



Daniel Vieites Torres

Desarrollo de Aplicaciones Web

USC

31/03/2024

# ÍNDICE

<u>Introducción</u> .....	3
<u>Inventario de contenido</u> .....	3
<u>Arquitectura de la información</u> .....	4
<u>Casos de uso</u> .....	5
<u>Mapa web</u> .....	6
<u>Bocetos</u> .....	8
<u>A mano</u> .....	8
<u>Wireframes</u> .....	9
<u>Mockups</u> .....	11
<u>Storyboard</u> .....	13
<u>Estructura de ficheros</u> .....	14
<u>Antes de empezar con el código</u> .....	15
<u>HTML</u> .....	15
<u>CSS</u> .....	19
<u>Archivos utilizados</u> .....	19
<u>Float</u> .....	20
<u>Multicol</u> .....	21
<u>Flex container</u> .....	22
<u>CSS Grid</u> .....	24
<u>Bootstrap</u> .....	26
<u>JavaScript</u> .....	28
<u>Acceso al DOM</u> .....	28
<u>Respuesta a eventos</u> .....	28
<u>jQuery</u> .....	29
<u>ES6</u> .....	30
<u>Carga de contenido</u> .....	31
<u>XML</u> .....	31
<u>JSON</u> .....	32

# Introducción

La idea de este proyecto es desarrollar una página web en la que se pueda obtener información de todo tipo sobre los gatos.

Los datos que se podrían consultar irán desde información general sobre ellos, como los tipos de razas; pasando por el tipo de nutrición y cuidados que deberían tener, hasta poder saber los refugios y veterinarios que se encuentran en una cierta zona. También se podrían encontrar enlaces a noticias actuales y anuncios sobre gatos que están desaparecidos.

En este caso, debido al reducido tiempo para realizar la página web, he realizado el contenido que me ha parecido más relevante y que me ha permitido crear una página web lo más dinámica posible. La cantidad de páginas iniciales se ha reducido finalmente con respecto a lo que se esperaba al principio, pero esto me ha permitido dedicarle más tiempo a los detalles. Igualmente, se han integrado todas las peticiones de diseño CSS y programación JavaScript que se solicitaban.

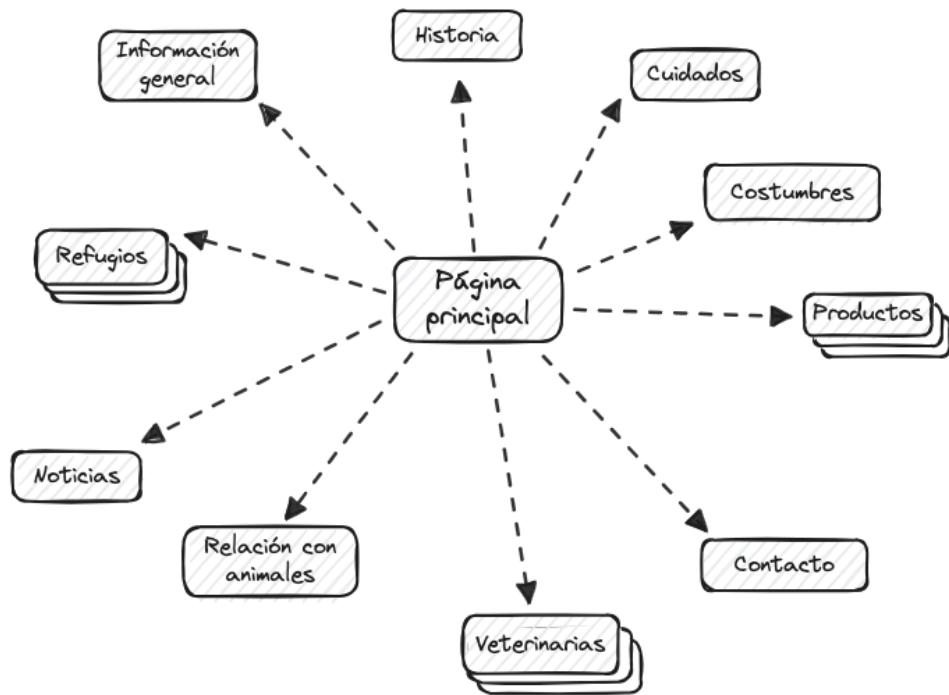
Como se podrá comprobar más adelante, no se ha incluido contenido real a la página, ya que se prioriza el estilo y la maquetación de la página por encima del propio contenido. Se podrá ver que tampoco existe un idioma concreto de la página debido a lo anteriormente comentado. Evidentemente, en el caso de que esta fuera una página web con un dominio en Internet, habría que buscar información y escoger un idioma, que presumiblemente serían tanto español como inglés, para darle más opciones al lector.

