona el caso de uso.

Los diferentes tipos de casos de uso son:

- Segundo su importancia:
  - Primarios: Procesos comunes más importantes.
  - Secundarios: Procesos menores o raros.
  - Opcionales: Procesos que pueden no abordarse.
- Segundo el nivel de detalle
  - Alto nivel: Descripción general del procesamiento.
  - Extendidos: Descripción de la secuencia de acción completa entre actores y sistema.
- Segundo el nivel de abstracción
  - Esencial: Expresado de forma abstracta, contiene poca tecnología y pocos detalles de diseño.
  - Real: Expresado en base al diseño actual, en el que aparecen relaciones con la interfaz de usuario.

<b>Caso de uso</b>	Dar de alta superadministrador	CU-1
<b>Actores</b>	Superadministrador	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-1.1	
<b>Precondición</b>	Debe existir previamente un superadministrador que pueda llevar a cabo esta función. Si no hay ninguno deberá añadirse por base de datos	
<b>Postcondición</b>	El superadministrador quedará registrado en el sistema	

<b>Propósito</b>
Dar de alta un nuevo superadministrador en el sistema

<b>Resumen</b>
Se darán de alta usuarios con rol superadministrador, los cuales podrán encargarse de la supervisión del sistema

<b>Caso de uso</b>	Dar de baja superadministrador	CU-2
<b>Actores</b>	Superadministrador	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-1.2	
<b>Precondición</b>	Debe existir previamente el superadministrador al que se quiera realizar la baja	
<b>Postcondición</b>	El superadministrador quedará eliminado del sistema	

<b>Propósito</b>
Dar de baja a un superadministrador del sistema

<b>Resumen</b>
Se darán de baja a usuarios con rol superadministrador, eliminando su información de la base de datos del sistema

<b>Caso de uso</b>	Consultar datos del superadministrador	CU-3
<b>Actores</b>	Superadministrador	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-1.3	
<b>Precondición</b>	Debe existir previamente el superadministrador para poder consultar sus datos	
<b>Postcondición</b>	El superadministrador no sufrirá modificaciones	

<b>Propósito</b>
Consultar los datos personales como usuario superadministrador del sistema

<b>Resumen</b>
Se podrá consultar los datos personales como usuario con rol superadministrador del sistema

<b>Caso de uso</b>	Modificar datos del superadministrador	CU-4
<b>Actores</b>	Superadministrador	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-1.4	
<b>Precondición</b>	Debe existir previamente el superadministrador para poder modificar sus datos personales	
<b>Postcondición</b>	Los datos del superadministrador sufrirán modificaciones	

<b>Propósito</b>
Modificar los datos personales como usuario superadministrador del sistema

<b>Resumen</b>
Se podrán modificar los datos personales como usuario con rol superadministrador del sistema

<b>Caso de uso</b>	Dar de alta trabajador	CU-5
<b>Actores</b>	Superadministrador y trabajador	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-2.1	
<b>Precondición</b>	Debe existir previamente un superadministrador o un trabajador del negocio que pueda llevar a cabo esta función. Si no hay ninguno deberá añadirse por base de datos	
<b>Postcondición</b>	El trabajador quedará registrado en el sistema	

<b>Propósito</b>
Dar de alta un nuevo trabajador en el negocio

<b>Resumen</b>
Se darán de alta usuarios con rol trabajador, los cuales podrán encargarse de la supervisión del negocio y atender citas

<b>Caso de uso</b>	Dar de baja trabajador	CU-6
<b>Actores</b>	Superadministrador y trabajador	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-2.2	
<b>Precondición</b>	Debe existir previamente el trabajador al que se quiera realizar la baja	
<b>Postcondición</b>	El trabajador quedará eliminado del sistema	

<b>Propósito</b>
Dar de baja a un trabajador del sistema

<b>Resumen</b>
Se darán de baja a usuarios con rol trabajador, eliminando su información de la base de datos del sistema

Caso de uso	Consultar datos del trabajador	CU-7
Actores	Superadministrador y trabajador	
Tipo	Primario, básico y esencial	
Referencias	RF-2.3	
Precondición	Debe existir previamente el trabajador para poder consultar sus datos	
Postcondición	El trabajador no sufrirá modificaciones	

<b>Propósito</b>
Consultar los datos personales como usuario trabajador del sistema de un negocio determinado

<b>Resumen</b>
Se podrá consultar los datos personales como usuario con rol trabajador del sistema de un negocio determinado

Caso de uso	Modificar datos del trabajador	CU-8
Actores	Superadministrador y trabajador	
Tipo	Primario, básico y esencial	
Referencias	RF-2.4	
Precondición	Debe existir previamente el trabajador para poder modificar sus datos personales	
Postcondición	Los datos del trabajador sufrirán modificaciones	

<b>Propósito</b>
Modificar los datos personales como usuario trabajador del sistema de un negocio determinado

<b>Resumen</b>
Se podrán modificar los datos personales como usuario con rol trabajador del sistema de un negocio determinado

<b>Caso de uso</b>	Dar de alta cliente	CU-9
<b>Actores</b>	Superadministrador, trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-3.1	
<b>Precondición</b>	Un cliente puede darse de alta en un negocio sin precondiciones	
<b>Postcondición</b>	El cliente quedará registrado en el sistema	

<b>Propósito</b>
Dar de alta un nuevo cliente en el negocio

<b>Resumen</b>
Se darán de alta usuarios con rol cliente, los cuales podrán realizar acciones en el negocio determinado

<b>Caso de uso</b>	Dar de baja cliente	CU-10
<b>Actores</b>	Superadministrador, trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-3.2	
<b>Precondición</b>	Debe existir previamente el cliente al que se quiera realizar la baja	
<b>Postcondición</b>	El cliente quedará eliminado del sistema	

<b>Propósito</b>
Dar de baja a un cliente del sistema

<b>Resumen</b>
Se darán de baja a usuarios con rol cliente, eliminando su información de la base de datos del sistema

<b>Caso de uso</b>	Consultar datos del cliente	CU-11
<b>Actores</b>	Superadministrador, trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-3.3	
<b>Precondición</b>	Debe existir previamente el cliente para poder consultar sus datos	
<b>Postcondición</b>	El cliente no sufrirá modificaciones	

<b>Propósito</b>
Consultar los datos personales como usuario cliente del sistema de un negocio determinado

<b>Resumen</b>
Se podrá consultar los datos personales como usuario con rol cliente del sistema de un negocio determinado

<b>Caso de uso</b>	Modificar datos del cliente	CU-12
<b>Actores</b>	Cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-3.4	
<b>Precondición</b>	Debe existir previamente el cliente para poder modificar sus datos personales	
<b>Postcondición</b>	Los datos del cliente sufrirán modificaciones	

<b>Propósito</b>
Modificar los datos personales como usuario cliente del sistema de un negocio determinado

<b>Resumen</b>
Se podrán modificar los datos personales como usuario con rol cliente del sistema de un negocio determinado

<b>Caso de uso</b>	Consultar citas	CU-13
<b>Actores</b>	Trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-2.5, RF-3.5	
<b>Precondición</b>	Debe existir previamente el usuario para consultar sus citas	
<b>Postcondición</b>	Las citas del usuario no sufrirán modificaciones	

<b>Propósito</b>
Consultar las citas de los usuarios de un negocio determinado

<b>Resumen</b>
Un usuario podrá consultar sus citas asignadas, ya sea un trabajador para ver sus citas atendidas y por atender, o un cliente para ver sus citas reservadas

<b>Caso de uso</b>	Consultar citas pendientes	CU-14
<b>Actores</b>	Trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-2.5, RF-3.5	
<b>Precondición</b>	Debe existir previamente el usuario para consultar sus citas pendientes	
<b>Postcondición</b>	Las citas del usuario no sufrirán modificaciones	

<b>Propósito</b>
Consultar las citas pendientes de los usuarios de un negocio determinado

<b>Resumen</b>
Un usuario podrá consultar sus citas pendientes, ya sea un trabajador para ver sus citas a atender, o un cliente para ver sus citas reservadas pendientes

<b>Caso de uso</b>	Consultar histórico de citas	CU-15
<b>Actores</b>	Trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-2.5, RF-3.5	
<b>Precondición</b>	Debe existir previamente el usuario para consultar su histórico de citas	
<b>Postcondición</b>	Las citas del usuario no sufrirán modificaciones	

<b>Propósito</b>
Consultar el histórico de citas de los usuarios de un negocio determinado

<b>Resumen</b>
Un usuario podrá consultar su histórico de citas, ya sea un trabajador o un cliente para ver sus citas registradas

<b>Caso de uso</b>	Reservar cita previa	CU-16
<b>Actores</b>	Trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-4.1, RF-4.3	
<b>Precondición</b>	Debe existir previamente el usuario del cliente y el trabajador para reservar una cita previa	
<b>Postcondición</b>	Se reserverá una cita para un cliente	

<b>Propósito</b>
Reservar una cita para un cliente y trabajador determinado

<b>Resumen</b>
Un usuario podrá reservar una cita previa disponible con un trabajador determinado. De la misma forma un trabajador disponible podrá reservar para un cliente existente que aún no disponga de cita reservada

<b>Caso de uso</b>	Cancelar cita previa	CU-17
<b>Actores</b>	Trabajador y cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-4.2, RF-4.4	
<b>Precondición</b>	Debe existir previamente el usuario con la cita reservada en estado pendiente	
<b>Postcondición</b>	Se cancelará una cita para un cliente	

<b>Propósito</b>
Cancelar una cita para un cliente determinado

<b>Resumen</b>
Un usuario podrá cancelar su cita previa pendiente. De la misma forma un trabajador podrá cancelar una cita a un cliente el cual se le había asignado.

<b>Caso de uso</b>	Alta de producto	CU-18
<b>Actores</b>	Trabajador y superadministrador	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-5.1	
<b>Precondición</b>	El producto no debe existir ya en el sistema	
<b>Postcondición</b>	Se dará de alta un nuevo producto	

<b>Propósito</b>
Dar de alta un nuevo producto para su venta

<b>Resumen</b>
Un trabajador o superadministrador del sistema podrá dar de alta un producto para su negocio determinado, estando este disponible para su venta online.

Caso de uso	Baja de producto	CU-19
Actores	Trabajador y superadministrador	
Tipo	Primario, básico y esencial	
Referencias	RF-5.2	
Precondición	El producto debe existir en el sistema	
Postcondición	Se dará de baja un producto	

**Propósito**

Dar de baja un producto para su venta

**Resumen**

Un trabajador o superadministrador del sistema podrá dar de baja un producto para su negocio determinado, eliminando la disponible para su venta online.

Caso de uso	Realizar pedido	CU-20
Actores	Cliente	
Tipo	Primario, básico y esencial	
Referencias	RF-6.1	
Precondición	El pedido no puede tener cero productos	
Postcondición	Se reducirá el stock correspondiente	

**Propósito**

Realizar un pedido de productos

**Resumen**

Un cliente podrá realizar un pedido de productos ofertados en la tienda online del negocio.

<b>Caso de uso</b>	Cancelación de pedido	CU-21
<b>Actores</b>	Cliente	
<b>Tipo</b>	Primario, básico y esencial	
<b>Referencias</b>	RF-6.2	
<b>Precondición</b>	Deberá existir un pedido en estado pendiente	
<b>Postcondición</b>	Se cancelará el pedido pendiente	

<b>Propósito</b>
Cancelar un pedido pendiente

<b>Resumen</b>
Un cliente podrá cancelar un pedido que se encuentre en estado pendiente.

<b>Caso de uso</b>	Consultar de pedido	CU-22
<b>Actores</b>	Cliente	
<b>Tipo</b>	Secundario, básico y esencial	
<b>Referencias</b>	RF-6.3	
<b>Precondición</b>	Deberá existir un pedido del cliente	
<b>Postcondición</b>	No se verá afectado el pedido	

<b>Propósito</b>
Consular el seguimiento y estado de un pedido

<b>Resumen</b>
Un cliente podrá consultar el estado y realizar un seguimiento de su pedido realziado.

#### 4.4. Diagrama de secuencia

Con los diagramas de secuencia vamos a obtener una previsualización de como el usuario interactuará directamente con el sistema por medio de pasos de mensajes. Vamos por tanto a representar los casos de uso definidos en el apartado anterior como diagramas de secuencia ayudándonos a la futura implementación de los mismos.

