

# DAW

## Proyecto Página Web

### Barbería VV



Cristian Novoa González  
Ivan Quintáns González

# ÍNDICE

<b>ÍNDICE.....</b>	<b>2</b>
<b>1. Introducción.....</b>	<b>4</b>
<b>2. Inventario de Contenido.....</b>	<b>5</b>
<b>3. Arquitectura de la Información.....</b>	<b>6</b>
<b>4. Casos de Uso.....</b>	<b>10</b>
<b>5. Mapa de Navegación.....</b>	<b>11</b>
5.1 Mapa de Navegación General.....	11
5.2 Mapa de Navegación Home.....	12
5.3 Mapa de Navegación Galería.....	13
5.4 Mapa de Navegación Ubicación.....	14
5.5 Mapa de Navegación Precios.....	15
5.6 Mapa de Navegación Conócenos.....	16
5.7 Mapa de Navegación Reserva.....	17
<b>6. Interfaces.....</b>	<b>17</b>
6.1. Bocetos.....	17
6.2. Wireframes.....	22
6.3. MockUps.....	27
<b>7. Storyboards.....</b>	<b>31</b>
7.1. Consultar Precios.....	31
7.2. Ver Cortes.....	32
7.3 Hacer Reserva.....	32
7.4 Consultar Historia.....	33
7.5 Obtener Localización.....	33
<b>8. Estructura de ficheros y carpetas.....</b>	<b>34</b>
<b>9.HTML.....</b>	<b>37</b>
<b>10.CSS.....</b>	<b>44</b>
10.1. Navbar y Footer.....	46
10.2 Float.....	49
10.3 Multicol.....	50
10.4 CSS Grid.....	54
10.5 Flex container.....	56
10.6 Bootstrap.....	59
<b>11.Javascript.....</b>	<b>61</b>
11.1 Servidor Nginx.....	61

11.2. Tema de Navbar Dinámico.....	61
11.3 Collapse Navbar.....	62
11.4 Imagen Fullscreen.....	63
11.5 Activación de Campos de Formulario.....	64
11.6 Validación de Fecha.....	65
11.7 Carga Dinámica de Catálogo.....	66
11.8 Publicar Formulario.....	68
11.9 Carga Dinámica de Horas Disponibles.....	69

## 1. Introducción

Este proyecto incorpora la creación de un documento que incorpore toda la información necesaria para el posterior desarrollo del sitio web creación de una página web de libre elección.

En nuestro caso mediante un proceso de pensamiento hemos elegido la creación de una página web de una barbería.

La Barberia VV es una barbería localizada en Baiona, Pontevedra, Galicia que nos ha solicitado la creación de una página web para su negocio. Esta página debe ser una página web que tenga la información básica sobre los precios, la localización, la galería de los cortes de pelo y trabajos realizados, información de contacto, trabajadores del negocio y finalmente un apartado para hacer reservas mediante un formulario que servirá a posteriori para la implementación del backend donde se proporcionará un sistema mediante el cual se puedan realizar citas al negocio.

Una vez con la propuesta, después de una larga investigación sobre otros trabajos realizados para otras barberías y de una investigación sobre otros sitios web del sector hemos encontrado diversas referencias para el diseño de la misma, que nos permitirán tener una base y una idea para los diseños y la estructuración de la página web.

Las referencias encontradas son las siguientes:

<https://newyorkbarbers.com.au/services/>

<https://www.barberia.es/BarberoNew/home/contacto.jsp>

Después de este proceso de investigación hemos desarrollado diversos puntos para la facilitación del desarrollo de la aplicación que serán desarrollados a como el inventario de contenido, la arquitectura de la información, el diagrama de casos de uso, mapas de navegación de la web y los bocetos, los wireframes y los mockups de las páginas de la aplicación a desarrollar.

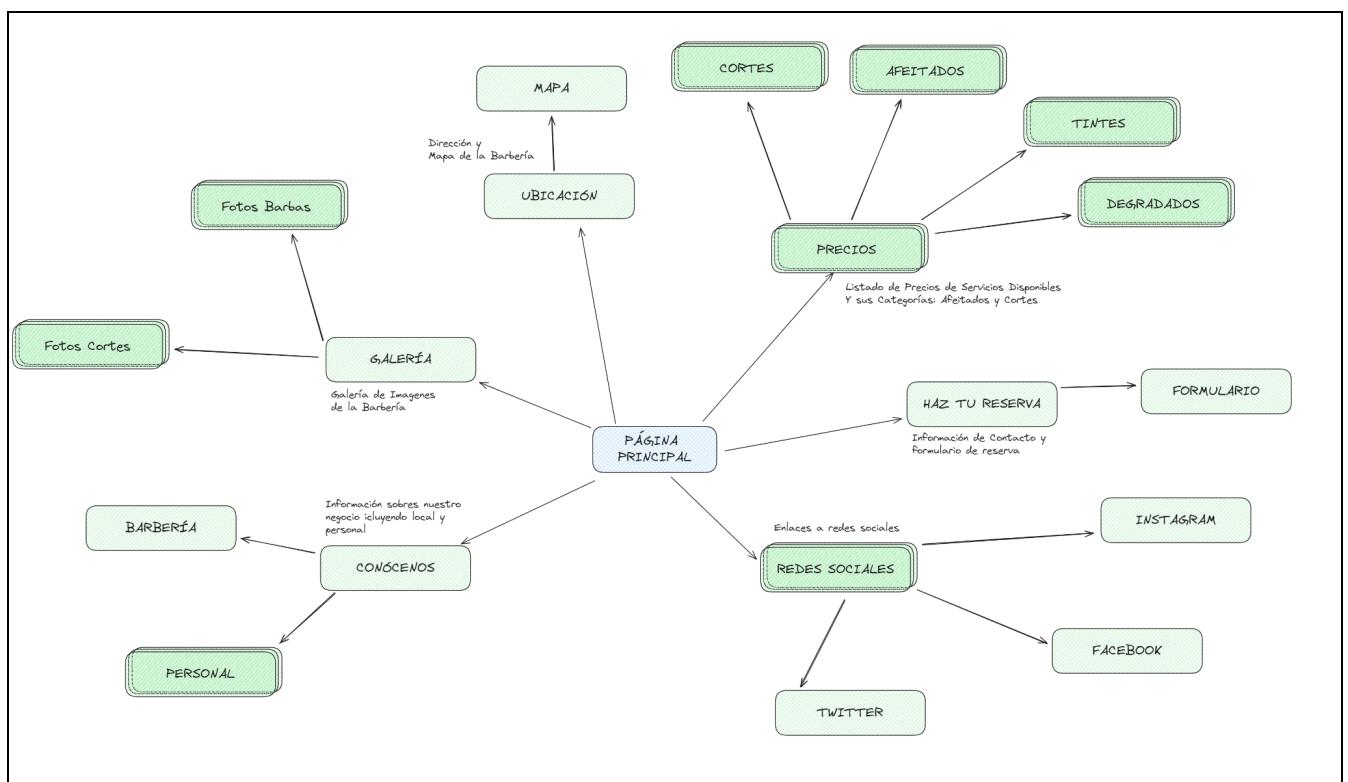
Posteriormente se realizará la página web en cuestión sobre la Barbería VV empleando los lenguajes HTML, CSS y Javascript además de la inclusión de información mediante JSON o XML para la realización de la página intentando realizar los contenidos de la manera más próxima posible a los mockups y bocetos realizados. Sin embargo es posible que durante la realización de la misma se realicen cambios en el diseño dado que realizandolos de esta nueva forma queden mejor o sean más adecuados para la experiencia del usuario. Para estos casos cuando se realicen los diversos apartados de documentación de las páginas finales se mencionan los cambios realizados con respecto a los mockups.

## 2. Inventario de Contenido

En primer lugar, a la hora de diseñar la aplicación se realizó un inventario de contenido para obtener la información que se quería representar en esta página .

En nuestro caso al tratarse de una barbería los principales contenidos a mostrar en la página son contenidos de información que permitan al cliente tener la información necesaria sobre la barbería de una manera rápida. Dentro de estos contenidos tendríamos, los diferentes servicios realizados y su correspondiente precio, la información de la ubicación del negocio y la información de contacto, las redes sociales del negocio, la galería de otros trabajos realizados por la barbería y finalmente la información sobre el resto de los barberos.

Con esta información se realizó el inventario de contenido y este se representa en la siguiente figura:



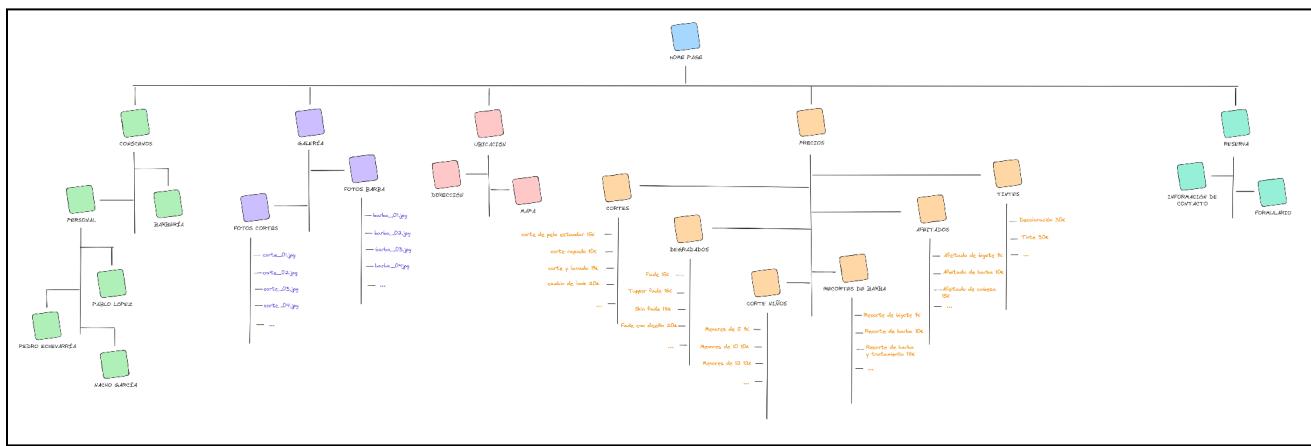
*Imagen 2. Inventario de Contenido*

### 3. Arquitectura de la Información

Una vez obtenida esta información que se quiere representar se realizó la arquitectura de la información para agrupar la información obtenida y de esta manera obtener una mayor claridad a la hora de representar la información de una manera ordenada.

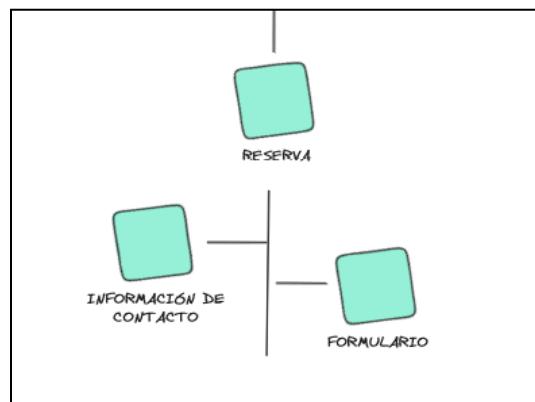
De esta manera aunque solamente estamos agrupando la podemos decir que se está realizando un acercamiento al número de páginas que se va a realizar en la página para representar esta información.

Obtenemos el siguiente diagrama:



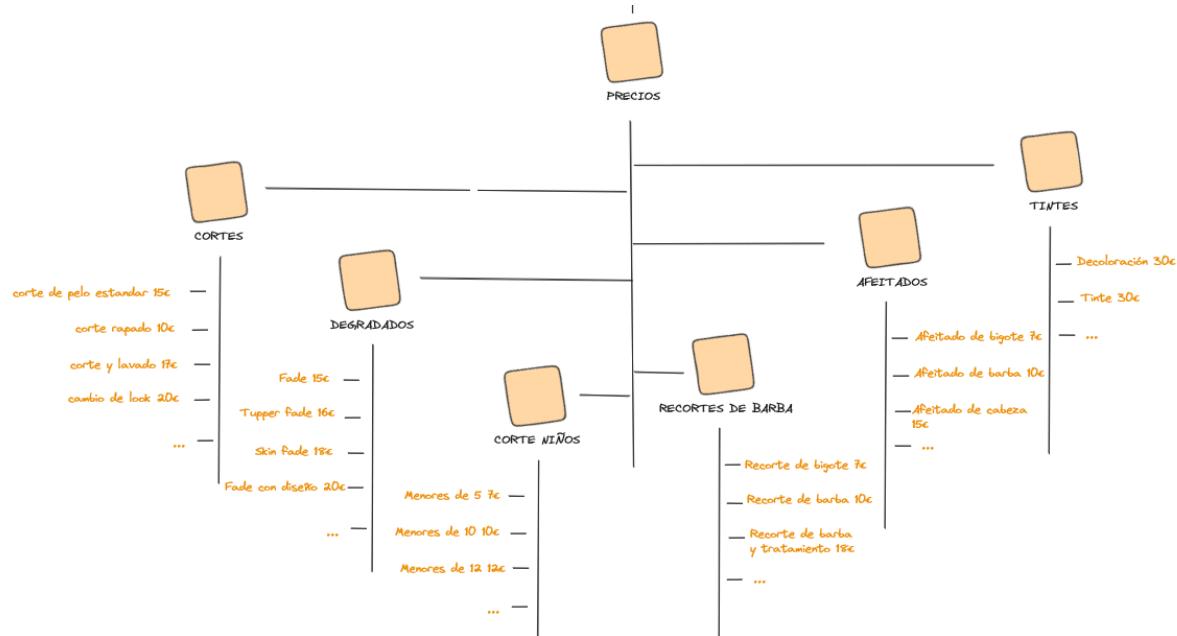
### ***Imagen 3. Arquitectura de la Información***

Con esta imagen nos hacemos una idea del mapa general pero no de cómo está representada la arquitectura por lo que vamos a realizar una visión más clara de cada una de las partes:



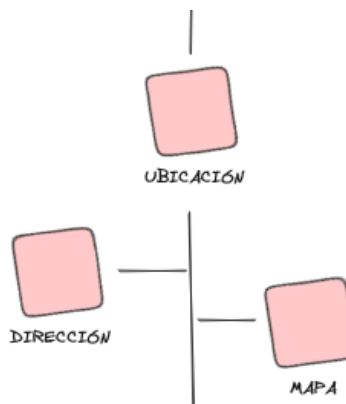
#### *Imagen 4. Reserva*

Como se puede apreciar en la Imagen 4, una de las partes a tener en cuenta en la arquitectura de la información de nuestra página es el caso de reserva donde es necesario mostrar la información de contacto y el formulario asociado para proporcionar al usuario una manera de solicitar una cita.



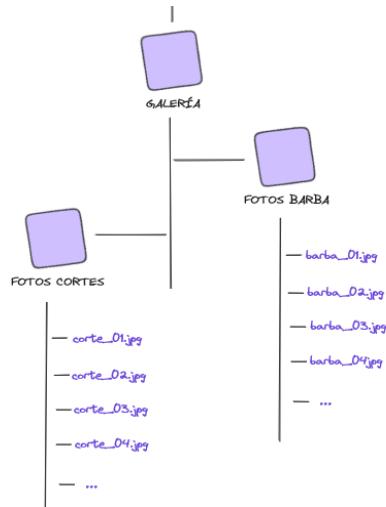
*Imagen 5. Precios*

En la Imagen 5, se aprecia el apartado de la arquitectura de la información referente a los precios y los servicios asociados a la barbería como pueden ser los cortes, afeitados, los tintes o los degradados y sus respectivos precios.



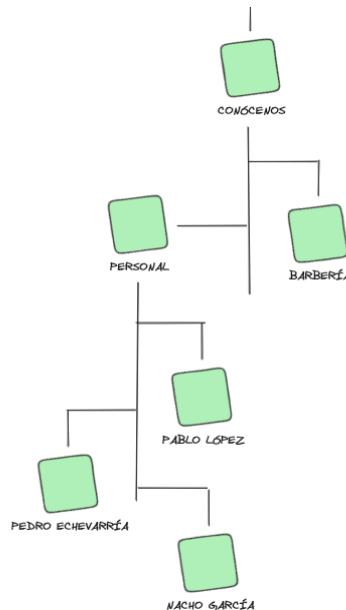
*Imagen 6. Ubicación*

En la Imagen 6, se muestra la información que se quiere mostrar sobre la ubicación y la localización del negocio ya que esta información permitirá a los usuarios conocer la localización del negocio.



**Imagen 7. Galeria**

En la Imagen 7, se muestra la información sobre otros trabajos y otros servicios realizados en el negocio de manera que el usuario obtenga una visual sobre los trabajos realizados



**Imagen 8. Conocenos**

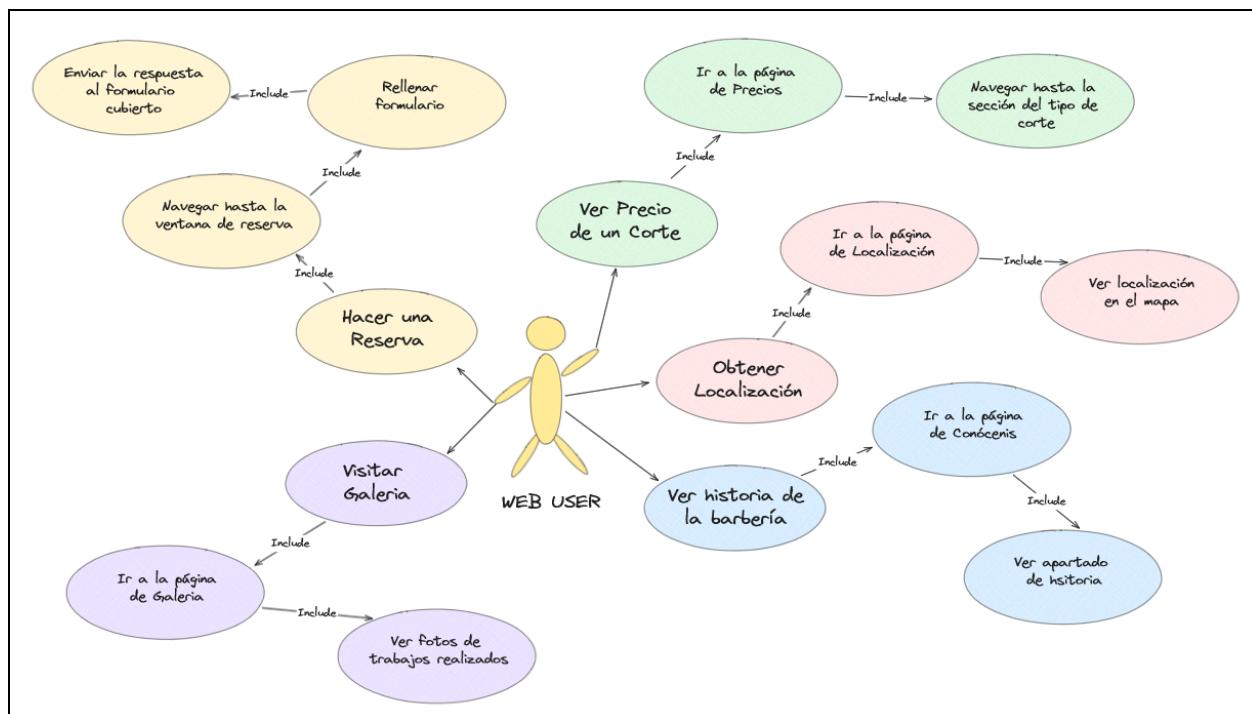
En la Imagen 8, se muestra la información sobre los trabajadores y sobre la historia de la barbería en general permitiendo al usuario conocer un poco sobre los barberos para poder elegir con quien cortarse el pelo.

## 4. Casos de Uso

En este apartado se realizó un diagrama de casos de uso sobre los casos de uso más relevantes en nuestra página web ya que estos permiten seguir la interacción principal del usuario con la página.

En nuestro caso los casos de uso escogidos fueron obtener localización, ver precio de un corte, hacer una reserva, visitar galería y ver historia de la barbería.

