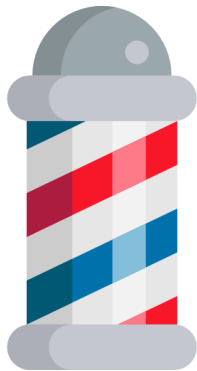


# NEW STYLE



## **Memoria del proyecto**

**CICLO FORMATIVO DE GRADO SUPERIOR  
DESARROLLO DE APLICACIONES WEB**

### **AUTORES**

Leonardo Calva

Jaime Martínez

### **TUTOR**

Alberto Garay

# ÍNDICE

<b>1.</b>	<b>INTRODUCCIÓN</b>	<b>3</b>
<b>2.</b>	<b>ALCANCE DEL PROYECTO</b>	<b>4</b>
2.1.	Análisis previo	5
<b>3.</b>	<b>ESTUDIO DE VIABILIDAD</b>	<b>8</b>
3.1.	Estado actual del sistema	8
3.2.	Resumen de requisitos del cliente	11
3.3.	Posibles soluciones	11
3.4.	Solución elegida	12
<b>4.</b>	<b>ANÁLISIS</b>	<b>13</b>
4.1.	Casos de uso	13
4.2.	Modelo de datos	22
4.3.	Requisitos funcionales	23
4.4.	Requisitos no funcionales	24
<b>5.</b>	<b>DISEÑO</b>	<b>25</b>
5.1.	Estructura de la aplicación	25
5.2.	Componentes del sistema	27
5.3.	Herramientas y tecnologías utilizadas	28
<b>6.</b>	<b>IMPLEMENTACIÓN</b>	<b>30</b>
6.1.	Implementación del modelo de datos	30
6.2.	Configuraciones realizadas en el sistema	30
6.3.	Despliegue	31
<b>7.</b>	<b>PRUEBAS</b>	<b>32</b>
7.1.	Casos de pruebas	32
<b>8.</b>	<b>EXPLOTACIÓN</b>	<b>44</b>
8.1.	Planificación	44
8.2.	Preparación para el cambio	44
<b>9.</b>	<b>DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN</b>	<b>45</b>
<b>10.</b>	<b>BIBLIOGRAFÍA</b>	<b>46</b>

# ÍNDICE DE TABLAS Y FIGURAS

Ilustración 1. Análisis previo	5
Ilustración 2. Marco Aldani 1	9
Ilustración 3. Marco Aldani 2	9
Ilustración 4. Leopoldo internacional 1	10
Ilustración 5. Leopoldo internacional 2	10
Ilustración 6. Casos de uso	13
Ilustración 7. Modelo de datos	22
Ilustración 8. Comparativa flujo de una SPA y tradicional	25
Ilustración 9. Prueba 001	32
Ilustración 10. Prueba 002	33
Ilustración 11. Prueba 003	33
Ilustración 12. Prueba 004	34
Ilustración 13. Prueba 005	34
Ilustración 14. Prueba 006	35
Ilustración 15. Prueba 007	35
Ilustración 16. Prueba 008	36
Ilustración 17. Prueba 009	36
Ilustración 18. Prueba 010	36
Ilustración 19. Prueba 011	37
Ilustración 20. Prueba 012	37
Ilustración 21. Prueba 013	38
Ilustración 22. Prueba 014	38
Ilustración 23. Prueba 015	39
Ilustración 24. Prueba 016	40
Ilustración 25. Prueba 017	40
Ilustración 26. Prueba 018	41
Ilustración 27. Prueba 019	41
Ilustración 28. Prueba 020	42
Ilustración 29. Prueba 021	42
Ilustración 30. Prueba 022	43
Ilustración 31. Prueba 023	43

# 1. INTRODUCCIÓN

Este documento recoge el trabajo realizado para el módulo de Proyecto del CFGS en DAW (Desarrollo de Aplicaciones Web).

Este módulo profesional complementa la formación establecida para el resto de los módulos profesionales que integran el título en las funciones de análisis del contexto, diseño o desarrollo del proyecto y organización de la ejecución.

## 2. ALCANCE DEL PROYECTO

El proyecto que se desarrollará a continuación es una aplicación web dirigida a cualquier negocio de peluquería en la que el usuario podrá acceder a la información de cortes y peinados además de reserva de citas permitiendo también que al mismo tiempo el administrador pueda gestionar estos mismo elementos.

El objetivo y los detalles de las principales características o utilidades de este aplicativo son las siguientes:

- Permitir al negocio mostrar al público sus trabajos, recogiendo en un único lugar personalizado y adaptado al estilo del negocio a través de un carrusel de imágenes del local, una galería con los cortes y peinados realizados con sus nombres.
- Permitir a los potenciales clientes registrarse en el sitio web para tener acceso a la reserva de cita previa, seleccionar el tipo de servicio que desean de los servicios disponibles, elegir el día y hora y permitir editar dicha reserva. El usuario recibirá un email de confirmación tanto al registrarse como al realizar reserva.
- El administrador podrá gestionar usuarios, citas, servicios, cortes y diseños y galería de bienvenida, pudiendo añadir, editar y borrar cada uno de los elementos según las necesidades y requerimientos de cada momento.
- La aplicación remitirá vía email al administrador cada una de las reservas registradas para su posterior gestión y confirmación.

El fin principal de esta aplicación es aportar a los negocios un medio de gestión más eficiente y centralizado que les sirva como herramienta para seguir mejorando en la calidad del servicio, evitando dedicar demasiado tiempo a la administración y pudiendo centrarse más en desarrollar el potencial del negocio.

Así mismo esta plataforma proporciona al negocio un escaparate virtual pudiendo llegar a potenciales clientes a los que previamente no podría alcanzar si no llegasen físicamente al propio negocio y a los que de esta manera mediante la difusión de este sitio virtual sí podrá llegar.

## 2.1 Análisis previo

Debido a la falta de tiempo para trabajar en nuestro proyecto, es probable que no podamos implementar todas las funcionalidades que nos gustaría. En la siguiente tabla se mostrarán las funcionalidades de la aplicación.

Entenderemos como visitante a cualquier persona que entre a la aplicación sin estar logado, se entenderá como usuario a la persona que esté o no logada, ya se ha registrado en algún otro momento y se entenderá como administrador o admin a la persona registrada con rol de administrador con acceso a la parte privada de la aplicación en la que se gestiona el contenido de la misma con privilegios para realizar una serie de tareas.

En la siguiente tabla se establecen unos puntos mínimos que cumplir y sus prioridades, que hemos dividido en objetos de negocio:

FUNCIONALIDAD		PRIORIDAD	MOTIVO
USUARIOS	Crear	Alta	Al poder registrarse, es posible ofrecer al usuario servicios que mejoren su experiencia como ofertas, promociones, servicio de fidelización... así como acceso a la información de todas sus citas y de los datos que almacenamos sobre él.
	Modificar	Alta	El usuario es dueño de sus datos y debe poder decidir sobre ellos
	Mostar	Media	El usuario debe poder acceder a sus datos.
	Borrar	Baja	Esta funcionalidad variará dependiendo de la decisión del propietario de la aplicación, pero en un principio los usuarios no se eliminarán salvo que el usuario lo pida o un administrador lo considere necesario.
ADMINISTRADORES	Crear	Alta	Solo otros administradores podrán crear nuevos administradores
	Modificar	Alta	Los administradores podrán modificar sus datos como quieran.
	Mostar	Media	Solo los administradores se van a poder ver entre ellos.
	Borrar	Baja	Los administradores se podrán eliminar por completo
CITAS	Crear	Alta	En una peluquería las citas son básicamente una herramienta de trabajo y de gestión y es con ellas con

			las que comienza el proceso de venta y servicio por lo que se crearán de forma constante.
	Modificar	Media	Modificar una cita tiene sentido si por cualquier casuística no se puede asistir, pero se continúa con la intención de obtener el servicio.
	Mostar	Media	Al igual que en el caso anterior, es necesario consultar la cita para poder gestionar el planning diario y no siempre es fácil de recordar.
	Borrar	Baja	No es muy probable ya que precisamente las citas se crean con la intención de obtener un servicio y una vez solicitado es raro el caso que no se lleve a cabo.
CORTES	Crear	Alta	Es la parte de la aplicación que le permite a la aplicación captar clientes mostrando la calidad de sus trabajos.
	Modificar	Baja	Aunque no se haga con demasiada frecuencia, es importante poder modificar la información
	Mostar	Alta	Si el usuario no tiene acceso a los visualizar los cortes realizados no podrá decidir acudir y la peluquería estaría perdiendo oportunidades
	Borrar	Baja	Aunque un corte no se suele realizar es interesante conservarlo con el fin de aportar variedad y opciones.
SERVICIOS	Crear	Alta	Al igual que los cortes, esta parte de la aplicación le permite al negocio captar clientes mostrando lo que ofrecen.
	Modificar	Baja	Aunque no se haga con demasiada frecuencia, es importante poder modificar la información
	Mostar	Alta	De la misma forma, si el usuario no tiene acceso a visualizar los servicios no podrá decidir acudir o no la peluquería y perderían clientes
	Borrar	Baja	Aunque un corte no se suele realizar es interesante conservarlo con el fin de aportar variedad y opciones.

GALERÍA	Crear	Baja	A no ser que el local se reforme la galería home no se modificará.
	Modificar	Baja	A no ser que el local se reforme la galería home no se modificará.
	Mostar	Alta	Debe mostrarse con el fin de que los usuarios se sientan atraídos e invitados a acudir.
	Borrar	Baja	A no ser que el local se reforme la galería home no se modificará.

Ilustración 1. Análisis previo



## 3. ESTUDIO DE VIABILIDAD

En esta fase se considera si el proyecto se puede realizar teniendo en cuenta las circunstancias internas y externas de la empresa, las diferentes soluciones posibles y los recursos de los cuales se dispone.

Para ello se hace una valoración del estado actual del sistema y de los requisitos del cliente, se presentará un estudio de soluciones alternativas y solución elegida por el cliente.

### 3.1 Estado actual del sistema

Actualmente pocas peluquerías disponen de una página web que ofrezca todas las posibilidades que ofrece este proyecto. Existen básicamente dos tipos de web diferentes en este ámbito, por un lado, las grandes cadenas ofrecen una web genérica de la marca, en la que navegando a través de menús puedes acceder a la dirección o formas de contacto de una u otra tienda y en otros casos no disponen de este listado.

Por otro lado, existen pequeñas peluquerías que por diversas razones no disponen de sitio web o que, si disponen de él, simplemente sirve de escaparate ofreciendo una serie de imágenes del negocio, y diseños, anunciando productos de otras marcas, etc.

En ninguno de los casos anteriores se integran eficientemente y a un tiempo la función de escaparate tanto de servicios, ofertas, productos, imágenes del local, junto con la posibilidad de reservar cita de forma online.

A continuación, se muestran algunos ejemplos de ambos casos donde se queda patente la falta de esta unificación de funciones en un único sitio web:

## Marco Aldani

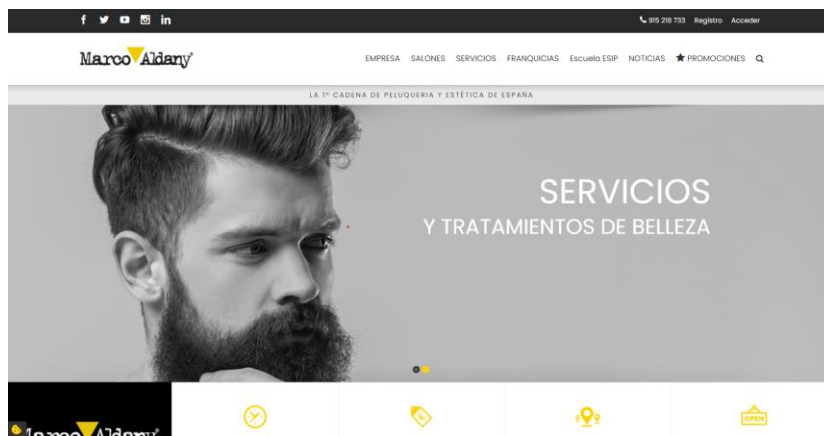


Ilustración 2. Marco Aldani 1

En esta imagen de hom3 de Marco Aldani el menú nos ofrece varias posibilidades, tanto servicios, como salones, como franquicias podrían darnos acceso a cada una de las peluquerías pero no es así, nos dan paso a galerías, a listas de servicios y demás donde se explican y muestran diversas imágenes sin dar acceso a tiendas o selección de los mismos.