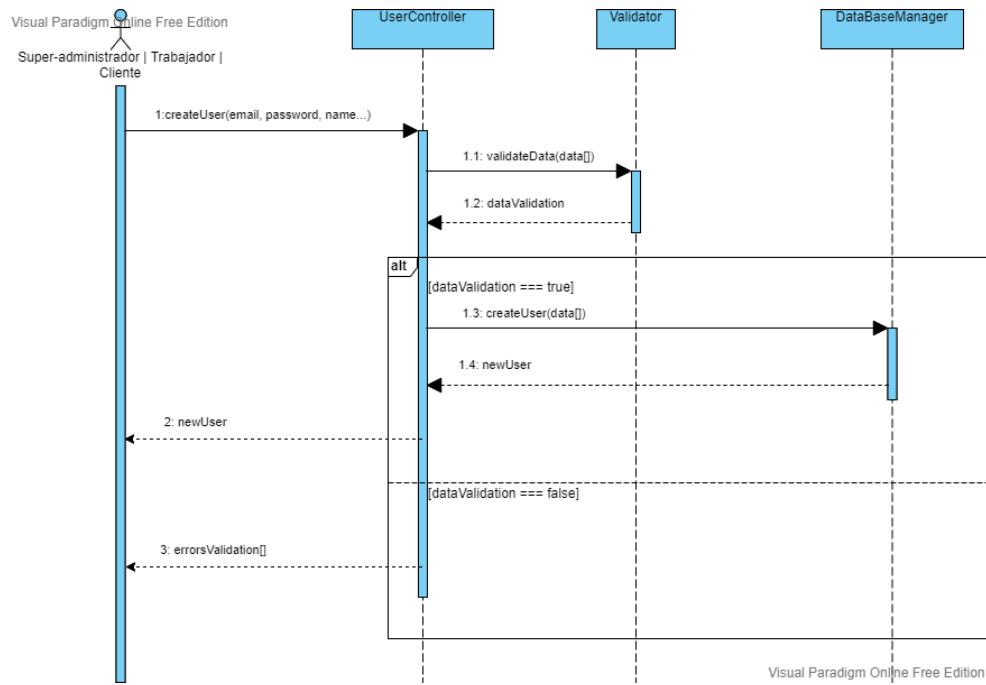


Figura 4.4: DS: Alta de cliente, trabajador o superadministrador.

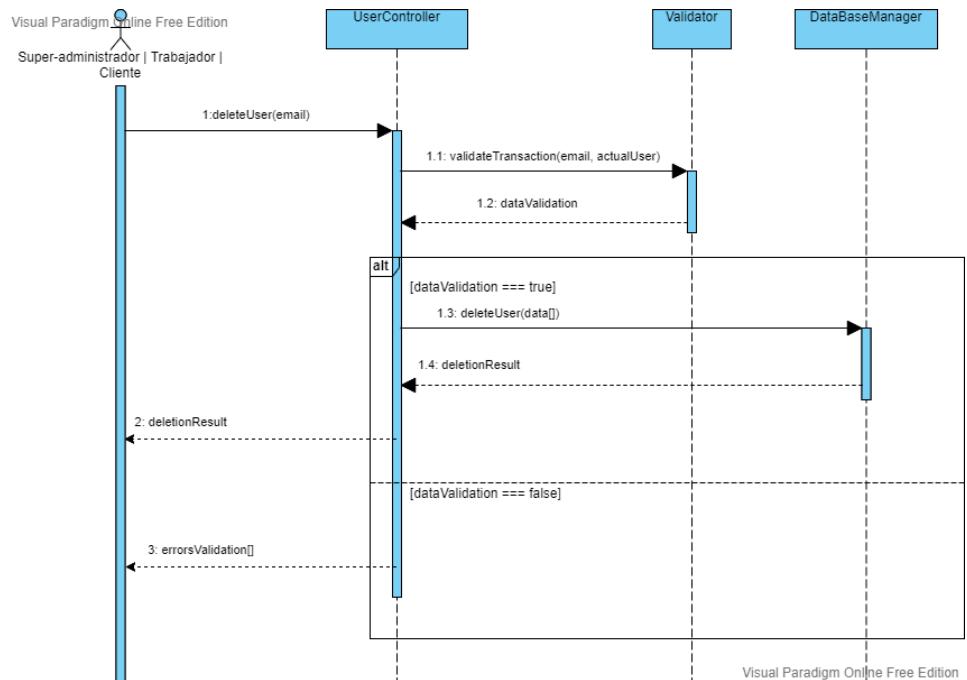


Figura 4.5: DS: Baja de cliente, trabajador o superadministrador.

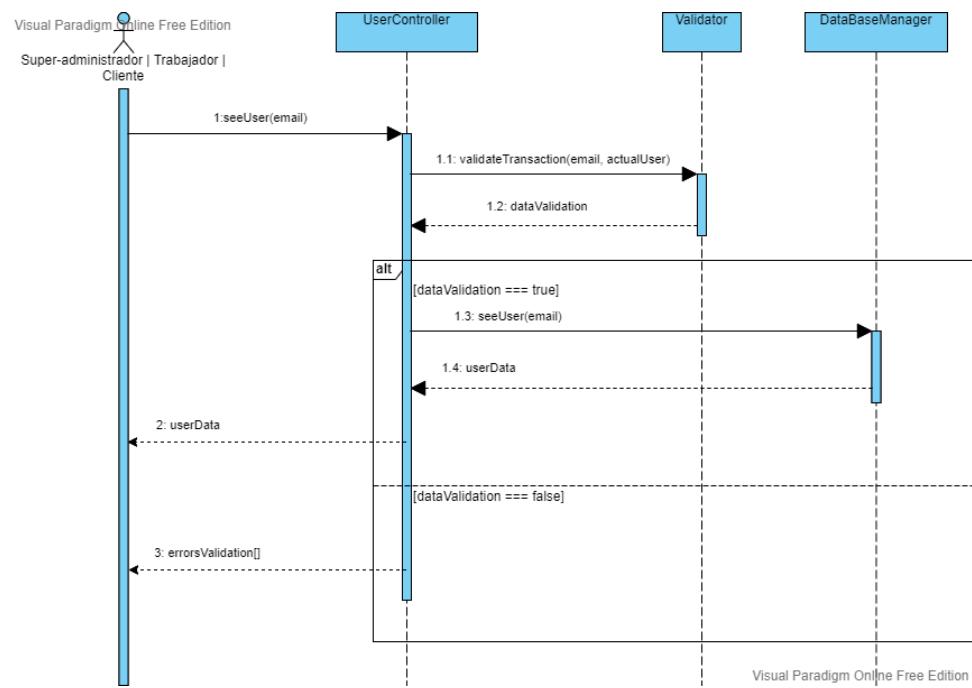


Figura 4.6: DS: Consultar datos de cliente, trabajador o superadministrador.

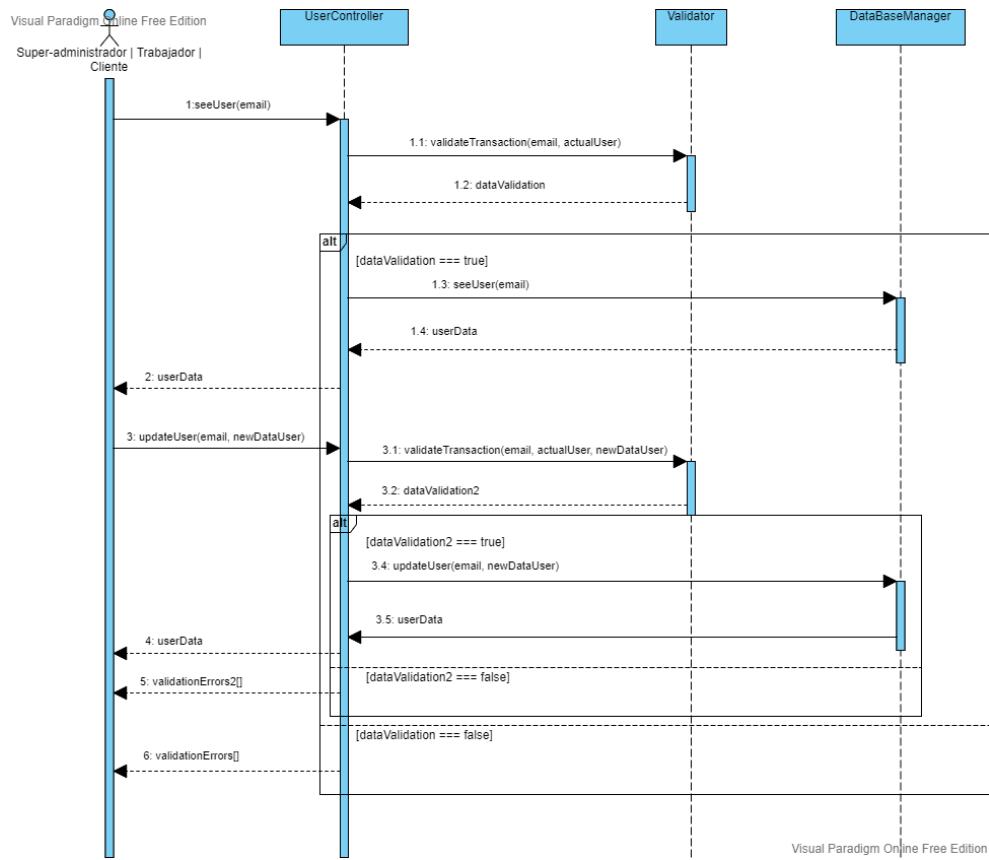


Figura 4.7: DS: Modificar datos de cliente, trabajador o superadministrador.

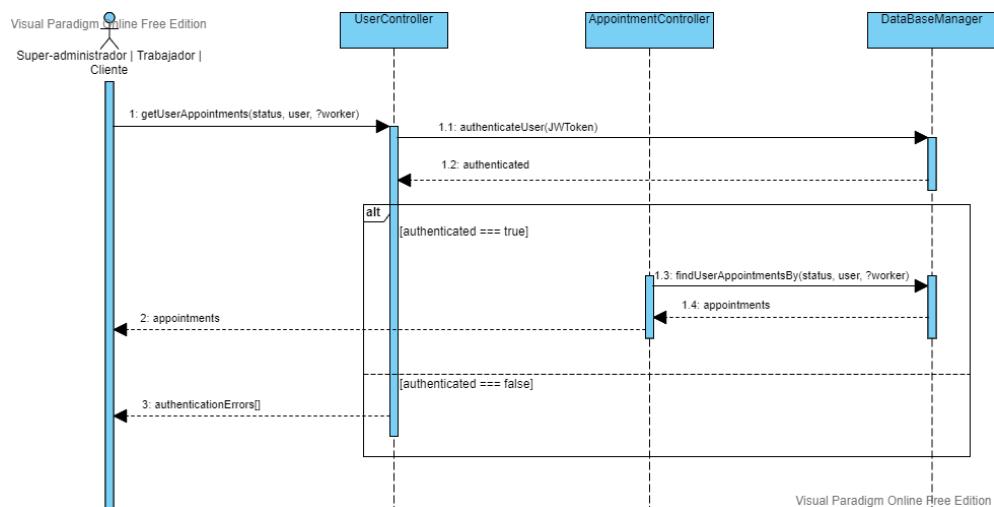


Figura 4.8: DS: Consultar citas por cliente, trabajador o superadministrador.

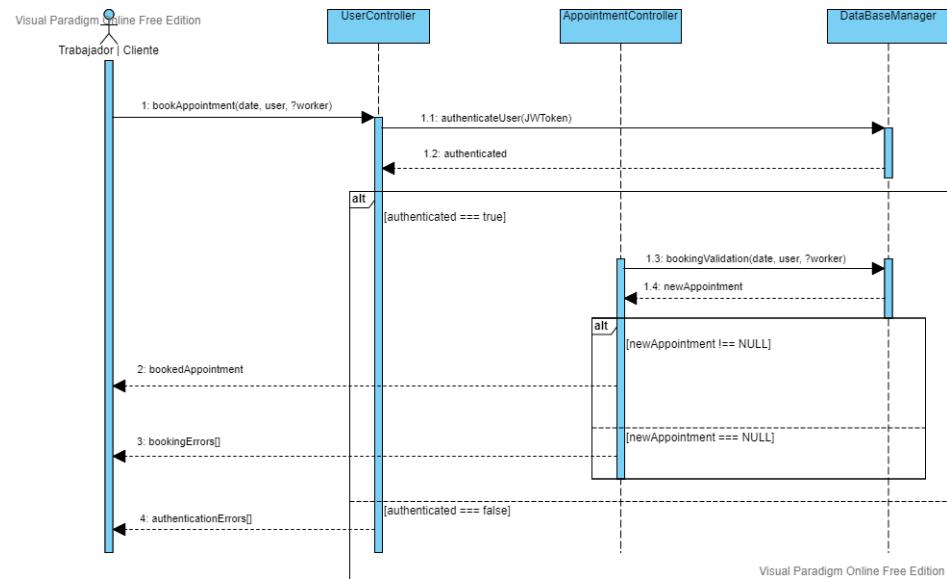


Figura 4.9: DS: Reserva de cita por cliente o trabajador.

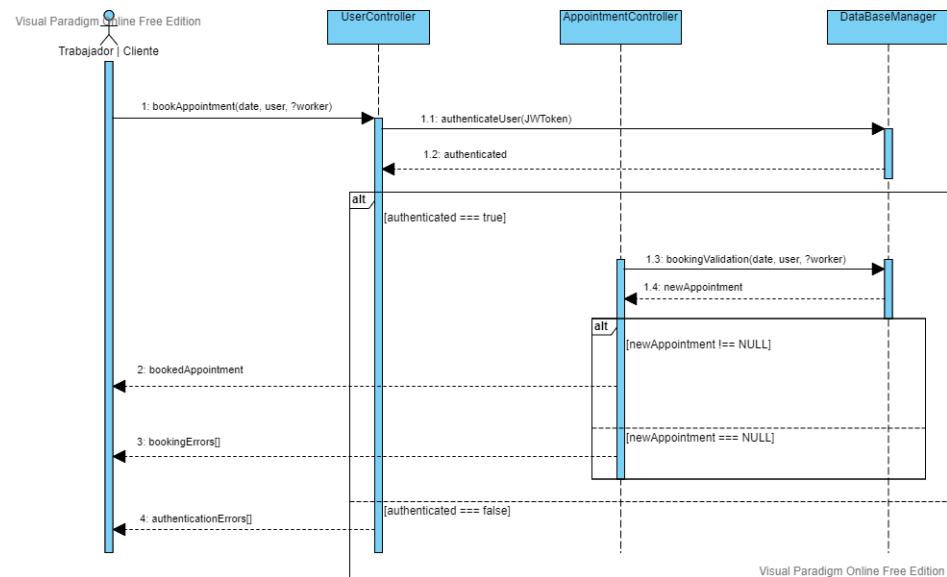


Figura 4.10: DS: Cancelación de cita por cliente o trabajador.