## Inventario de contenido

Crear un inventario de contenido consiste en obtener un listado detallado de todos los elementos de contenido de un sitio web. Esto incluye textos, imágenes, videos, documentos, enlaces y cualquier otro tipo de recurso multimedia o interactivo que forme parte del sitio.

El inventario de contenido es una herramienta muy importante para la organización y estructuración del contenido, facilitando su análisis, actualización y mantenimiento. Este proceso ayuda a planificar el desarrollo futuro y mantener una

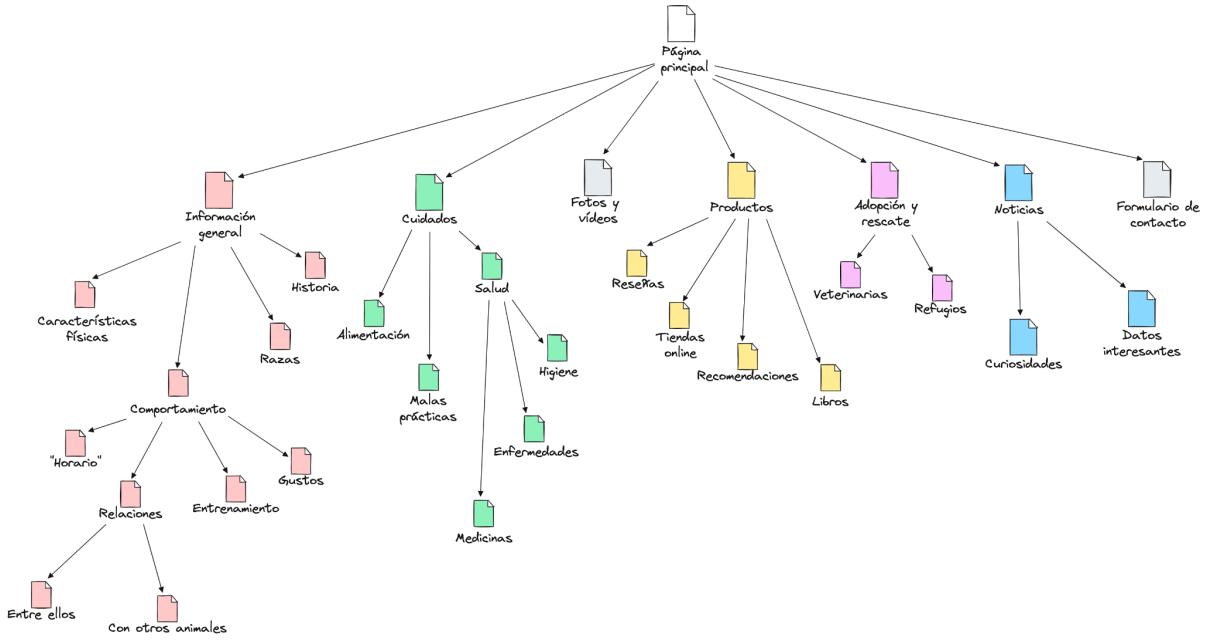
alta consistencia y calidad. Además, es crucial para mejorar el SEO y la accesibilidad del sitio, ya que un inventario de contenido bien organizado optimiza el sitio tanto para los usuarios como para los motores de búsqueda, lo que es especialmente relevante.



## Arquitectura de la información

Un diagrama de arquitectura de la información es una herramienta utilizada para organizar y estructurar la información en un sitio de manera eficaz. Esta estructuración facilita la creación de un sistema de navegación lógico y accesible, esencial para una buena usabilidad.