Véase aquí el siguiente diagrama:



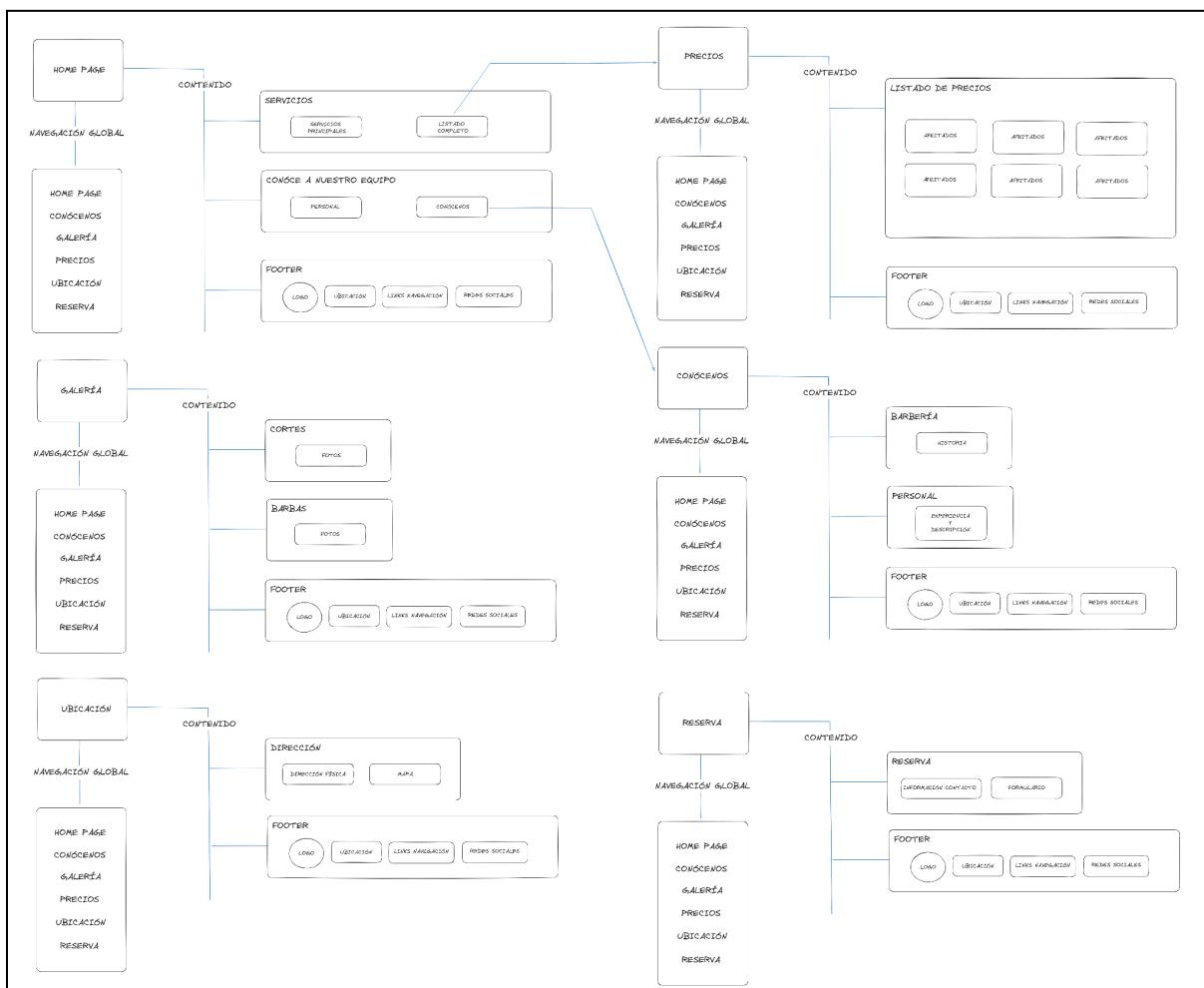
**Imagen 4. Diagrama de Casos de Uso**

## 5. Mapa de Navegación

Una vez recopilada y estructurada la información como se hizo en la Arquitectura de Información, se empiezan a diseñar las diversas páginas y la información que tienen estas, además de como poder acceder a las páginas a través de clicks y botones, y cual es el camino para llegar a una información.

Para ello se realizaron los siguientes mapa de navegación. Debido a su gran magnitud, en los siguientes apartados se desglosaron.

### 5.1 Mapa de Navegación General



**Imagen 5.1. Mapas de Navegación**

## 5.2 Mapa de Navegación Home

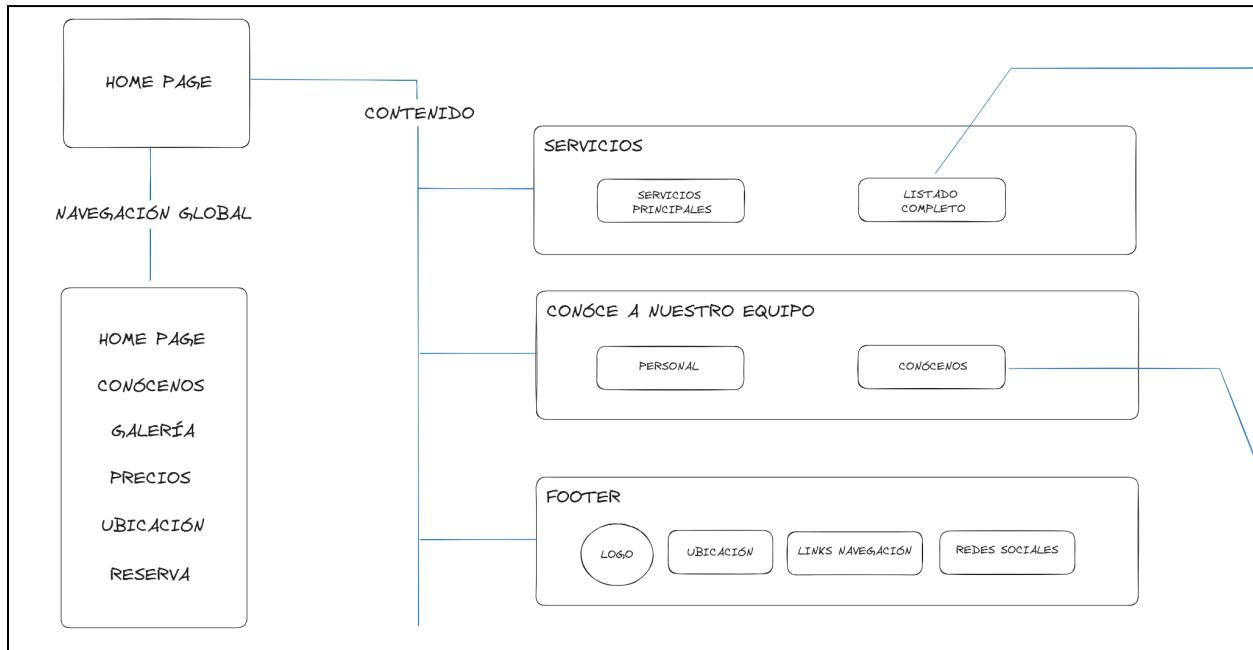


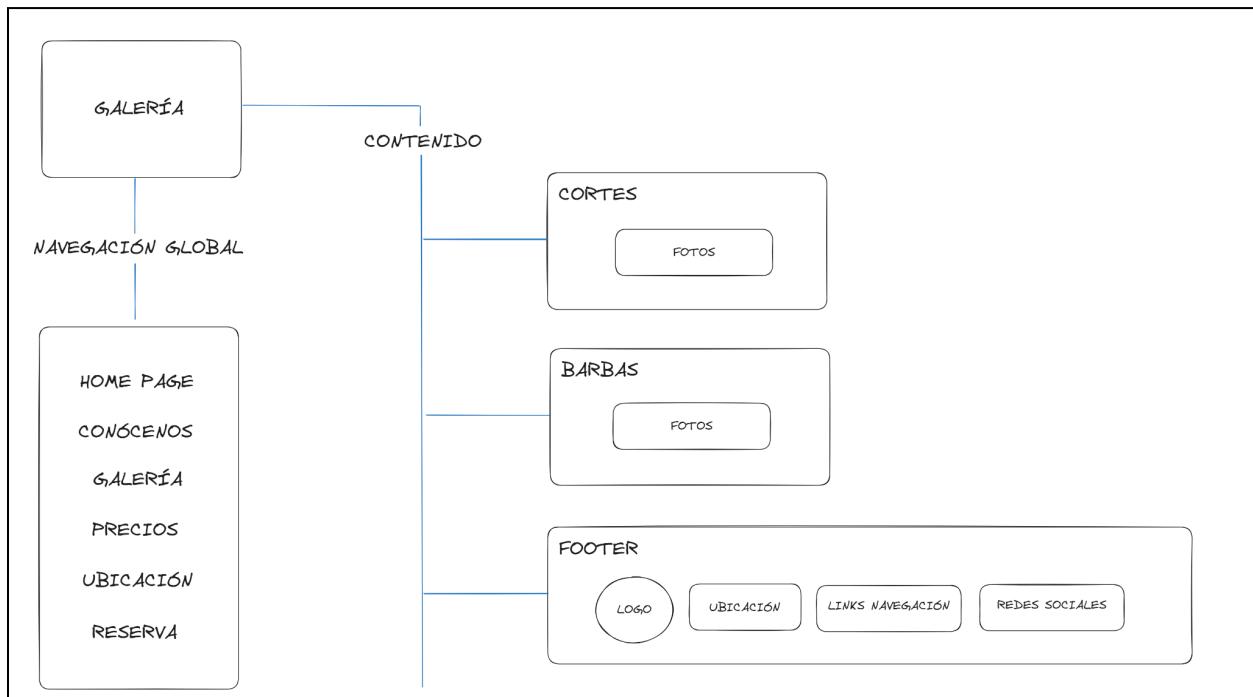
Imagen 5.2. Mapa de Navegación Home

La página home es la página principal de la web de la barbería y por tanto abarca contenido general ilustrativo de la web completa. Cuenta con una navbar y footer que permiten accesos rápidos y completos de navegación para todas las páginas así como enlaces externos a redes sociales o páginas relevantes.

La home contiene 3 secciones principales:

- La portada. No reflejada en el mapa de navegación ya que es puramente estética
- Los servicios. Muestran brevemente los servicios disponibles en la barbería y permiten la navegación directa a la sección correspondiente en la página de *precios*.
- El Equipo. Presenta a los miembros principales de la barbería y permite la navegación a información más detallada sobre estos en la página de *conócenos*.

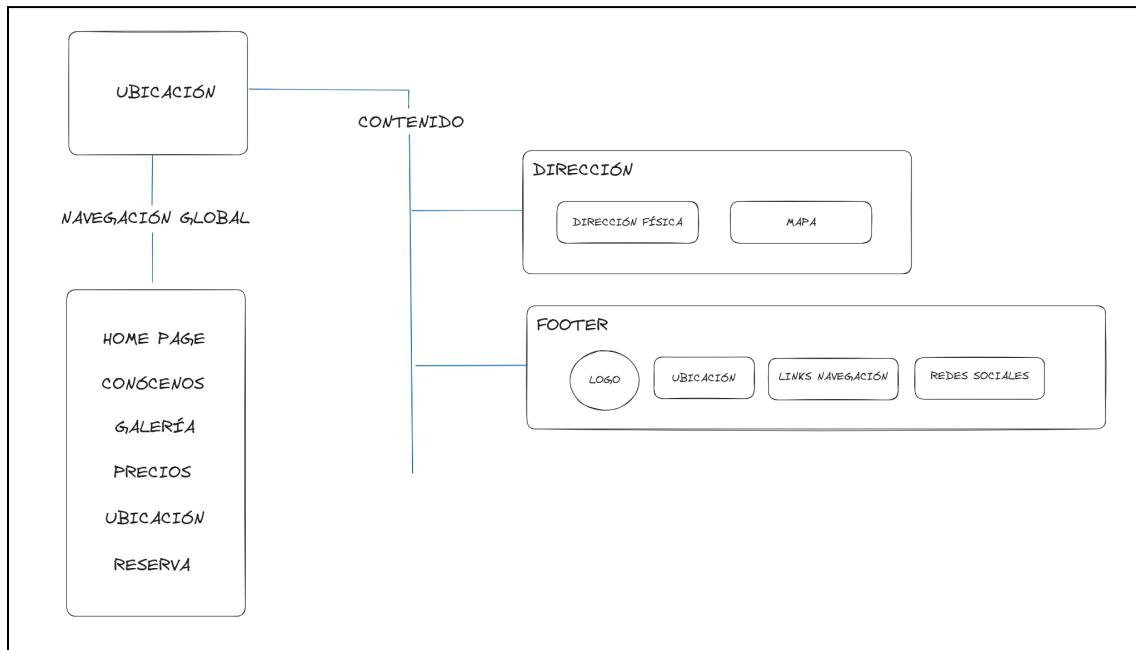
## 5.3 Mapa de Navegación Galería



*Imagen 5.3. Mapa de Navegación Galería*

La galería es accesible mediante los enlaces de navegación generales de la barra de navegación y del footer. La propia página en si no presenta navegación hacia el exterior o hacia otras páginas teniendo como posible excepción la capacidad de visualizar las imágenes presentes en pantalla completa y descargarlas.

## 5.4 Mapa de Navegación Ubicación

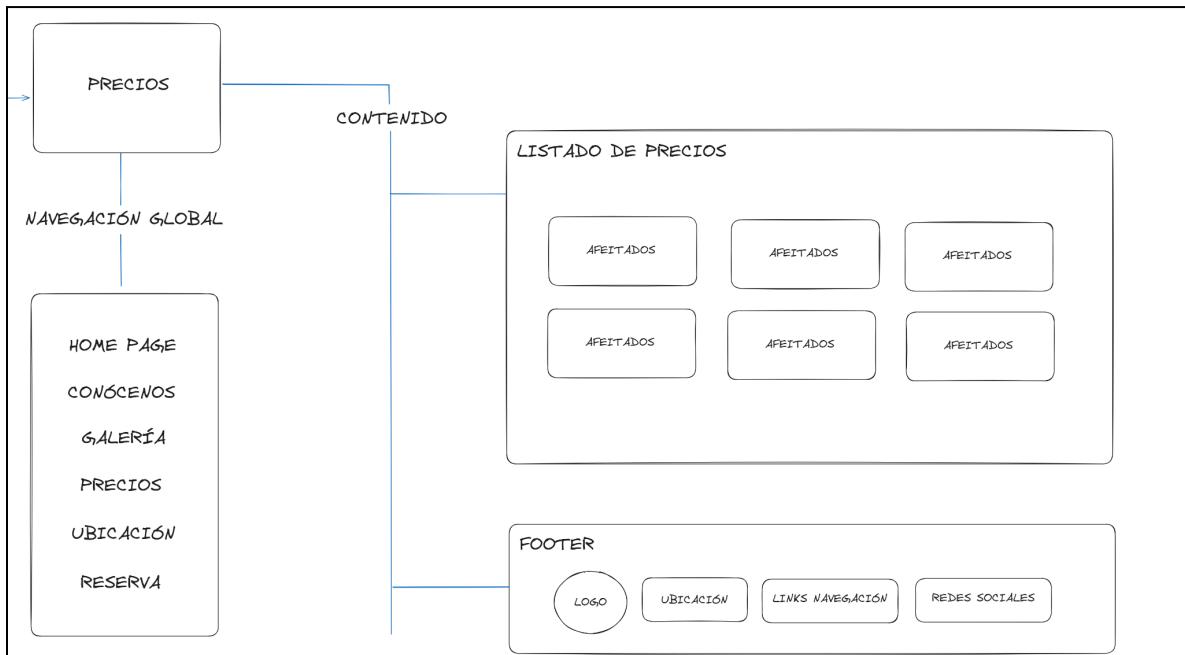


*Imagen 5.4. Mapa de Navegación Ubicación*

La página de ubicación al igual que la *galería* y la página de *reserva* son páginas aisladas únicamente relacionadas con el resto a través de los enlaces de navegación general.

Esta página sí que contiene un enlace de navegación externa hacia la web de Google Maps, en concreto, la dirección correspondiente a la barbería.

## 5.5 Mapa de Navegación Precios

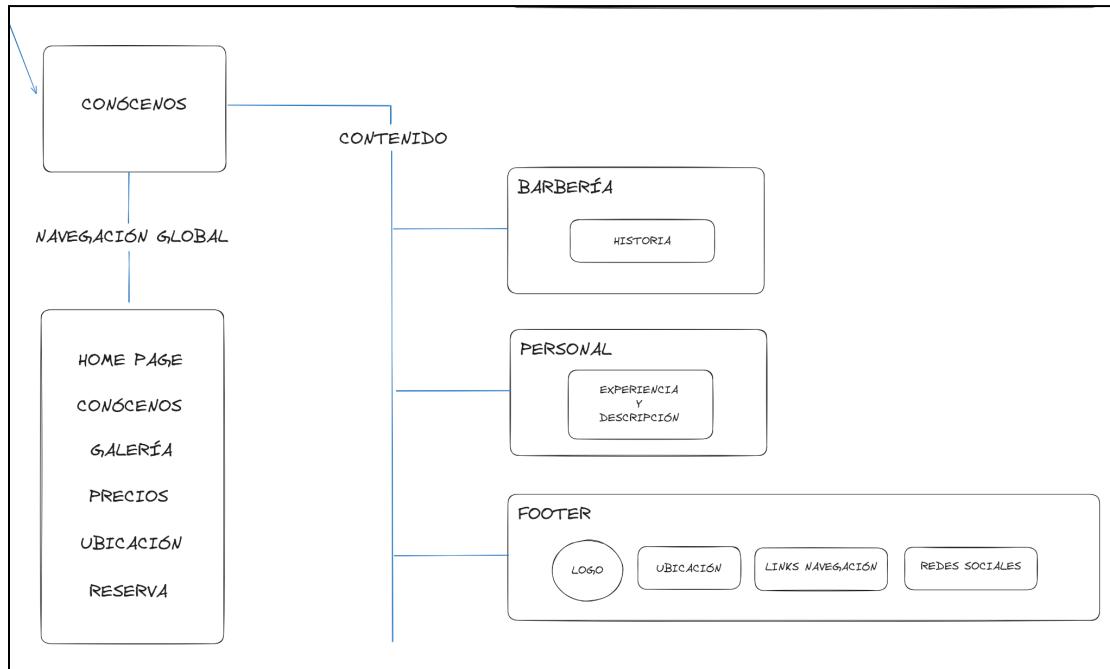


*Imagen 5.5. Mapa de Navegación Precios*

Como se mencionó en la home, además de la navegación global mediante la navbar y footer, el acceso a precios es posible mediante la sección de servicios en la home.

La propia página en sí no contiene ningún link de navegación.

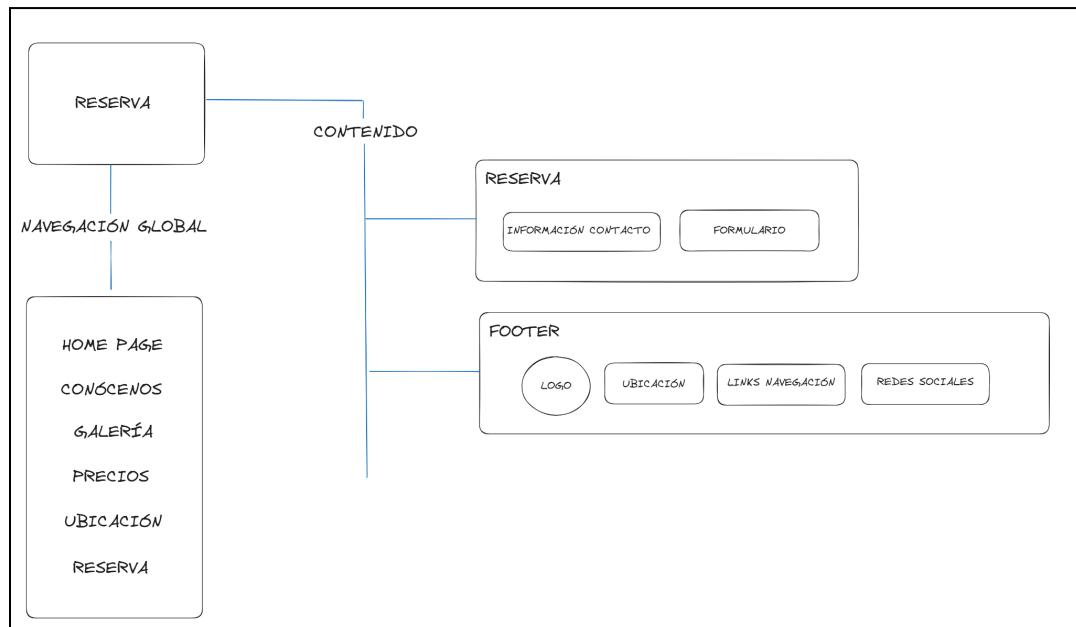
## 5.6 Mapa de Navegación Conócenos



*Imagen 5.6. Mapa de Navegación Conócenos*

Al igual que la página de *precios*, **conócenos** es accesible mediante navegación directa en la sección de *equipo* de la home. Esta navegación permite solo acceder directamente a la sección de *personal* teniendo por lo que solo se puede llegar a la sección de *barbería* mediante scroll del usuario.