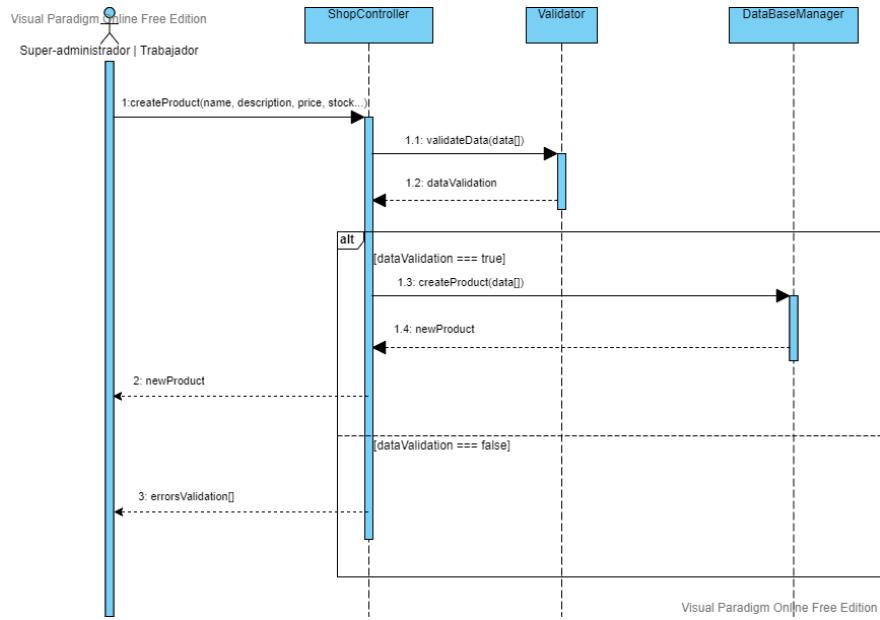


Figura 4.11: DS: Alta de producto para venta online por trabajador o superadministrador.

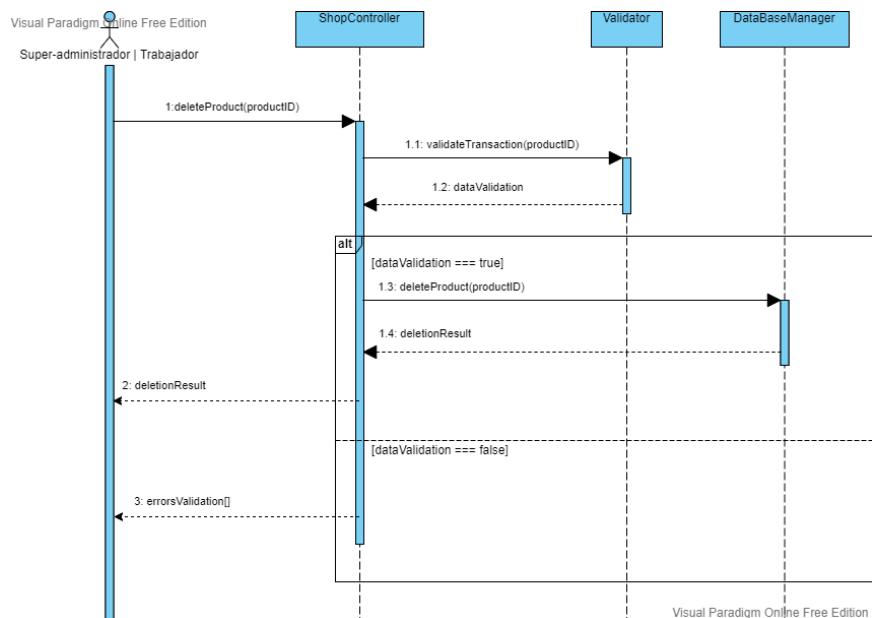


Figura 4.12: DS: Baja de producto para venta online por trabajador o superadministrador.

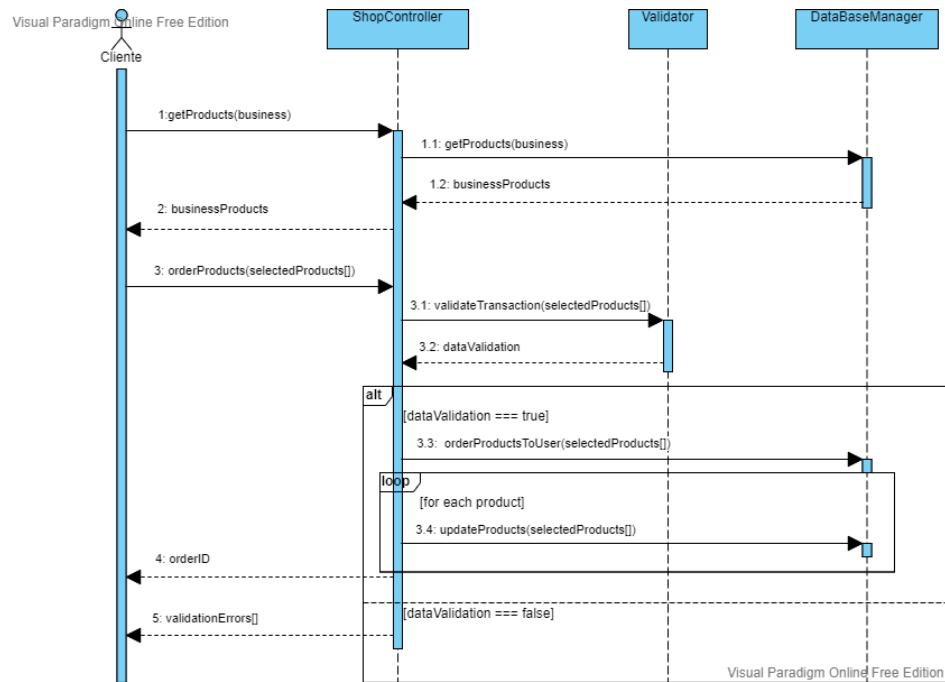


Figura 4.13: DS: Realizar pedido de productos por cliente.

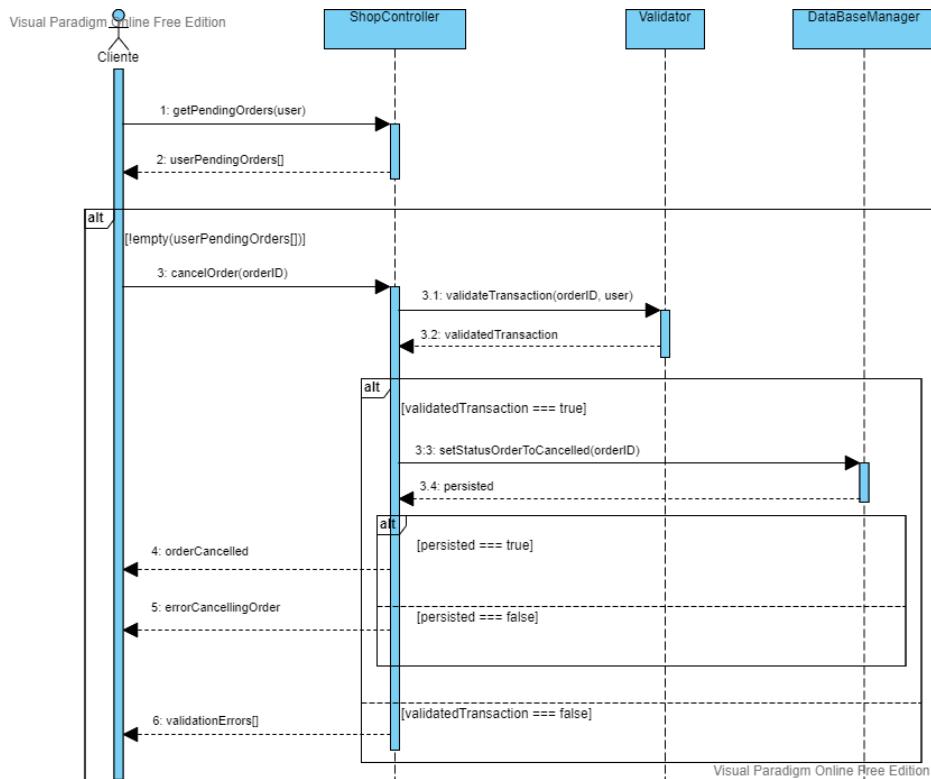


Figura 4.14: DS: Cancelar pedido por cliente.

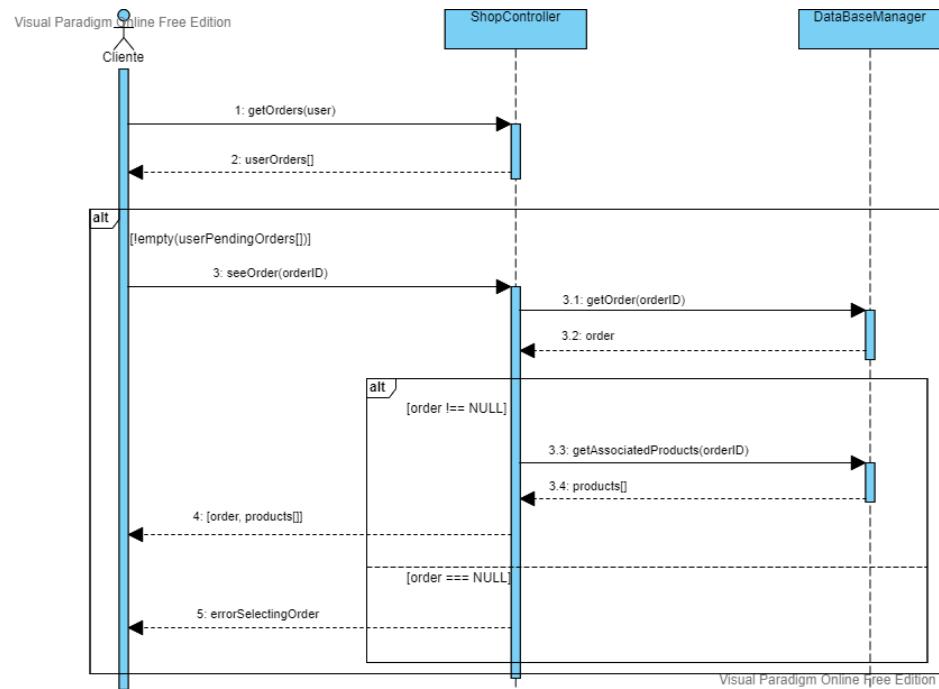


Figura 4.15: DS: Consultar pedido de productos por cliente.

Capítulo

# 5

## Diseño

En este capítulo nos centraremos en el diseño elegido después de haber analizado los requisitos de nuestro sistema. De aquí obtendremos la estructura de base de datos idónea para un correcto desarrollo del aplicativo, además de la arquitectura necesaria para llevar a cabo todos estos procesos.

### 5.1. Base de datos

El sistema puede ser diferenciado en 4 grandes bloques.

Por una parte tenemos a los usuarios. Estos pueden tener varios roles como son los clientes, los trabajadores y los superadministradores. Todos podrán interactuar con las funcionalidades (según su rol) proporcionadas por el web service a los dominios Front-End de cada negocio, sin embargo el rol diferenciador entra en juego en la parte de administración de negocios y el propio sistema mencionado. Un usuario deberá estar ligado a un único negocio, ya que, aunque la información se almacenen en la misma base de datos el usuario final no tiene por qué saber que negocios pertenecen al sistema.

Por otra parte tenemos los negocios, que, para que todo tenga sentido, el sistema permita una rápida implementación de nuevos clientes, donde se crearán configuraciones para las distintas secciones. De esta forma, aunque se separen el Front-End del Back-End, podremos configurar y personalizar nuestra web de la forma que deseemos (dentro de lo permitido por la configuración implementada). Sin olvidar la configuración de horarios, tiempos entre cita y cita, etc.

Las citas son también uno de los bloques principales de este sistema. Se deberá guardar información relevante a la cita, la cual deberá estar asignada a un usuario y a un trabajador del negocio determinado. Guardar datos referentes a la fecha de registro de la cita puede ser interesante de cara al análisis estadístico de los negocios que lo deseen.

Por último, pero no menos importante tenemos la parte de la venta de productos online. Esto conlleva el registro de productos para el negocio y de pedidos

para los clientes que deseen acceder a este servicio. Un producto deberá estar asociado a un negocio, un pedido deberá tener uno o muchos productos además de estar asociado a un único usuario. No podemos olvidar que para realizar el envío de los pedidos realizados el usuario deberá indicar una dirección. Consecuentemente añadiremos una entidad de direcciones que se relacionará con el usuario de forma que un usuario pueda tener cero o más direcciones y una dirección pueda pertenecer a uno o varios usuarios.