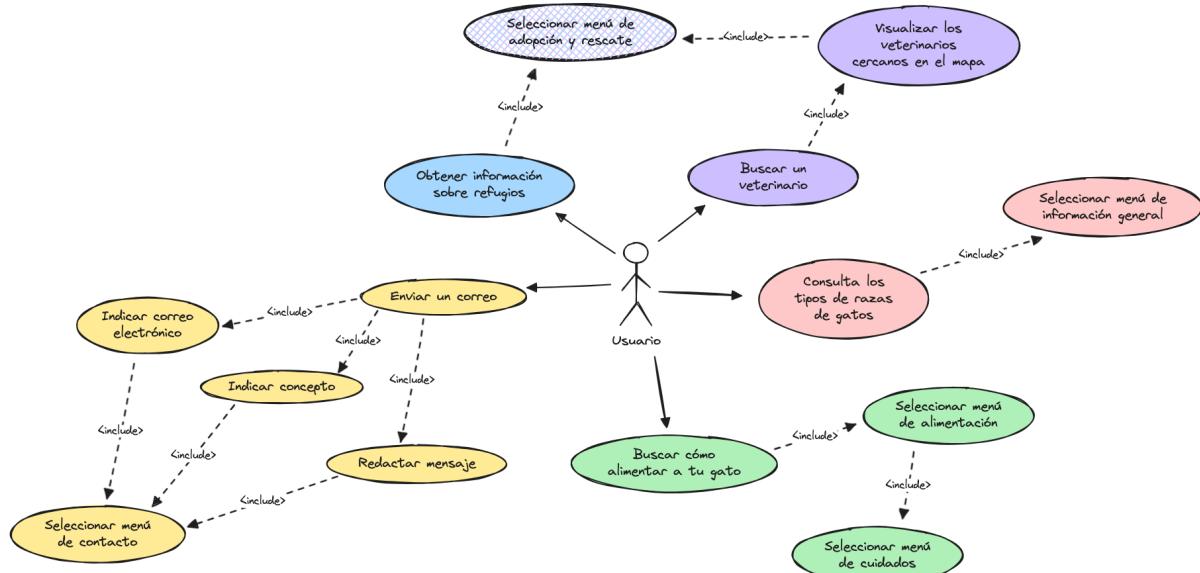
Este diagrama es clave en la planificación del contenido y en la colaboración del equipo de desarrollo. Proporciona una visión clara de dónde y cómo se debe organizar el contenido, lo que es especialmente importante en las fases iniciales del desarrollo de un sitio web.



## Casos de uso

Los casos de uso permiten definir cómo los usuarios interactúan con un sistema para alcanzar objetivos específicos. Sirven principalmente para identificar y documentar los requerimientos del sistema desde la perspectiva del usuario.

Además, facilitan la comunicación entre los desarrolladores y diseñadores al describir las funcionalidades del sistema de forma clara y comprensible. Son fundamentales para la elaboración de pruebas del sistema, permitiendo asegurar que todas las funcionalidades se comporten como se espera en situaciones reales.