## 5.7 Mapa de Navegación Reserva



*Imagen 5.7. Mapa de Navegación Reserva*

Finalmente, la página de *reserva* es la última de las páginas *aisladas* accesible únicamente mediante los enlaces de navegación global. La página en sí no presenta ninguna navegación y cuenta con una sección de información y un formulario de reserva.

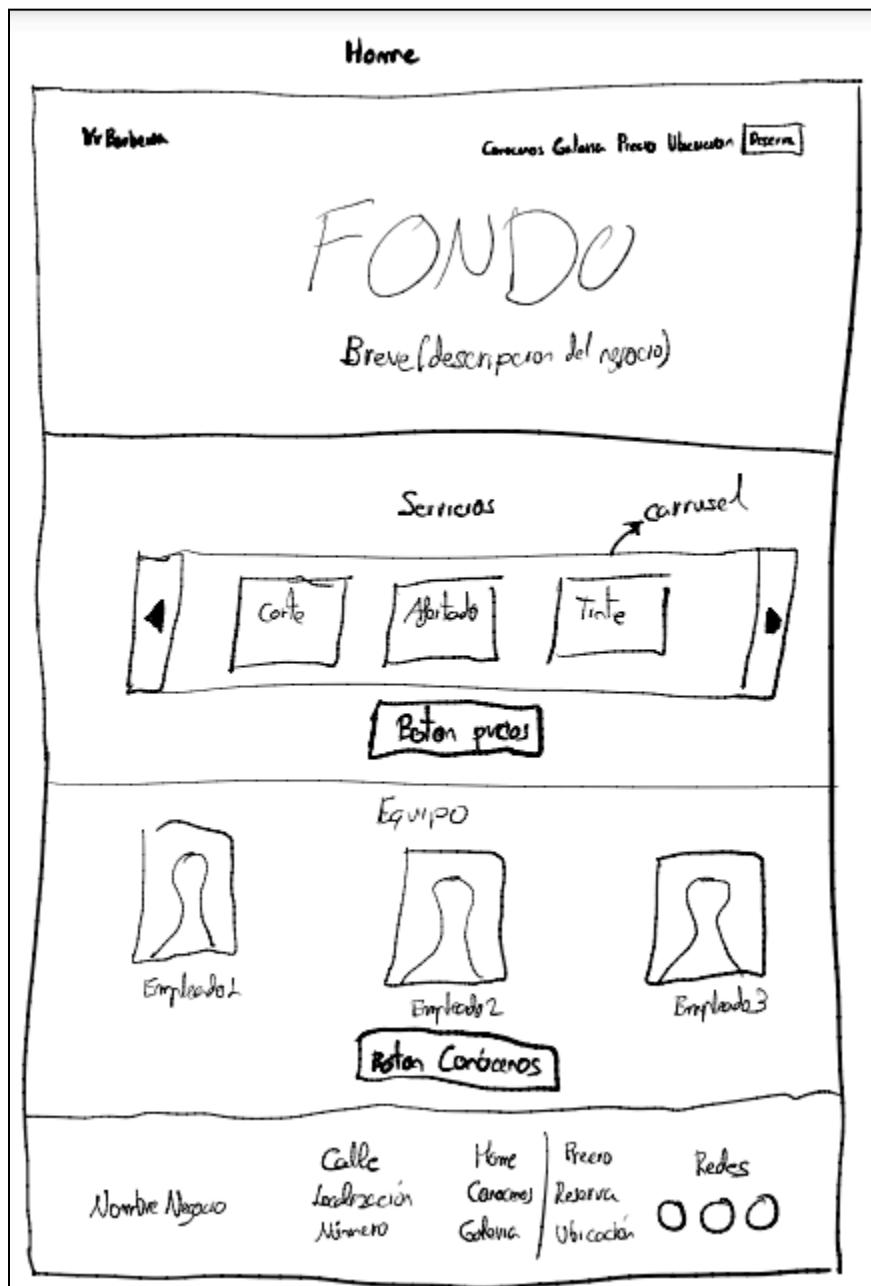
## 6. Interfaces

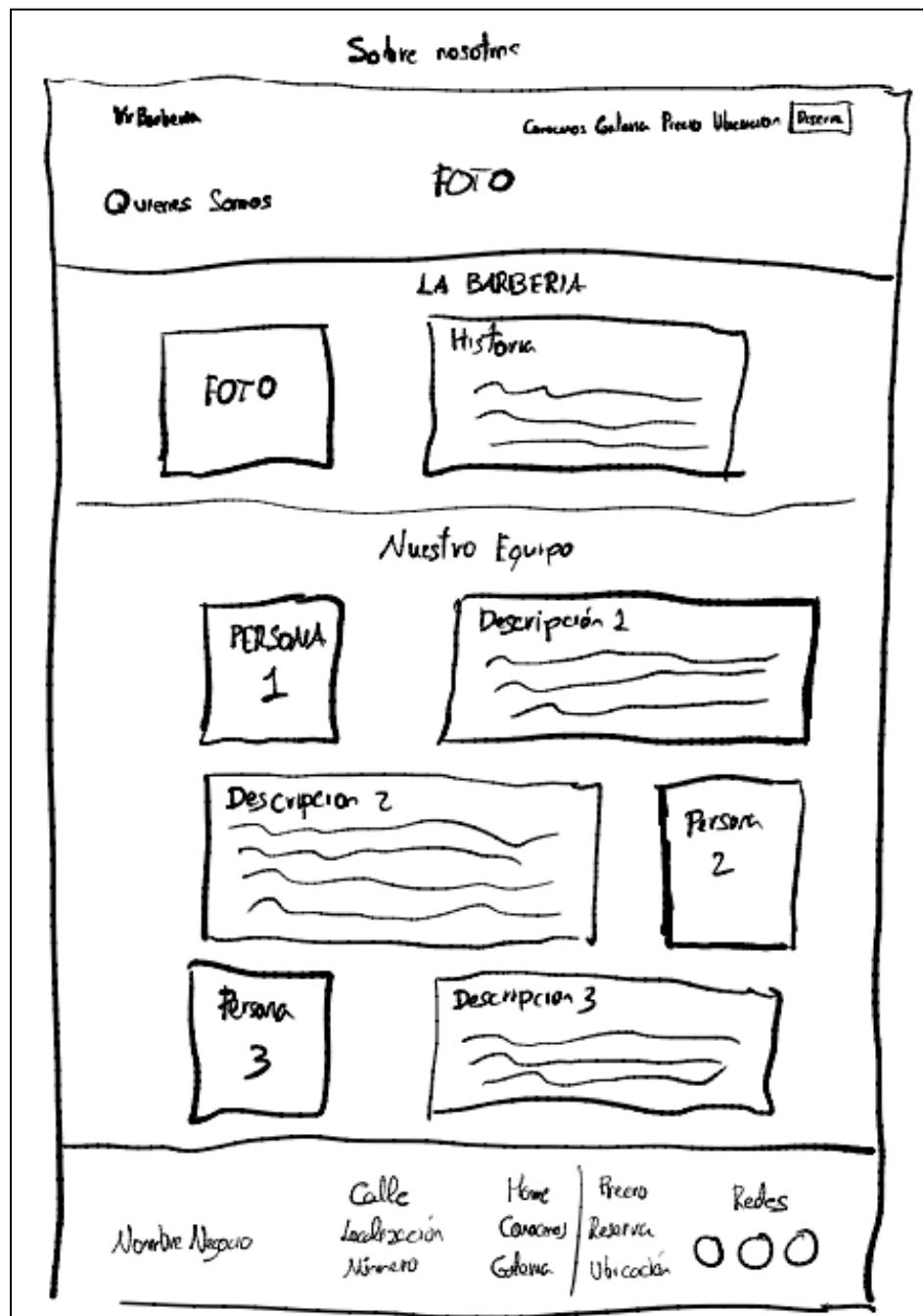
En este apartado una vez organizada la información y teniendo una principal idea de las páginas se realizaron en primer lugar un boceto preliminar de las páginas de la web, después se realizó un wireframe detallando de una mayor manera y finalmente los mockups con el diseño de la aplicación.

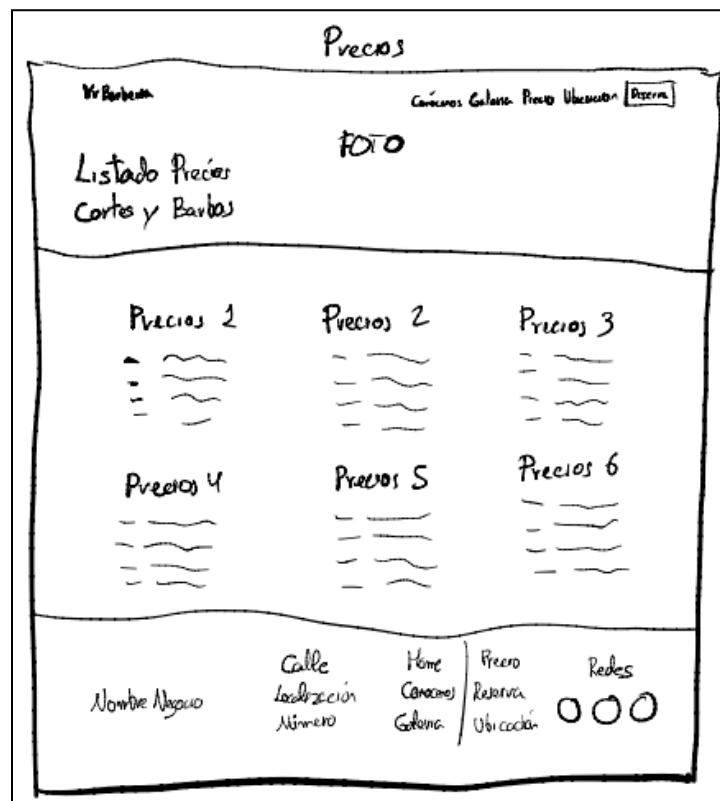
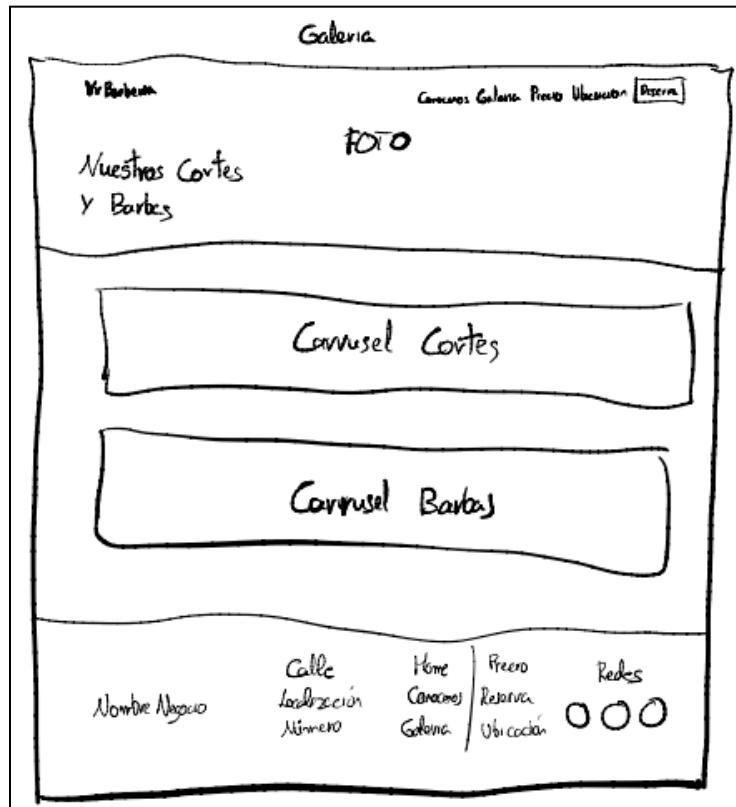
### 6.1. Bocetos

Los siguientes bocetos fueron diseñados para plasmar la esencia de la web: su contenido y estructura general. La finalidad de estos bocetos es permitirnos tener un punto de partida y expresar de forma visual lo que difícilmente se puede expresar mediante palabras. Es por esta razón que los trazos son rápidos, imperfectos y, en ciertos momentos, feos. Son una herramienta para fomentar la creatividad y probar conceptos de forma rápida.

A continuación se muestran los bocetos finales que se pusieron en común antes de la siguiente fase de diseño.







Reserva

Mr Barbería Cancún, Gómez, Puerto Vallarta [Reserva]

FOTO

Reserve

Datos Contacto

Nombre  Dirección   
Apellido  Horario   
Teléfono

FORMULARIO DE RESERVA

Enviar

Nombre Apellido Calle Localización  Home Correo  Preco Reserva  Redes   
Número Gómez, Gómez, Gómez, Ubicación  000

Ubicación

Mr Barbería Cancún, Gómez, Puerto Vallarta [Reserva]

FOTO

Visítanos

Datos Básicos

Nombre  Avenida ...

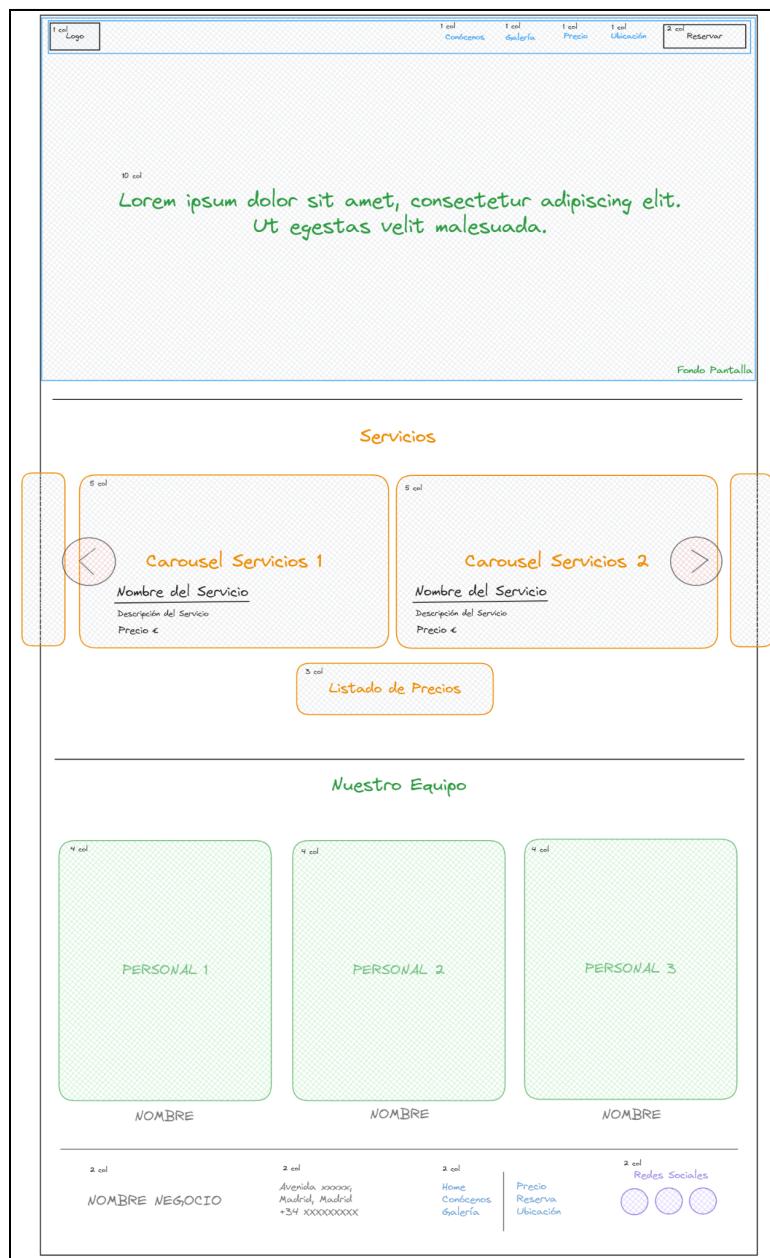
MAPA

Nombre Apellido Calle Localización  Home Correo  Preco Reserva  Redes   
Número Gómez, Gómez, Gómez, Ubicación  000

## 6.2. Wireframes

Una vez finalizados los bocetos y captado el estilo y líneas generales que va a seguir el diseño, el siguiente paso es darle estructura. Esta estructura, aunque variable a futuro, tiene la misma función que tendría la creación de un plano en un diseño arquitectónico: otorga detalles sobre la implementación y la estructura.

Por esta razón y partiendo de los bocetos, a continuación se muestran los *wireframes* correspondientes a cada uno, detallando información fundamental para la esquemática y arquitectura de la web final.