Vamos ahora por tanto a representar el diagrama de Entidad-Relación:

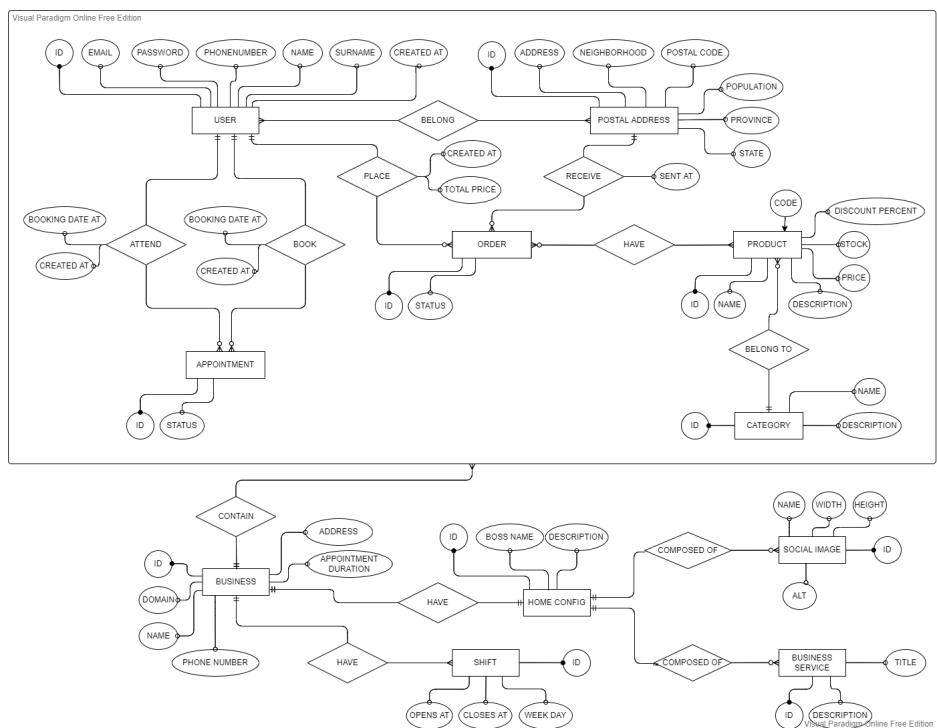


Figura 5.1: Diagrama Entidad-Relación

### 5.1.1. Paso a tablas y fusión

Una vez definido nuestro esquema Entidad-Relación procedemos al paso a tabla y la fusión de las mismas. Primero definiremos las tablas de las entidades del sistema junto con sus relaciones (1-12). En la fusión de las anteriores tablas podemos encontrar la unión de las citas previas (appointment) junto con los trabajadores y clientes que atienden y reciben estas citas respectivamente, quedando así una tabla con todos los datos respectivos a una cita. Ocurre lo mismo con la tabla de pedidos y sus relaciones, ya que uniendo 4-9-11 conseguimos aunar la información del usuario, a donde va dirigido y los datos relativos al mismo como son el estado, el precio, la fecha, etc.

Por último para la agregación del negocio con las respectivas entidades y

relaciones, se añadirá un ID identificativo de la tienda que relacione a esta con cada una de las tablas que son almacenadas en el sistema.



Figura 5.2: Paso a tablas y fusión

### 5.1.2. Normalización

El proceso de normalización es un proceso de descomposición de los esquemas de relación hasta que todas las relaciones alcancen la forma normal deseada.[11] Analizaremos por tanto las diferentes Formas Normales:

- **Primera Forma Normal (1FN):** Una relación está en primera forma normal cuando todos sus atributos son atómicos.[13] Como se observa nuestra base de datos ya se encuentra en la Primera Forma Normal, ya que cumple

con todos los requisitos: sus atributos son atómicos, las tablas contienen una clave primaria única y sin atributos nulos, existe independencia con el orden de filas y columnas.

- **Segunda Forma Normal (2FN):** Una relación está en 2FN si está en 1FN y si los atributos que no forman parte de ninguna clave dependen de forma completa de la clave principal. Es decir que no existen dependencias parciales. (Todos los atributos que no son clave principal deben depender únicamente de la clave principal).[18] Podemos afirmar también que nuestras tablas se encuentran en Segunda Forma Normal debido a la dependencia completamente funcional entre los campos de las mismas.
  
- **Tercera Forma Normal (3FN):** La tabla se encuentra en 3FN si es 2FN y si no existe ninguna dependencia funcional transitiva entre los atributos que no son clave.[21] Por consecuente nuestra base de datos se encuentra en Tercera Forma Normal, ya que no presenta relaciones transitivas (no hay tablas que presenten ningún campo no clave dependiente de ningún otro campo no clave).

## 5.2. Arquitectura del Sistema

Como se ha indicado ya en varios puntos de esta documentación nuestro sistema estará separado en Front-End y Back-End. Aún así el diseño de la arquitectura se basa en el clásico Modelo-Vista-Controlador (MVC) pero dividiéndolo en lo que podríamos llamar módulos.

Por una parte el Web Service implementado será nuestro Modelo-Controlador, que será el encargado de manejar los datos del aplicativo y devolverlos o procesarlos a través de los controladores a los solicitantes. La vista se separa del modelo controlador por cada una de las páginas webs que estén integradas en el sistema, por lo que no habrá únicamente una vista, habrá múltiples vistas que haga uso de la arquitectura de Back-End del Modelo-Controlador.

A su vez en la parte Back-End se implementará una administración para los trabajadores del negocio completando así un nuevo MVC para la gestión de los datos almacenados.

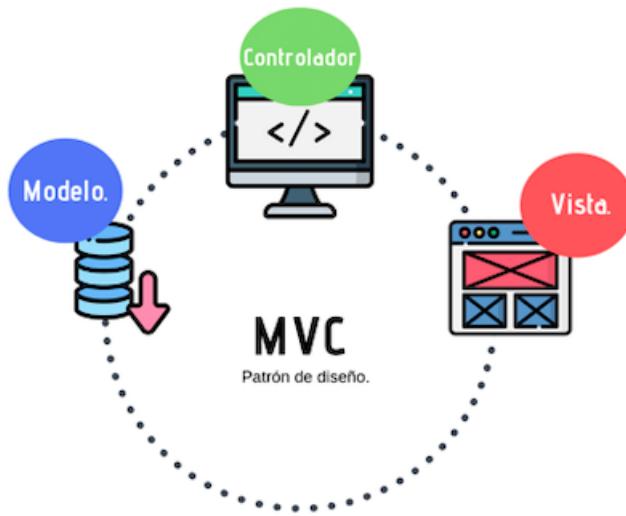


Figura 5.3: Modelo Vista Controlador

### 5.3. Interfaz de Usuario

Previo al comienzo de la implementación y desarrollo de la aplicación esbozaremos unos bocetos de como será la interfaz con la que tengan que interactuar los usuarios de los negocios que se encuentren en el sistema. Un diseño de interfaz [8] puede marcar la diferencia a la hora de las conversiones para el negocio. Por ejemplo, un diseño demasiado complejo o poco llamativo para una tienda online puede hacer que sus ventas bajen considerablemente.

Vamos a definir por tanto los diseños de las páginas principales de los negocios del sistema.

#### 5.3.1. Página de Inicio

Esta será la página principal del negocio. En ella podremos encontrar diferentes secciones con información relativa a este (descripciones del funcionamiento del negocio, de los trabajadores, imágenes de las tareas desempeñadas en el negocio, etc.). Se pretende que el diseño sea llamativo y moderno para que llame la atención del usuario y decida seguir navegando por la web.

Contendrá también una barra de navegación, que acompañará en todas las páginas como podremos ver, y un footer con información importante como puede ser la dirección, teléfono o datos de contacto del personal del negocio.

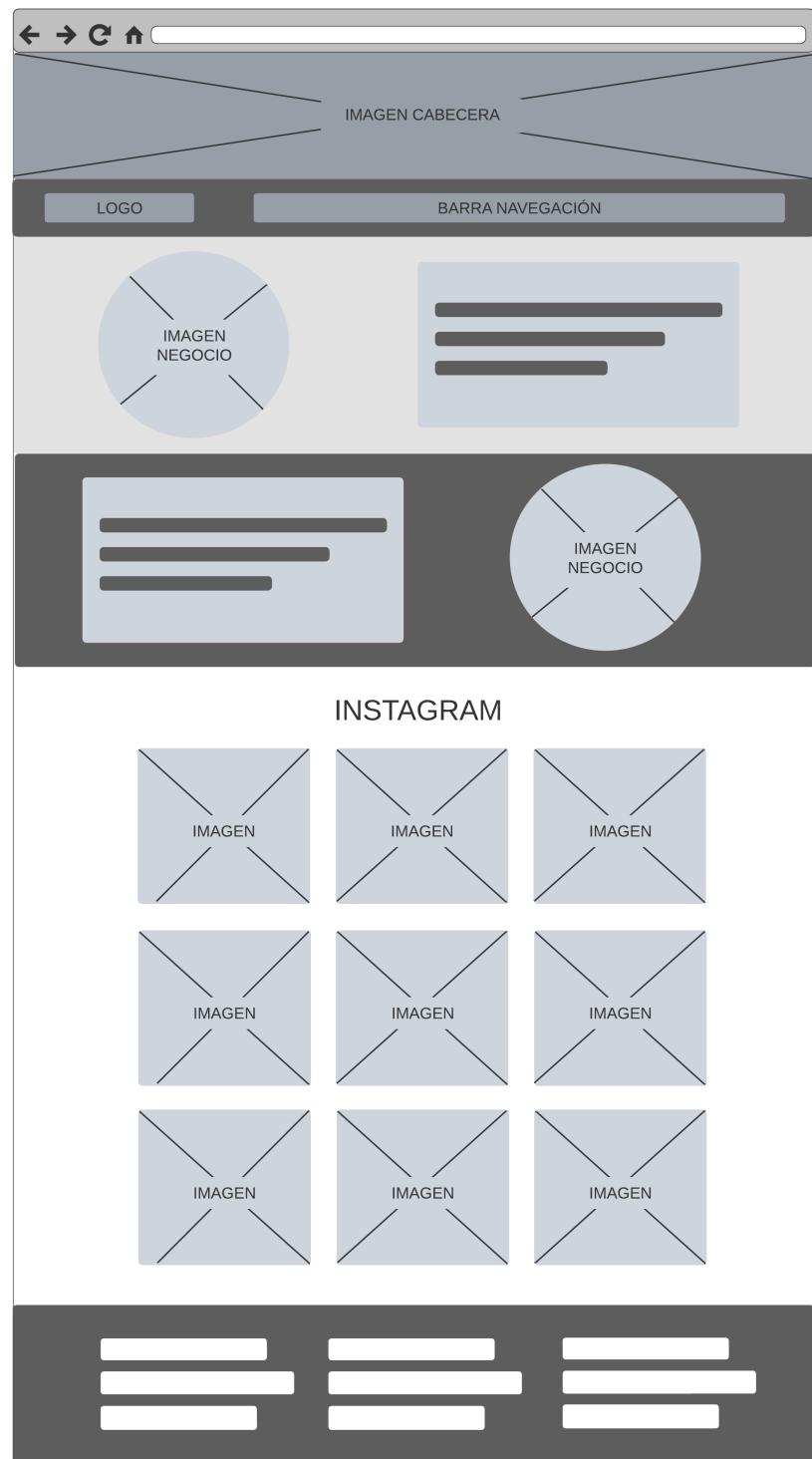


Figura 5.4: Página de Inicio de Web

### 5.3.2. Página de Tienda

La página de Tienda tendrá un formato sencillo y simple. Como hemos indicado, una web demasiado compleja puede llevar a un bajo nivel de tráfico. En un lateral encontraremos la sección de filtros para los productos y en la parte principal encontraremos los productos paginados. La paginación es importante debido a que evitara que el usuario tenga que hacer un scroll desmesurado para encontrar el producto deseado.

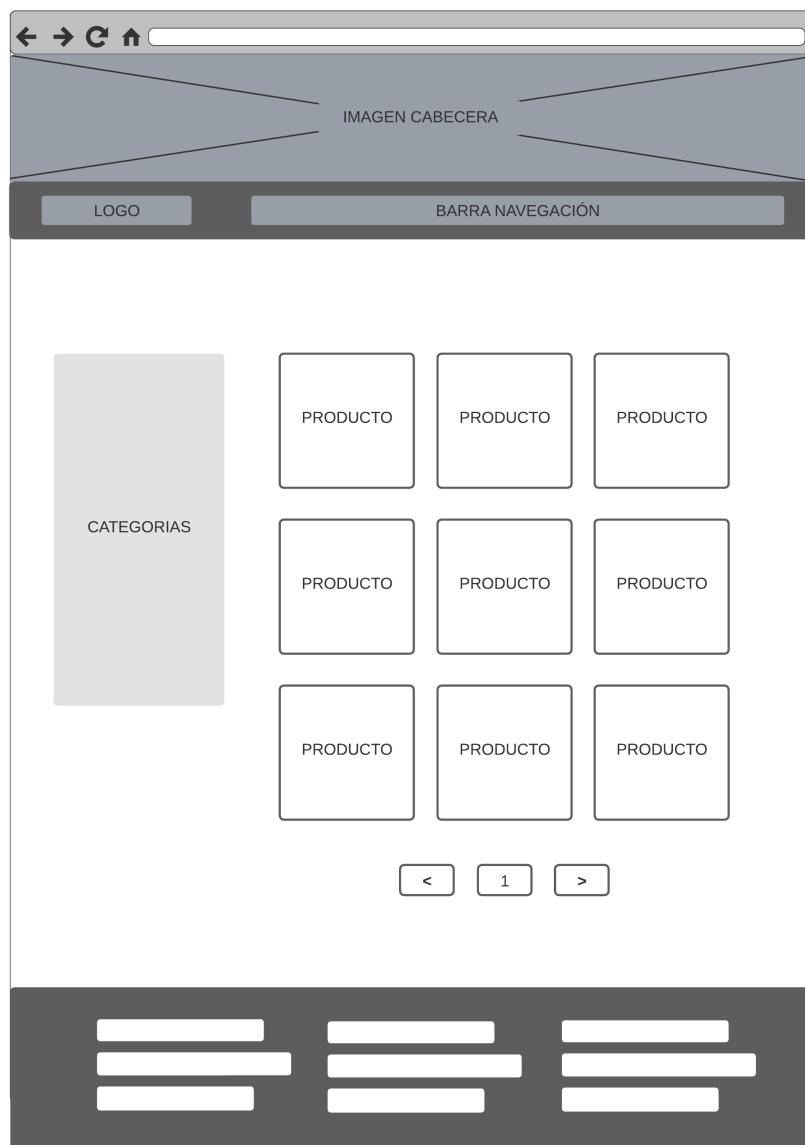


Figura 5.5: Página de Tienda de Web



Figura 5.6: Página de Tienda de Web con Carrito de Compra

### 5.3.3. Página de Reserva de Citas

Al igual que en la página de Tienda, se implementará un diseño simple e intuitivo para la reserva de citas del negocio. En la página podremos ver un calendario, el cual podremos elegir el mes en el que situarnos, y los días del mes marcados con colores en los que el usuario pueda apreciar si el día está disponible o no para reservar una cita sin necesidad de acceder a él.