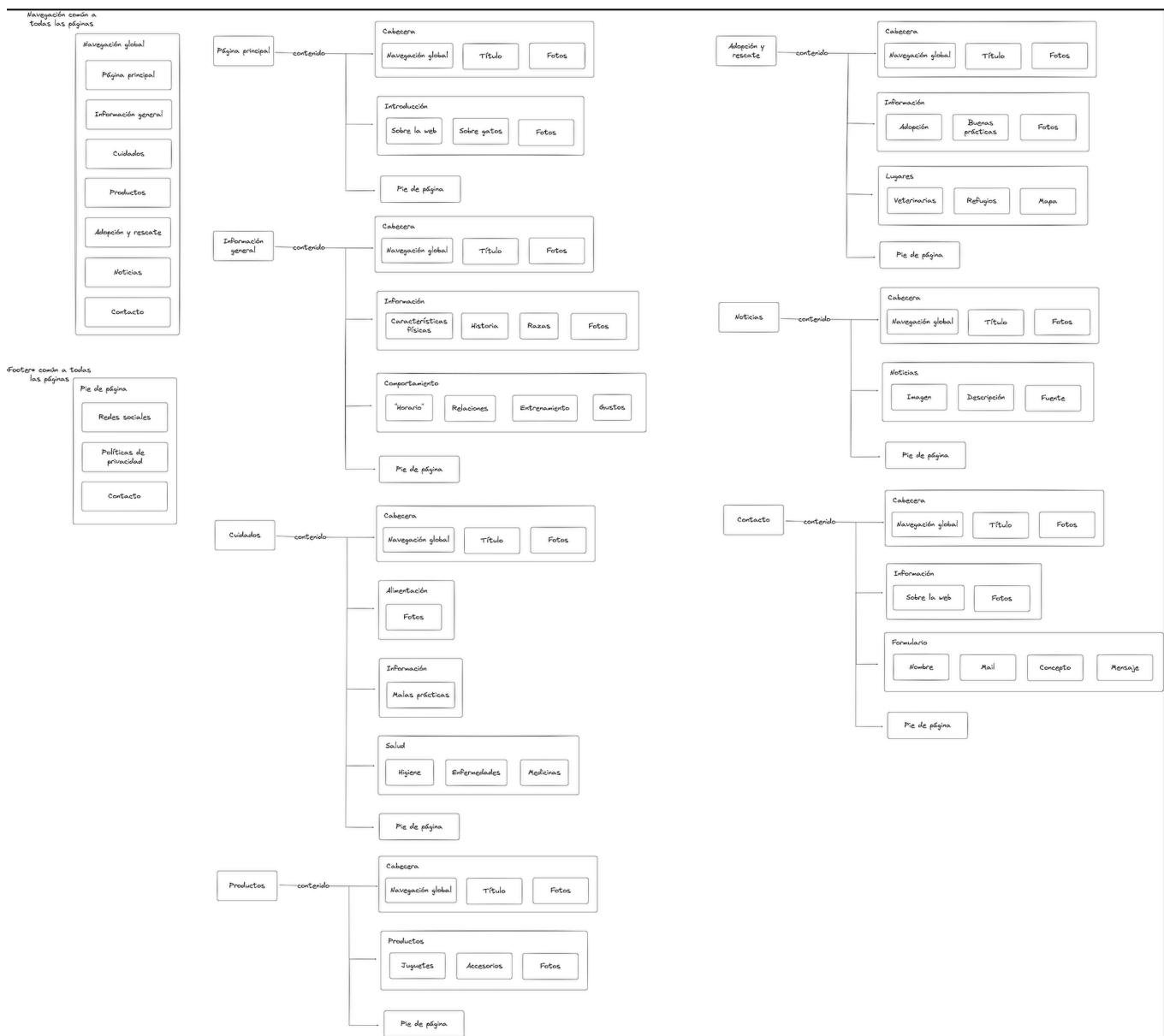


Estos serían algunos ejemplos de casos de uso que se podrían hacer con la página web completamente terminada. Como se ha comentado antes, no todos los casos de uso que se muestran se pueden hacer a estas alturas, ya que no existe contenido real en la página. Sin embargo, sí que se podría dar el caso de uso de visualizar veterinarios en un mapa; se ha introducido una pequeña funcionalidad en la que se muestran en un mapa diferentes ubicaciones, entre las que se encuentran consultas veterinarias.