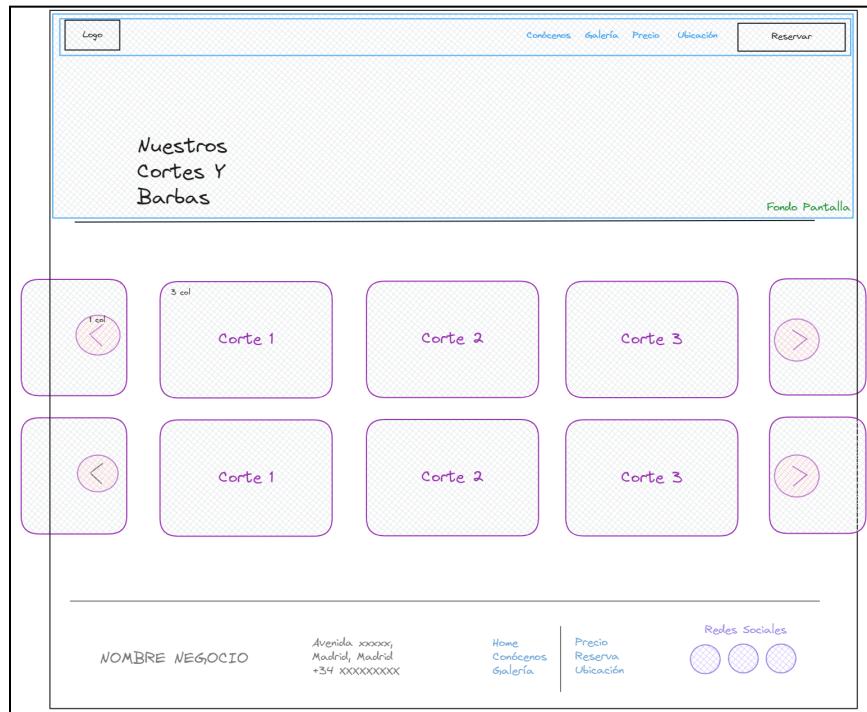


**Imagen 6.2.1. Wireframe Home**

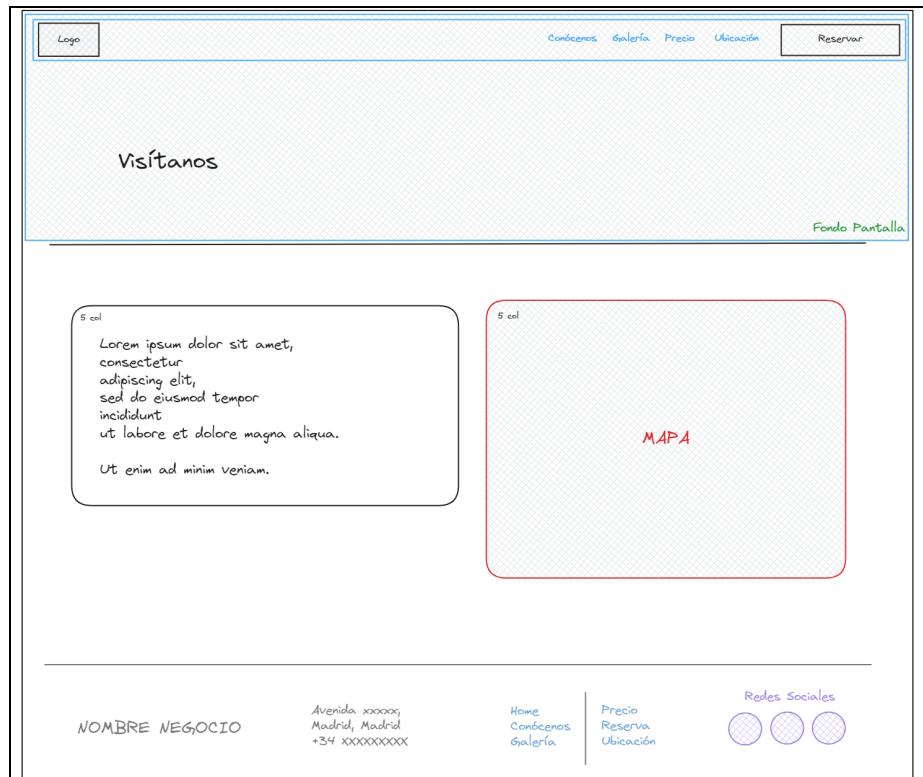
The wireframe for the Home page is structured as follows:

- Header:** Logo, Navegación (Conócenos, Galería, Precio, Ubicación), Reservar.
- Section 1: Quienes Somos**
- Section 2: LA BARBERÍA**
  - Left column (5 col): FOTO BARBERÍA
  - Right column (5 col):
    - Historia:** Placeholder text.
    - Photography by Aro Ha**
- Section 3: NUESTRO EQUIPO**
  - Left column (3 col): PERSONAL 1
  - Middle column (4 col):
    - Nombre Personal 1:** Placeholder text.
    - Photography by Aro Ha**
  - Right column (3 col): PERSONAL 2
  - Bottom row (3 col):
    - Nombre Personal 2:** Placeholder text.
    - Photography by Aro Ha**
    - PERSONAL 3**
  - Bottom row (3 col):
    - Nombre Personal 3:** Placeholder text.
    - Photography by Aro Ha**
    - PERSONAL 3**
- Footer:**
  - NOMBRE NEGOCIO
  - Avenida, 00000, Madrid, Madrid  
+34 XXXXXXXXXX
  - Links: Home, Conócenos, Galería, Precio, Reserva, Ubicación
  - Redes Sociales: Placeholder for social media icons.

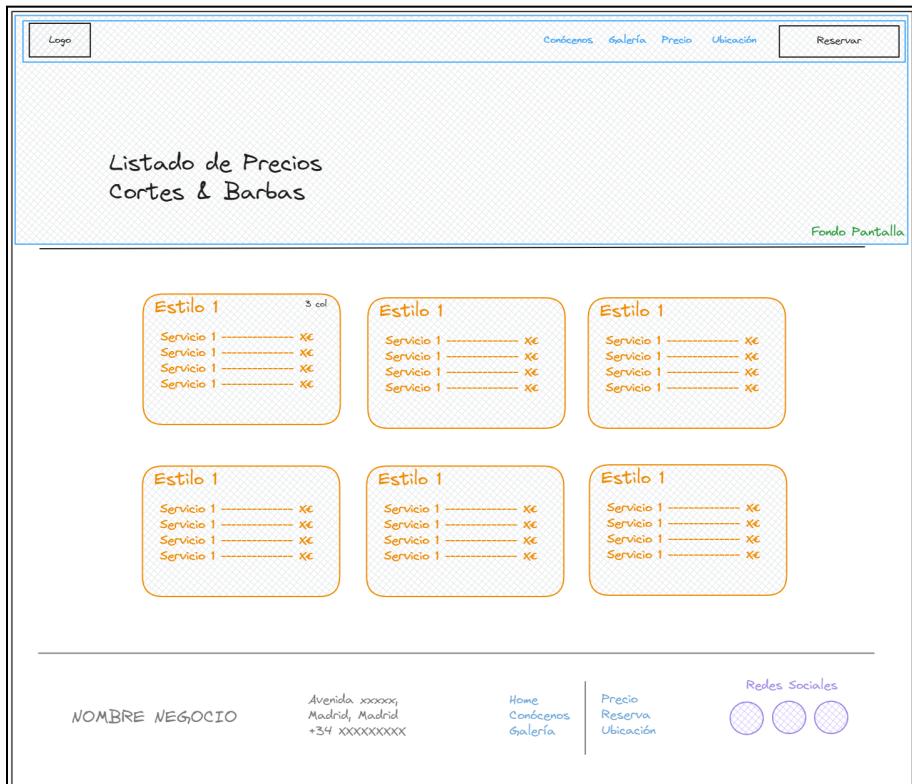
**Imagen 6.2.2. Wireframe Conócenos**



**Imagen 6.2.3. Wireframe Galería**



**Imagen 6.2.4. Wireframe Ubicación**



*Imagen 6.2.5. Wireframe Precios*

The wireframe illustrates the layout of a reservation page. At the top, there is a header bar with a logo, navigation links (Conócenos, Galería, Precio, Ubicación), and a 'Reservar' button. Below the header, the main content area features a title 'Haz tu Reserva' and a background image labeled 'Fondo Pantalla'. The central part of the page contains two main sections: a contact information section on the left and a reservation form on the right.

**Contact Information Section (Left):**

- Section title: '5 col'
- Text area: 'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.'
- Text below: 'Información de Contacto'

**Reservation Form Section (Right):**

- Section title: '5 col 6 col'
- Text area: 'Con quién te cortas?'
- List: 'Barbero 1', 'Barbero 2', 'Barbero 3'
- Text area: 'Qué día te viene bien?'
- Date input: '01/01/1970'
- Text area: 'Qué hora te va bien?'
- Time input: '00:00'
- Text area: 'Cómo te llamas?'
- Text input: 'Nombre' (15 col)
- Text input: 'Apellidos' (2.5 col)
- Text area: 'Reservar' (2 col)
- Text below: 'Formulario'

**Page Footer:**

- Text: 'NOMBRE NEGOCIO'
- Text: 'Avenida xxxxxx, Madrid, Madrid +34 XXXXXXXXX'
- Links: 'Home', 'Conócenos', 'Galería', 'Precio', 'Reserva', 'Ubicación'
- Text: 'Redes Sociales' followed by three circular icons.

*Imagen 6.2.6. Wireframe Reserva*

## 6.3. MockUps

Finalmente los mockups tienen como finalidad reflejar la web final de forma prematura al realizar la implementación. Esto no solo permite un desarrollo de *observar y replicar* si no que nos permite validar, mejorar o cambiar el diseño sin tener que realizar la implementación.

Es de notar que para la realización de los mockups empleamos la herramienta externa de creación de páginas web sin código **squarespace**.

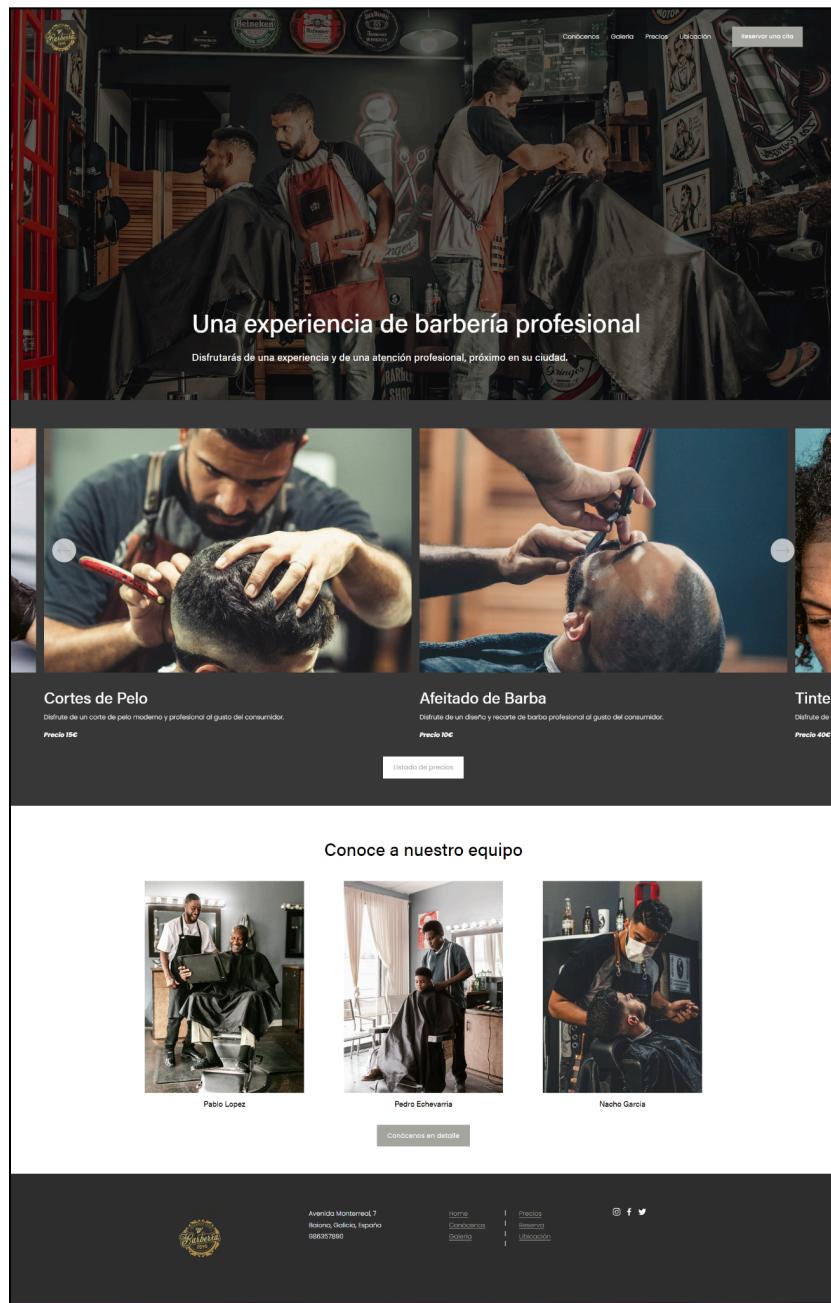


Imagen 6.3.1. MockUp Home



## Quienes somos

## LA BARBERIA



**Historia**

Fundada en el año 2023, VvBarbería se erige como un santuario para aquellos que buscan más que un simple corte de pelo; es un lugar donde la destreza artesanal y la atención personalizada convergen para crear looks excepcionales.

En VvBarbería, nuestra misión es más que solo cortar el cabello; se trata de resaltar la individualidad de cada cliente, restaurando su estilo personal con cortes y acabados precisos. Nuestro equipo de barberos altamente capacitados no solo son expertos en los últimos tendencias, sino que también abrazan la herencia de la barbería clásica, ofreciendo servicios que combinan lo moderno con lo atemporal.

**Pablo Perez**



Es un talentoso y apasionado barbero en VvBarbería. Con una habilidad excepcional para crear cortes de cabello y estilos modernos, Pablo se destaca por su atención meticulosa a los detalles y su enfoque personalizado para satisfacer las necesidades de cada cliente. Su amabilidad y profesionalismo crean un ambiente acogedor en la barbería, donde los clientes disfrutan de una experiencia única. Con años de experiencia en la industria, Pablo Pérez es un activo invaluable para VvBarbería, brindando servicios de calidad y dejando a sus clientes con una sensación de confianza y estilo renovado.

**Pedro Echevarría**



Pedro López, un talentoso barbero en VvBarbería, se distingue por su creatividad y pasión por la estilización capilar. Con un enfoque artístico, Pedro transforma el cabello de sus clientes en obras maestras personalizadas, fusionando las últimas tendencias con toques únicos. Su habilidad para comprender las preferencias individuales y adaptar cada corte a la personalidad de quien lo lleva, crea una experiencia auténtica para cada cliente. La energía positiva y el trato amigable de Pedro hacen que la visita a VvBarbería sea más que una simple cita de belleza, convirtiéndola en una experiencia memorable. Con una sólida trayectoria en la industria, Pedro López es un colaborador valioso que eleva el estándar de la barbería, ofreciendo no solo servicios de alta calidad, sino también un toque distintivo que refleja su destreza y estilo único.

**Nacho García**



Nacho García, un talentoso artista de la barbería en VvBarbería, se destaca por su enfoque vanguardista y su creatividad sin límites. Con una habilidad excepcional para fusionar las últimas tendencias con su propia toque distintivo, Nacho transforma cada corte de cabello en una expresión única de estilo. Su atención personalizada a las necesidades y preferencias de sus clientes crea una conexión auténtica, convirtiendo cada visita a VvBarbería en una experiencia personalizada y memorable. Nacho no solo aporta su destreza técnica, sino también una energía vibrante y amigable que contribuye a un ambiente acogedor en la barbería. Con una sólida experiencia en la industria, Nacho García es un miembro esencial del equipo, elevando el estándar de la barbería y dejando a sus clientes no solo con un nuevo look, sino con una sensación renovada de confianza y estilo.



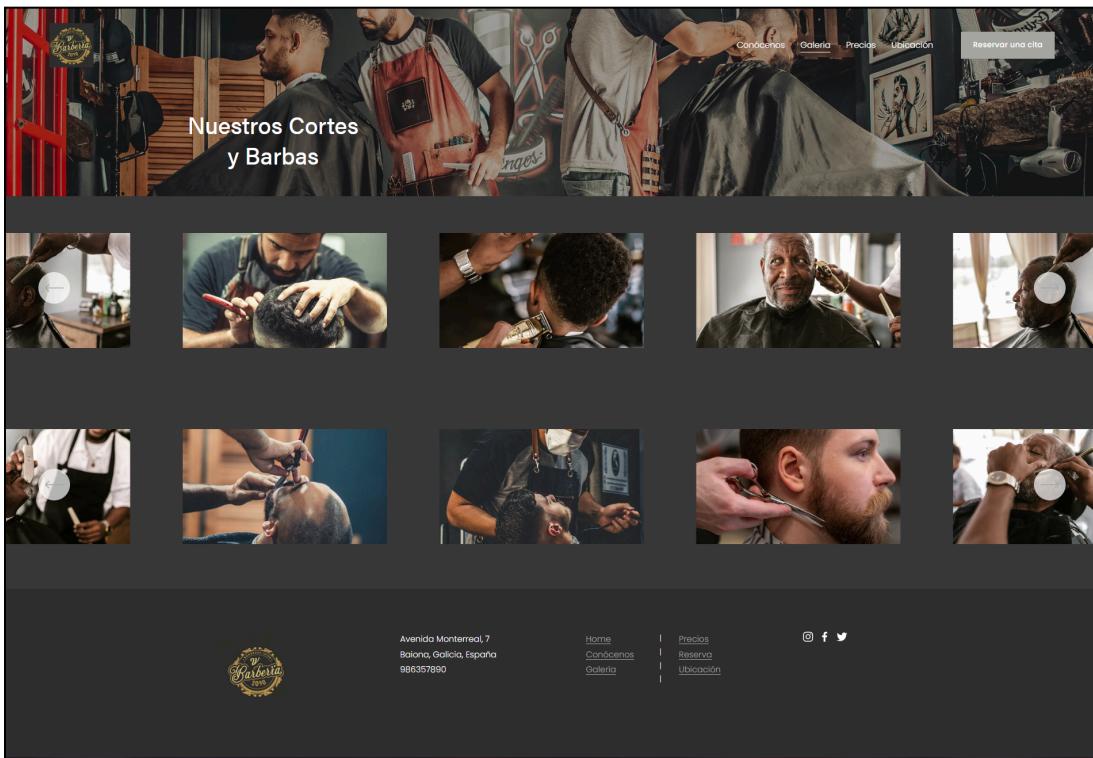
Avenida Monterrey, 7  
 Babia, Galicia, España  
 98257860

[Home](#) | [Conócenos](#) | [Reserva](#) | [Precios](#) | [Ubicación](#)

[Reservar una cita](#)

[Facebook](#) | [Twitter](#)

*Imagen 6.3.2. MockUp Conocenos*

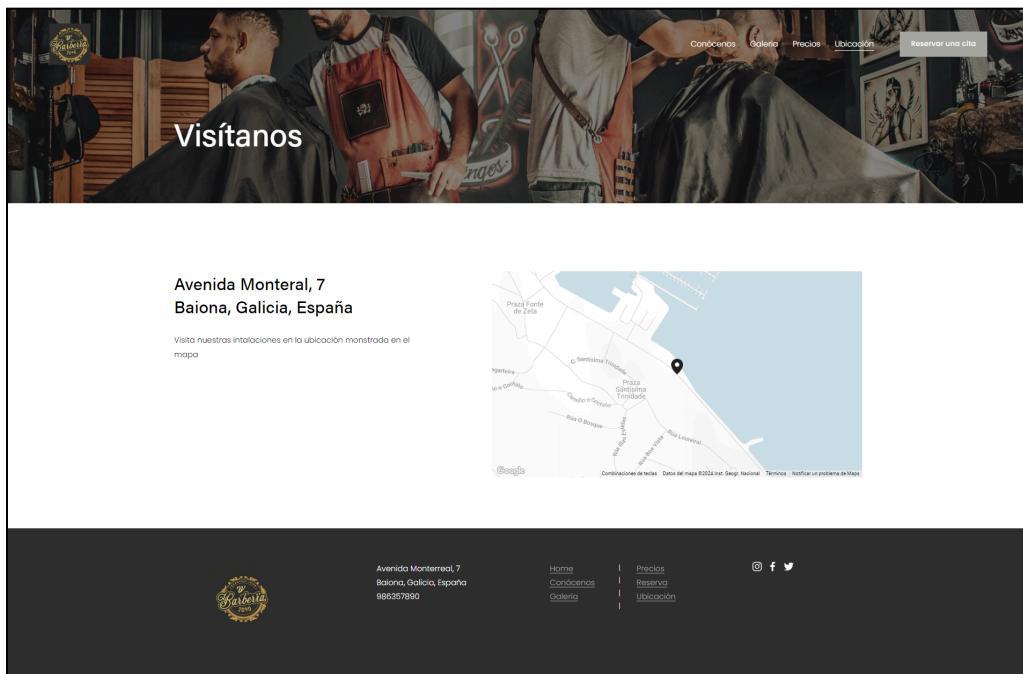


*Imagen 6.3.3. MockUp Galería*

CORTES DE PELO		DEGRADADOS		CORTE PARA NIÑOS	
Corte de pelo estandar	15€	Fade	15€	Menores de 5	7€
Corte Rapado	10€	Tupper Fade	18€	Menores de 10	10€
Corte y Lavado	17€	Skin Fade	18€	Menores de 12	12€
Cambio de look	20€	Fade con diseño	20€		

RECORTES DE BARBA		AFEITADOS		TINTES	
Recorte de bigote	7€	Afeitado de bigote	7€	Decoloración	30€
Recorte de barba	10€	Afeitado de barba	10€	Tinte	30€
Recorte de barba y tratamiento	18€	Afeitado de cabeza	15€		

*Imagen 6.3.4. MockUp Precios*



*Imagen 6.3.5. MockUp Ubicación*

Le proporcionamos los siguientes datos para que pueda contactar con nosotros:

**Dirección:**  
Avenida Monterreal, 7 Baiona, Galicia, España

**Horario**  
lunes-viernes  
10 a.m.-6 p.m.

**Teléfono**  
986357890

**1. Con quien te cortas el pelo?**

Pablo Lopez  
 Pedro Echevarria  
 Nacho Garcia

**2. Qué día te viene bien?**  
 dd/mm/aaaa

**3. A que hora te viene bien?**  
 \_\_\_\_\_   
en hora de Europa central

**4. Como te llamas?**  
 Nombre \_\_\_\_\_ Apellidos \_\_\_\_\_

*Imagen 6.3.6. MockUp Reserva*

## 7. Storyboards

En este apartado se representará el camino que hay que llevar en la web para poder realizar los casos de uso del diagrama de casos de uso especificados.