<https://www.marcoaldany.com/>

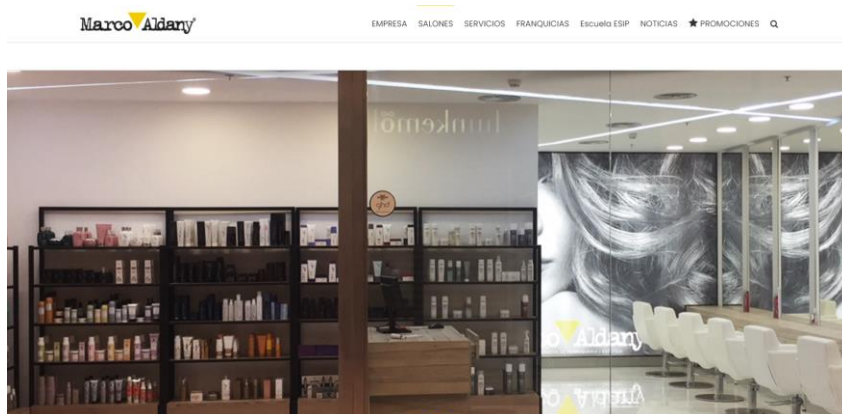


Ilustración 3. Marco Aldani 2



Existen muchos más ejemplos en los que este factor se repite y se hace palpable, se echa en falta continuamente y hace necesario recurrir a otras aplicaciones como Google Maps para localizar el lugar exacto e incluso aporta la misma información que el sitio propio de la empresa y aún más accesible situando los datos de contacto mucho más visualmente y con menos clics que en la web del negocio que debería ser la más interesada en facilitar y mostrar estos datos.

Otra opción es utilizar buscadores o metabuscadores en los que te permite utilizar un filtro para localizar un centro en el que realicen el servicio deseado y que muestra como resultado un listado de peluquerías con dirección y contacto que igual que en casos anteriores se echa en falta la reserva online.

### 3.2 Resumen de requisitos del cliente

El cliente busca una manera de acercar a los usuarios sus servicios al tiempo que pueden conocer el resultado final de estos servicios, eliminando la barrera física y los desplazamientos ya que no siempre es posible por distancia o por tiempo el poder llegar físicamente y en hora al propio comercio.

A su vez pretende agilizar el servicio de citas previas evitando que el cliente deba esperar a ser atendido y obligando al trabajador a dejar de trabajar para atender y apuntar una nueva cita. De esta manera. La gestión de estas citas se hará concentrada al final o comienzo del día evitando más interrupciones.

### 3.3 Posibles soluciones

Las soluciones actuales pasan por el uso de webs intermediarias para los trabajos realizados, así como redes sociales utilizándose Facebook como escaparate principal, pero esto genera una dependencia de una plataforma de terceros y en cierto modo limita la personalización y desenfoca el objetivo al estar en el mismo sitio que relaciones sociales, etc.

Por otro lado, Facebook o Instagram no son la mejor manera de guardar las fotos y de administrarlas, ya que no permiten una edición avanzada ni de fotos ni de mensajes.

Una segunda opción, también muy utilizada, es el uso de un blog. En el caso de elegir esta opción, corremos el riesgo de que las entradas antiguas pierdan visibilidad y solo generemos tráfico en las tres o cuatro primeras entradas.

Por último, la gestión de reservas se suele realizar vía telefónica adjuntando en redes el número de contacto, algo que depende de la recepción o no de esa llamada pudiendo limitar el servicio.

### 3.4 Solución elegida

La solución elegida, una vez evaluadas las opciones citadas anteriormente, es crear una página web con un sistema de administración en el que se registrarán los clientes mediante un sencillo

formulario, que también permitirá gestionar al administrados la base de datos de los mismo y tener acceso a sus datos, y a los propios clientes les permitirá gestionar sus citas una vez registrados.

Así mismo, permitirá un acceso directo de los usuarios con la peluquería, mediante solicitudes de cita y fotos de peinados y cortes actualizadas. En cuanto a los productos, se mostrará una selección orientada a poder solventar las necesidades derivadas de los propios servicios de peluquería pudiendo adquirirlos una vez se acuda a la cita.

## 4. ANÁLISIS

Para poder usar esta aplicación no es necesario que la peluquería disponga de un ordenador específico, es posible utilizarla y gestionarla desde cualquier teléfono móvil o, en el caso de su gestión, puede realizarse desde cualquier lugar que disponga de ordenador, Tablet o móvil con navegador web actualizado.

Por otro lado, sería necesaria la adquisición de un dominio bajo el que realizar el despliegue además de un servidor web donde realizar este despliegue. Tanto el mantenimiento, como el despliegue como la adquisición de estos servicios puede realizarse con terceros o a través de nosotros.

### 4.1 Casos de uso



Ilustración 6. Casos de uso

**Caso de uso (UC001).**

**Escenario de éxito:** El visitante se registra en la aplicación.

**Actores:** Visitante no registrado.

**Precondición:** El visitante no debe estar registrado ni logado en la aplicación.

1. El visitante indica al sistema que quiere registrarse.
2. El sistema muestra el formulario de registro.
3. El visitante rellena el formulario de registro.
4. El visitante confirma los datos.
5. El sistema registra los datos introducidos.

**Extensiones:**

- 3a. Alguno de los campos introducidos no cumple los requisitos de seguridad.
  1. El sistema qué campo no es correcto.
  2. El visitante introduce un nuevo valor correcto en el campo indicado.
  3. Se repiten los puntos 1 y 2 de esta extensión hasta que en el valor en el campo indicado sea correcto.
  4. Continúa en el punto 4.
- 3b. La dirección de email introducida ya está registrada.
  1. El sistema indica que el email introducido ya está en uso.
  2. El visitante introduce una nueva dirección de email.
  3. Se repiten los puntos 1 y 2 de esta extensión hasta que la dirección de email sea correcta.
  4. Continúa en el punto 4.

**Caso de uso (UC002).**

**Escenario de éxito:** El visitante lista los servicios.

**Actores:** visitante o usuario.

**Precondición:** El visitante puede estar logado o no en la aplicación.

1. El usuario indica al sistema que quiere listar los servicios.
2. El sistema muestra la lista de los servicios.

**Caso de uso (UC003).**

**Escenario de éxito:** La visitante lista datos de contacto.

**Actores:** visitante o usuario.

**Precondición:** El visitante puede estar logado o no en la aplicación.

1. El usuario indica al sistema que quiere listar los datos de contacto.
2. El sistema muestra la lista de los datos de contacto.

**Caso de uso (UC004).**

**Escenario de éxito:** El visitante lista los tipos de corte.

**Actores:** visitante o usuario.

**Precondición:** El visitante puede estar logado o no en la aplicación.

1. El usuario indica al sistema que quiere listar los tipos de corte.
2. El sistema muestra la lista de los tipos de corte.

**Caso de uso (UC005).**

**Escenario de éxito:** El usuario modifica la información de su perfil.

**Actores:** Usuario.

**Precondición:** El usuario tiene una cuenta en la aplicación y está logado.

1. El usuario indica al sistema que quiere visualizar su perfil.
2. El sistema muestra la información que puede modificar.
3. El usuario modifica los datos.
4. El usuario acepta.

**Extensiones:**

3a. El usuario introduce un email que ya está en uso.

1. El sistema indica que los datos introducidos no son válidos.
2. El usuario introduce de nuevo los datos.
3. Se repiten los puntos 1 y 2 de hasta que los datos sean válidos.
4. Continúa en el punto 4.

3b. El usuario introduce un Nombre de Usuario que ya está en uso.

1. El sistema indica que los datos introducidos no son válidos.
2. El usuario introduce de nuevo los datos.
3. Se repiten los puntos 1 y 2 de hasta que los datos sean válidos.
4. Continúa en el punto 4.

**Caso de uso (UC006).**

**Escenario de éxito:** El usuario genera una nueva cita.

**Actores:** usuario.

**Precondición:** El usuario debe estar logado.

1. El usuario indica al sistema que quiere generar una cita.
2. El sistema muestra el formulario de registro de citas.
3. El usuario rellena el formulario.
4. El sistema registra la cita.



**Caso de uso (UC007).**

**Escenario de éxito:** El usuario modifica una cita.

**Actores:** usuario.

**Precondición:** El usuario debe estar logado.

1. El usuario indica al sistema que quiere modificar una cita.
2. El sistema muestra el registro de citas.
3. El usuario indica al sistema que cita quiere modificar.
4. El sistema muestra el formulario de registro de citas.
5. El usuario modifica el formulario.
6. El sistema registra la cita.

**Caso de uso (UC008).**

**Escenario de éxito:** El usuario borra una cita.

**Actores:** usuario.

**Precondición:** El usuario debe estar logado.

1. El usuario indica al sistema que ver las citas.
2. El sistema muestra el registro de citas.
3. El usuario indica al sistema que cita quiere borrar.
4. El sistema borra la cita.

**Caso de uso (UC009).**

**Escenario de éxito:** El usuario se logea.

**Actores:** Visitante previamente registrado.

**Precondición:** El visitante tiene una cuenta en la aplicación

1. El visitante indica al sistema que quiere logarse.
2. El sistema muestra el formulario de logeo.
3. El visitante rellena el formulario.
4. El sistema comprueba los datos introducidos.
5. El sistema logea al usuario.

**Extensiones:**

- 3a. La contraseña no es correcta.
  1. El sistema indica que el correo o la contraseña introducida no es válida.
  2. El visitante introduce de nuevo la contraseña.
  3. Se repiten los puntos 1 y 2 de hasta que la contraseña sea correcta.
- 3b. La dirección de correo no es correcta.
  1. El sistema indica que el correo o la contraseña introducida no es válida.
  2. El visitante introduce de nuevo el correo electrónico.
  3. Se repiten los puntos 1 y 2 hasta que el correo sea correcto.

**Caso de uso (UC010).**

**Escenario de éxito:** El administrador borra imágenes home.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que ver las imágenes de home.
2. El sistema muestra las imágenes del home.
3. El usuario indica al sistema que cita imagen borrar.
4. El sistema borra la imagen.

**Caso de uso (UC011).**

**Escenario de éxito:** El administrador añade imágenes home.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere añadir imágenes al home.
2. El sistema muestra el formulario.
3. El usuario indica al sistema qué imagen añadir.
4. El sistema añade la imagen.

**Caso de uso (UC012).**

**Escenario de éxito:** El administrador lista imágenes home.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar imágenes home.
2. El sistema muestra la lista.

**Caso de uso (UC013).**

**Escenario de éxito:** El administrador edita imágenes de home.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar imágenes de home.
2. El sistema lista las imágenes de home.
3. El usuario indica al sistema qué imagen quiere editar.
4. El sistema muestra el formulario.
5. El usuario editar la imagen.
6. El sistema guarda los cambios.

**Extensiones:**

5a. El nombre ya está en uso.

1. El sistema indica que nombre está en uso.
2. El visitante introduce de nuevo el nombre.
3. Se repiten los puntos 1 y 2 de hasta que el nombre sea correcto.

**Caso de uso (UC014).**

**Escenario de éxito:** El administrador borra imágenes de cortes.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que ver las imágenes de cortes.
2. El sistema muestra las imágenes de cortes.
3. El usuario indica al sistema qué imagen borrar.
4. El sistema borra la imagen.

**Caso de uso (UC015).**

**Escenario de éxito:** El administrador añade imágenes de cortes.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere añadir imágenes de cortes.
2. El sistema muestra el formulario.
3. El usuario indica al sistema qué imagen añadir.
4. El sistema añade la imagen.

**Caso de uso (UC016).**

**Escenario de éxito:** El administrador lista imágenes de cortes.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar imágenes de cortes.
2. El sistema muestra la galería de cortes.

**Caso de uso (UC017).**

**Escenario de éxito:** El administrador edita imágenes de cortes.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar imágenes de cortes.
2. El sistema lista las imágenes de cortes.
3. El usuario indica al sistema qué imagen quiere editar.
4. El sistema muestra el formulario.
5. El usuario edita la imagen.
6. El sistema guarda los cambios.

**Extensiones:**

- 5a. El nombre ya está en uso.
  1. El sistema indica que nombre está en uso.
  2. El visitante introduce de nuevo el nombre.
  3. Se repiten los puntos 1 y 2 de hasta que el nombre sea correcto.

### ***Caso de uso (UC018).***

**Escenario de éxito:** El administrador borra servicio.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que ver la lista de servicios.
2. El sistema muestra la lista de servicios.
3. El administrador indica al sistema que servicio borrar.
4. El sistema borra el servicio.

### ***Caso de uso (UC019).***

**Escenario de éxito:** El administrador añade servicio.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere añadir un servicio.
2. El sistema muestra el formulario.
3. El administrador rellena el formulario.
4. El sistema añade el servicio.