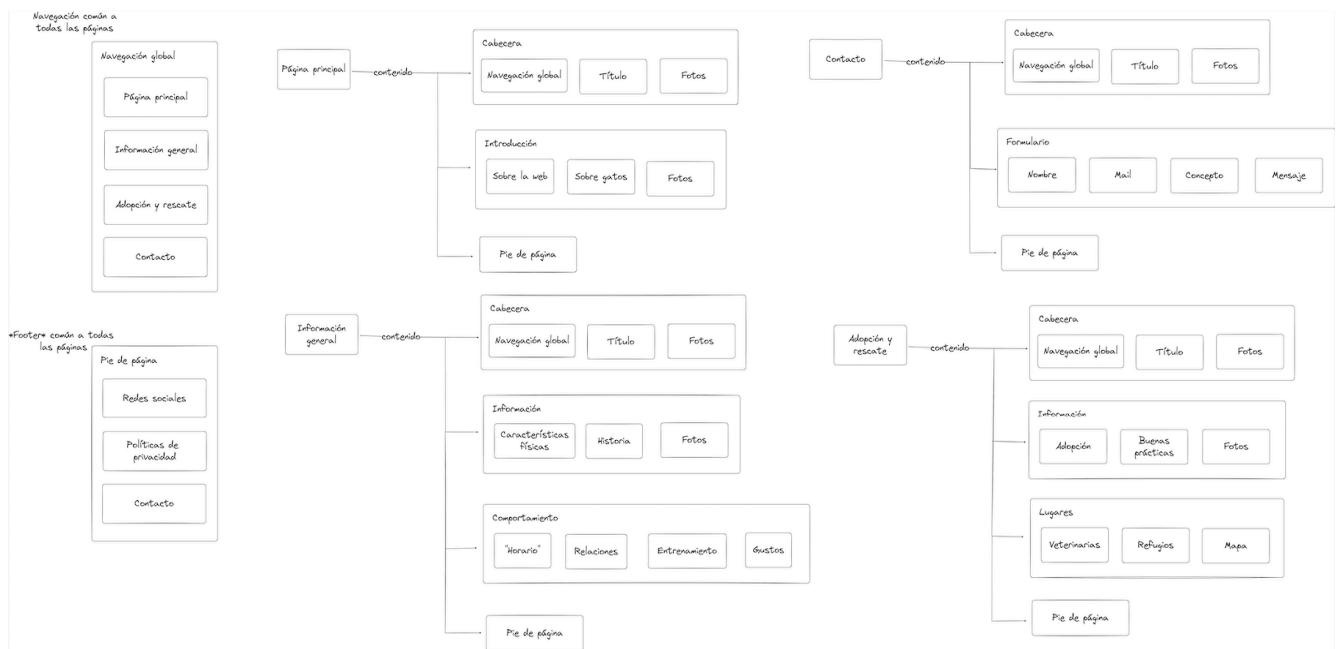
## Mapa web

Un mapa web es una representación esquemática o lista de las páginas de un sitio web, mostrando cómo están organizadas y cómo se relacionan entre sí.

Los mapas web mejoran la navegación y la usabilidad del sitio al proporcionar una guía de cómo se accede a las distintas secciones y páginas. Son una herramienta valiosa para la planificación y organización del contenido. Además, facilitan el mantenimiento y las actualizaciones del sitio, ya que los desarrolladores y diseñadores pueden identificar más fácilmente dónde se necesitan cambios o adiciones.



Este es el mapa web original, que se ha visto bastante reducido finalmente.  
Ahora se muestra el mapa web resultante tras haber terminado la web.