Al acceder a un día deseado por el usuario podremos ver un Popup similar al de carrito de compra que nos indicará el día seleccionado y las citas disponibles de las que disponemos según el turno de trabajo (mañana o tarde).

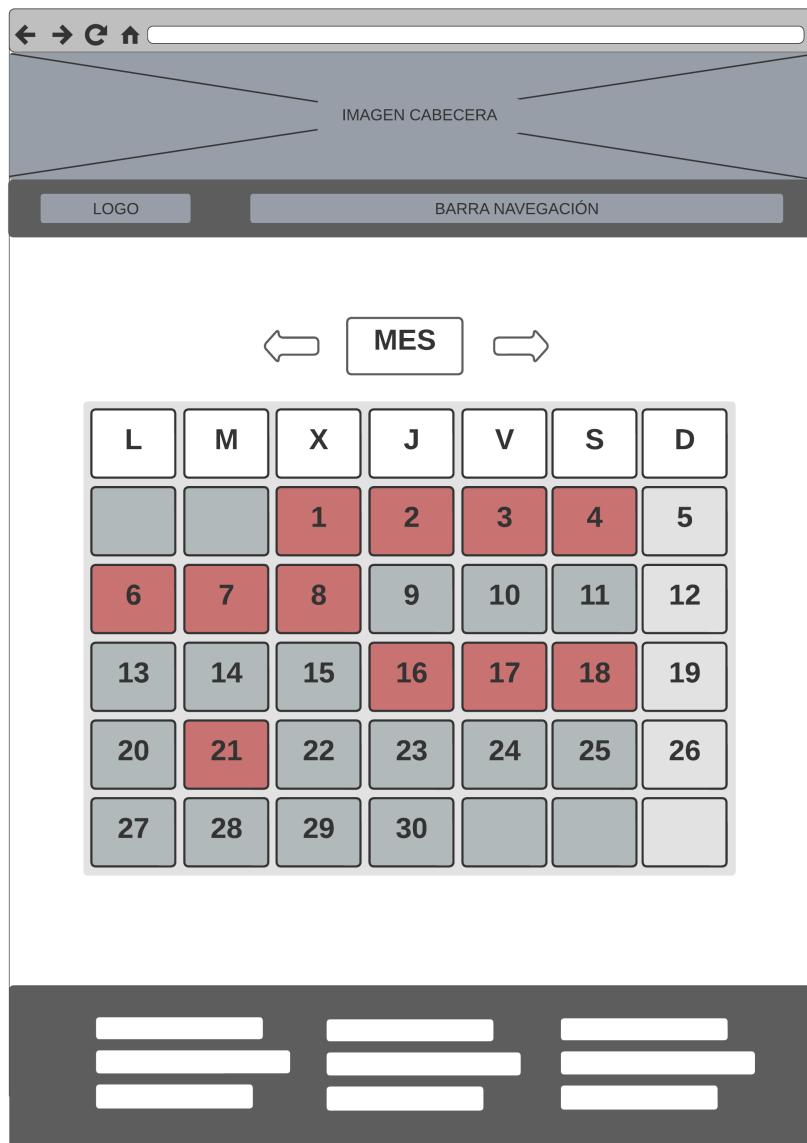


Figura 5.7: Página de Reserva de Citas de Web

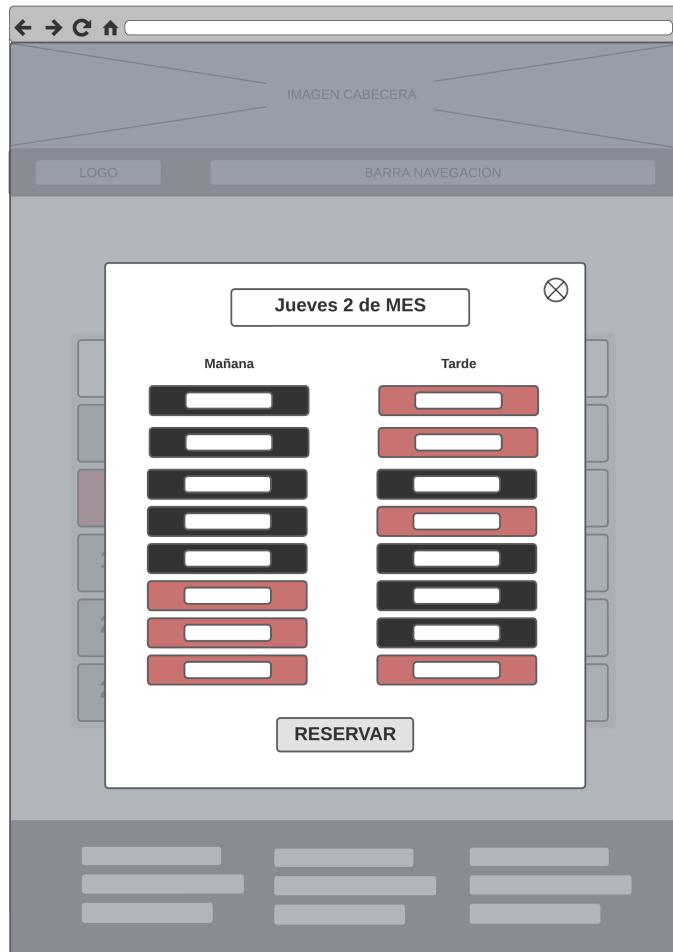


Figura 5.8: Página de Reserva de Citas (selección de día)

#### 5.3.4. Página de Perfil de Usuario

Por último vamos a mostrar el diseño de la página de Perfil de Usuario. En esta podremos ver 3 secciones relativas a toda la información relacionada con los perfiles de los clientes como son los datos personales, las citas reservadas y los pedidos realizados. Las maquetaciones serán sencillas y sin demasiada carga visual para el cliente (dentro de lo posible para mostrar la información importante).

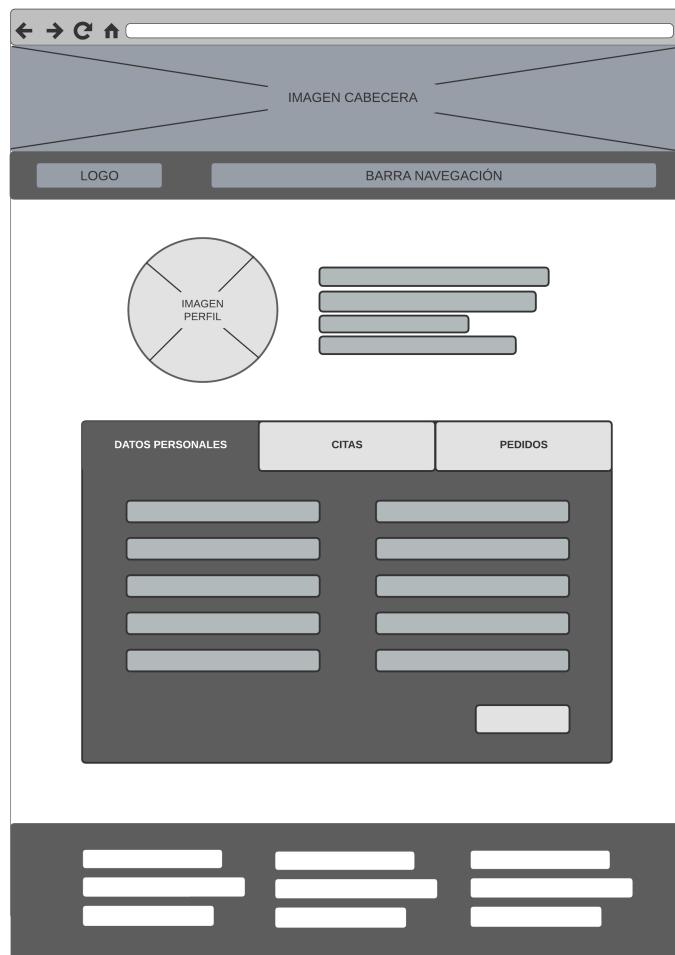


Figura 5.9: Página de Perfil de Usuario (datos personales)

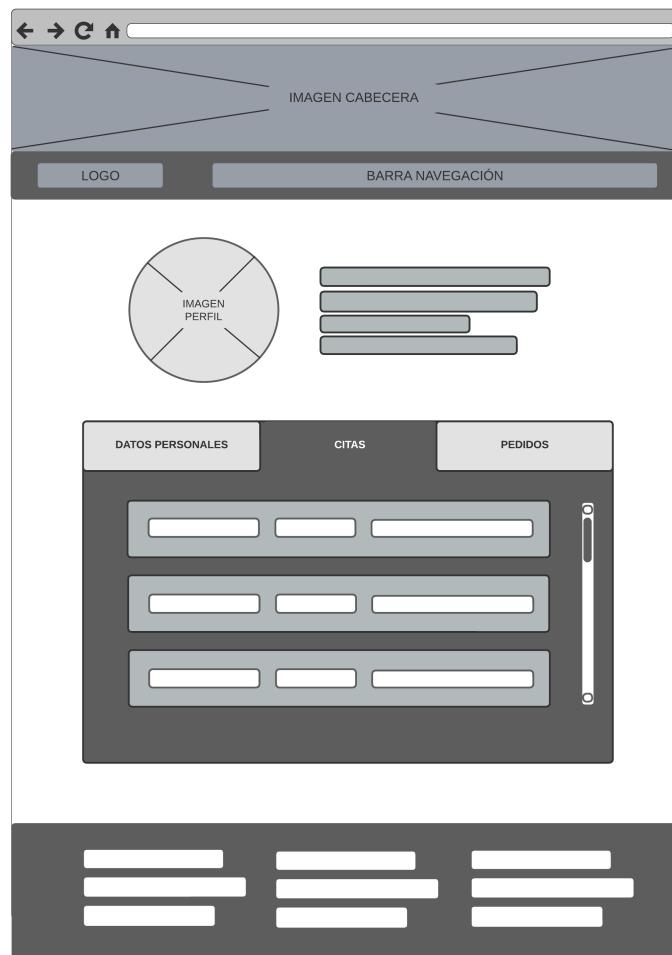


Figura 5.10: Página de Perfil de Usuario (citas)

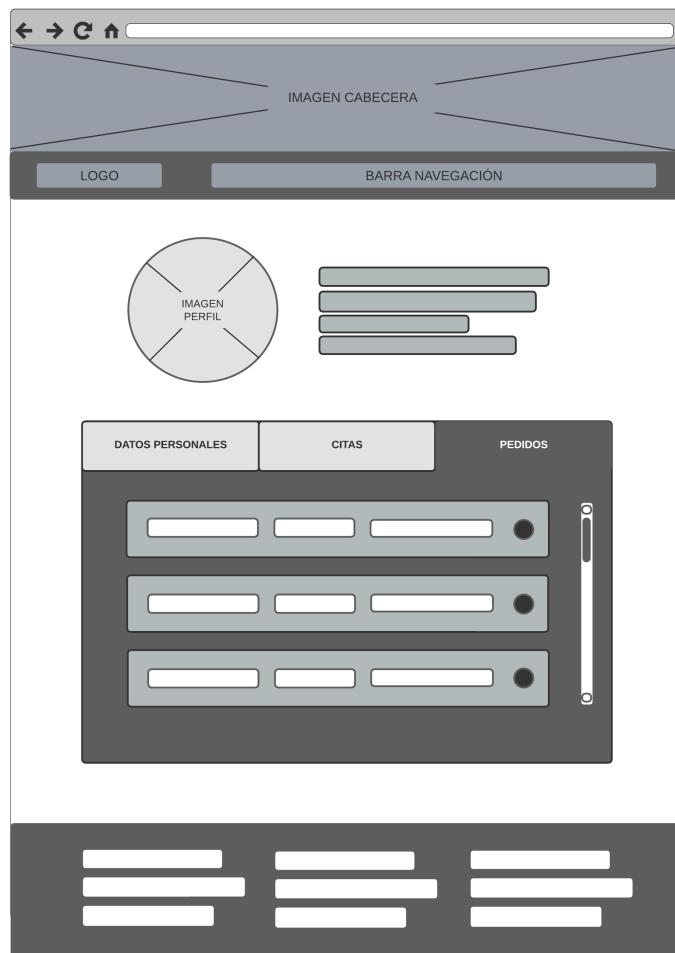


Figura 5.11: Página de Perfil de Usuario (pedidos)