**Extensiones:**

- 3a. El nombre ya está en uso.
  1. El sistema indica que nombre está en uso.
  2. El administrador introduce de nuevo el nombre.
  3. Se repiten los puntos 1 y 2 de hasta que el nombre sea correcto.

**Caso de uso (UC020).**

**Escenario de éxito:** El administrador lista los servicios.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar los servicios.
2. El sistema muestra la lista de servicios.

**Caso de uso (UC021).**

**Escenario de éxito:** El administrador edita un servicio.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar los servicios.
2. El sistema lista los servicios.
3. El usuario indica al sistema qué servicios quiere editar.
4. El sistema muestra el formulario.
5. El usuario edita el servicio.
6. El sistema guarda los cambios.

**Extensiones:**

- 5a. El nombre ya está en uso.
  1. El sistema indica que nombre está en uso.
  2. El administrador introduce de nuevo el nombre.
  3. Se repiten los puntos 1 y 2 de hasta que el nombre sea correcto.

**Caso de uso (UC022).**

**Escenario de éxito:** El administrador borra cita.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere ver la lista de citas.
2. El sistema muestra la lista de citas.
3. El administrador indica al sistema qué cita borrar.
4. El sistema borra la cita.

**Caso de uso (UC023).**

**Escenario de éxito:** El administrador lista las citas.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar las citas.
2. El sistema muestra la lista de citas.

**Caso de uso (UC024).**

**Escenario de éxito:** El administrador borra usuario.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere ver la lista de usuarios.
2. El sistema muestra la lista de usuarios.
3. El administrador indica al sistema qué usuario borrar.
4. El sistema borra el el usuario.

**Extensiones:**

- 4a. El usuario tiene una cita.
  1. El sistema indica que el usuario tiene una cita.
  2. El administrador borra la cita.
  3. Se repiten los puntos 1 y 2 hasta que no queden citas.

**Caso de uso (UC025).**

**Escenario de éxito:** El administrador lista los usuarios.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar los usuarios.
2. El sistema muestra la lista de usuarios.

**Caso de uso (UC026).**

**Escenario de éxito:** El administrador edita un usuario.

**Actores:** administrador.

**Precondición:** El administrador debe estar logado.

1. El administrador indica al sistema que quiere listar los usuarios.
2. El sistema lista los usuarios.
3. El administrador indica al sistema qué usuario quiere editar.
4. El sistema muestra el formulario.
5. El administrador editar el usuario.
6. El sistema guarda los cambios.

#### Extensiones:

5a. El nombre ya está en uso.

1. El sistema indica que nombre está en uso.
2. El administrador introduce de nuevo el nombre.
3. Se repiten los puntos 1 y 2 de hasta que el nombre sea correcto.

## 4.2 Modelo de datos

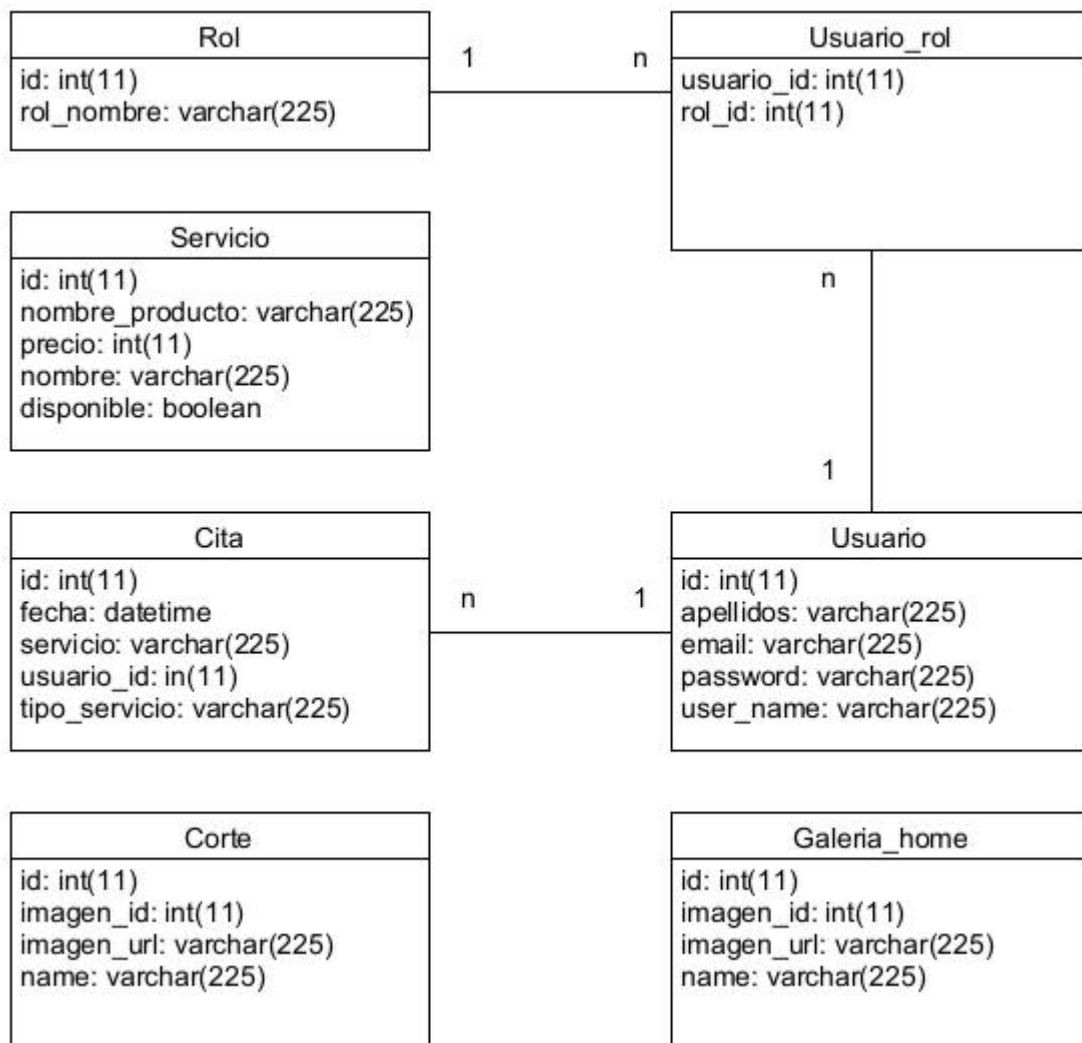


Ilustración 7. Modelo de datos

### 4.3 Requisitos funcionales

Son aquellos que determinan qué tareas tiene que hacer el sistema.

#### **Independizar usuarios registrados de la aplicación de administradores.**

Por seguridad y rapidez los usuarios de la aplicación son independientes de los usuarios administradores, cada parte tiene su acceso independiente y en ningún momento se cruzan.

#### **Restringir la creación de nuevos administradores.**

Sólo el súper administrador puede crear nuevas cuentas de administrador, no existe formulario de registro si no está autenticado con esta cuenta.

#### **Restringir la creación de nuevos servicios.**

El administrador puede registrar nuevos servicios en la aplicación. Sólo los usuarios administradores podrán crear un servicio o borrar los que ya existan.

#### **Restringir la creación de nuevos cortes.**

Sólo los usuarios administradores podrán crear un corte o borrar los que ya existan.

#### **Restringir la subida de nuevas imágenes.**

Sólo los usuarios administradores podrán añadir o borrar las imágenes existentes.

#### **Registrar citas de forma eficaz.**

El proceso de una cita se simplificará para que no genere confusión y se pueda hacer de la manera más sencilla posible.

#### **Control de eliminación de usuarios registrados.**

Para evitar errores y problemas, un administrador no puede borrar un usuario que tiene citas registradas. Se informa de esto si intenta hacerlo.

#### **Gestionar galería de cortes.**

La galería muestra los cortes y diseños realizados. El administrador selecciona las imágenes que quiere que aparezcan.

#### **Gestionar carrusel de home.**

Carrusel de home muestra las fotos del local y/o diseños realizados. El administrador selecciona las imágenes que quiere que aparezcan.

#### **Mostrar los servicios disponibles.**

Al entrar en citas el usuario puede ver un formulario con los servicios disponibles.



#### 4.4 Requisitos no funcionales

Son las propiedades o cualidades que el sistema debe cumplir. En este caso se ha priorizado el diseño y la usabilidad de cara al usuario, y facilitar la gestión de cara al administrador.

##### **Diseño atractivo y fácil de usar.**

Normalmente este tipo de aplicaciones tiene un aspecto muy agradable y llamativo, pero poco intuitivo porque se centran en servir de escaparate y las opciones son confusas.

En este caso el usuario se encontrará con una página principal que en un clic le permitirá acceder a lo que está buscando sin necesidad de navegar en menús ni buscar por la página.

En este caso el diseño es responsive, para que se pueda adaptar tanto a monitores como a portátiles o móviles, ya que la aplicación podrá usarse en el propio negocio o en un dispositivo móvil para pedir cita o consultar la ubicación o diseños.

##### **Panel de administrador sencillo y con opciones concisas.**

El administrador tendrá un panel sencillo, con opciones concisas que le permitirá realizar todas las gestiones de forma fácil.

## 5. DISEÑO

### 5.1. ESTRUCTURA DE LA APLICACIÓN

La aplicación que se ha desarrollado consiste en una Single Page Application, o página de aplicación única (SPA). La principal característica de este tipo de aplicaciones es que la información con la que interactúa el usuario se va cargando de manera dinámica, recargando y sobrescribiendo el contenido en vez de cargar páginas nuevas cada vez. Esto supone una mejora en la experiencia de usuario, manteniéndose el estilo a lo largo de toda la web, como si de una aplicación de escritorio se tratara.

Un SPA puede definirse como una aplicación web que se ejecuta en una sola página, dando al usuario una sensación de estar usando una aplicación de escritorio. El usuario navega a través de la página no de la forma tradicional de enlaces entre componentes de la página, sino con un sistema de uso del HTML, AJAX o JavaScript e incluso combinando todas ellas haciendo que se actualice únicamente lo que ve el usuario, no la página entera.

Además, una SPA puede funcionar si se pierde la conexión con internet y puede contar con apartados que funcionen en tiempo real.

Comparativa del diagrama de flujo de una SPA y una página tradicional:

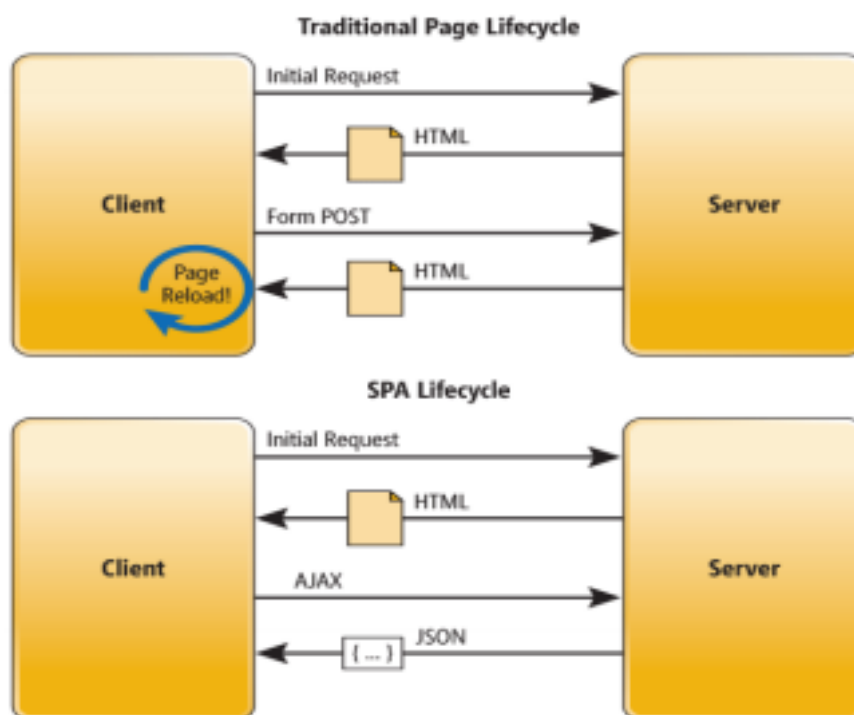


Ilustración 8. Comparativa flujo de una SPA y tradicional

El modelo de Single Page Application surgió de la necesidad de mejorar la experiencia de usuario, y aunque lleve usándose desde mediados de los 2000, no ha sido hasta hace años cuando ha empezado a convertirse en una constante en la web. Algunas de las empresas más grandes de internet usan SPA para sus webs, como Google, Facebook o Netflix.