### 7.1. Consultar Precios

En el caso de consultar precios para poder acceder a la sección con los servicios disponibles y sus precios debemos de realizar los siguientes clicks en la página:

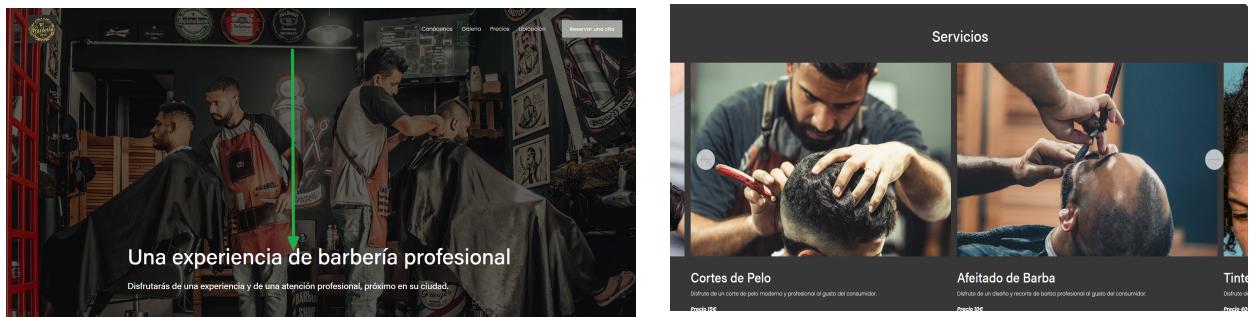
The storyboard illustrates the user journey for viewing price lists. It begins with a screenshot of the website's homepage, which includes a banner for 'Listado precios Cortes y barbas' (Price List Cuts and Beards). A red arrow points from this banner down to a second screenshot, which shows a detailed list of services and their corresponding prices. This list is organized into several categories: Cortes de Pelo, Degradados, Corde para Niños, Recortes de Barba, Afeitados, and Tintes. Each category contains specific service names and their prices.

CATEGORÍA	SERVICIO	PRECIO
CORTES DE PELO	Corte de pelo estandar	15€
	Corte Rapido	10€
	Corte y Lavado	17€
	Cambio de look	20€
DEGRADADOS	Fade	15€
	Tupper Fade	16€
	Skin Fade	18€
CORTE PARA NIÑOS	Menores de 5	7€
	Menores de 10	10€
	Menores de 12	12€
RECORTES DE BARBA	Recorte de bigote	7€
	Recorte de barba	10€
	Recorte de barba y tratamiento	18€
AFEITADOS	Afeitado de bigote	7€
	Afeitado de barba	10€
	Afeitado de cabeza	15€
TINTES	Decoloración	30€
	Tinte	30€

En el caso de este storyboard después de la realización de la página web, se ha mejorado el diseño haciendo una sección diferente con una foto para cada servicio por lo que el usuario debería scrollear hasta la sección de los servicios deseados.

## 7.2. Ver Cortes

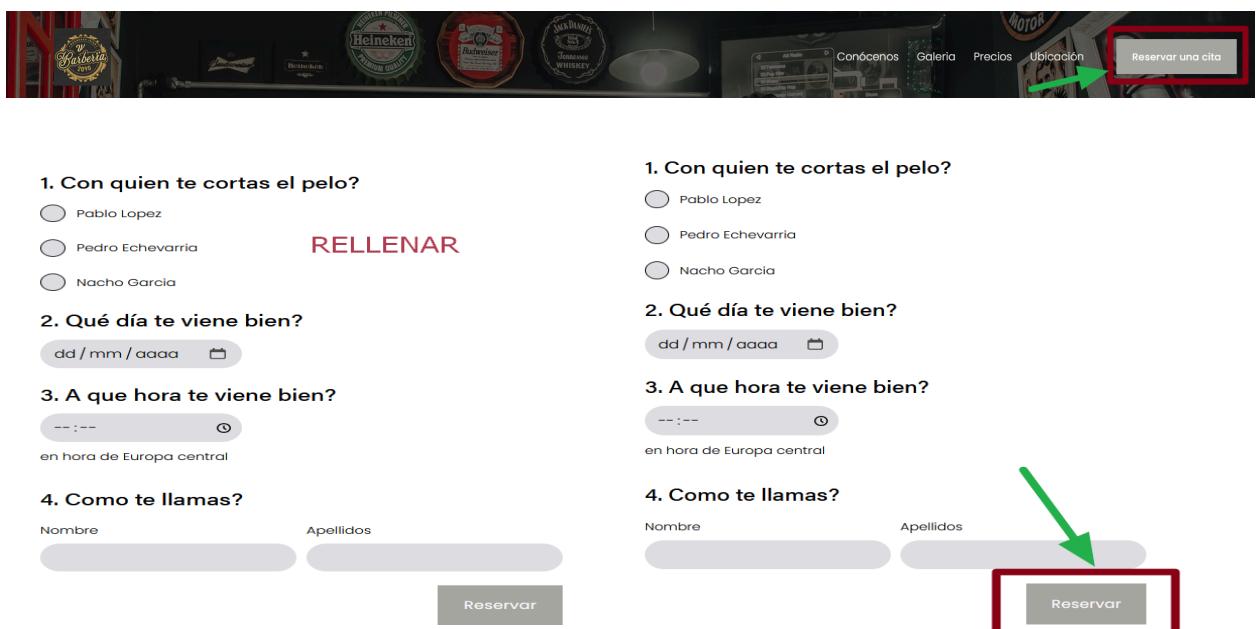
Para ver los cortes una vez en la página principal, el usuario debe de scrollar hasta el carrusel donde se muestran los diferentes servicios disponibles:



En este caso después de la realización de la página web una vez nos encontramos en el apartado de servicios, nos podemos desplazar por el carrusel y en caso de clicar en alguno de los servicios el usuario será redireccionado a la ventana de precios centrado en el servicio clicado

## 7.3 Hacer Reserva

Para realizar una reserva, en la página principal el usuario debe de ir al apartado de reservar una cita y será redireccionado a la página de reservas donde encontrará un formulario que el usuario debe cubrir y una vez cubierto deberá darle a enviar.



1. Con quien te cortas el pelo?

Pablo Lopez  
 Pedro Echevarria  
 Nacho Garcia

**RELEÑAR**

2. Qué día te viene bien?

dd / mm / aaaa

3. A que hora te viene bien?

-- : --

en hora de Europa central

4. Como te llamas?

Nombre   
Apellidos

**Reservar**

1. Con quien te cortas el pelo?

Pablo Lopez  
 Pedro Echevarria  
 Nacho Garcia

2. Qué día te viene bien?

dd / mm / aaaa

3. A que hora te viene bien?

-- : --

en hora de Europa central

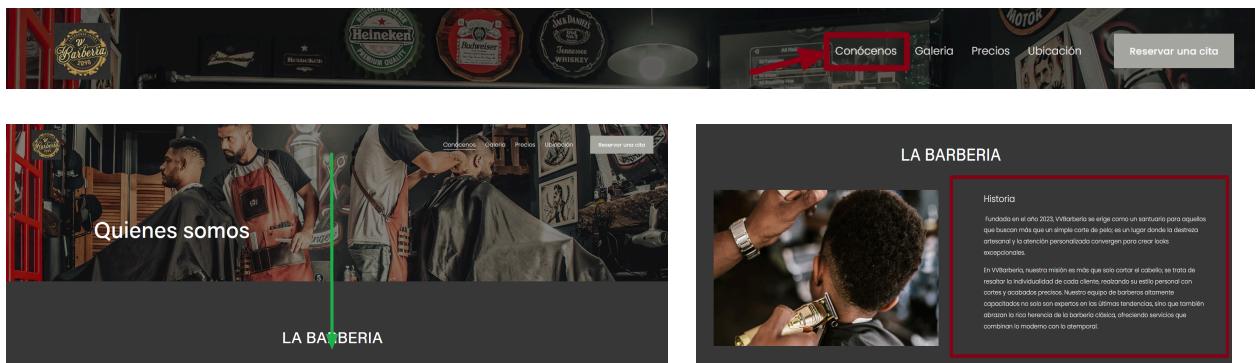
4. Como te llamas?

Nombre   
Apellidos

**Reservar**

## 7.4 Consultar Historia

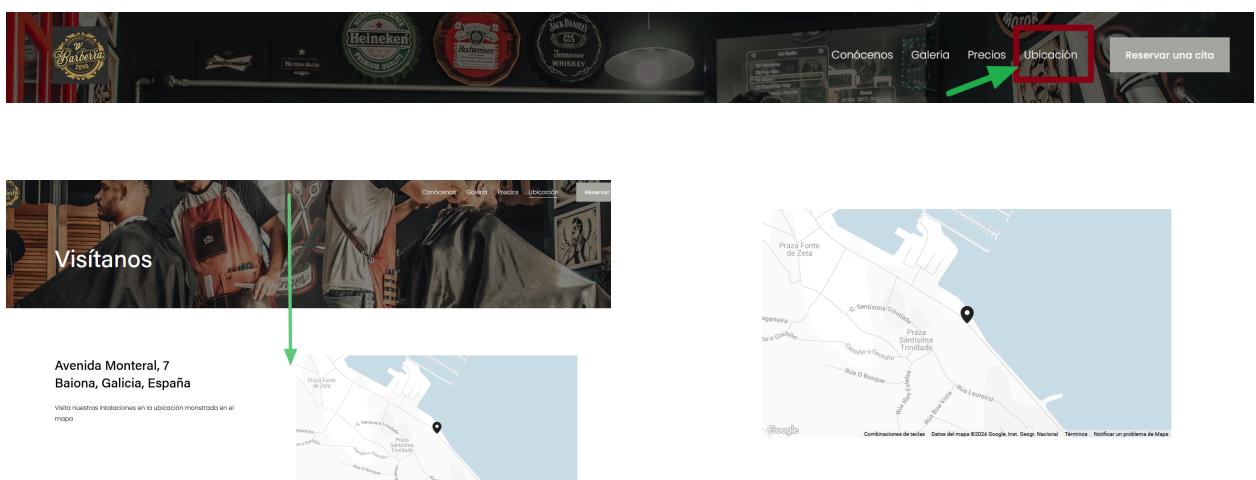
Para consultar la historia de la barbería el usuario debe de seleccionar en la barra de navegación la página de conócenos y finalmente scrollear en esta página hasta el apartado de historia.



The first screenshot shows the website's header with various brand logos (Heineken, Budweiser, Jack Daniels) and a navigation bar with links: 'Conócenos' (highlighted with a red box and arrow), 'Galería', 'Precios', 'Ubicación', and 'Reservar una cita'. The second screenshot is a wireframe showing a banner with the text 'Quienes somos' and 'LA BARBERIA'. The third screenshot is a wireframe of the 'Historia' section, which contains a paragraph about the barbershop's mission and history.

## 7.5 Obtener Localización

Para obtener la localización de la barbería, es necesario seleccionar en la barra de navegación la página de ubicación y en el wireframe proporcionado, estará la ubicación de la barbería.



The first screenshot shows the website's header with various brand logos and a navigation bar with links: 'Conócenos', 'Galería', 'Precios', 'Ubicación' (highlighted with a green arrow), and 'Reservar una cita'. The second screenshot is a wireframe showing a banner with the text 'Visítanos' and a map of Baiona, Galicia, Spain. The third screenshot is a wireframe of a Google Map showing the location of the barbershop on Avenida Monreal, 7.

## 8. Estructura de ficheros y carpetas

Cabe destacar que la estructura de ficheros y carpetas de la página ha cambiado notoriamente desde el inicio hasta la final realización de la práctica siendo en primer lugar la estructura de los archivos como se muestra en la siguiente figura:

```
VVBarberia/
└── public/
    ├── index.css
    ├── index.html
    ├── index.js
    └── media/
        ├── images/
        │   ├── barber1.png
        │   ├── barber2.png
        │   ├── barber3.png
        │   └── ...
        └── video/
            ├── intro.mp4
            └── ...
    └── pages/
        ├── conocenos.html
        ├── galeria.html
        ├── precios.html
        ├── reservar.html
        └── ubicacion.html
    └── styles/
        └── main.css
```

Después de la inclusión de todos los archivos HTML, CSS y JavaScript, y de además de la realización de los añadidos que no estaban pensados en un primer lugar obtenemos que la estructura de los ficheros a avanzado de la forma en la que se aprecia en la siguiente figura:

Se muestra la estructura de archivos tal cual se encuentra en el repositorio de github.

Primero de todo mencionar los archivos independientes:

- Barber\_Document.pdf. Es este mismo documento, presente en la carpeta para aplicarle el controlado de versiones.
- LICENSE. La licencia MIT aplicable a la web desarrollada
- README.md. Archivo empleado por el equipo de desarrollo con información relevante a este y gestión de tareas internas
- Docker-compose.yaml. Archivo en formato yaml con la especificación del despliegue del servidor web Nginx empleado en la web mediante docker. Su contenido, funcionamiento y utilidad se explica en detalle en [11.1 Servidor Nginx](#)

El código y assets relativos a la web se encuentran ubicados en la carpeta de *src*. Esta carpeta contiene:

- **index.html**. Documento raíz html correspondiente a la página home.
- **index.css**. Documento con los estilos css globales, utilizado en todas las páginas de la web.
- Directorio **data**. Este directorio contiene los archivos xml y json correspondientes a los datos de las horas de apertura del local y el catálogo de precios respectivamente.
- Directorio **media**. Contiene los *assets* empleados en las distintas páginas de la web categorizadas en vídeo e imágenes y estas a su vez divididas en el tipo de imagen. A mayores existen las imágenes del logo de la web, y el background de la portada.

```
VVBarberia/
├── Barber_Document.pdf
├── docker-compose.yaml
├── LICENSE
└── README.md
└── src/
    ├── data/
    │   ├── hours.xml
    │   └── prices.json
    ├── index.css
    ├── index.html
    └── media/
        ├── images/
        │   ├── background.jpg
        │   ├── barbas/
        │   │   ├── barba_001.jpg
        │   │   └── ...
        │   ├── barbers/
        │   │   ├── nacho_garcia.jpg
        │   │   └── ...
        │   ├── cortes/
        │   │   ├── cortes_000.jpg
        │   │   └── ...
        │   ├── logos/
        │   │   ├── instagram.png
        │   │   └── ...
        │   ├── logo.svg
        │   └── services/
        │       ├── service_000.png
        │       └── ...
        └── video/
            └── background.mp4
```

- Directorio **pages**. Aquí se encuentran los documentos *html* correspondientes a las distintas páginas de la web y el código correspondiente al *navbar* y *footer* que se cargan mediante *JQuery* en el resto de páginas.
- Directorio **styles**. Contiene un archivo de estilo *css* por cada archivo *html* del directorio *pages*.
- Directorio **js**. Contiene el código *javascript* empleado en las distintas páginas. Cabe destacar los archivos *toggle.js*, *navbar.js* y *footer.js* que no corresponden a ninguna página si no a componentes de ella.

```
VVBarberia/
└── ...
   └── src/
      ├── ...
      └── js/
          ├── galeria.js
          ├── navbar.js
          ├── precios.js
          ├── reserva.js
          └── toggle.js
      └── pages/
          ├── conocenos.html
          ├── footer.html
          ├── galeria.html
          ├── navbar.html
          ├── precios.html
          ├── reserva.html
          └── ubicacion.html
      └── styles/
          ├── conocenos.css
          ├── footer.css
          ├── galeria.css
          ├── home.css
          ├── navbar.css
          ├── precios.css
          ├── reserva.css
          └── ubicacion.css
```

## 9.HTML

Una vez realizados los bocetos y recopilada la información necesaria para la realización de la página web, se comenzó con el primer proceso encargado de darle la estructura a nuestra página web, todo esto realizado mediante el uso de HTML.

Para ello se realizó un diseño las más parecido posible a los diseños y a los mockups realizados encontrando en las primeras sesiones de clases la siguiente disposición del HTML que se muestra mediante un mapa de etiquetas sobre cada mockup:

The screenshot displays the Barbería website with several handwritten annotations:

- Header:** A large image of a barbershop interior with two men getting haircuts. The word "Header" is written above the image.
- Section:** A green box highlights the main heading "Una experiencia de barbería profesional". Inside this box, the word "hgroup" is written above "h1". Below "h1" is the text "Disfrutarás de una experiencia v de una atención profesional. próximo en su ciudad." followed by a price icon "P".
- Section:** A blue box highlights the "Servicios" section. Inside, there are two images: one of a barber cutting hair and another of a barber shaving a beard. The word "Section" is written above the second image.
- Section:** A red box highlights the "Cortes de Pelo" service. It shows a barber cutting hair with clippers. The word "li" is written above the image. Below the image is the service name "Cortes de Pelo" and a brief description: "Disfrute de un corte de pelo moderno y profesional al gusto del consumidor.". A price icon "P" and the price "Precio 15€" are shown.
- Section:** An orange box highlights the "Afeitado de Barba" service. It shows a barber shaving a beard. The word "li" is written above the image. Below the image is the service name "Afeitado de Barba" and a brief description: "Disfrute de un diseño y recorte de barba profesional al gusto del consumidor.". A price icon "P" and the price "Precio 10€" are shown.
- Section:** A green box highlights the "Conoce a nuestro equipo" section. It features three images of barbers: Pablo Lopez, Pedro Echevarria, and Nacho Garcia. The word "Section" is written above the third image.
- Section:** A grey box at the bottom left contains the text "Conócenos en detalle".
- Footer:** The footer includes the logo "Sigure", the address "Avenida Monterreal, 7 Balona, Gótica, España 986357990", a menu with "Home", "Conócenos", "Galería", "Precios", "Reserva", and "Ubicación", social media links for "Instagram", "Facebook", and "Twitter", and the word "Footer".



**Quienes somos**

## LA BARBERIA

**Sigüe**



**Historia**

Fundada en el año 2023, VVBarbería se erige como un santuario para aquellos que buscan más que un simple corte de pelo; es un lugar donde la destreza artesanal y la atención personalizada convergen para crear looks excepcionales.

En VVBarbería, nuestra misión es más que solo cortar el cabello; se trata de resaltar la individualidad de cada cliente, realizando su estilo personal con cortes y acabados precisos. Nuestro equipo de barberos altamente capacitados no solo son expertos en las últimas tendencias, sino que también abrazan la rica herencia de la barbería clásica, ofreciendo servicios que combinan lo moderno con lo atemporal.

## Nuestro Equipo

**ul**

**Pablo Perez**



**P**

Es un talentoso y apasionado barbero en VVBarbería. Con una habilidad excepcional para crear cortes de cabellera y estilos modernos, Pablo se destaca por su atención meticulosa a los detalles y su enfoque personalizado para satisfacer las necesidades de cada cliente. Su amabilidad y profesionalismo crean un ambiente acogedor en la barbería, donde los clientes disfrutan de una experiencia única. Con años de experiencia en la industria, Pablo Pérez es un activo invaluable para VVBarbería, brindando servicios de calidad y dejando a sus clientes con una sensación de confianza y estilo renovado.

**li**

**Pedro Echevarría**

**h2**

**P**

Pedro López, un talentoso barbero en VVBarbería, se distingue por su creatividad y pasión por la estilización capilar. Con un enfoque artístico, Pedro transforma el cabello de sus clientes en obras maestras personalizadas, fusionando las últimas tendencias con toques únicos. Su habilidad para comprender las preferencias individuales y adaptar cada corte a la personalidad de quien lo lleva, crea una experiencia auténtica para cada cliente. La energía positiva y el trato encantador de Pedro hacen que la visita a VVBarbería sea más que una simple cita de belleza; convertiéndola en una experiencia memorable. Con una sólida trayectoria en la industria, Pedro López es un colaborador valioso que eleva el estándar de la barbería, ofreciendo no solo servicios de alta calidad, sino también un toque distintivo que refleja su destreza y estilo único.