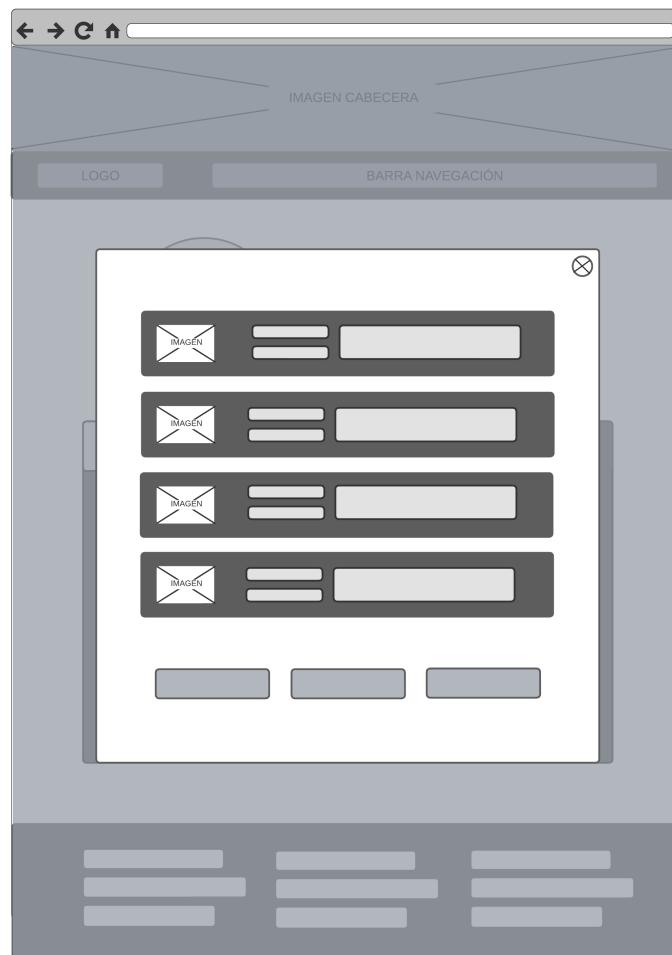


Figura 5.12: Página de Perfil de Usuario (datos de pedido)

Capítulo

# 6

## Implementación

La implementación del software se ha dividido en hitos. Estos han sido definidos en Github ([2], [3]) y cada uno de ellos contiene un grupo de *issues* que se corresponden con las distintas mejoras que se han ido incorporando al software a lo largo de su desarrollo.

### 6.1. Tecnologías

Podemos hacer la división en Front-End y Back-End con las tecnologías utilizadas en este aplicativo.

Para la parte de Front-End se ha utilizado una tecnología algo reciente como es ReactJS de la compañía de Facebook, aunque al ser de software libre da la posibilidad de ser mantenida por la comunidad [15]. React es una biblioteca de JavaScript para construir interfaces de usuario. Está siendo actualmente altamente demandada en puestos de trabajo y es una nueva forma de llevar la programación de la lógica de negocio al Front-End. Con este framework tenemos la posibilidad de construir una página web dinámica y más optimizada para el usuario, ya que React realiza una renderización por componentes y de DOM completo.

Aunque no es requerido la sintaxis utilizada con React para la construcción del Front es JSX [7]. Esta es una extensión de la sintaxis de JavaScript con la que podremos combinar código JavaScript con código HTML facilitando en gran medida la maquetación web y la construcción de componentes. Esto lo acompañaremos con las hojas de estilo en cascada (CSS) con las que daremos la apariencia deseada a los portales de los negocios.

En la parte de Back-End la tecnología utilizada ha sido PHP bajo el framework Symfony [12]. El uso de un framework para el Back-End es casi necesario en estos momentos, debido a que facilita el montaje de un sistema MVC si está enfocado a ello. Además, el uso de capas para la gestión de los modelos y la base

de datos aumenta la seguridad del sistema como por ejemplo, evitar la inyección de código, ya que se sanitizan las entradas de datos. Otra de las ventajas que ofrece Symfony con respecto a otros frameworks es la posibilidad de trabajar a más bajo nivel según deseemos y tener el control de cualquier proceso que implementemos en el proyecto. Sin olvidar tampoco la gran cantidad de Bundles (paquetes) desarrollados por la comunidad con funcionalidades muy interesantes como por ejemplo las sesiones con JSON Web Tokens.

## 6.2. Librerías y Bundles

Le sacaremos el máximo partido a las tecnologías haciendo uso de librerías y paquetes ya desarrollados con funcionalidades específicas. Vamos a destacar a continuación algunos de ellos.

### 6.2.1. EasyAdminBundle

Para la administración del sistema que implementaremos para el Back-End utilizaremos el Bundle EasyAdmin [5] desarrollado para Symfony. Con este paquete podremos fácilmente montar una vista de administración con permisos de usuarios para la gestión completa del sistema o gestión parcial de un único negocio. Esto es posible debido a que podemos montar un Modelo-Vista-Controlador en nuestro Back-End usando como modelo las entidades ya creadas para el aplicativo. Se podrá montar el CRUD de entidades invirtiendo muy poco tiempo y esfuerzo obteniendo una vista de estas ya maquetadas.

### 6.2.2. LexikJWTAuthenticationBundle

Es obvio que en el sistema se deberá implementar una forma de autenticación de usuarios para poder acceder a los datos almacenados en el aplicativo. Se va a hacer uso por tanto de un Bundle desarrollado para Symfony como es LexikJWTAuthenticationBundle [9]. Con éste tendremos la opción de manejar los inicios de sesión con JSON Web Tokens, los cuales se enviarán en las cabeceras de las peticiones de los clientes. De una forma sencilla tendremos la generación automática de estos JWT y comprobación con el usuario al cual corresponde.

### 6.2.3. Symfony mailer

La autenticación de usuarios será un punto necesario para los nuevos clientes que se registren en el sistema. Para ello haremos uso del Bundle Mailer y GoogleMailer para poder hacer envíos de emails, desde una cuenta de Gmail configurada para el sistema, a los usuarios recién registrados. Se implementará un servicio encargado de instanciar el mailer del Bundle y se desarrollarán funcionalidades de envío de email según las necesidades dadas. La única configuración externa al proyecto que necesitaremos será habilitar, en la cuenta de Gmail que

queramos utilizar, la contraseña de 16 caracteres con la que el cliente de mail podrá acceder sin problemas.

Además será necesario realizar la maquetación de un email de validación junto con la lógica de negocio para no dejar al usuario iniciar sesión hasta que su cuenta haya sido verificada.

#### 6.2.4. Stripe

En este caso haremos uso de las librerías tanto en el Front como en el Back-End debido a la dependencia de estos a la hora de realizar y gestionar los pagos. Con esta plataforma gestionaremos **intentos de pago** cuyos pasos hasta ser completados son:

- **Notificación de pedido:** Para que un intento de pago pueda darse, primero, un cliente deberá acceder a su carrito de la compra (no vacío), seleccionar una de sus direcciones postales y pulsar en el botón de continuar. Una vez hecho esto se realizará una llamada al Web Service, creando un pedido en el sistema con estado pendiente.
- **Creación de intento de pago:** Si el pedido se ha creado correctamente, se procederá a crear inmediatamente un intento de pago haciendo uso del **StripeClient** proporcionado por el Bundle. Si no ha ocurrido ningún error, se generarán dos valores: el ID del intento de pago en Stripe y el ClientSecret necesario para realizar el pago en la parte de Front. Estos valores se establecerán en el pedido creado y se devolverá como respuesta de Web Service al usuario.
- **Pago:** Una vez recibido el pedido cargaremos el formulario donde se introducirá la tarjeta de crédito. Tras completar los datos, podremos confirmar el pago junto con el ClientSecret establecido en el pedido haciendo uso de un hook de Stripe. Si no ha ocurrido ningún error en el pago, se notificará de nuevo al Web Service que ese pedido ha sido pagado con éxito para cambiar su estado en base de datos.

```

const stripe = useStripe();

const onSubmitPayment = async (e) => {
    e.preventDefault();
    setPaying( value: true );
    setMessageAlert( value: undefined );
    if (!complete) {
        setMessageAlert( value: {type: 'danger', 'text': 'No se han completado todos los datos de pago.'} )
    } else {
        const card = elements.getElement(CardElement);
        const payload = await stripe.confirmCardPayment(props.order.clientSecret, data: {payment_method: {card}});

        if ('paymentIntent' in payload
            && 'status' in payload.paymentIntent
            && payload.paymentIntent.status === 'succeeded'
        ) {
            Promise.all( values: [
                notifyPaymentOrder(payload.paymentIntent.id)
            ]).then(response => {
                const data = response.length > 0 ? response[0] : [];

                if(data.result){
                    setMessageAlert( value: {
                        'type': 'success',
                        'text': 'El pago se ha realizado con éxito! Puedes ver tu pedido en la página de ' +
                            'perfil. Estamos redirigiéndote a la página de Perfil...'
                    });
                }
            });
        }
    }
}
    
```

Figura 6.1: Confirmación de pago en ReactJS con Stripe

Pagos		Pagos			
Todas las transacciones		Todos		Exportar + Crear pago	
Fraudes y riesgos		Importe	DESCRIPCIÓN	CLIENTE FECHA	
Facturas		68,49 €	Incompleto	Intento de pago para el usuario antonio112@gmail.com.	5 sept. 10:00 ...
Suscripciones		189,70 €	Exito ✓	Intento de pago para el usuario antonio112@gmail.com.	4 sept. 12:07 ...
Presupuestos		89,40 €	Exito ✓	Intento de pago para el usuario antonio112@gmail.com.	4 sept. 11:29 ...
Enlaces de pagos		99,64 €	Exito ✓	Intento de pago para el usuario antonio112@gmail.com.	3 sept. 22:02 ...
		89,40 €	Exito ✓	Intento de pago para el usuario antonio112@gmail.com.	3 sept. 19:35 ...

Figura 6.2: Historial de pagos en la plataforma Stripe

### 6.2.5. Axios

Realizaremos las llamadas al Back-End desde la parte del cliente con Axios [1]. Con esta librería podremos hacer llamadas asíncronas a los datos de nuestro sistema para eliminar los tiempos de carga completos de nuestras páginas webs. Esto es una parte importante del planteamiento de Front, ya que el objetivo es hacer páginas dinámicas donde el usuario no se vea imposibilitado de interactuar con la web mientras espera la renderización de los productos de la tienda por ejemplo. También las llamadas asíncronas mejorarán la velocidad de carga de estas gracias al DOM se irá cargando poco a poco de manera asíncrona y no estará en su mayor parte bloqueado por la renderización completa.

#### 6.2.6. Reporte de errores con Telegram

Todo sistema informático está expuesto a posibles errores no esperados por los desarrolladores. Detectar estos problemas a tiempo puede ser fundamental para el correcto funcionamiento del aplicativo o la defensa ante posibles patrones de ataques. Para ello usaremos la API de TelegramBot [20] con la que podremos enviarnos notificaciones desde el servidor hasta un Bot de Telegram previamente configurado.

Podremos hacer uso del servicio **TelegramService**:

```
/**
 * TelegramService constructor.
 */
public function __construct(ParameterBagInterface $parameterBag)
{
    /**
     * @noinspection MissingService
     */
    $this->setBot($parameterBag->get('app.telegram_bot_api_token'))
        ->setChatID($parameterBag->get('app.telegram_chat_id_developer'));
}
```

Figura 6.3: Constructor de TelegramService: inicialización de parámetros

```
/**
 * @inheritDoc
 * @return bool bool
 */
public function sendNotificationAppError(AppError $error): bool
{
    $message = $this->formatAppErrorToHTML($error);

    return $this->sendNotification($message);
}

/**
 * @inheritDoc
 * @return bool bool
 */
public function sendNotification(string $message): bool
{
    try {
        $this->getBot()->sendMessage($this->chatID, $message, parseMode: 'HTML');
        $sent = TRUE;
    } catch (TelegramException $e) {
        $sent = FALSE;
    }

    return $sent;
}
```

Figura 6.4: Métodos públicos de TelegramService: enviar mensajes y AppErrors

```

/**
 * Method to format the AppError to HTML.
 *
 * @param AppError $error The AppError to format.
 *
 * @return string string
 */
protected function _formatAppErrorToHTML(AppError $error): string
{
    $message = '<b>' . $error->getCreatedAt()->format('d/m/Y h:i:s') . '</b>' . chr( codepoint: 10 ) .
    '<b>' . $error->getMethod() . '</b>' . chr( codepoint: 10 ) . chr( codepoint: 10 ) .
    $error->getMessage();

    if ($error->getExceptionCode() != NULL):
        $message .= chr( codepoint: 10 ) . '<b>Excepción</b>' . chr( codepoint: 10 ) .
        '<b>Código: </b>' . $error->getExceptionCode() . chr( codepoint: 10 ) .
        '<b>Mensaje: </b>' . $error->getExceptionMessage() . chr( codepoint: 10 );

    endif;

    return $message;
}

```

Figura 6.5: Método privado de TelegramService: formatear mensaje de error

### 6.3. Estructuración de código en Symfony

En este proyecto Symfony vamos a tener los siguientes tipos de archivos:

- **Controller:** Los controladores serán los encargados de recibir las peticiones realizadas al servidor y sanitizar y validar los datos recibidos por el Request.
- **Entity:** Las entidades serán nuestro modelo de datos. Dentro de las entidades haremos uso de una herramienta denominada Annotations de Doctrine. Con esto definiremos nuestras tablas, atributos, relaciones y restricciones de la base de datos del Sistema.
- **Traits:** Los trait son trozos de código los cuales podremos injectar en nuestros diferentes ficheros según necesitemos. Principalmente los usaremos para definir propiedades comunes de entidades y así evitar la duplicidad de código en el proyecto.
- **Repository:** Los repositorios serán los encargados de recuperar las entidades de datos que tengamos almacenados. Con estos se podrán hacer consultas a la base de datos sin necesidad de realizar sentencias SQL directamente.
- **Service:** Los servicios serán la capa bajo los controladores. Estos se encargarán de realizar la lógica de negocio correspondiente a cada petición de los usuarios. Será la única capa con acceso a los repositorios de las entidades en la que se podrán persistir objetos ORM.