JavaScript, HTML 5 y CSS3 son los ingredientes para cualquier SPA. En la actualidad se han popularizado librerías y frameworks de desarrollo web SPA, tales como React, Angular o Vue. En nuestro caso, nos hemos inclinado por el uso de Angular.

Algunas de las ventajas de usar una SPA son:

- **Reducen los tiempos de respuesta**, ya que se delegan procesos del servidor al navegador web. El objetivo es que solo queden en remotos procesos tales como autenticación, validación e inserción de datos.
- **Reducen el consumo de ancho de banda** debido a que las conexiones de un SPA con el servidor deben ser en pequeños tramos muy puntuales, y los datos, en general, serán transferidos en objetos manejables tipo JSON.
- **Reduce las fases de trabajo en la arquitectura** al trabajar sobre una misma página, reduciendo con ello la cantidad de código que hay que escribir.
- **Simplifica el análisis de la web** dado que sólo analizamos una página.
- **Facilita el diseño general de la página** dado que es único de la página, haciendo que podamos centrarnos en los detalles de la página, o poder probar varios tipos de diseño para la página.

Algunos de sus inconvenientes son:

- **La seguridad de los sitios SPA es menor**, ya que al centrar el diseño en la parte frontend, dado que JavaScript no es un lenguaje compilado, y está abierto a código malicioso proveniente de usuarios malintencionados, aunque existen medios para aumentar la seguridad. utilizando indicadores de Javascript para reforzar la seguridad.
- **Son páginas difíciles de optimizar para el SEO**, ya que, por lo general, tienen menos contenido y palabras clave que una web tradicional y debido a esto la optimización SEO puede ser menos efectiva, aunque actualmente hay rastreadores de SPA que facilitan el trabajo.
- **Al tratarse de una única página es fundamental centrarse en su diseño.**
- **Pierdes oportunidades de mejorar el tráfico** a través del long-text ya que tienes menos información.
- **El contenido suele estar supeditado al diseño** general de la página lo que puede provocar que no se pueda subir cierto contenido a la página.

## 5.2 Componentes del sistema / arquitectura de red

### Servidor web

Los servidores web son los que hacen posible el Web hosting, es decir, la posibilidad de alquilar un espacio en un servidor para almacenar los datos de nuestro sitio web. La principal función de un servidor Web es almacenar los archivos de un sitio y emitirlos por Internet para poder ser visitado por los usuarios. Básicamente, un servidor Web es una gran computadora que guarda y transmite datos. Cuando un usuario entra en una página de Internet, su navegador se comunica con el servidor enviando y recibiendo datos que determinan qué es lo que ve en la pantalla. Por eso, decimos que los servidores Web están para almacenar y transmitir datos de un sitio según lo solicita el navegador de un visitante.

El uso de esta plataforma nos ha sido necesario para desplegar la aplicación, aunque por falta de tiempo no hemos podido realizar este despliegue, por lo que la aplicación se ejecutará en local teniendo la base de datos en la nube.

### Sistema gestor de bases de datos (BBDD)

Un sistema gestor de base de datos (a partir de ahora SGDB) es, a groso modo, un sistema que permite la creación, gestión y administración de bases de datos, así como la elección y manejo de las estructuras necesarias para el almacenamiento y búsqueda de la información del modo más eficiente posible.

En nuestro proyecto hemos optado por el uso de Google Cloud que es una plataforma de servicios de nube que proporciona una variedad de servicios de infraestructura tales como almacenamiento, redes, bases de datos, servicios de aplicaciones, potencia de cómputo, mensajería, inteligencia artificial, servicios móviles, seguridad, identidad y conformidad, entre otros.

Como característica principal y que lo hace muy atractivo es que es gratuito. Además es muy sencillo de utilizar, únicamente sabiendo unos pocos comandos podemos utilizarlo, es rápido, posee buena seguridad y es compatible con varios SO.

### Navegadores web

Un navegador web es un software, aplicación o programa que permite acceder a la web interpretando el conjunto de archivos proporcionados por un sitio web para que el usuario sea capaz de visualizarlos. La funcionalidad básica de un navegador web es permitir la visualización de documentos de texto, generalmente escritos en lenguaje HTML o JSP. Además, permite visitar páginas web y hacer actividades en ella, es decir, enlazar un sitio con otro, imprimir, enviar y recibir correo, entre otras funcionalidades más.

Actualmente existen numerosos navegadores web, aunque las preferencias de los usuarios están

muy definidas, pero cabe destacar los siguientes:

- **Google Chrome:** Es actualmente el navegador más utilizado debido a su interfaz amigable, su rapidez a la hora de cargar archivos del servidor, y sobre todo por su adaptabilidad (es de Google, claro) a todos los servicios Google que prácticamente acaparan hoy en día cualquier servicio online que usemos, ya sea Gmail, YouTube, Drive, etc. Es compatible con muchos dispositivos, incluidos smartphones y tablets y con todos los sistemas operativos. Asimismo, es el origen de Node y posee herramientas de desarrollo impecables. Además, cuenta con innumerables extensiones que permiten personalizar el navegador y hacerlo aún más llamativo desde el punto de vista de la usabilidad. En definitiva, es prácticamente la mejor opción hoy en día. El único pero que puede atribuírsele es la gran cantidad de memoria RAM que consume, lo que puede hacer que, en algunos dispositivos, pasado el tiempo, fluya más lento.
- **Mozilla Firefox:** Cada año mejoran un poco más. Actualmente trabajan en la optimización de su código haciendo que sea cada vez más veloz y fiable. Posee al igual que Chrome un conjunto muy grande de extensiones de todo tipo y es muy compatible con varios dispositivos y sistemas operativos. Posee una interfaz muy personalizable y un gestor independiente de descargas. Usan la filosofía de software libre lo que lo hace muy atractivo para desarrolladores. Los contras que tiene pueden ser su innegable comparación con Chrome (en la que suele salir perdiendo) o su interfaz que en algunos casos se hace demasiado ostentosa. Aun así, sigue siendo uno de los preferidos por los usuarios.
- **Safari:** Es el navegador por excelencia en sistemas Mac OS e iOS puesto que su integración es sublime. Tiene una interfaz limpia, clara y sencilla lo que lo hace muy atractivo. Utiliza de forma muy inteligente el touchpad y para usuarios de Mac OS es, sin duda, el navegador más rápido y eficaz en este SO. Los contras que tiene es que es código propietario (como todo el software Apple), suele estar desactualizado lo que lo hace algo más inseguro, extensiones básicas, pero sobre todo y lo más perjudicial para él es su única compatibilidad con sistemas Apple (y probablemente siempre sea así) lo que hace que sea una opción sólo válida para dispositivos de esta marca. Aun así, ha de ser mencionado.

### 5.3 Herramientas y tecnologías utilizadas

Para la parte frontend hemos utilizado las siguientes tecnologías:

#### Angular

Hemos elegido este framework de aplicaciones web ya que permite crear páginas SPA. Además, su estructura permite tener dividida la página a nivel interno en pequeños componentes, haciéndola más mantenible y ampliable. Otra de las razones por las que se ha usado Angular es porque permite el uso de librerías.

#### Material Angular

Se trata de una librería para Angular que ayuda a construir sitios web atractivos, consistentes y funcionales. Está inspirada en Google Material Design.

### **Bootstrap**

Es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Es la librería más utilizada a lo largo de la web, lo que nos ha hecho decidimos por ella en vez de por otras, tales como BULMA, FOUNDATION o SEMANTIC UI.

En cuanto a la parte backend, se han utilizado:

### **JWT**

Para nuestro proyecto JWT juega un papel importante ya que es la herramienta que utilizamos para manejar la autenticación en el caso de los usuarios en la aplicación o de los administradores en el panel de administrador. JWT genera un token que guarda las credenciales necesarias para validar la autenticación del usuario, este token se registra en una cookie que se mantiene durante un tiempo determinado, o bien se borra en el caso de que el usuario haga logout.

### **Cloudinary**

Cloudinary es un gestor de imágenes para aplicaciones móviles y sitios web en la que se almacenan las imágenes en la nube, permite su manipulación y ayuda a la optimización de estas para la web y entrega. Permite almacenar un número ilimitado de imágenes de forma privada y segura. Las imágenes se sirven a través del CDN mundial de Akamai a fin de lograr una entrega rápida y optimizada para cualquier dispositivo. También permite acceder a analíticas e informes avanzados para optimizar y comprender el rendimiento de las imágenes Por estas características nos parece una herramienta fundamental.

## 6. IMPLEMENTACIÓN

Una vez realizado el diseño de la aplicación este se convierte en la base que permite construir el sistema siguiendo los puntos fijados, en esta segunda fase es donde se lleva a la práctica, donde se implementa.

### 6.1. Implementación del modelo de datos

Como se ha usado el Spring, se han ido definiendo y generando las tablas en la base de datos de forma programática, permitiendo así poder migrar a otros sistemas sin tener que copiar la estructura o la base de datos anterior y migrar de SGBD cambiando la configuración en el API.

### 6.2. Configuraciones realizadas en el sistema

Para poder desarrollar la aplicación y poder realizar pruebas en local, se ha tenido que crear un entorno idóneo y que nos permitiera poder realizar la implementación:

#### Parte general

- Instalación de GIT
- Instalación Visual Studio Code, programa que hemos elegido para hacer el desarrollo debido a su gran cantidad de extensiones que hace más fáciles o cómodas muchas tareas repetitivas.
- Instalación de Spring Boot

#### Front End

- Instalación de NodeJS y npm
- Instalación de Angular CLI

#### Back End: Instalación XAMPP / MAMP

- Enlace simbólico al API del repositorio dentro de la carpeta htdocs para poder montar el servidor Virtual
- Creación de la base de datos con MySQL.

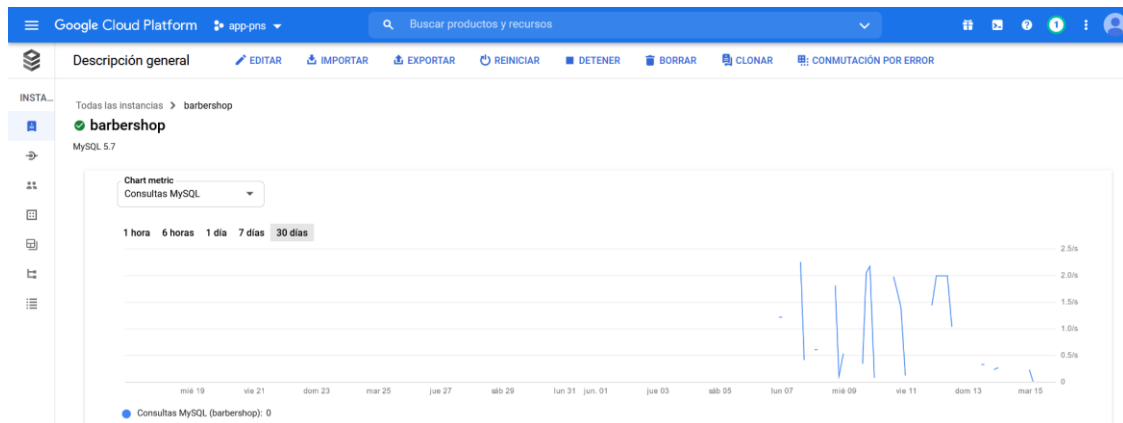
Una vez instalados estos programas, ya tendríamos un entorno listo para poder ejecutar la aplicación localmente, arrancando los servidores de Apache y MySQL en XAMPP y ejecutando el comando `ng serve` por parte de Angular.

## 6.3 Despliegue

Debido a la falta de tiempo e inexperiencia no hemos podido realizar el despliegue de la aplicación, aunque sí hemos realizado pruebas Amazon Web Services, Heroku, y Google Cloud.

Lo que sí hemos conseguido es crear una BBDD MySQL, adjuntamos capturas:

Panel de Google Cloud -> SQL:



En Spring añadimos la IP de la instancia que nos asignan para poder acceder a la BBDD, con usuario y contraseña creada al crear la BBDD.