**Sigüe**



**Nacho García**

Nacho García, un talentoso artista de la barbería en VVBarbería, se destaca por su enfoque vanguardista y su creatividad sin límites. Con una habilidad excepcional para fusionar las últimas tendencias con su propio toque distintivo, Nacho transforma cada corte de cabello en una expresión única de estilo. Su atención personalizada a las necesidades y preferencias de sus clientes crea una conexión auténtica, convirtiendo cada visita a VVBarbería en una experiencia personalizada y memorable. Nacho no solo apunta su destreza técnica, sino también una energía vibrante y amigable que contribuye a un ambiente acogedor en la barbería. Con una sólida experiencia en la industria, Nacho García es un miembro esencial del equipo, elevando el estándar de la barbería y dejando a sus clientes no solo con un nuevo look, sino con una sensación renovada de confianza y estilo.

Nuestros Cortes y Barbas

Avenida Monterreal, 7  
Baiona, Galicia, España  
986357890

Home | Conócenos | Galería | Precios | Reserva | Ubicación

Instagram | Facebook | Twitter



h2

Avenida Monteral, 7  
Baiona, Galicia, España

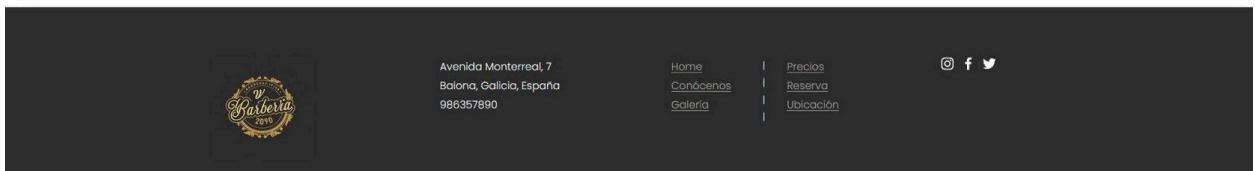
div

Visita nuestras instalaciones en la ubicación mostrada en el mapa

P

section

iframe





hz

Le proporcionamos los siguientes datos para que pueda contactar con nosotros:

de

dirección: Avenida Monterreal, 7 Baiona, Galicia, España

Horario: Lunes-viernes 10 a. m.-6 p. m.

Teléfono: 986357890

aside

section

1. Con quien te cortas el pelo?

Pablo Lopez

Pedro Echevarria

Nacho Garcia

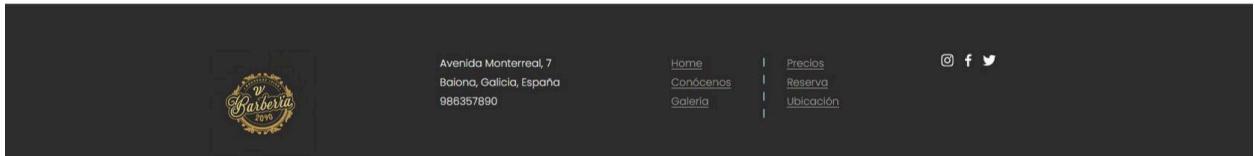
2. Qué día te viene bien?  dd/mm/aaaa  geldset

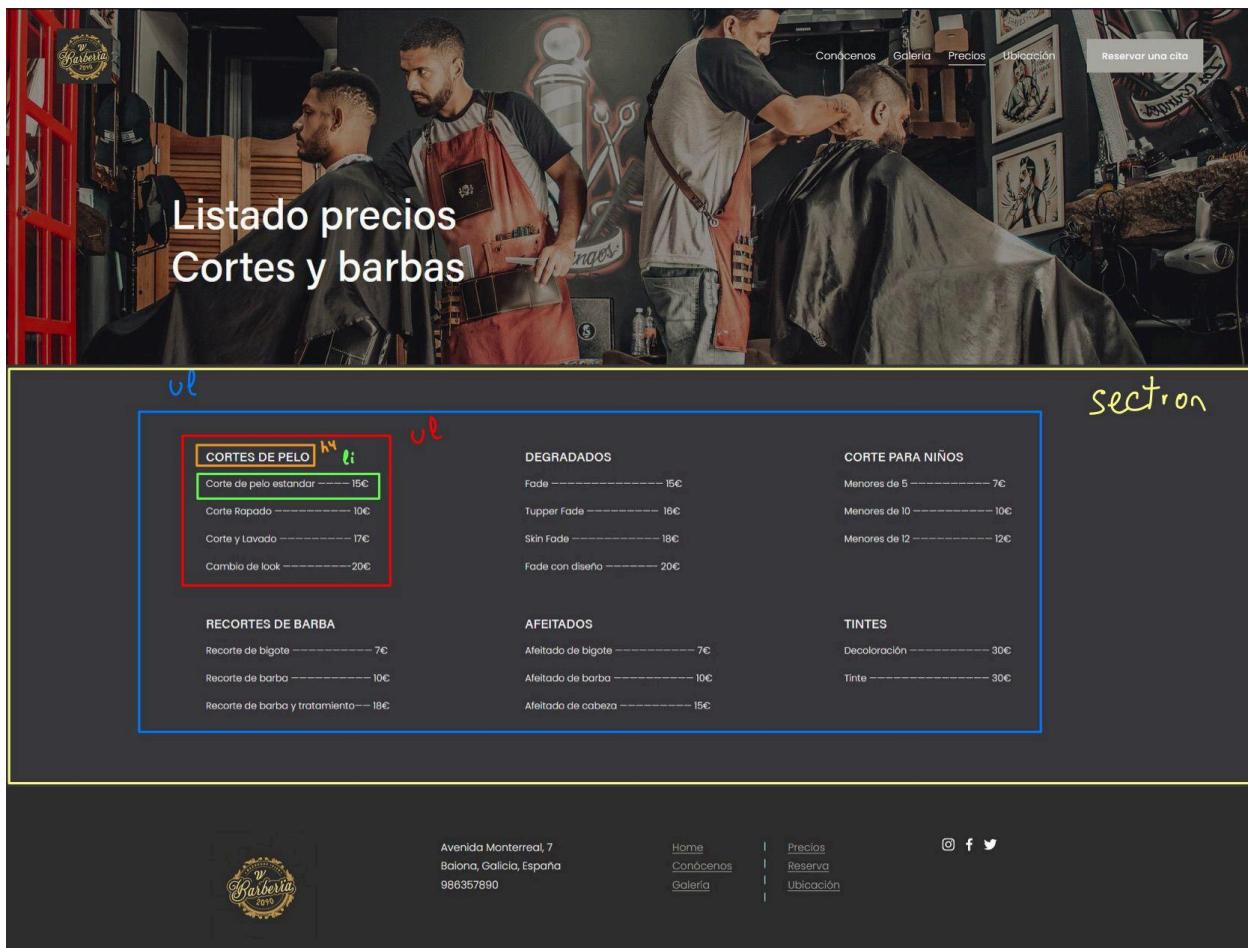
3. A que hora te viene bien?

--:--  en hora de Europa central

4. Como te llamas?

Nombre  Apellidos





The screenshot shows the 'Precios' (Prices) section of the Barbería website. The page features a large background image of a barber working on a client's hair. At the top right, there are navigation links: 'Conócenos', 'Galería', 'Precios', 'Ubicación', and a 'Reservar una cita' button. Below the image, the title 'Listado precios' and 'Cortes y barbas' is displayed. The main content area is divided into several sections:

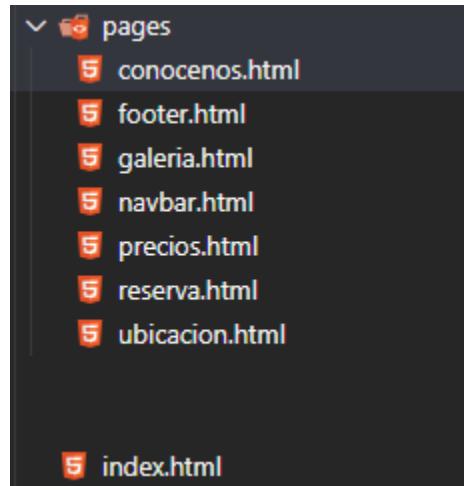
- CORTES DE PELO**
  - Corte de pelo estandar — 15€
  - Corte Rapado — 10€
  - Corte y Lavado — 17€
  - Cambio de look — 20€
- DEGRADADOS**
  - Fade — 15€
  - Tupper Fade — 16€
  - Skin Fade — 18€
  - Fade con diseño — 20€
- CORTE PARA NIÑOS**
  - Menores de 5 — 7€
  - Menores de 10 — 10€
  - Menores de 12 — 12€
- RECORTES DE BARBA**
  - Recorte de bigote — 7€
  - Recorte de barba — 10€
  - Recorte de barba y tratamiento — 18€
- AFEITADOS**
  - Afeitado de bigote — 7€
  - Afeitado de barba — 10€
  - Afeitado de cabeza — 15€
- TINTES**
  - Décoloración — 30€
  - Tinte — 30€

At the bottom of the page, there is a footer with the Barbería logo, address (Avenida Monterreal, 7, Balona, Galicia, España, 986357890), and links to 'Home', 'Conócenos', 'Galería', 'Precios', 'Reserva', and 'Ubicación'. Social media icons for Instagram, Facebook, and Twitter are also present.

Las imágenes anteriores se realizaron en las primeras sesiones de clase cuando el objetivo solo era crear documentos HTML que servirían como soporte de la web. Sin embargo dado que durante el desarrollo se han realizado diversos cambios, debido a añadidos, o a cambios hacia una forma más correcta de representación del código estos mapas de etiquetas han cambiado significativamente.

La gran mayoría de los cambios se centran en la incorporación de etiquetas como 'div' que permiten la agrupación de etiquetas para un más sencillo estilado de la página.

Los archivos html creados para el desarrollo de la página web son los siguientes:



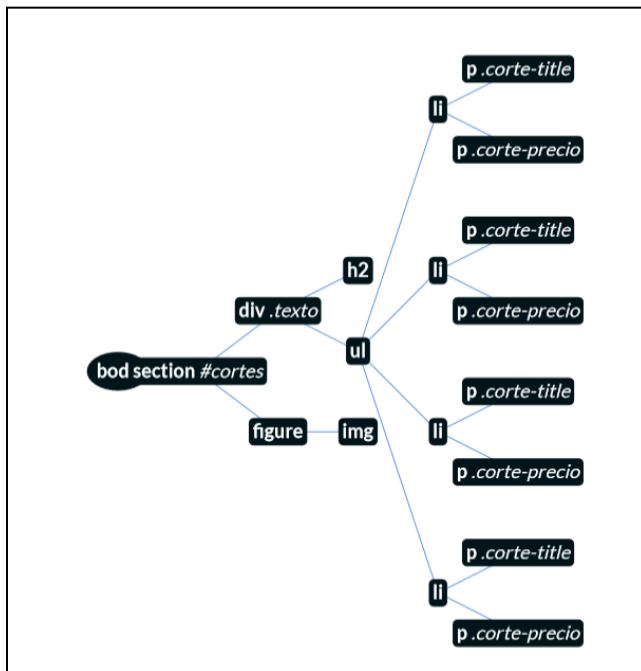
Como se puede apreciar tenemos el archivo index.html encargado de la realización de la página principal de nuestra página web y la carpeta pages, que contiene los archivos de las otras cinco páginas en cuestión. Además en esta carpeta también se añade el código html empleado para el header y el footer ya que este es el mismo en todas las páginas del sitio y es cargado en cada página dinámicamente mediante JQuery.

Además como se comentó anteriormente se realizaron diferentes cambios en algunas de las páginas como es en el caso de la galería en la que en primer lugar íbamos a emplear dos carruseles por lo que teníamos dos listas de elementos con imágenes, mientras que finalmente se optó por el empleo de una galería colocada a lo largo de la ventana mediante el empleo de css grid por lo que se emplearon en este caso divs con la imagen de fondo para una más fácil colocación de los elementos y de la responsividad.

También en el caso de precios, se optó por una diferente representación de las etiquetas empleando una sección diferente para cada apartado de precios. De esta forma obtenemos la siguiente disposición de etiquetas:

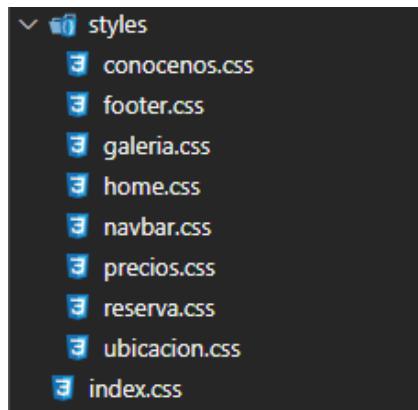
## 10.CSS

En este apartado se incluye la información más relevante sobre el proceso de estilado realizado en nuestra página.



Una vez se creó la estructura básica mediante el uso de HTML en este apartado mediante el empleo de las hojas de estilo CSS se proporcionó un estilo a la página haciendo que esta se pareciera lo máximo posible a los mockups realizados.

Las hojas de estilos creadas en esta página web son las que se muestran en la siguiente figura:



Como se puede apreciar en la figura, la hoja `index.css` es la hoja que proporciona un estilizado general para todas las páginas de la web y en el resto de archivos de la carpeta `conócenos` se realiza el estilizado de las páginas correspondientes así como del header y del footer.

Para la disposición de los elementos es necesario emplear la responsividad ya que nos permite que si se visualiza el sitio web en otros dispositivos de un menor tamaño como es el caso de los dispositivos móviles los elementos se coloquen de una manera adecuada.

Para la responsividad se emplearon diversas técnicas las cuales debían de ser empleadas una en cada una de las páginas de la web siendo estas técnicas las siguientes:

### 10.1. Navbar y Footer

Cabe destacar que como en el caso de el navbar y el footer se cargan dinámicamente en la página web y también se le cargan sus estilos, aunque estén en páginas en los que se emplean diferentes técnicas para la responsividad consideramos como casos aislados estos dos elementos, mientras que en el navbar empleamos un layout fijo para su disposición por toda la página, en el caso del footer empleamos flex container para su responsividad.

La navbar tiene dos estilos (cuya razón de ser se explica en [11.2. Tema de Navbar Dinámico](#)). Un estado inicial translúcido en la portada y uno con fondo claro en el resto de las páginas.



El código que permite el layout fijo se declara en la clase `vv-navbar` y su tema blanco en `nav-active` mediante el siguiente código `css`. Nótese el uso de la variable `css --color-light` para permitir la modificación rápida de los colores principales.

```
.vv-navbar {
    width: 100%;
    height: 100px;
    display: flex;
    justify-content: space-between;
    align-items: center;
    position: fixed;
    padding: 10px;
    z-index: 1000;
}

.nav-active {
    background-color: var(--color-light);
}
```

Además, de la navbar cabe destacar el enlace de *Hacer una Reserva* ya que está estilado de forma distinta que el resto de enlaces y también cuenta con un tema blanco.

```
#reserva a {  
    color: black;  
    background-color: var(--color-light);  
    padding: 5px 15px;  
    text-decoration: none;  
    border-radius: 20px;  
    border: none;  
    transition: transform 0.1s ease;  
}  
  
.nav-active #reserva > a:hover {  
    transform: scale(1.2);  
    color: var(--color-light);  
    background-color: black;  
}
```

Finalmente, cuando el tamaño de la pantalla es inferior a **768px** se activa la media query que, junto con un poco de javascript para permitir la acción *toggle* la navbar se convierte en la siguiente.

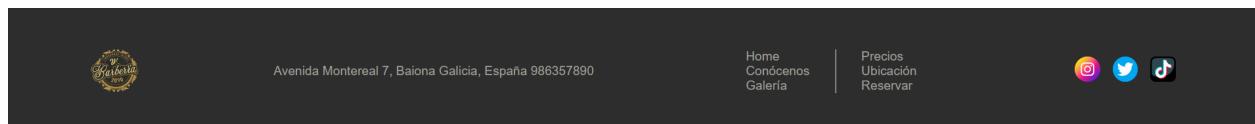


Para el footer, en nuestro contenedor flex tenemos las siguientes propiedades:

```
.footer {  
    background-color: #2e2e2d;  
    height: max(10rem, 20vh);  
    display: flex;  
    flex-direction: row;  
    justify-content: space-around;  
    align-items: center;  
    padding: 1rem;  
}
```

Se encarga de que todos los elementos del contenedor se coloquen mediante flex y además hacemos que la disposición de los elementos sea en columna mediante flex-direction: row. Además empleamos la propiedad align-items: center para alinear los elementos horizontalmente dentro de su espacio.

De esta manera obtenemos la siguiente disposición de elementos en pantalla normal:



Sin embargo, para realizar la responsividad de los dispositivos móviles empleamos una media query para que cuando el tamaño sea menor al especificado los elementos se coloquen mediante flex column y no mediante flex row, obteniendo así un diseño óptimo para los dispositivos móviles.

```
@media screen and (max-width: 768px) {  
    .footer {  
        flex-direction: column;  
        height: 100%;  
    }  
  
    .footer > * {  
        margin: 0.25rem 0;  
    }  
}
```

Con este código obtenemos el siguiente resultado:



## 10.2 Float

La técnica 'float' en CSS se utiliza para posicionar elementos en una página web, permitiendo que los elementos fluyan a su alrededor.

El uso de 'float' en esta web se encuentra en la página ubicación donde para colocar el iframe de la localización de la barbería a la derecha de la página:

Nuestras Instalaciones

Visita nuestras instalaciones en [Avenida Monreal, 7 en Baiona, España](#)

Lorem ipsum dolor sit amet consectetur adipisicing elit. Iusto aliiquid error ipsum, consequatur, neque temporibus inventore saepe aut maiores enim, in minima delectus repudiandae? Natus illo asperiores aspernatur odio qui.

Lorem ipsum dolor sit amet consectetur adipisicing elit. Iusto aliiquid error ipsum, consequatur, neque temporibus inventore saepe aut maiores enim, in minima delectus repudiandae? Natus illo asperiores aspernatur odio qui.

Lo que se realiza en este caso es realizar una disposición normal de los elementos cuando estamos en un tamaño menor a 960px y cuando este tamaño pasa a ser mayor que 960px mediante una media query hacemos que el iframe flote a la derecha

```
@media (min-width: 960px) {  
    main * {  
        max-width: 40vw;  
    }  
    main iframe {  
        float: right;  
    }  
  
    main div {  
        position: absolute;  
        top: 50%;  
        transform: translateY(-50%);  
    }  
}
```

Como se aprecia en la imagen cuando el tamaño de la ventana es mayor de 960px hacemos que el iframe flote a la derecha de sección del main.

De esta forma cuando el tamaño es mayor obtenemos el resultado de la imagen anterior y en caso de estar en menos de 960px obtenemos el siguiente resultado:



## 10.3 Multicol

Con `multicol` nos referimos a una técnica de CSS, utilizada para dividir el contenido de un elemento en múltiples columnas, facilitando la creación de layouts tipo periódico.

El uso de 'multicol' en esta web se encuentra en la página precios donde cada una de las secciones del body las dividimos en dos columnas teniendo de esta manera en una de las columnas el texto asociado a los precios y en la otra columna se representa la imagen asociada, obteniendo de esta forma la siguiente disposición:



En la imagen se muestra un ejemplo de dos de las secciones de la página y como influye el uso del multicol.

Para conseguir este efecto en primer lugar hacemos que el contenedor section tenga la propiedad column-count:2 y además hacemos que dentro del texto en cada uno de los li de la lista también aparezca esta propiedad para colocar los precios de los cortes todos en la misma columna y lo mismo para todos los servicios.

Para conseguir la responsividad en los dispositivos móviles empleamos una media query de manera que cuando el tamaño de la pantalla sea inferior a 767px la sección se represente solamente con una columna:

```
@media (max-width: 767px) {  
  
    section{  
        column-count: 1;  
        padding: 3rem 1rem 3rem 1rem;  
    }  
  
    .texto{  
        top:none;  
        transform: none;  
    }  
  
    section figure {  
        margin: 5% auto ;  
    }  
}
```

Mediante el uso de la media query obtenemos el siguiente resultado en la página:



CORTES DE PELO

<u>Corte de pelo básico</u>	15€
<u>Corte Rapado</u>	10€
<u>Corte y Lavado</u>	17€
<u>Cambio de look</u>	20€





DEGRADADOS

<u>Fade</u>	15€
<u>Tupper Fade</u>	16€
<u>Skin Fade</u>	18€
<u>Fade con diseño</u>	20€

## 10.4 CSS Grid

CSS Grid permite crear diseños complejos y versátiles con facilidad. Se activa usando `display: grid` en un contenedor, permitiendo definir filas y columnas y de esta forma disponer los contenedores dentro del número de filas y de columnas representadas.