- **Interfaces:** Habrá interfaces por cada archivo del proyecto para tener una documentación completa de todas las funcionalidades implementadas.
- **Helpers:** Los helpers serán archivos con funcionalidades de ayuda como su nombre indica. Funcionalidad fuera de la lógica de negocio que nos puede facilitar cálculos repetitivos.

Con esta estructura podríamos decir que nuestro flujo sería el siguiente: el cliente realiza una petición al servidor. Esta llega a la ruta definida en uno de los controladores donde se tratarán los datos obtenidos del Request de manera que si no son válidos, reporte el correspondiente error al cliente. En el caso de ser válidos pasarán al respectivo Servicio (normalmente un controlador deberá tener injectado un servicio). En él se ejecutará la lógica de negocio definida donde se creará, recuperará, modificará o eliminará una entidad definida. Una vez procesado se devolverán los datos correspondientes al controlador (ya sean errores o los datos solicitados) y el controlador los devolverá al cliente con una respuesta en formato JSON.

```
/**
 * @inheritDoc
 * @return JsonResponse JsonResponse
 */
public function createJsonResponse($data, array $validationErrors, AppServiceInterface $service,
                                    int $code = 200): JsonResponse
{
    if (empty($validationErrors)):
        $serviceErrors = $service->getErrors();
        $response = array(
            'data' => $data,
            'result' => empty($serviceErrors),
            'code' => $code,
        );

        if (!empty($serviceErrors)):
            $response['code'] = $serviceErrors[0]->getExceptionCode() ?? $serviceErrors[0]->getType();
            $response['message'] = $serviceErrors[0]->getExceptionMessage() ?? $serviceErrors[0]->getMessage();
        endif;
    else:
        $response['code'] = Response::HTTP_BAD_REQUEST;
        $response['message'] = 'Error en la validación';
        $response['errors'] = $validationErrors;
    endif;

    return new JsonResponse($response);
}
```

Figura 6.6: Funcionalidad de AppController: creación de respuesta JSON para API

Todo esto proceso registrará, en caso de producirse, errores en el aplicativo que serán notificados por Telegram como bien hemos comentado en un apartado anterior. Gracias a estas notificaciones y registro de errores el control para el superadministrador del aplicativo será mucho mayor pudiendo solucionar errores en un menor periodo de tiempo.

## 6.4. Estructuración de código en ReactJS

El objetivo de usar ReactJS es el trabajar con componentes con los que facilitarnos la maquetación y el control de los estados de la aplicación. Para ello vamos a estructurar nuestro código de la siguiente manera:

- **App.js:** Será el archivo principal de la aplicación donde definiremos el routing. Además añadiremos los componentes comunes como pueden ser el Header y el Footer que no recibirán modificaciones en las diferentes vistas.
- **Services:** Aquí almacenaremos los diferentes servicios de nuestra aplicación. Estos serán los encargados de intercambiar información con el Web Service. Separando la lógica de comunicación de lo que son los componentes ganamos en abstracción, ya que, si en algún momento recuperamos los datos de un sitio solamente deberemos actualizar el servicio sin que nada afecte al funcionamiento de la aplicación.
- **Components:** Tendremos un directorio con componentes comunes. El encapsulamiento de estos nos ahorrará trabajo a la hora del diseño Web con lo que conseguiremos no repetir código y tener una mayor homogeneidad a los ojos del usuario.
- **Directorios de páginas:** Por último tendremos nuestros directorios referentes a cada página del proyecto así como los distintos componentes de los que se compone.

## 6.5. Uso y funcionamiento

En esta sección vamos a repasar el funcionamiento de la herramienta, a nivel de Front-End y Back-End, y ver en como afecta la configuración registrada en la administración a la aplicación Web.

### 6.5.1. Página de Home

Como bien se ha explicado, la página de Home se ha hecho de una forma configurable para los trabajadores del negocio, pudiendo mostrar así la información que deseen. Vamos a ver a continuación las comparativas de configuración en la administración y la visualización en la Web:

### 6.5.2. Sección de Introducción

Figura 6.7: Configuración de la sección de Introducción en el Back-End



Figura 6.8: Renderización de la configuración en la Web

### 6.5.3. Sección de Red Social

Figura 6.9: Configuración de la sección de Red Social en el Back-End



Figura 6.10: Renderización de la configuración en la Web

#### 6.5.4. Sección de Servicios

A screenshot of the BusinessAPP back-end interface. On the left, there's a sidebar with navigation links like 'Inicio', 'Control Negocio', 'Negocios', 'Turnos', 'Horas', 'Red Social', 'Servicios', 'Config. Tienda', 'Categoría', 'Productos', 'Config. Clientes', 'Citas', 'Pendientes', 'Config. Pedidos', 'Pedidos', 'Pendientes', 'En Preparación', 'Config. Usuarios', and 'Usuarios'. The main area has a search bar at the top right. Below it, there's a table titled 'Servicios de Negocio' with columns for 'ID', 'Título', and 'Descripción'. There are three rows: 'Tinte' (with a note about leaving hair longer than the rest), 'Corte con Barba' (with a note about leaving the hair on the sides longer), and 'Corte Normal' (with a note about leaving the hair on the sides shorter). To the right of the table, there's a 'Configuración de Horas' section with a 'Nuevo Servicio' button and a list of services offered by 'Barbería Adrián Carrillo'. At the bottom, there are navigation buttons for 'Anterior' and 'Siguiente'.

Figura 6.11: Configuración de la sección de Servicios ofrecidos en el Back-End

A screenshot of the Barbería Adrián Carrillo website. At the top, there's a header with the logo 'Barbería Adrián Carrillo' and a menu with options like 'Home', 'Services', 'About', 'Contact', and 'Blog'. Below the header, there's a section titled '¿Qué ofrecemos?' with three cards: 'Corte Normal' (ideal for fine hair), 'Corte con Barba' (ideal for留长 sides), and 'Tinte' (ideal for留长 hair). Each card has a small icon of a person with a beard. At the bottom, there's a dark footer bar with the text 'Contacta con Barbería Adrián Carrillo' and 'Dirección: Calle Ruiz Moreno N°'.

Figura 6.12: Renderización de la configuración en la Web

### 6.5.5. Sección de Información del negocio

Figura 6.13: Configuración de la sección de Información del negocio en el Back-End

Figura 6.14: Configuración de los turnos del negocio en el Back-End



Figura 6.15: Renderización de la configuración en la Web

### 6.5.6. Página de Productos

En la página de productos también estará la posibilidad de crear, editar o eliminar lo ofertado, pudiendo crear también a su vez las categorías de estos

para realizar filtros en la Web. A continuación veremos las comparativas de esta sección:

The screenshot shows the 'Categorías' (Categories) section of the BusinessAPP Back-End. On the left, a sidebar menu includes 'Inicio', 'Negocios', 'Turnos', 'Home', 'Red Social', 'Servicios', 'COSING, TENDA', 'Categorías', 'Productos', 'COSING, COSAS', 'Citas', 'Pedidos', 'COSING, PERRITOS', 'Citas', 'Pedidos', 'En Preparación', and 'COSING, USUARIOS'. The 'Categorías' item is selected. The main content area has a search bar ('Q Buscar') and a table titled 'Categorías' with columns: ID, Nombre, Código, Total (0), Stock, Descuento (%), Categoría, Negocio, and Imagen. There are three rows: 'Corte' (ID 3), 'Barba' (ID 2), and 'Limpieza' (ID 1). To the right, a list of businesses ('Negocios') is shown: Barbería Adrián Carrillo (checked), Barbería Adrián Carrillo (unchecked), and Barbería Adrián Carrillo (unchecked). Navigation buttons 'Anterior' and 'Siguiente' are at the bottom.

Figura 6.16: Configuración de las categorías de productos del negocio en el Back-End

The screenshot shows the 'Productos' (Products) section of the BusinessAPP Back-End. The sidebar menu is identical to Figura 6.16. The main content area displays a table titled 'Productos' with columns: ID, Nombre, Código, Total (0), Stock, Descuento (%), Categoría, Negocio, and Imagen. Four products are listed: 'Maquinilla Profesional' (ID 21), 'Barber Shave' (ID 22), 'Kit de Limpieza Barba' (ID 21), and 'Shampoo Anticáida' (ID 20). Each product row includes an 'Imagen' thumbnail. Navigation buttons 'Anterior' and 'Siguiente' are at the bottom.

Figura 6.17: Configuración de los productos vendidos por el negocio en el Back-End

The screenshot shows a web catalog titled 'CATÁLOGO'. It features a search bar ('Busca en nuestro catálogo los productos que deseas y disfruta de la mayor calidad con un simple click!'), a 'Ver carrito' (View cart) button, and a sidebar with filters: 'FILTROS' (Ordenar por: 'Más recientes'), 'DISPONIBILIDAD' (En stock: checked, No disponible: checked), and 'CATEGORÍAS' (Corte, Barba, Limpieza). The main area displays four product cards: 'MAQUINILLA PROFESIONAL' (39,99€), 'BARBER SHAVE' (9,95€), 'KIT DE LIMPIEZA BARBA' (31,25€), and 'SHAMPOO ANTICÁIDA' (6,45€). Each card includes an 'Añadir al carrito' (Add to cart) button. Navigation buttons 'Anterior' and 'Siguiente' are at the bottom.

Figura 6.18: Renderización de la configuración en la Web

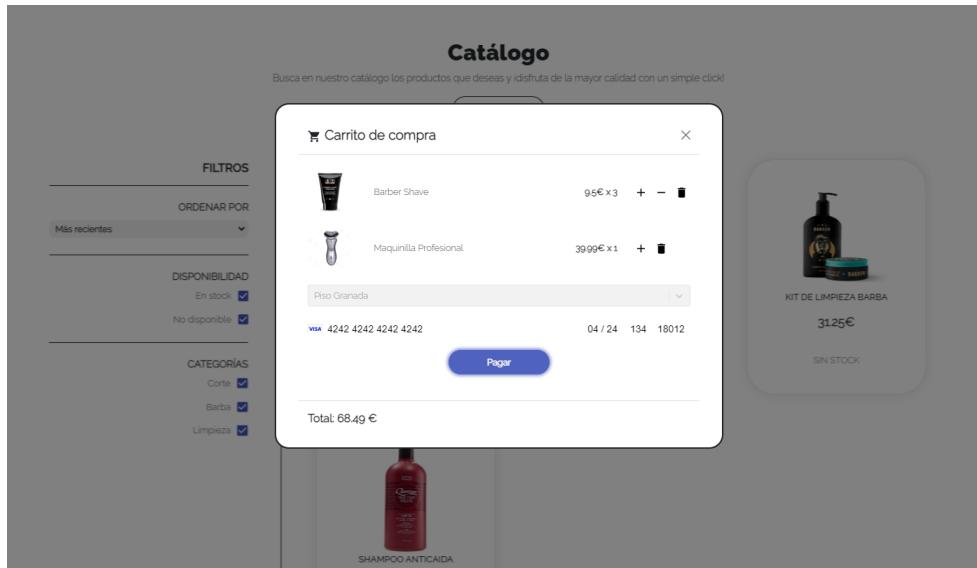


Figura 6.19: Carrito de compra en proceso de pago

### 6.5.7. Página de Citas

La página de citas usará la información de configuración de turnos del negocio para la renderización del número de citas por día. Además consultará al Web Service las citas ya reservadas en un día seleccionado para habilitar o deshabilitar en la página Web aquellas horas que estén o no disponibles. Si un usuario tiene una reserva activa no podrá realizar otra reserva hasta completarse o cancelarse la actual. Se informará con una alerta en los días seleccionados.