### Conectarse a esta instancia

Dirección IP pública

34.89.90.64

Nombre de la conexión

app-pns-304215:europa-west2:barbershop

```

1 # URL MySQL
2 spring.datasource.url = jdbc:mysql://34.89.90.64:3306/barbershop
3 #spring.datasource.url = jdbc:mysql://localhost:3306/barbershop?useSSL=false&serverTimezone=UTC&useLegacyDateTimeCode=false
4 #spring.cloud.gcp.sql.database-name=barbershop
5 #spring.cloud.gcp.sql.instance-connection-name=app-pns-304215:europa-west2:barbershop
6
7 # Username and password
8 spring.datasource.username = root
9 spring.datasource.password = 9kBqsPBkh17PBaNF
10 #spring.datasource.password =
11
12 # mostrar el SQL por consola
13 spring.jpa.show-sql = true
14
15 # con update cada vez que se crea una entidad JPA crea una nueva tabla
16 # si se modifica la entidad también se modifica su respectiva tabla
17 spring.jpa.hibernate.ddl-auto = update
18
19 # permite a Hibernate generar SQL optimizado
20 spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5Dialect
21 spring.datasource.driverClassName = com.mysql.cj.jdbc.Driver
22 spring.jpa.database-platform = org.hibernate.dialect.MariaDBDialect
  
```



## 7. PRUEBAS

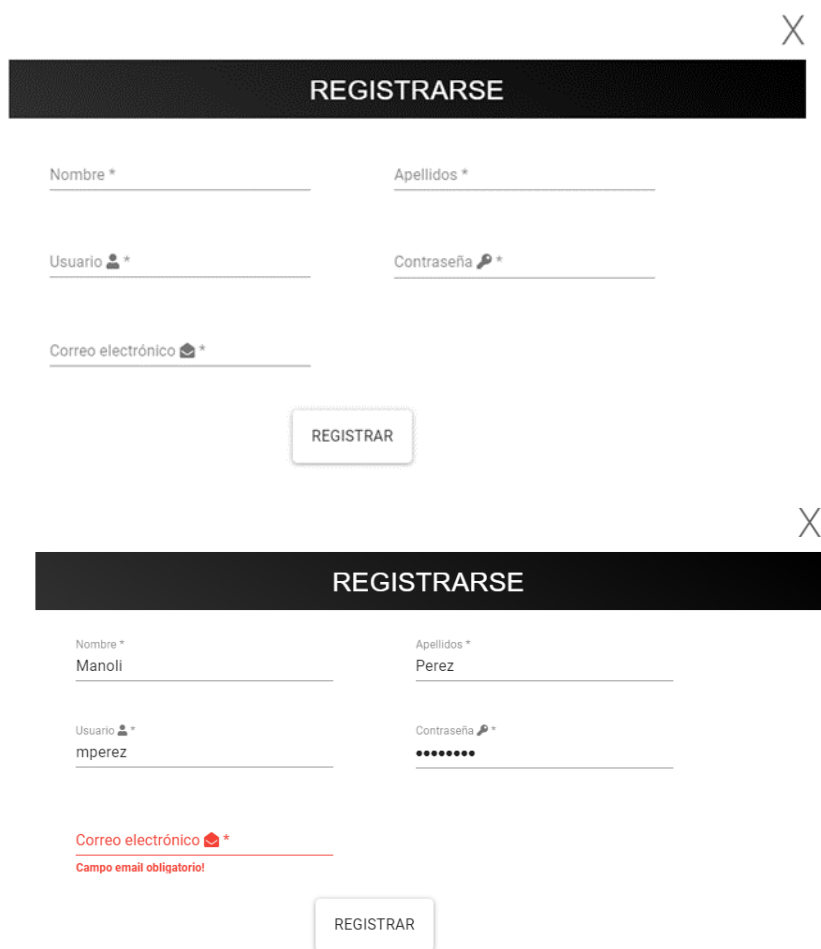
Son muchas las pruebas que pueden realizarse en un proyecto para eliminar los posibles errores y garantizar su correcto funcionamiento. Los casos de prueba establecen las condiciones/variables que permitirán determinar si los requisitos establecidos se cumplen.

A continuación, se detallan algunos de los casos de prueba que se ejecutarán para comprobar la correcta construcción de este proyecto.

### 7.1 Casos de pruebas

#### Prueba 001

Descripción: Si alguno de los campos está vacío en el formulario de registro o es o no cumple con los requisitos necesarios se pondrá en rojo al darle al botón de registro.



The image displays two versions of a registration form titled "REGISTRARSE".

**Top Screenshot:** The form has five input fields: "Nombre \*" (empty), "Apellidos \*" (empty), "Usuario" with a person icon (empty), "Contraseña" with a key icon (empty), and "Correo electrónico" with an envelope icon (empty). A "REGISTRAR" button is at the bottom.

**Bottom Screenshot:** The form is partially filled. "Nombre \*" contains "Manoli", "Apellidos \*" contains "Perez", "Usuario" contains "mperez", and "Contraseña" contains masked characters. The "Correo electrónico" field is empty and highlighted with a red border. Below it, a red error message reads "Campo email obligatorio!". The "REGISTRAR" button remains at the bottom.

Ilustración 9. Prueba 001

### Prueba 002

Descripción: Si todos los campos son correctos saldrá un mensaje al pulsar el botón de registro.

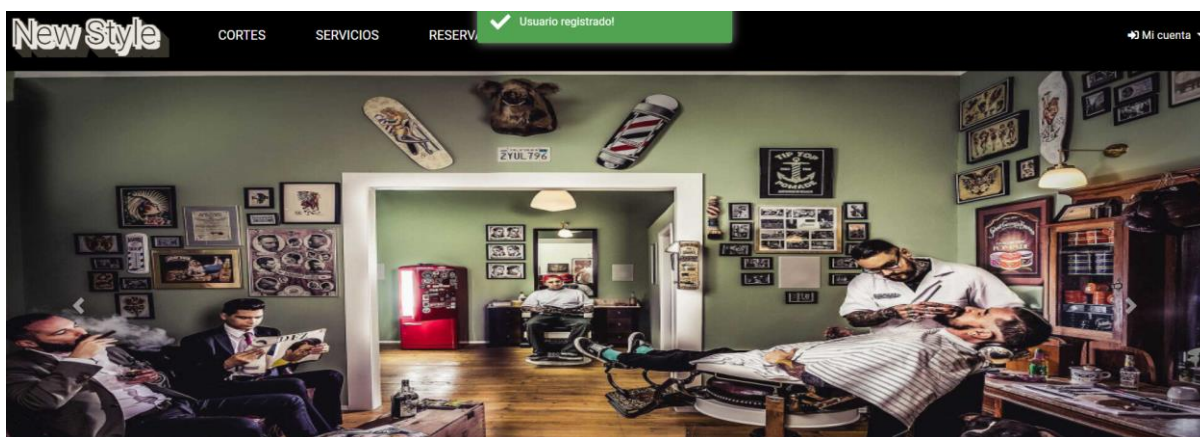


Ilustración 10. Prueba 002

### Prueba 003

Descripción: Aparece un mensaje de error si el usuario o email ya están registrados.

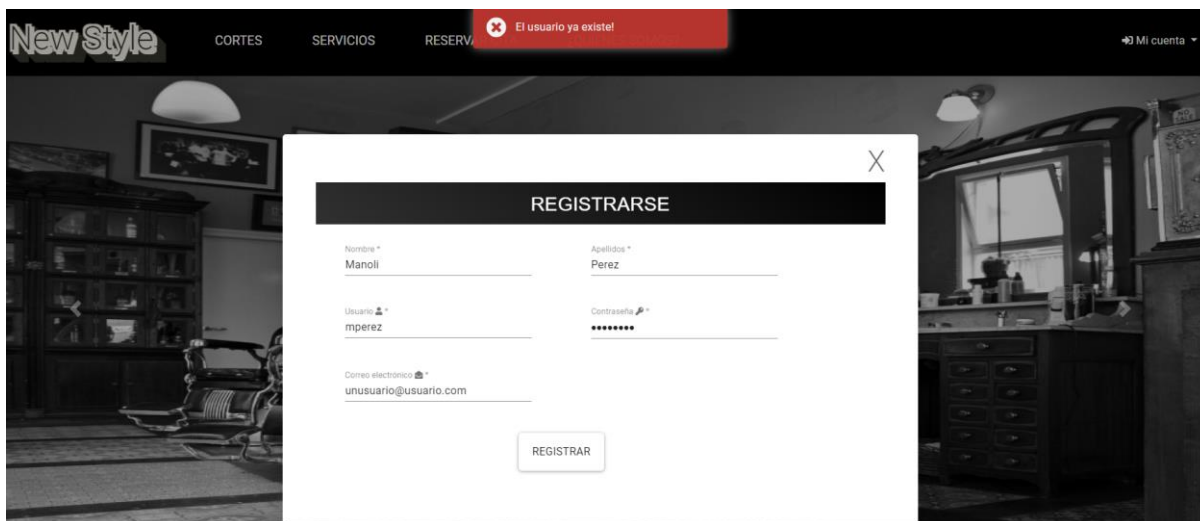


Ilustración 11. Prueba 003

### Prueba 004

Descripción: Aparece un mensaje de error si el usuario o email ya son incorrectos en el formulario de login.

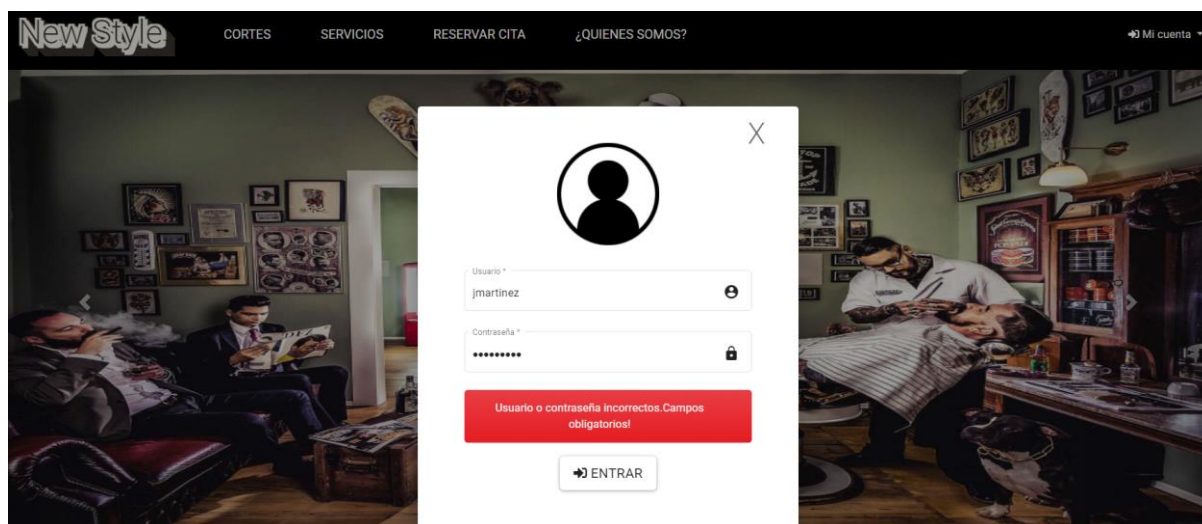


Ilustración 12. Prueba 004

### Prueba 005

Descripción: Aparece el nombre del administrador logado y el menú de administrador.

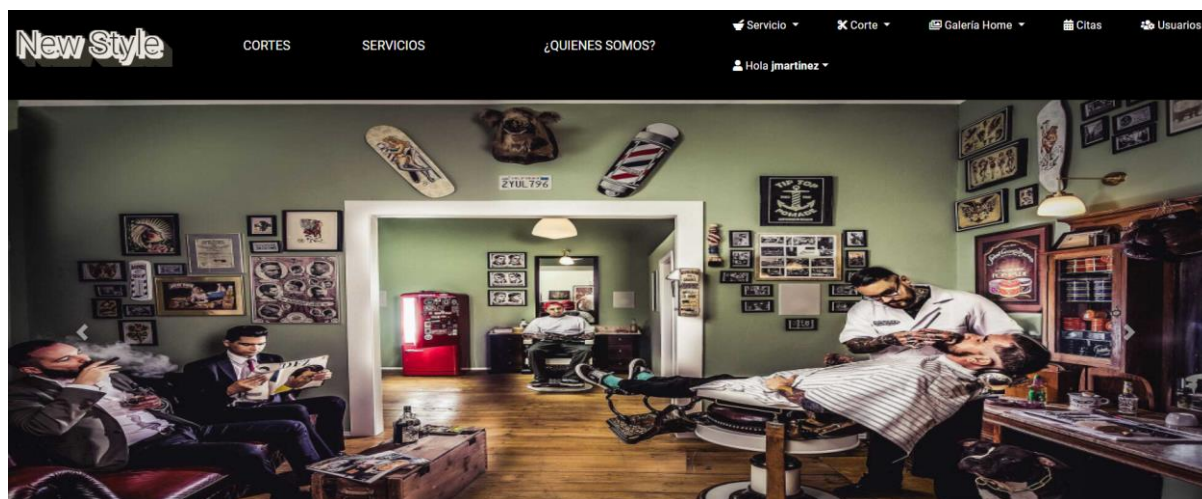


Ilustración 13. Prueba 005

## Prueba 006

Descripción: Aparece el nombre del usuario logado y el menú de usuario.