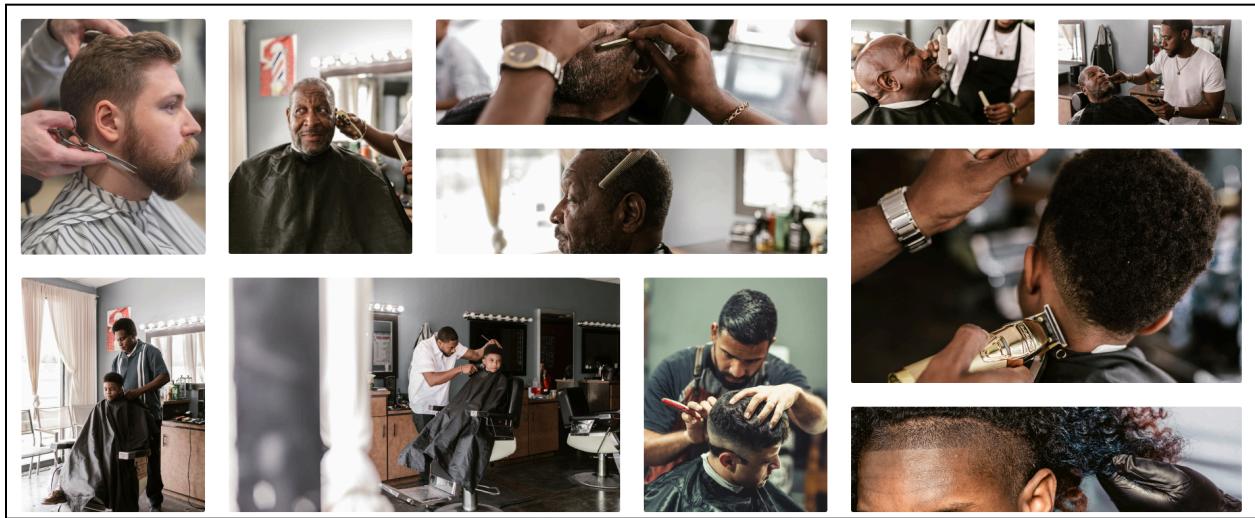
En el caso de nuestro sitio web, el grid se emplea en la página de galería donde mediante el siguiente código de css hacemos que el contenedor se divida en las filas y columnas especificadas:

```
main {  
    background-color: white;  
    display: grid;  
    grid-template-columns: repeat(auto-fill, minmax(320px, 1fr));  
    grid-auto-rows: minmax(200px, auto);  
    grid-auto-flow: dense;  
    gap: 45px;  
    padding: 30px;  
}
```

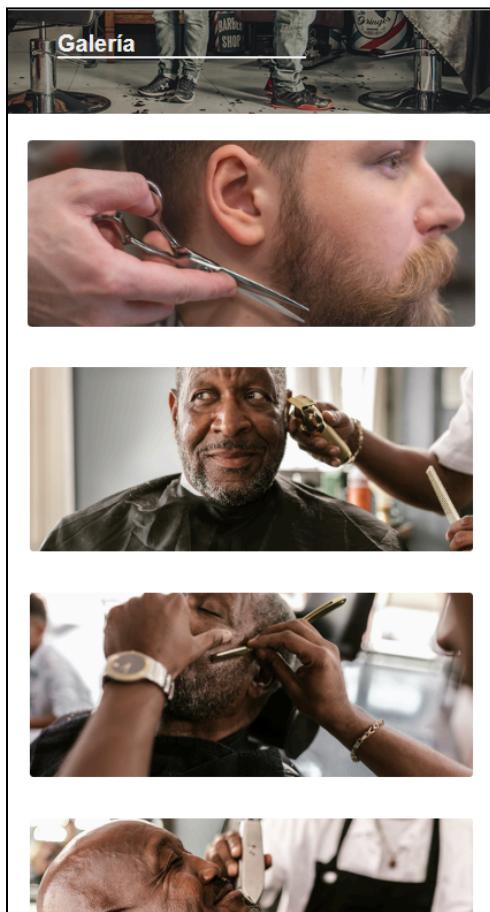
De esta forma con la propiedad grid-template-columns asignamos la plantilla de columnas que vamos a emplear en el contenedor, y en este caso hacemos que el contenedor se rellene de columnas con un tamaño mínimo de 320px y de máximo de 1fr. Hacemos mediante grid-auto-rows que se creen las filas necesarias y que estas tengan un tamaño mínimo de 200px de modo que mientras haya elementos estos se colocaran en la primera fila y cuando no entren en esta saltará a la segunda fila. Este comportamiento se consigue mediante el grid-auto-flow: dense.

Además se crearon dos clases alto y ancho dentro de una media query que se encargan de que cuando el tamaño de la pantalla es mayor a 600px los elementos que tengan la clase alto ocupen dos filas de alto y los que tengan la propiedad ancho ocupen dos columnas de ancho.

Este diseño con grid nos permite una responsividad automática y un resultado como el que se muestra en la siguiente figura:



En caso de estar en dispositivos móviles, las clases alto y ancho no funcionan y obtenemos el siguiente resultado:



## 10.5 Flex container

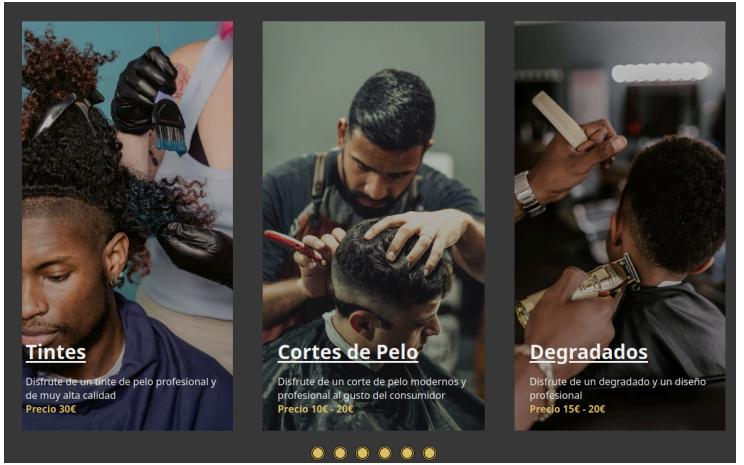
Los contenedores flex destacan por su facilidad de uso, *flexibilidad* y responsividad innata que los hacen del tipo de contenedor más popular a día de hoy.

Este contenedor se emplea en las páginas *home*, *conócenos* y en los componentes *footer* y *navbar*. Estos últimos componentes ya fueron explicados en la sección [10.1. Navbar y Footer](#)

En la página home, se destaca su uso en:

- El carrusel de servicios
- Las cartas de los miembros del equipo

La creación del carrusel de servicios mediante el uso de overflow y flex direction row sin wrapping en pantallas superiores a **768px** y su display en lista mediante el uso de wrap. Del propio carrusel también cabe destacar la gestión del scroll para que *snappee* las imágenes en el centro de la pantalla mediante los atributos css *scroll-snap-type* en el contenedor padre y *scroll-snap-align* en los hijos (los *service-cards*).



```
.carrusel {
    display: flex;
    width: 95vw;
    flex-flow: row nowrap;

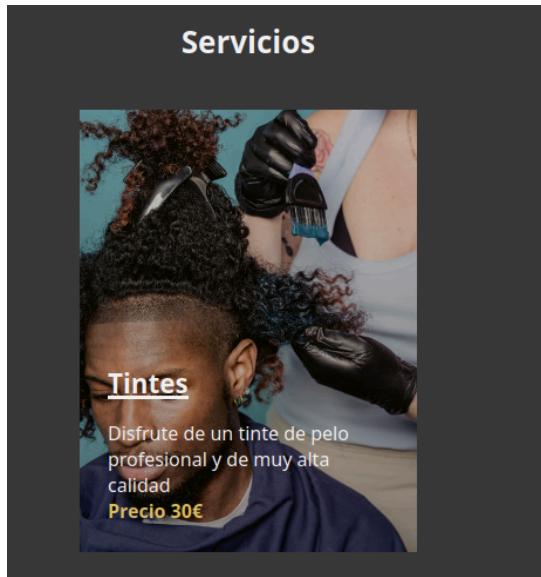
    /*overflow */
    overflow-x: auto;
    scroll-behavior: smooth;
    scroll-snap-type: x mandatory;
}

.service-card {
    width: 30vw;
    height: 70vh;
    background-position: center;
    background-repeat: no-repeat;
    background-size: cover;
    position: relative;
    display: flex;
    flex-flow: column nowrap;
    justify-content: flex-end;
    padding: 1.5rem;

    /*Scrolling */
    scroll-snap-align: center;
}
```

Para tamaños de pantalla inferiores, se emplea la siguiente media query desactivando el overflow y activando el wrapping.

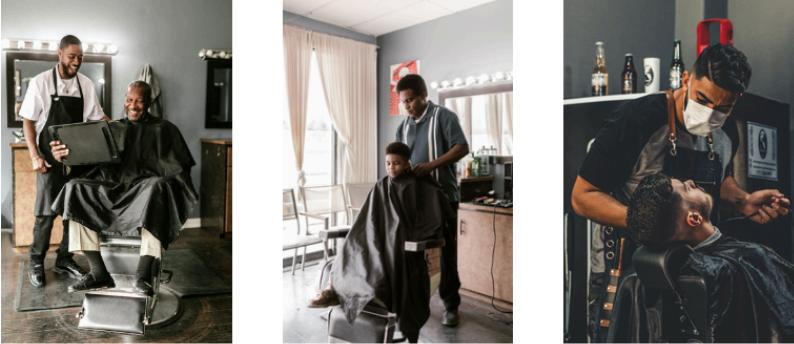
```
@media screen and (max-width: 768px) {  
    .carousel {  
        flex-flow: row wrap;  
        justify-content: center;  
        overflow-x: none;  
    }  
    .service-card {  
        min-width: 40vw;  
        height: 40vh;  
        margin: 0.5rem;  
    }  
    .carousel-links {  
        display: none;  
    }  
}
```



Las cartas de los miembros del equipo se aprovechan del wrapping de los contenedores flex que, junto con una media query para reducir el tamaño de las imágenes en dispositivos pequeños, permiten mostrar los miembros en filas de 3, 2 o 1.

```
section.equipo {  
    display: flex;  
    align-items: center;  
    flex-direction: column;  
    margin: 3rem 0;  
}  
  
section.equipo ul {  
    display: flex;  
    flex-flow: row wrap;  
    align-items: center;  
    justify-content: center;  
}  
  
.barber-card {  
    display: flex;  
    flex-flow: column;  
    justify-content: center;  
    align-items: center;  
    margin: 0 1.5rem;  
    padding-bottom: 1rem;  
}  
  
@media screen and (max-width: 900px) {  
    .barber-card img {  
        height: 35vh;  
    }  
}
```

### Conoce a nuestro equipo



Pablo Lopez      Pedro Echevarría      Nacho Garcia



Pablo Lopez      Pedro Echevarría  
Nacho Garcia      Pablo Lopez

La página de *conócenos* emplea las mismas técnicas que la home con el añadido de el uso de *flex-direction: column-reverse* en los miembros impares de la sección equipo. Esto se consiguió añadiendo la clase *barber-reverse* a los miembros del equipo correspondiente y en esta clase modificando la *flex-direction*. Fué necesario el uso de *!important* para sobreescribir la dirección establecida en la clase *barber*.



**Pablo Lopez**

Es un talentoso y apasionado Barbero en Villaverde. Con una habilidad excepcional para crear cortes de cabelllo y estilos modernos, Pablo se destaca por su atención minuciosa a los detalles y su enfoque personalizado. Ofrece servicios de corte y diseño para hombres de todos los gustos y edades. Su ambiente acogedor y profesional es perfecto para relajarse y sentirse bien.



**Pedro Echevarría**

Pedro Echevarría, un talentoso Barbero en Villaverde, se destaca por su creatividad y pasión por la belleza masculina. Con una habilidad excepcional para crear cortes de cabelllo y estilos modernos, Pedro se destaca por su atención minuciosa a los detalles y su enfoque personalizado. Ofrece servicios de corte y diseño para hombres de todos los gustos y edades. Su ambiente acogedor y profesional es perfecto para relajarse y sentirse bien.

```
.barber {
  display: flex;
  align-items: center;
  justify-content: center;
  margin: 1.5rem 0;
}
.barber-reverse {
  flex-direction: row-reverse !important;
}
```

## 10.6 Bootstrap

Bootstrap es un framework de desarrollo front-end de código abierto que proporciona herramientas y estilos predefinidos para crear interfaces web modernas y responsivas. Con una combinación de HTML, CSS y JavaScript, Bootstrap permite a los desarrolladores crear rápidamente sitios web y aplicaciones que se adaptan automáticamente a diferentes dispositivos y tamaños de pantalla.

Este framework fué empleado en la página de reservar cita. Esta página tiene por contenido una sección con información general de contacto y un formulario. Estos elementos son alineados en pantalla mediante el sistema de grid de bootstrap con 6 columnas cada uno.

El principal uso del framework en la página es en el formulario mediante el uso de sus clases de *input-groups*, *disabled*, *btn*, *btn-primary*, *form-select* y el alineado mediante rows y cols.

Barbero\*

Elige tu barbero

Dia\*

mm / dd / yyyy

Hora\*

Selecciona la Hora de Tu Cita

Nombre\*

Sergio

Apellidos

Vadillo Vilar

Barbero\*

Elige tu barbero

Elige tu barbero

Pablo Lopez

Pedro Echevarría

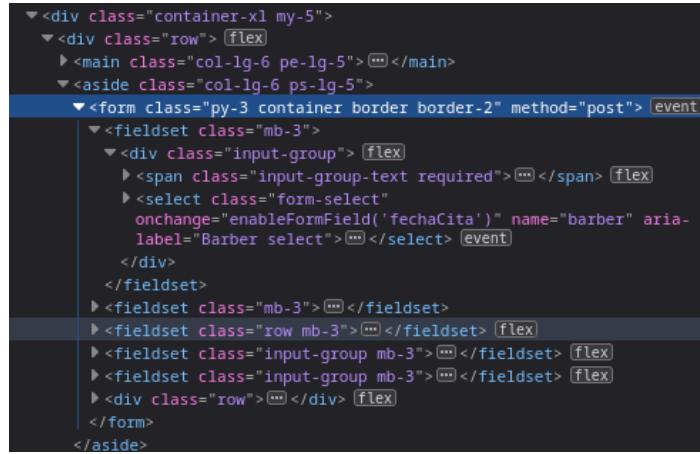
Nacho García

Nombre\*

Sergio

Apellidos

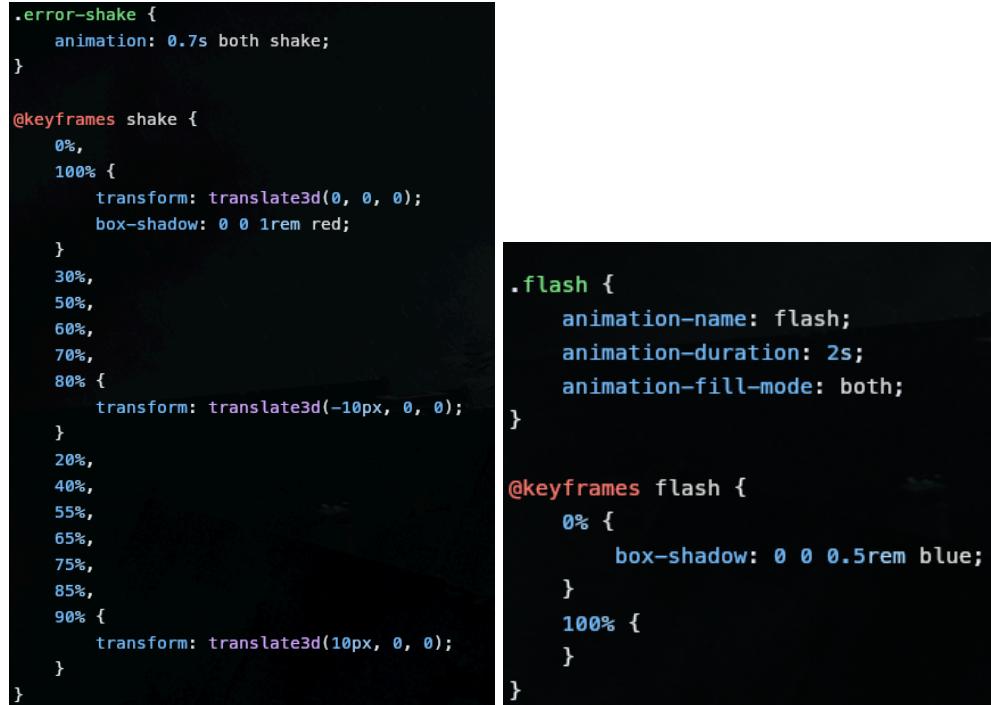
Vadillo Vilar



A screenshot of a browser's developer tools element inspector. It shows a portion of the DOM tree under an 'aside' element. The visible code includes:

```
<div class="container-xl my-5">
  <div class="row"> [flex]
    > <main class="col-lg-6 pe-lg-5">[...]</main>
    <aside class="col-lg-6 ps-lg-5">
      <form class="py-3 container border border-2" method="post"> [event]
        <fieldset class="mb-3">
          <div class="input-group"> [flex]
            > <span class="input-group-text required">[...]</span> [flex]
            > <select class="form-select" onchange="enableFormField('fechaCita')" name="barber" aria-label="Barber select">[...]</select> [event]
          </div>
        </fieldset>
        <fieldset class="mb-3">[...]</fieldset>
        <fieldset class="row mb-3">[...]</fieldset>
        <fieldset class="input-group mb-3">[...]</fieldset>
        <fieldset class="input-group mb-3">[...]</fieldset>
        <div class="row">[...]</div>
      </form>
    </aside>
```

Además de las clases proporcionadas por bootstrap, también se definieron ciertas animaciones (usadas en la validación del formulario) para mostrar campos activados y error en los inputs:



Two side-by-side snippets of CSS code for validation animations.

The left snippet defines the 'shake' animation:

```
.error-shake {
  animation: 0.7s both shake;
}

@keyframes shake {
  0%,
  100% {
    transform: translate3d(0, 0, 0);
    box-shadow: 0 0 1rem red;
  }
  30%,
  50%,
  60%,
  70%,
  80% {
    transform: translate3d(-10px, 0, 0);
  }
  20%,
  40%,
  55%,
  65%,
  75%,
  85%,
  90% {
    transform: translate3d(10px, 0, 0);
  }
}
```

The right snippet defines the 'flash' animation:

```
.flash {
  animation-name: flash;
  animation-duration: 2s;
  animation-fill-mode: both;
}

@keyframes flash {
  0% {
    box-shadow: 0 0 0.5rem blue;
  }
  100% {
  }
}
```

## 11. Javascript

JavaScript es un lenguaje de programación versátil y dinámico utilizado principalmente en el desarrollo web para agregar interactividad y funcionalidad a las páginas. JavaScript se ejecuta en el navegador del cliente, permitiendo la manipulación del DOM y la creación de aplicaciones web complejas.

En nuestra página web, lo empleamos para manipulación compleja de elementos del DOM y asignado de clases y atributos de forma dinámica y responsive a las acciones del usuario, validación de campos de formulario, carga dinámica de contenido, efectos visuales y notificaciones al usuario mediante alerts.

### 11.1 Servidor Nginx

Para poder desarrollar de forma local la web y que la navegación mediante hyperlinks y las cargas de ficheros html y xml mediante AJAX y XMLHttpRequest funcionasen es necesario el uso de un servidor web. Para el proyecto de la barbería se desplegó un servidor nginx contenerizado mediante un fichero *docker-compose.yaml* y empleando la tecnología de contenedores docker.