Figura 6.20: Página de citas: vista inicial

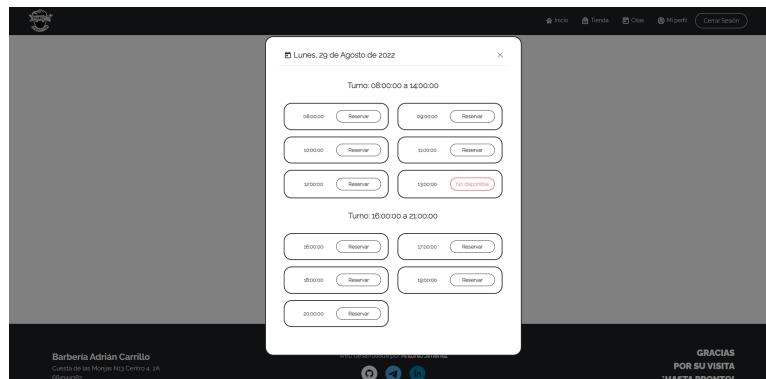


Figura 6.21: Página de citas: día seleccionado

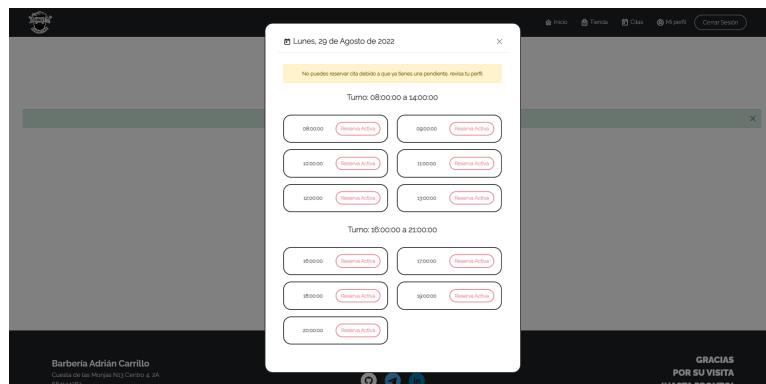


Figura 6.22: Página de citas: usuario con cita pendiente

### 6.5.8. Página de Perfil

Por último tenemos la página de Perfil. En esta el usuario tendrá la posibilidad de editar sus datos personales; crear, editar o eliminar sus direcciones postales (que deberá usar para realizar pedidos online); y ver sus historiales de pedidos y citas así como realizar las cancelaciones pertinentes.

**Mi Perfil**

Busca en nuestro calendario un hueco para ti ¡No tardes en reservar tu cita!

*User icon*    *Home icon*    *Calendar icon*    *Cart icon*

**Datos personales**

Puedes actualizar tus datos personales aquí cada vez que lo necesites.

<input type="text" value="antonio112@gmail.com"/>	<input type="text" value="Antonio"/>
<input type="text" value="Contraseña.."/>	<input type="text" value="Business"/>
<input type="text" value="Repite la contraseña.."/>	<input type="text" value="66663334"/>

**Actualizar**

Figura 6.23: Página de Perfil: datos personales

**Mi Perfil**

Busca en nuestro calendario un hueco para ti ¡No tardes en reservar tu cita!

*User icon*    *Home icon*    *Calendar icon*    *Cart icon*

**Direcciones de envío**

Puedes ver todas tus direcciones de envío desde aquí. Además podrás crear nuevas y editar las existentes.

<input type="text" value="Casa Loja"/>	<input type="text" value="Loja"/>
<input type="text" value="Cuesta de las Morjas Núñez 2A"/>	<input type="text" value="Granada"/>
<input type="text" value="Barrio.. (Opcional)"/>	<input type="text" value="España"/>
<input type="text" value="18300"/>	<b>Editar</b> <i>Lock icon</i>

**Tus direcciones**

<input type="text" value="Casa Loja"/> <i>Edit icon</i>
<input type="text" value="Piso Granada"/> <i>Edit icon</i>

Figura 6.24: Página de Perfil: direcciones postales

## Antonio Jiménez Rodríguez

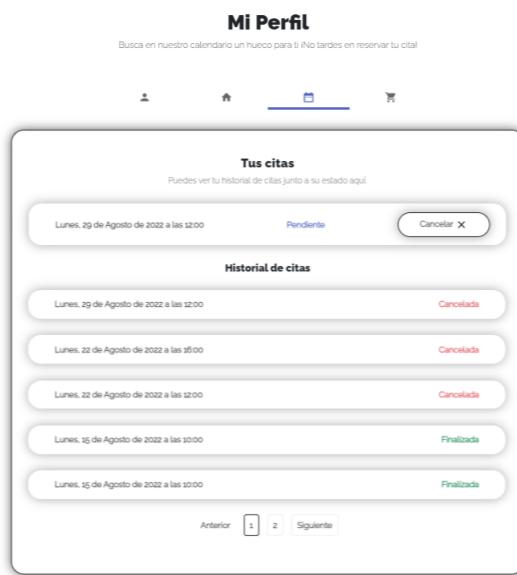


Figura 6.25: Página de Perfil: historial de citas

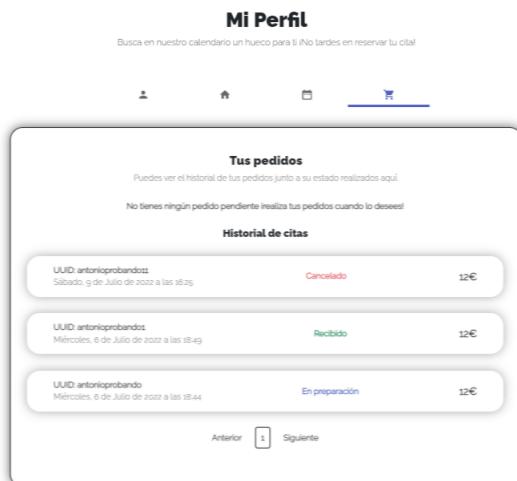


Figura 6.26: Página de Perfil: historial de pedidos

Antonio Jiménez Rodríguez

Capítulo

7

# Conclusiones y trabajos futuros

## 7.1. Temporización final

Sobre la temporización final, se ha cumplido el orden del desarrollo de las fases del proyecto marcadas, sin embargo por motivos de trabajo y finalización de los estudios, los plazos se han ido alargando más de lo deseado.

Aun así no ha distado mucho de la primera planificación. Si es verdad que en la fase final del desarrollo, mientras se iba implementando la aplicación Web, se han tenido que realizar refactorizaciones en las diferentes rutas del Web Service, ya que aparecían casuísticas que no se habían tenido en cuenta. Al igual que en el desarrollo de la administración, que se debía adaptar las entidades del sistema al CRUD proporcionado por el Bundle utilizado.

## 7.2. Objetivos alcanzados

Finalmente, se han conseguido la mayoría de los objetivos alcanzados para este proyecto, los cuales son:

- **Declaración de las entidades para la Base de Datos:** Se han implementado las diferentes entidades para el manejo de la Base de Datos del sistema. Se han añadido todas las entidades necesarias para facilitar la gestión del sistema tanto en la parte de Front como en la parte de Back.
- **Desarrollo del Web Service:** Se ha desarrollado el Web Service encargado de servir/gestionar los datos solicitados/enviados por las aplicaciones de Front-End.
- **Administración Back-End:** Para facilitar la gestión del sistema se ha implementado una administración por permisos de usuario, donde los trabajadores podrán acceder a todos los datos de su negocio y editar la

configuración del mismo como pueden ser imágenes del Home, productos de la tienda, turnos de trabajo, etc. Además para los superadministradores les será más sencillo arreglar posibles incidencias que puedan ocurrir en el sistema.

- **Aplicación Web Front-End:** Se ha desarrollado la aplicación Web con la que un cliente podrá tener acceso a todo lo ofertado por el negocio, como puede ser la venta de productos online, la reserva de citas previas y la gestión de su perfil.

### 7.3. Aprendizaje

Gracias a haber creado una herramienta completa con un objetivo que cubrir, he podido enfrentarme a problemas, resolverlos y ganar la experiencia para el futuro:

- **Comunicación entre Web Service y APP Web:** El primer obstáculo, aunque el más pequeño, fue la comunicación entre la aplicación Web y el Web Service. Los problemas de CORS y la recuperación de parámetros fueron los protagonistas.

Para resolverlo se hizo uso de un Bundle de Symfony con el que se configura el CORS del servidor permitiendo conexiones con el mismo desde aplicaciones externas.

Para el segundo, se implementó una funcionalidad común para los controladores con los que se obtiene el valor pasado por Request a la ruta llamada. Así no hay problema si los datos vienen por GET o por POST, aunque para la securización de la aplicación nosotros siempre haremos uso de POST.

- **Securización del Token de Autenticación:** Antes del desarrollo de la aplicación Web se implementó la gestión de sesión de usuario con lo que se conoce como JSON Web Token. En primera instancia se realizó una implementación básica de esta, la cual, tras realizar el inicio de sesión devolvía el token generado para el usuario. Es aquí donde apareció el problema, ¿cómo almacenamos ese token de una forma segura en el Front-End?. Los primeros pensamientos fueron almacenarlo en el local storage del navegador o en una cookie pero estos lugares son susceptibles de ataques de Cross-Site Scripting, ya que son zonas accesibles desde el JavaScript del navegador.

Investigando sobre cuál podría ser la mejor forma tomé la decisión de almacenar el token dividido en 2 cookies [17]. Una de ellas accesible por el JavaScript, con el que podremos comprobar si el usuario tiene la sesión activa o no en el sistema. La otra parte, que contiene la firma del token, será almacenada en una cookie Http-Only la cual no hay manera de acceder y es enviada en las cabeceras en las peticiones HTTP. Gracias a la

configuración del Bundle utilizado para la gestión de este JWT ha sido posible implementar este método para la gestión de sesión de usuario.

- **ReactJS:** Además de los problemas resueltos, uno de los aprendizajes en este proyecto ha sido el de los frameworks de Front-Ent, en este caso ReactJS. Gracias a él la maquetación y desarrollo de front no ha sido tan pesada como puede ser en el caso del uso de HTML, CSS y JavaScript vanilla. Además, la optimización que realiza el framework sobre el código hace posible un mejor rendimiento para una Web dinámica como la desarrollada.

## 7.4. Desarrollo futuro

Existen varias iteraciones que desearía haber realizado en este proyecto de fin de carrera, sin embargo, debido al tiempo y otros inconvenientes para su desarrollo no ha sido posible.

Entre ellas cabe mencionar:

- **Despliegue:** Otro punto clave hubiera sido el despliegue de la aplicación Back-End y varios despliegues del Front-End para tener un buen testeo de la gestión multi-negocio. La principal idea era hacer uso de contenedores para esto, pero ahondando un poco más en materia la mayoría de servidores online ofrecía servicio para **Kubernetes** por lo que no ha sido posible realizar este despliegue a tiempo.
- **Limpieza de código y estructuración:** Aunque para este proyecto se han intentado seguir las mejores prácticas posibles, siempre se pueden mejorar las cosas. Conforme se avanza en la carrera profesional se van aprendiendo nuevas técnicas y cambia la visión de las cosas, por lo que alguna refactorización de zonas y mejoras de la lógica de negocio podrían optimizar mucho más el rendimiento de los aplicativos.
- **Integración Continua:** Este punto considero que es esencial en un proyecto de este estilo, ya que podría convertirse en un proyecto colaborativo debido a sus dimensiones a futuro. Por tiempo no ha sido posible el desarrollo de los test funcionales y unitarios, junto con la integración continua en el repositorio de Github mediante las Github Actions, pero será la primera iteración a desarrollar después de la finalización de este periodo.

Como he comentado y pienso, este proyecto podría ser comercializable o usado por negocios para su gestión online. Para poder llegar a esto deberán completarse los puntos mencionados.

Antonio Jiménez Rodríguez

# Bibliografía

- [4] Daniel Ramos Cardozzo. *Desarrollo de Software: Requisitos, Estimaciones y Análisis*. 3.<sup>a</sup> ed. CreateSpace Independent Publishing Platform, 2018. ISBN: 1720896550.
- [8] Steve Krug. *No me hagas pensar*. 1.<sup>a</sup> ed. ANAYA MULTIMEDIA, 2015. ISBN: 8441537275.
- [12] Fabien Potencier. *Symfony 5: The Fast Track*. 1.<sup>a</sup> ed. Symfony SAS, 2019. ISBN: 2918390372.
- [14] Jonathan Rasmusson. *The Agile Samurai: How Agile Masters Deliver Great Software*. 1.<sup>a</sup> ed. Pragmatic Programmers, 2017. ISBN: 1934356581.

## Páginas de consulta

- [1] Axios. <https://github.com/axios/axios>.
- [2] BusinessAppWeb. <https://github.com/Antobio17/BusinessAppWeb>.
- [3] BusinessAppWS. <https://github.com/Antobio17/BusinessAppWS>.
- [5] EasyAdmin. <https://symfony.com/bundles/EasyAdminBundle/current/index.html>.
- [6] Free Software Foundation. *GNU General Public License*. <http://www.gnu.org/licenses/gpl.html>. Ver. 3.
- [7] JSX. <https://es.reactjs.org/docs/introducing-jsx.html>.
- [9] LexikJWTAuthenticationBundle. <https://github.com/lexik/LexikJWTAuthenticationBundle>.
- [10] miCita. <https://micita.app/>.
- [11] Normalización. [http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/42\\_normalizacin\\_de\\_tablas\\_relacionales.html](http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/42_normalizacin_de_tablas_relacionales.html).
- [13] Primera Forma Normal. [http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/421\\_primera\\_forma\\_normal.html](http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/421_primera_forma_normal.html).

- [15] *ReactJS.* <https://es.reactjs.org>.
- [16] *Reservio.* <https://www.reservio.com/>.
- [17] *Securización de JWT.* <https://dev.to/gkoniaris/how-to-securely-store-jwt-tokens-51cf>.
- [18] *Segunda Forma Normal.* [http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/422segunda\\_forma\\_normal.html](http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/422segunda_forma_normal.html).
- [19] *Shopify.* <https://www.shopify.es/>.
- [20] *TelegramBot.* <https://github.com/TelegramBot/Api>.
- [21] *Tercera Forma Normal.* [http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/423tercera\\_forma\\_normal.html](http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/423tercera_forma_normal.html).