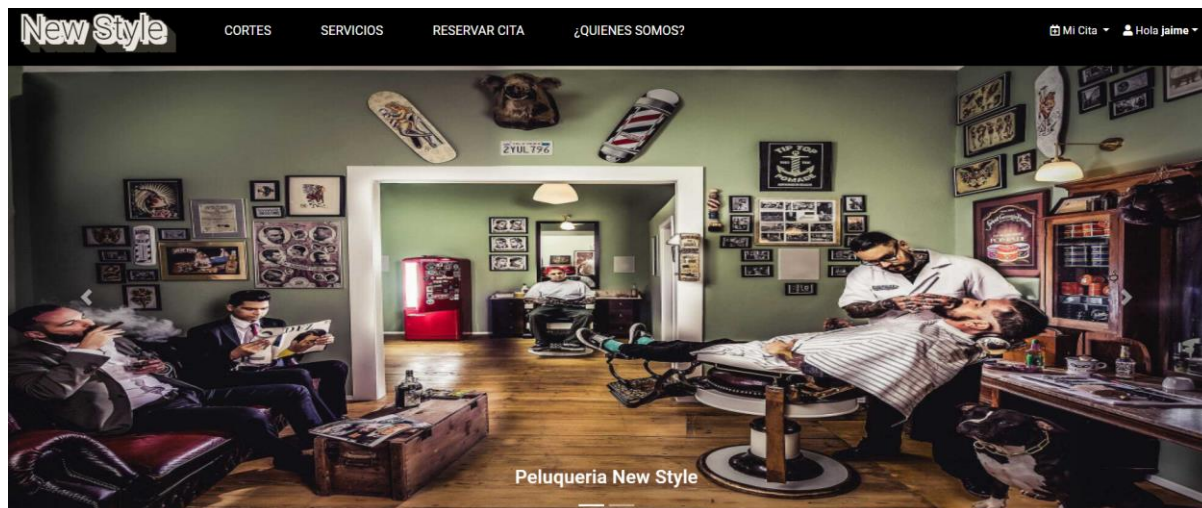


Ilustración 14. Prueba 006

## Prueba 007

Descripción: se listan los usuarios una vez logado como administrador

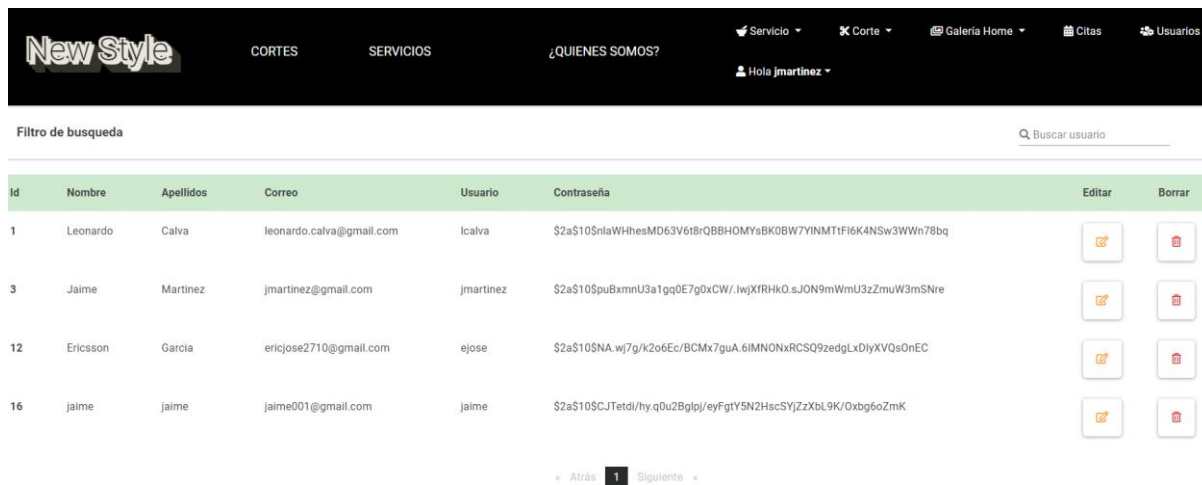


Ilustración 15. Prueba 007

## Prueba 008

Descripción: lista las citas del usuario logado



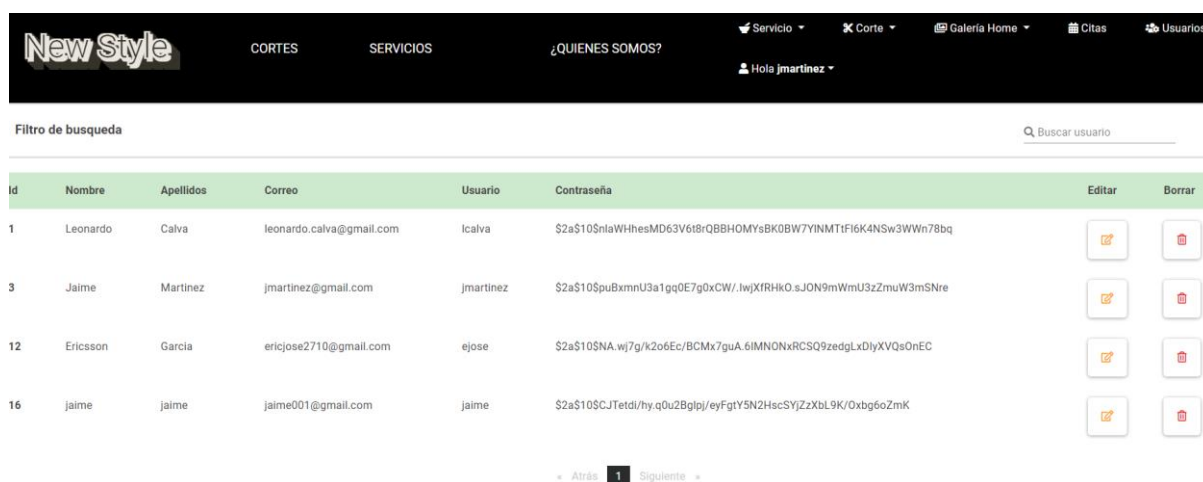
ID	Nombre	Apellidos	Email	Hora	Fecha	Servicio
6	jaime	jaime	jaime001@gmail.com	12:00	2021-06-10	Afeitado barba

« Atrás 1 Sigüente »

Ilustración 16. Prueba 008

## Prueba 009

Descripción: lista todas las citas de los usuarios una vez logado como administrador y tras pulsar el botón de citas.



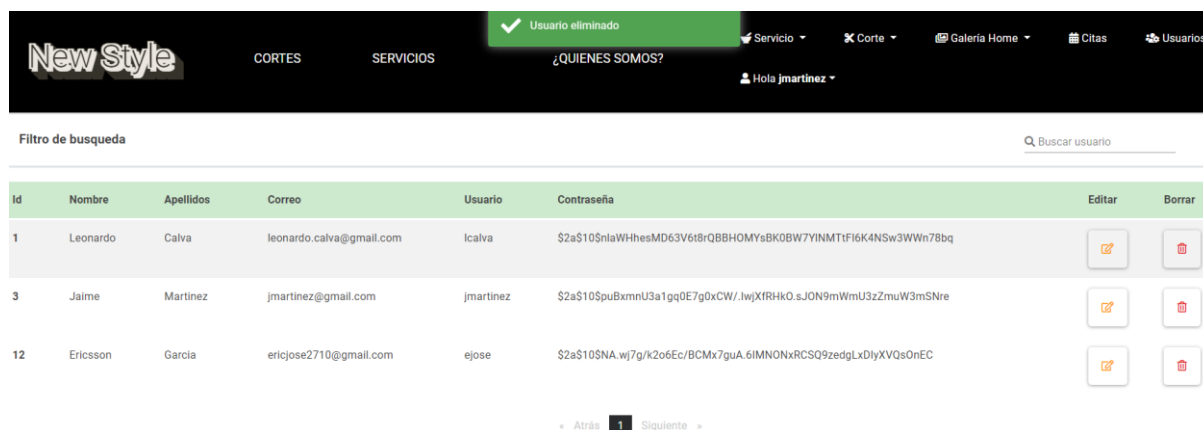
Id	Nombre	Apellidos	Correo	Usuario	Contraseña	Editar	Borrar
1	Leonardo	Calva	leonardo.calva@gmail.com	lcalva	\$2a\$10\$niaWHhesMD63V6t8rQBHOMYsBK0BW7YINMTfI6K4NSw3WWn78bq		
3	Jaime	Martinez	jmartinez@gmail.com	jmartinez	\$2a\$10\$puBxmnU3a1gq0E7g0xCW/.lwjXfRhK0.sJON9mWmU3zZmuW3mSNre		
12	Ericsson	Garcia	ericjose2710@gmail.com	ejose	\$2a\$10\$NA.wj7g/k2o6Ec/BCMx7guA.6IMNONxRCSQ9zedgLDlyXVQsOnEC		
16	jaime	jaime	jaime001@gmail.com	jaime	\$2a\$10\$CJTetdi/hy.q0u2Bgipi/eyFgtY5N2HscSYjZzXbL9K/Oxbg6oZmK		

« Atrás 1 Sigüente »

Ilustración 17. Prueba 009

## Prueba 010

Descripción: muestra mensaje de confirmación una vez está eliminado el usuario.



Id	Nombre	Apellidos	Correo	Usuario	Contraseña	Editar	Borrar
1	Leonardo	Calva	leonardo.calva@gmail.com	lcalva	\$2a\$10\$niaWHhesMD63V6t8rQBHOMYsBK0BW7YINMTfI6K4NSw3WWn78bq		
3	Jaime	Martinez	jmartinez@gmail.com	jmartinez	\$2a\$10\$puBxmnU3a1gq0E7g0xCW/.lwjXfRhK0.sJON9mWmU3zZmuW3mSNre		
12	Ericsson	Garcia	ericjose2710@gmail.com	ejose	\$2a\$10\$NA.wj7g/k2o6Ec/BCMx7guA.6IMNONxRCSQ9zedgLDlyXVQsOnEC		

« Atrás 1 Sigüente »

Ilustración 18. Prueba 010



### Prueba 011

Descripción: muestra la información del usuario logado.

**New Style** CORTES SERVICIOS RESERVAR CITA ¿QUIENES SOMOS?

Mi Cita Hola jaimem

### MI PERFIL

Nombre: Jaime Apellidos: Martinez

Usuario: jaimem Contraseña: \*\*\*\*\*

Correo electrónico: jaime001@jaime.com

ACTUALIZAR PERFIL

▲ Se enviará correo con la modificación de los datos! ▲

Ilustración 19. Prueba 011

### Prueba 012

Descripción: muestra un mensaje de confirmación al actualizar los datos correctamente.

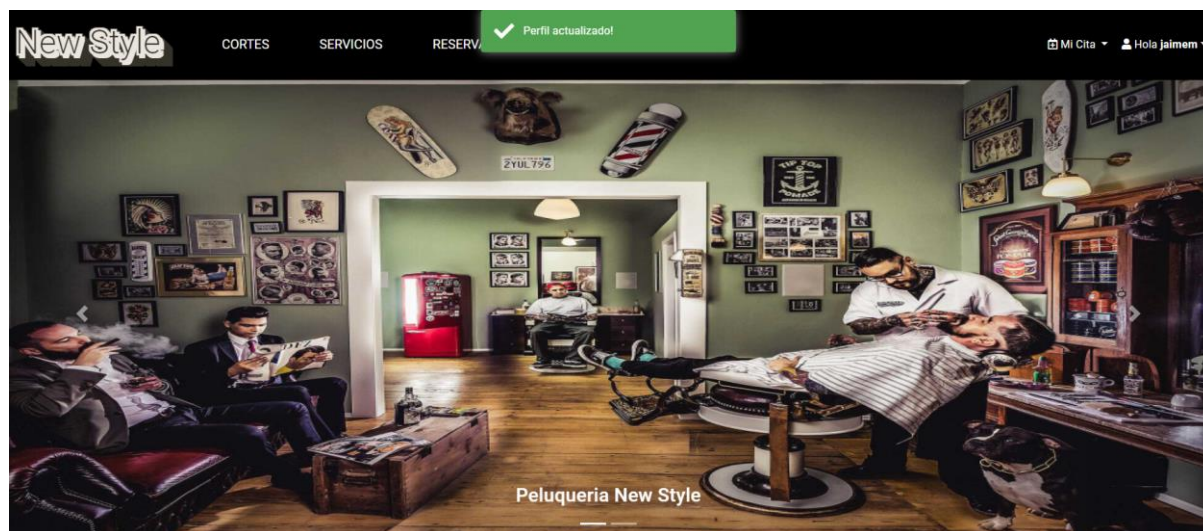


Ilustración 20. Prueba 012

### Prueba 013

Descripción: se listan las imágenes que se muestran en la galería de home una vez logados como administrador y pinchando sobre listar galería home.