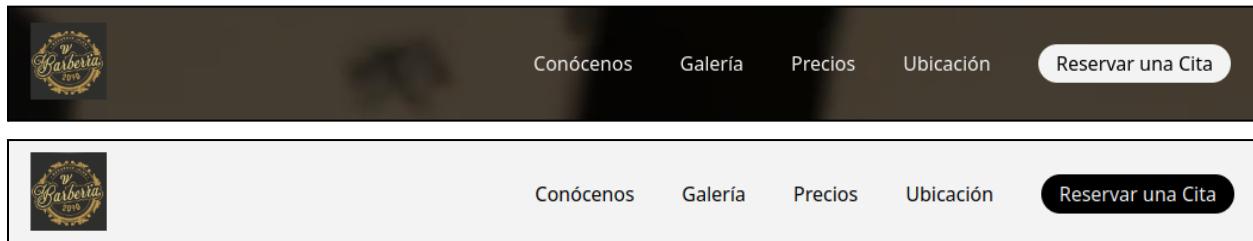
Para facilitar el desarrollo, se mapeo como volumen del directorio de la imagen */usr/share/nginx/html* los contenidos de la dirección del proyecto *src*. De esta forma, al ejecutar *docker compose up* se puede acceder a la página web en la dirección *localhost* (ya que se levanta en el puerto http 80).

```
services:
  nginx-server:
    image: nginx:1-alpine3.17-slim
    volumes:
      - ./src:/usr/share/nginx/html
    ports:
      - 443:443
      - 80:80
```

### 11.2 Tema de Navbar Dinámico

El estado inicial de la navbar es de tener un fondo translúcido permitiendo ver la imagen o video empleado de fondo en la portada. El problema de este estilo es que, debido a su naturaleza fija, si el fondo de la sección actual es oscuro puede interferir con la correcta visualización de los elementos de navegación. Por esta razón se optó por cambiarle el color del fondo una vez esta

deje de estar en la portada. Esto se implementa mediante un event listener de la navba en *scroll* en el archivo *navbar.js*.



```
const nav = document.querySelector("nav");

window.addEventListener("scroll", () => {
    nav.classList.toggle("nav-active", window.scrollY > 0);
});
```

### 11.3 Collapse Navbar

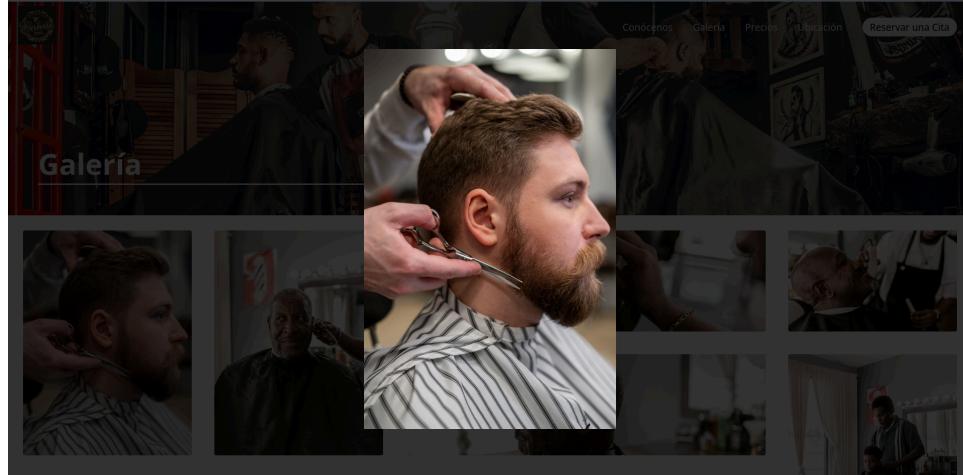


En tamaños de dispositivos pequeños no es posible tener una barra de navegación con los links por lo que se optó collapsarlos en un *menú hamburguesa* como se explicó en la sección css correspondiente. Para el correcto funcionamiento del toggle de este menú, se creo un event listener *onclick* en el fichero *toggle.js*

```
const menu = document.getElementsByClassName("nav-links")[0];
const toggle = document.getElementById("toggle");

toggle.addEventListener("click", () => {
    menu.classList.toggle("visible");
});
```

## 11.4 Imagen Fullscreen



La página de galería ofrece la posibilidad de ver distintos cortes, afeitados y tintes realizados por los miembros de nuestro equipo. Es solo lógico que, en una página de estas características, esté habilitada la posibilidad de ver en pantalla completa las imágenes.

Esta funcionalidad se implementa en el archivo *galeria.js* y su funcionamiento es el siguiente.

- En *galeria.html* hay una div vacía de la clase *fs-wrapper*. Esta div que inicialmente está oculta con *display: none* es de layout fixed, del tamaño de la pantalla completa y con un color negro semitransparente. En esta div se introducirán las imágenes que queremos visualizar a pantalla completa y el fondo negro permite centrar el foco de atención.
- Un event listener *onclick* es añadido a cada una de las imágenes de la página y cuando se ejecuta se crea una copia del nodo de la imagen, se le establece la clase *fullscreen* (y se elimina su clase *card*) y se añade como hijo del wrapper.

```
const imgs = document.querySelectorAll(".card");
const fullpage = document.getElementsByClassName("fullscreen")[0];

for (let img of imgs) {
    img.addEventListener("click", () => {
        let element = img.cloneNode(true);
        let fullscreenWrapper = document.getElementById("fs-wrapper");

        element.classList.add("fullscreen");
        element.classList.remove("card");
        fullscreenWrapper.appendChild(element);
        fullscreenWrapper.style.display = "block";
    });
}
```

- Finalmente, para poder salir de fullscreen, la clase wrapper tiene implementado un event listener *onclick* que elimina sus hijos y establece la oculta (*display: none*).

```
const wrapper = document.getElementById("fs-wrapper");
wrapper.addEventListener("click", () => {
    let children = wrapper.children;
    for (child of children) {
        wrapper.removeChild(child);
    }
    wrapper.style.display = "none";
});
```

## 11.5 Activación de Campos de Formulario

The figure consists of three side-by-side screenshots of a web-based booking form. The first screenshot shows the initial state where all input fields are grayed out and inactive. The second screenshot shows the 'Barbero' field populated with 'Pablo Lopez' and the date and time pickers now active and showing options. The third screenshot shows the date field set to '04/10/2024' and the time picker now active and showing options. A blue 'Reservar' button is visible at the bottom of each form.

El formulario permite la introducción de datos personales e independientes como el nombre, pero también de datos relativos al barbero, fecha y hora de la cita. Estos últimos datos, debido a los horarios de la barbería y de los miembros de nuestros equipos, son dependientes los unos sobre los otros.

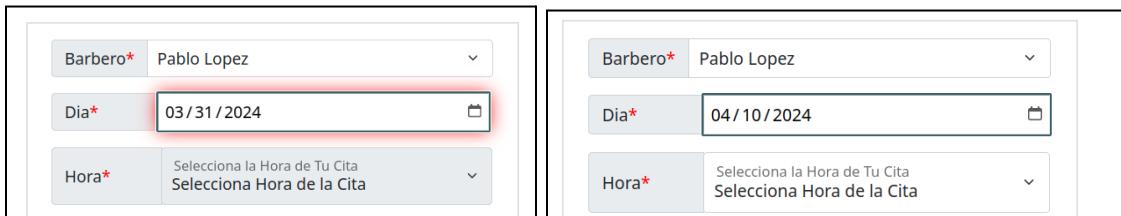
Por esta razón, el formulario comienza con los campos dia y hora deshabilitados. El campo día se habilita una vez se escoge un barbero (ya que es necesario tener en cuenta el barbero para comprobar su disponibilidad) y una vez escogida una fecha válida, se muestran las horas disponibles (en caso de haberlas).

También es de notar que, cada vez que un elemento se habilita, se le activa una clase *flash* la cual se elimina automáticamente mediante un event listener del propio elemento sobre el evento *animationend*.

```
const enableFormField = (fieldName) => {
    let field = document.getElementsByName(fieldName)[0];
    if (!field.hasAttribute("disabled")) {
        field.setAttribute("disabled", "disabled");
    }
    field.removeAttribute("disabled");
    field.classList.remove("disabled");
    field.classList.add("flash");
    field.addEventListener("animationend", function () {
        field.classList.remove("flash");
    });
};

field.addEventListener("animationend", function () {
    field.classList.remove("flash");
});
```

## 11.6 Validación de Fecha



La validación de la fecha cumple 2 criterios:

- La fecha elegida no puede ser pasada, es decir, no se pueden reservar citas en el pasado
- La fecha elegida no puede ser domingo o festivo ya que esos días la barbería está cerrada.

En caso de que la validación sea negativa da como resultado la captura de la izquierda implementando la animación error-shake. En caso de que la fecha sea válida se activa el siguiente campo de seleccionar la hora. Ambas funciones se implementan en el archivo *reserva.js*.

La validación de la fecha se realiza mediante la siguiente función que verifica los criterios nombrados anteriormente.

```
const isValidDate = (fieldName) => {
    let field = document.getElementsByName(fieldName)[0];
    let [year, month, day] = field.value.split("-");
    let inputDate = new Date(year, month - 1, day);
    let today = new Date();
    today.setHours(0, 0, 0, 0);
    let isValid = inputDate >= today && inputDate.getDay() != 0;
    if (!isValid) {
        field.classList.add("error-shake");
        field.addEventListener("animationend", function (event) {
            field.classList.remove("error-shake");
        });
    }
    return isValid;
};
```

La activación del siguiente campo se realiza al igual que en el apartado anterior [11.5 Activación de Campos de Formulario](#) y estableciendo en el atributo *onclick* del input date en *reserva.html* como la función `isValidDate` el operador lógico AND y la función de activación del siguiente campo. Esto permite que, si la función de validación de fecha devuelve un valor falso no se ejecuta el resto de la sentencia.

```
<fieldset class="mb-3">
  <div class="input-group">
    <span class="input-group-text w-25 required">Dia</span>
    <input
      required
      onchange="isValidDate('fechaCita') && enableFormField('horaCita')"
      name="fechaCita"
      disabled
      type="date"
      class="ps-1 w-75 border-1 disabled">
  />
</div>
</fieldset>
```

## 11.7 Carga Dinámica de Catálogo

**CORTE PARA NIÑOS**

<u>Menores de 5</u>	7€
<u>Menores de 10</u>	10€
<u>Menores de 12</u>	12€



**RECORTES DE BARBA**

El catálogo de productos y precios se carga de forma dinámica a partir de los datos guardados en *prices.json*. Este archivo json es un array de objetos con la siguiente estructura:

```
{  
    "titulo": "CORTES DE PELO",  
    "servicios": [  
        { "titulo": "Corte de pelo básico", "precio": "15€" },  
        { "titulo": "Corte Rapado", "precio": "10€" },  
        { "titulo": "Corte y Lavado", "precio": "17€" },  
        { "titulo": "Cambio de look", "precio": "20€" }  
    ],  
    "imagen": "/media/images/cortes/cortes_006.jpg",  
    "id": "cortes"  
},
```

Estos datos se cargan mediante una XMLHttpRequest en el archivo *precios.js*. La petición se realiza y como callback se establece la función **generatePriceList** que itera cada uno de los objetos del fichero json (una vez parseado) y genera las etiquetas html correspondientes así como con sus clases.

```
const xhr = new XMLHttpRequest();  
  
xhr.onreadystatechange = function () {  
    if (xhr.readyState === XMLHttpRequest.DONE && xhr.status === 200) {  
        const jsonData = JSON.parse(xhr.responseText);  
        console.log(jsonData);  
        generatePriceList(jsonData);  
    }  
};  
  
xhr.open("GET", "/data/prices.json", true);  
xhr.send();
```

La generación de estas etiquetas es trivial y se basa en replicar la estructura que habíamos generado de forma manual en un principio. La única complejidad añadida que tiene es el control de si la sección es par o impar para aplicar las clases y orden de elementos pertinentes:

- Necesidad de incluir una clase *bg-dark* o no
- Añadir primero la div de clase *texto* o la figure

```
// Estilo para los pares (fondo claro)
// e estilo para los impares (fondo oscuro)
if (i % 2 === 0) {
    section.appendChild(div);
    section.appendChild(figure);
} else {
    section.classList.add("bg-dark");
    div.classList.add("light");
    section.appendChild(figure);
    section.appendChild(div);
}
```

## 11.8 Publicar Formulario

The screenshot shows a web form on the left and a confirmation dialog on the right. The form fields are:

- Barbero\*: Pedro Echevarría
- Dia\*: 04/10/2024
- Hora\*: Selecciona la Hora de Tu Cita  
18:00-18:30
- Nombre\*: Sergio Christian
- Apellidos: Vadillo Vilar Novoa Gonzalez

A blue "Reservar" button is at the bottom. To the right, a dark modal window titled "localhost" displays the reservation details:

Reservada Cita para el 2024-04-10 a las 18:00-18:30  
Cliente: Christian Novoa Gonzalez  
Barbero: Pedro Echevarría

An "OK" button is at the bottom right of the modal.

Debido a la falta de un servidor, el evento de publicar el formulario, manejado como un event listener de *onsubmit* del propio elemento *form*, simplemente realiza una alerta en el navegador que actúa a modo de notificación e informa del contenido de la reserva. Para ello convertimos el formulario en un *FormData* y mapeamos todas sus entradas en un objeto de javascript cuyos valores usamos posteriormente para formatear la alerta.

```
document.getElementsByTagName("form")[0].addEventListener("submit", function (event) {
    event.preventDefault(); // Prevent form submission
    const form = new FormData(event.target); // Using 'this'
    const formValues = {};
    for (const [key, value] of form.entries()) {
        formValues[key] = value;
    }
    console.log(formValues);
    alert(`Reservada Cita para el ${formValues.fechaCita} a las ${formValues.horaCita}\n
          Cliente: ${formValues.clientName} ${formValues.clientApellidos}\n
          Barbero: ${formValues.barber}`);
});
```

## 11.9 Carga Dinámica de Horas Disponibles

Finalmente, el efecto más complejo de toda la página web es la carga dinámica de las horas disponibles en el formulario. Esto se debe a:

- Leer los horarios del fichero xml y cargarlos.
- Filtrar las horas no válidas. En el pasado, es solo aplicable cuando el día de la cita es igual al día actual.
- Manejar la posibilidad de que no haya horas disponibles en ese día.
- Refrescar las horas disponibles cada vez que se modifica la fecha de la cita.

La carga de los horarios a partir del fichero xml se realiza mediante una petición XMLHttpRequest y la llamada a la función `loadAvailableHours` con el documento XML parseado para poder realizar queries sobre él como si fuese el DOM. Este código es muy similar al empleado en la carga del archivo JSON cambiando simplemente el parseado y la forma de acceso a los elementos.

```
const xhr = new XMLHttpRequest();
xhr.open("GET", "/data/hours.xml", true);

// Set up a callback function to handle the response
xhr.onreadystatechange = function () {
    if (this.readyState === XMLHttpRequest.DONE && this.status === 200) {
        const parser = new DOMParser();
        const xmlDoc = parser.parseFromString(this.responseText, "application/xml");
        loadAvailableHours(xmlDoc);
    }
};

// Send the request
xhr.send();
```

Una vez parseado el documento XML (y obviando comprobaciones que se explicaran a continuación) en la función *loadAvailableHours* se iteran cada uno de los *slots* del día correspondiente y se generan las options del form select con su contenido.

```
// Process XML Document
let timeSlots = xmlDoc.querySelectorAll("weekday slot");
if (selectedDate.getDay() === 6) {
    timeSlots = xmlDoc.querySelectorAll("saturday slot");
}

let selectionForm = document.getElementsByName("horaCita")[0];
selectionForm.innerHTML = "<option selected>Selecciona Hora de la Cita</option>";

let selectionForm = document.getElementsByName("horaCita")[0];
selectionForm.innerHTML = "<option selected>Selecciona Hora de la Cita</option>";

let index = 0;
for (slot of timeSlots) {
    let slot = timeSlots[index];
    let start = slot.querySelector("start").textContent;
    let [h, m] = start.split(":");
    if (isToday && (h < hrs || (h == hrs && m < min))) {
        continue;
    }

    let end = slot.querySelector("end").textContent;
    let option = document.createElement("option");
    option.setAttribute("value", `${start}-${end}`);
    option.innerText = `${start}-${end}`;
    selectionForm.appendChild(option);
    index++;
}
```

En el código anterior se puede observar la comprobación *if(isToday...)* esta comprobación es la que maneja si la hora que se está procesando en el momento es válida (a futuro) o no (pasada). Básicamente comprueba que, si la cita es para el dia actual, tiene que asegurarse que la hora de la cita sea mayor que la actual o que, en caso de ser iguales, los minutos lo sean. La variable *isToday* es el resultado de una comprobación realizada anteriormente entre la fecha actual y la

seleccionada en el campo del formulario date. Se calcula al comienzo de la función de la siguiente forma:

```
const loadAvailableHours = (xmlDoc) => {
    // Get Selected Date for Comparison
    let selectedDate = document.getElementsByName("fechaCita")[0];
    let [selYear, selMonth, selDay] = selectedDate.value.split("-");
    selectedDate = new Date(selYear, selMonth - 1, selDay);

    // Get Current date and time for comparison
    const today = new Date();
    let [day, hrs, min] = [today.getDay(), today.getHours(), today.getMinutes()];

    const isToday =
        today.getFullYear() === selectedDate.getFullYear() &&
        today.getMonth() === selectedDate.getMonth() &&
        today.getDate() === selectedDate.getDate();
```

Llegados a este punto, se puede dar lugar que un usuario intente pedir cita para el día actual, pero no quedan horas disponibles (o son pasadas o, si el fichero lo genera un servidor, están todas reservadas). Esta comprobación se realiza al final de la iteración y, de ser el caso, actualiza la option por defecto del form select y establece su clase a *disabled*.

```
// If only one child means there are no available dates
if (selectionForm.childNodes.length === 1) {
    selectionForm.innerHTML = "<option selected>No Available Hours on this date</option>";
    selectionForm.classList.add("disabled");
    selectionForm.setAttribute("disabled", "self");
}
```

Finalmente, se puede observar que en esa función se añade la clase disabled y el atributo disabled con valor *self*. El atributo disabled ignora el valor que se le de, pero este lo empleamos nosotros para validar cuando recargar los datos del archivo y cuando no.

Esto último se debe a que leemos los datos del archivo *hours.xml* regenerando las horas disponibles cada vez que se modifica el atributo disabled del campo horaCita del formulario. Para esto y su correcto funcionamiento son necesarios:

- Crear un observer del campo del formulario horaCita que cada vez que se modifique su atributo *disabled* actualice el listado.

```
const hoursSelect = document.getElementsByName("horaCita")[0];
// Load Valid Hourse after day selection
// Create a MutationObserver instance
const observer = new MutationObserver(function (mutationsList, observer) {
    for (let mutation of mutationsList) {
        if (mutation.type === "attributes") {
            // Check if the 'disabled' attribute has been modified
            if (
                mutation.attributeName === "disabled"
                &&
                mutation.target.getAttribute("disabled") !== "self"
            ) {
                const xhr = new XMLHttpRequest();
                xhr.open("GET", "/data/hours.xml", true);

                // Set up a callback function to handle the response
                xhr.onreadystatechange = function () {
```

- Cuando se cambia la fecha, hay que actualizar el listado de horas con la nueva fecha. Esto se consigue eliminando el atributo *disabled* como ya se hace pero surge un problema, que ocurre si no lo tiene? Es por ello que en caso de no tenerlo, se le añade para luego eliminarlo.

```
if (!field.hasAttribute("disabled")) {
    field.setAttribute("disabled", "disabled");
}
field.removeAttribute("disabled");
```

- Finalmente, es necesario saber distinguir cuando el campo está *disabled* debido a la selección de una fecha y cuando está *disabled* debido a que no hay horas disponibles. Ahí es donde entra el valor *self* mencionado anteriormente y usado en la comprobación del observer.

```
// Check if the 'disabled' attribute has been modified
if (
    mutation.attributeName === "disabled"
    &&
    mutation.target.getAttribute("disabled") !== "self"
) {
```