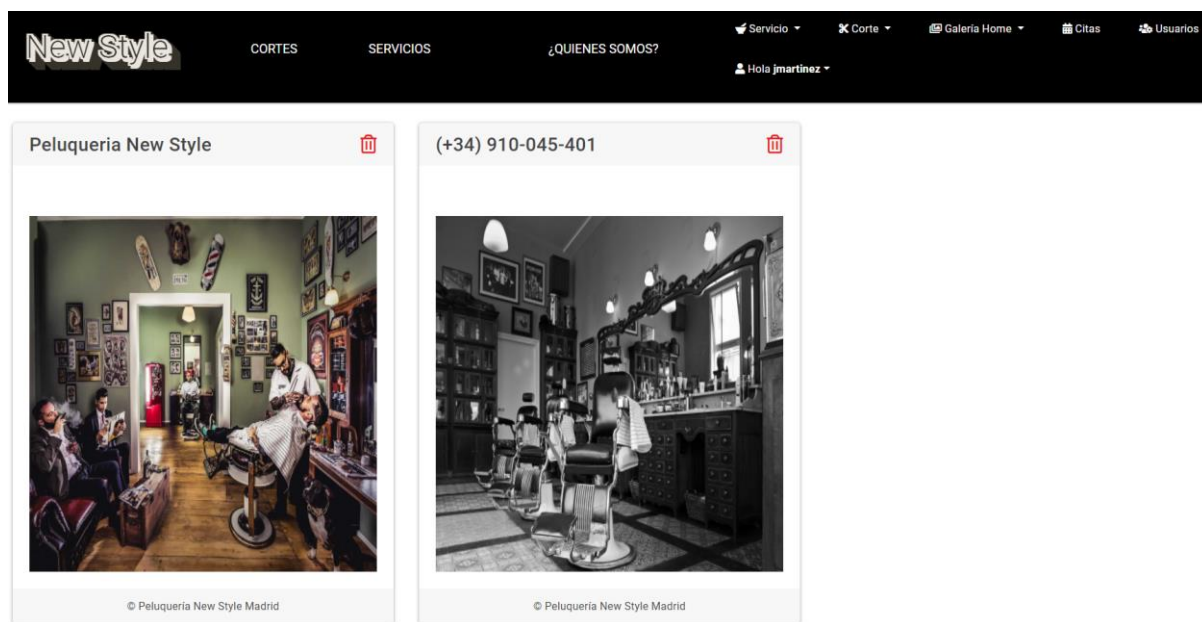


Ilustración 21. Prueba 013

### Prueba 014

Descripción: se listan las imágenes que se muestran en la galería de cortes una vez logados como administrador y pinchando sobre listar cortes y se puede ver el icono de borrar sobre cada una de ellas.

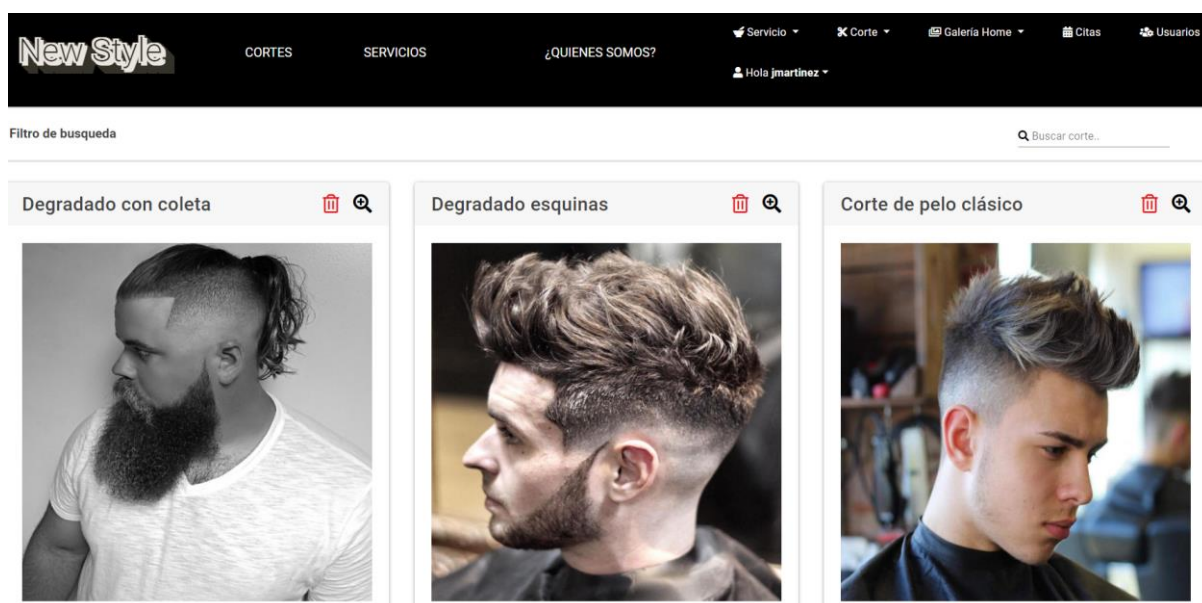


Ilustración 22. Prueba 014

## Prueba 015

Descripción: se elimina la imagen al pulsar el icono de papelera y aparece un mensaje de confirmación.

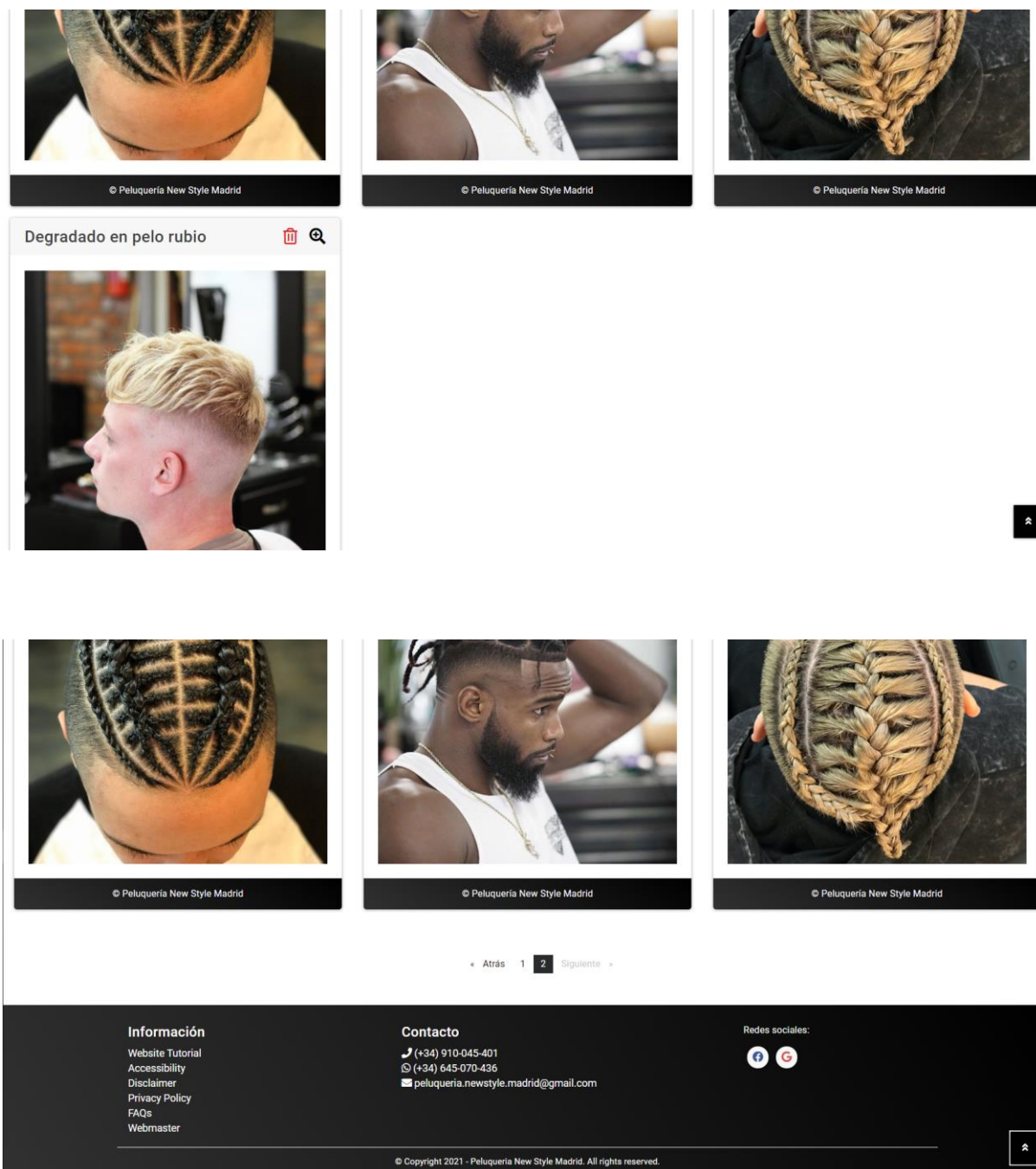


Ilustración 23. Prueba 015



## Prueba 016

Descripción: se amplía la imagen al pulsar sobre el icono de lupa.

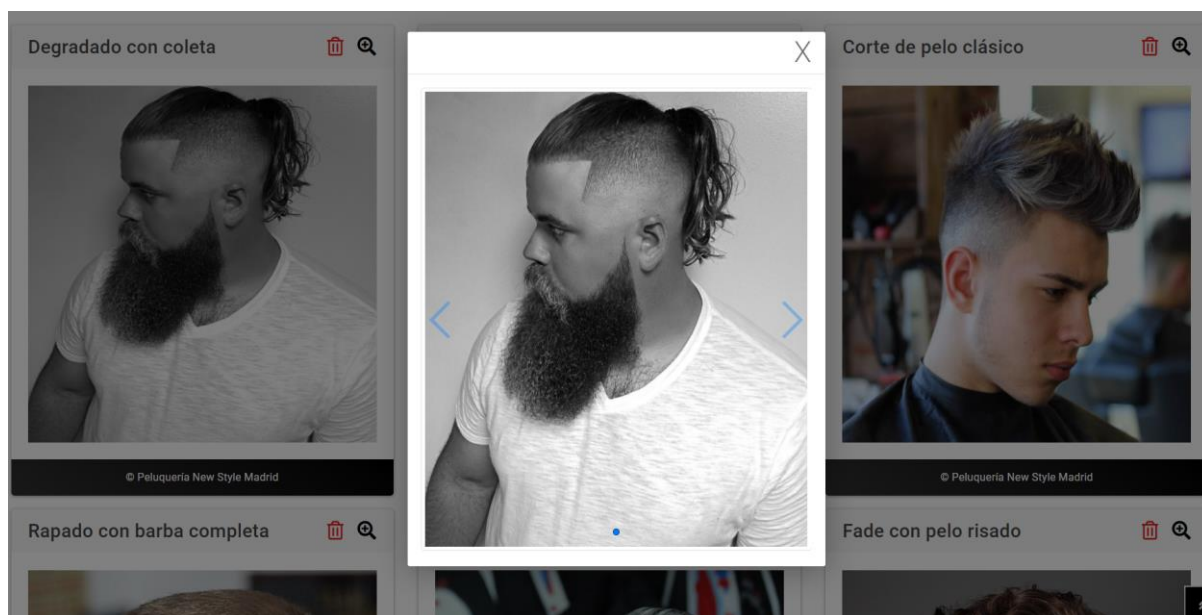


Ilustración 24. Prueba 016

## Prueba 017

Descripción: se listan los servicios al listar sobre el botón de servicios.

CORTES
SERVICIOS
¿QUIENES SOMOS?
Servicio
Corte
Galería Home
Citas
Usuarios
Hola jmartinez

TIPOS DE SERVICIOS QUE OFRECEMOS

ID	SERVICIOS	MUJERES	HOMBRES	NIÑOS	TERCERA EDAD
1	CORTE DE PELO	15 €	10 €	8 €	8 €
2	AFEITAR BARBA	No disponible	5 €	No disponible	5 €
3	CORTE MÁS AFEITADO	No disponible	12 €	No disponible	10 €

**Información**  
Website Tutorial  
Accessibility  
Disclaimer  
Privacy Policy  
FAQs  
Webmaster

**Contacto**  
(+34) 910-045-401  
(+34) 645-070-436  
peluqueria.newstyle.madrid@gmail.com

**Redes sociales:**

© Copyright 2021 - Peluqueria New Style Madrid. All rights reserved.

Ilustración 25. Prueba 017

## Prueba 018

Descripción: se muestra información de contacto al pulsar sobre el botón ¿quiénes somos?

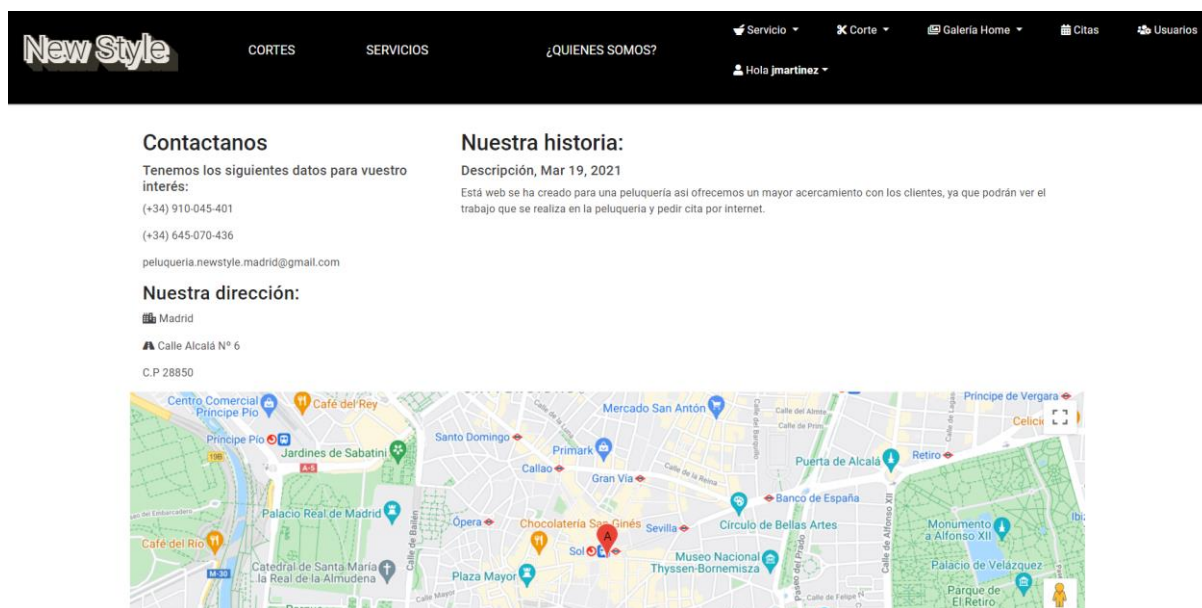


Ilustración 26. Prueba 018

## Prueba 019

Descripción: aparece el formulario de nuevo servicio al pulsar sobre nuevo servicio.

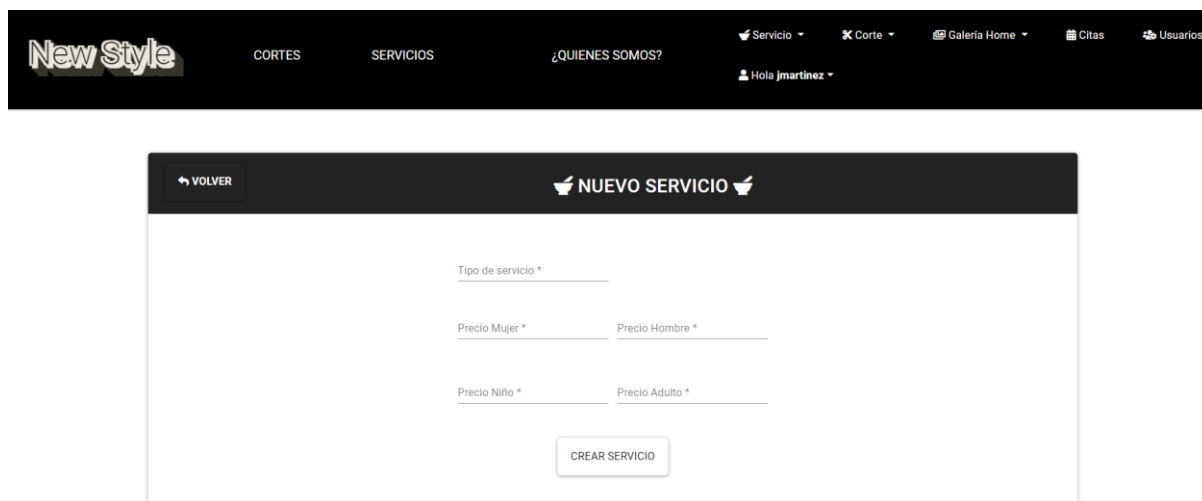


Ilustración 27. Prueba 019

### Prueba 020

Descripción: aparece el formulario de nueva cita al pulsar sobre el botón nueva cita estando registrado como usuario.

Ilustración 28. Prueba 020

### Prueba 021

Descripción: aparece un select desplegable al pulsar sobre servicios en el formulario de nueva cita.

Ilustración 29. Prueba 021

### Prueba 022

Descripción: los campos del formulario de nueva cita se ponen en rojo si no están rellenos.

Ilustración 30. Prueba 022

### Prueba 023

Descripción: se listan los cortes al pulsar sobre el botón cortes y no aparece ningún icono de papelera ni de lupa ya que se trata de un usuario y no de administrador.

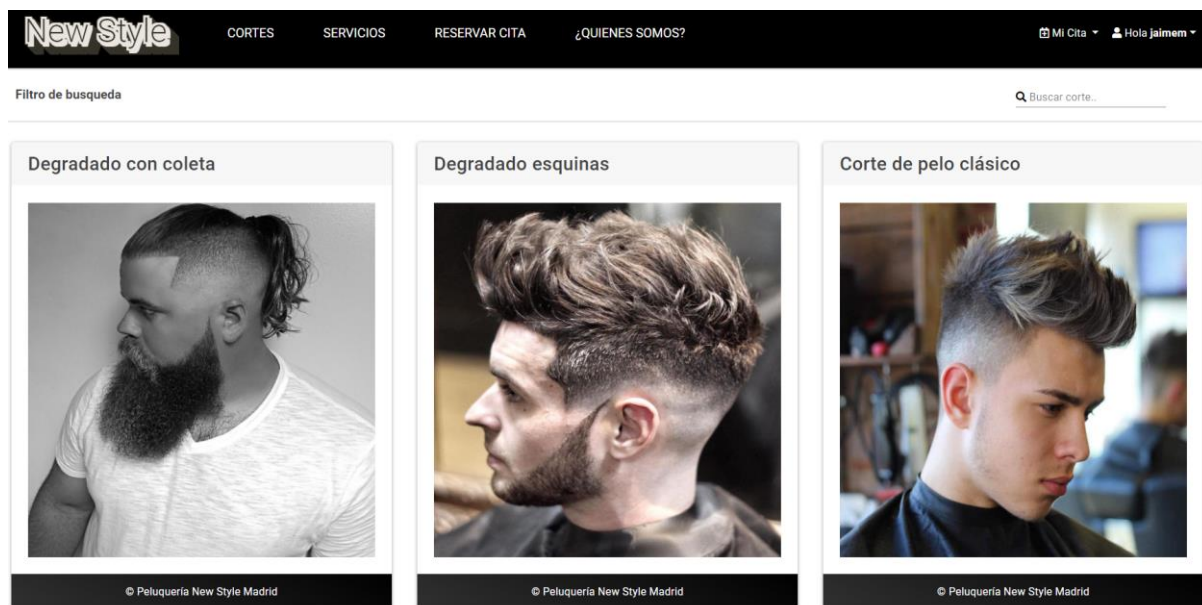


Ilustración 31. Prueba 023

## 8. EXPLOTACIÓN

La implantación es la fase más crítica del proyecto ya que el sistema entra en producción, es decir, opera en un entorno real con usuarios reales.

A continuación, se detallan algunos de los casos de prueba que se ejecutarán para comprobar la correcta construcción de este proyecto.

### 8.1 Planificación

Una vez alcanzados unos objetivos mínimos en cuanto a funcionalidad de la aplicación esta debería subirse al servidor elegido, pero como hemos dicho anteriormente, en nuestro caso no se realizará y se ejecutará en un entorno local.

Los objetivos mínimos que debe cumplir la aplicación antes de pasar a producción, es decir, antes de ser lanzada son:

1. Debe poder registrar usuarios y almacenarlos en la base de datos de forma correcta.
2. Debe permitir acceder con un usuario registrado al panel de usuario.
3. Debe permitir acceder con un administrador registrado al panel de administrador.
4. Debe permitir crear citas una vez logado como usuario.
5. Debe mostrar todos los servicios creados, tomándose de la base de datos.

### 8.2 Preparación para el cambio

El repositorio se configurará de la siguiente manera para realizar una subida al servidor con la última versión probada y estable:

- La rama “master” será la que contenga la versión definitiva y estable, que se subirá al servidor.
- La rama “dev” contiene la versión que se está probando actualmente y donde se están metiendo los últimos cambios. Es una rama que funciona, pero no es estable. Cuando se terminan de meter funcionalidades en esta rama y se ha comprobado que la aplicación funciona correctamente realizando un ciclo completo de la misma, se subirá el código a “master” como código estable.

Con este tipo de organización, lo que permitimos es que el servidor siempre tenga la última versión estable, y los desarrolladores puedan encargarse de depurar funcionalidades o meter otras nuevas sin que esto influya en el funcionamiento de la aplicación.

## 9. DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN

### Gestión de sprints y tareas

En nuestro caso para la gestión del proyecto no se ha utilizado ninguna herramienta externa, sino que, partiendo de la base de los casos de uso y el modelo de datos, nos hemos apoyado en una muy buena comunicación facilitada por ser tan solo dos integrantes en el equipo de trabajo.

Hemos ido generando una serie de documentos en Google drive, tales como:

- Guión con los pasos a seguir.
- Diario con los cambios realizados por cada uno de los integrantes del grupo especificando en qué parte del proyecto se han realizado dichas modificaciones, qué queda pendiente de esa tarea y cuál es el siguiente paso a dar para dejar cerrada la misma.

Por otro lado, para la parte de Angular se han utilizado PostMan, que es una utilidad que nos permite evaluar el código a través de una serie de consultas y del terminal y nos devuelve los errores y problemas que presenta nuestro código con posibles soluciones en el caso de error.

### Control de versiones

Para el control de versiones, se ha utilizado GitHub. Github es una plataforma de desarrollo colaborativo que permite alojar proyectos utilizando el sistema de control de versiones Git. Se ha creado un repositorio público configurado de la siguiente manera:

- La rama “master” que contiene la última versión estable, donde todas las funcionalidades del aplicativo están probadas y en funcionamiento.
- La rama “dev” contiene la versión que se está probando actualmente y donde se están realizando los últimos cambios. Es una rama que funciona, pero no es estable. Se realizan merges a “master” cuando el aplicativo tiene la calidad suficiente y se ha realizado un ciclo completo de la aplicación sin fallos.

A la hora de realizar las subidas al repositorio, se han usado diferentes herramientas, como GitHub Desktop o la herramienta que tienen integradas tanto Visual Studio como Spring Boot para el control de versiones.

## 10. BIBLIOGRAFÍA

### **Cloudinary 1:**

[https://cloudinary.com/documentation/java\\_integration](https://cloudinary.com/documentation/java_integration)

[https://cloudinary.com/documentation/java\\_image\\_and\\_video\\_upload](https://cloudinary.com/documentation/java_image_and_video_upload)

### **Angular Guards:**

<https://www.codigocorrecto.com/programacion/angular-2020-como-usar-guard-canactivate-ejemplo-de-uso/>

<https://codingpotions.com/angular-seguridad>

### **Relaciones JPA:**

<https://www.adictosaltrabajo.com/2020/04/02/hibernate-one-to-one-one-to-many-many-to-one-y-many-to-many/>

<https://attacomsian.com/blog/spring-data-jpa-one-to-many-mapping>

### **Angular:**

<https://angular.io/>

<https://ng-bootstrap.github.io/#/home>

<https://www.npmjs.com/package/@angular/material>

<https://angular.io/guide/pipes>

<https://material.angular.io/components/dialog/overview>

### **Jason Web Token:**

<https://jwt.io/>

<https://www.adictosaltrabajo.com/2017/09/25/securizar-un-api-rest-utilizando-json-web-tokens/>

<https://adamofig.medium.com/spring-boot-y-google-cloud-sql-cloud-app-engine-la-forma-m%C3%A1s-f%C3%A1cil-ca1c3f5a051>

### **Spring Boot:**

<https://www.javatpoint.com/angular-spring-crud-example>

### **Google Cloud:**

<https://www.youtube.com/watch?v=D1GJKd62l-A>