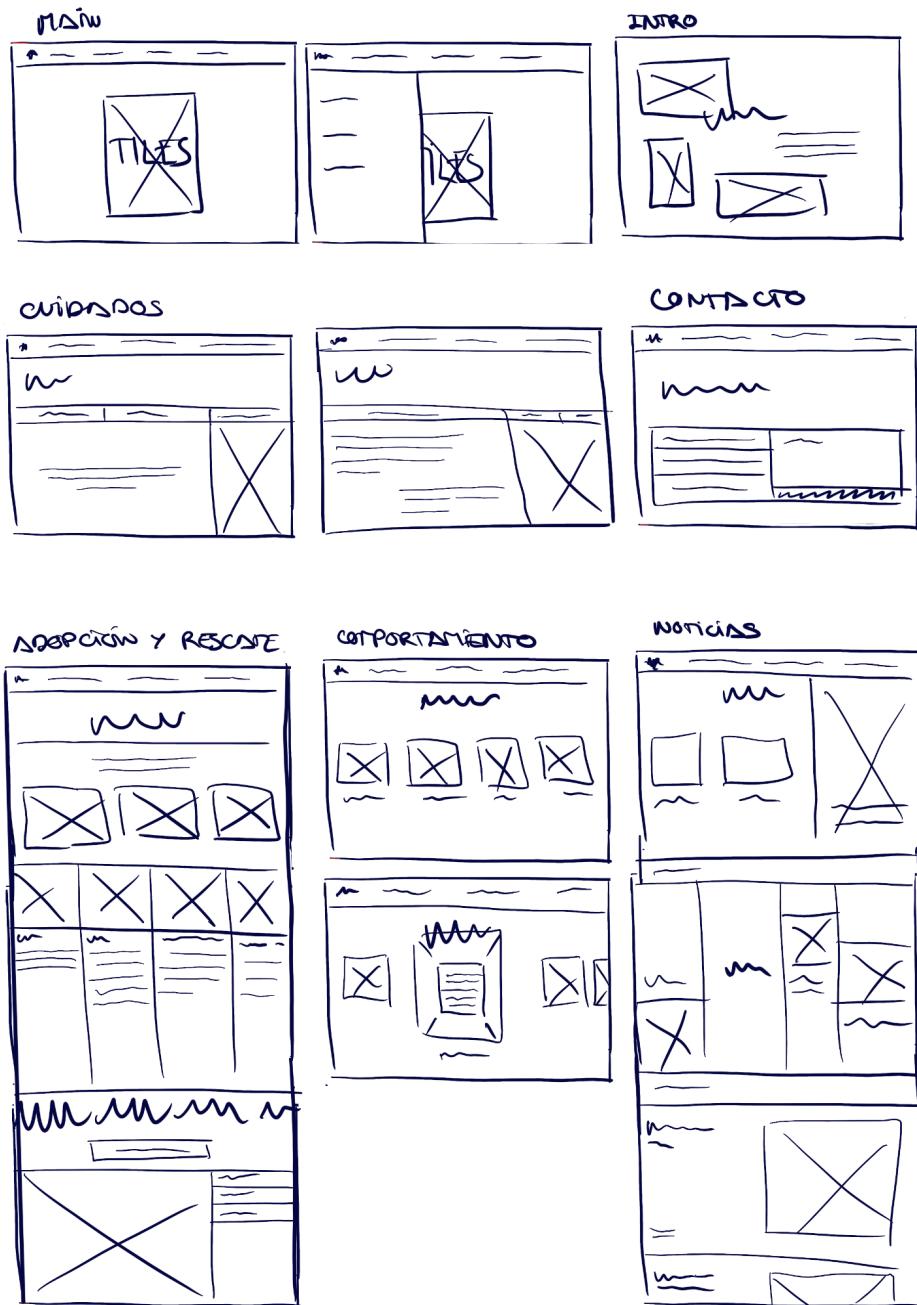


## Bocetos

El diseño de un sitio web implica varios tipos de bocetos, cada uno con un propósito específico. Estos bocetos van desde dibujos a mano hasta *wireframes* y *mockups*.

### A mano

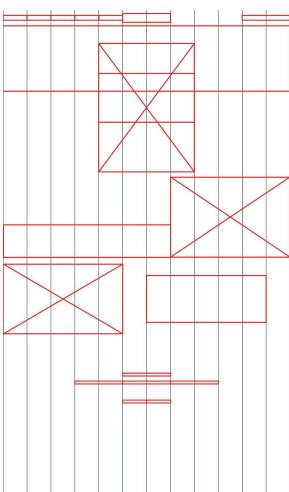
Estos bocetos son dibujos preliminares hechos a mano, rápidos y sin demasiada técnica, que se utilizan para capturar ideas básicas del diseño del sitio. Son muy útiles en sesiones de *brainstorming*, en las primeras etapas de planificación, para obtener una visión general del diseño y la disposición del sitio.



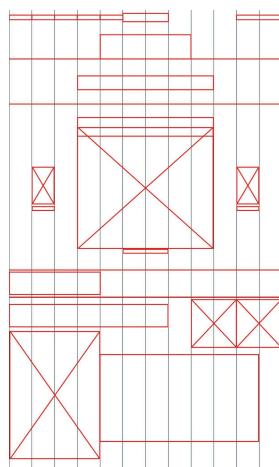
## Wireframes

Son esquemas que muestran los elementos básicos de una página web y su disposición, pero sin detalles como colores o imágenes. Se utilizan después de los bocetos iniciales para solidificar la estructura del sitio y ayudan a los equipos a discutir las características y la disposición antes de pasar a detalles más específicos.

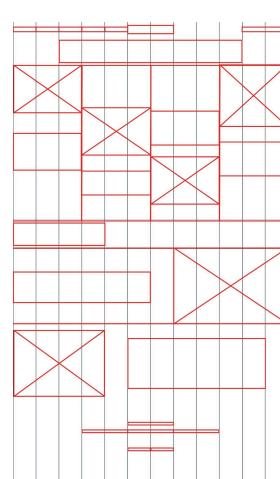
‘index.html’



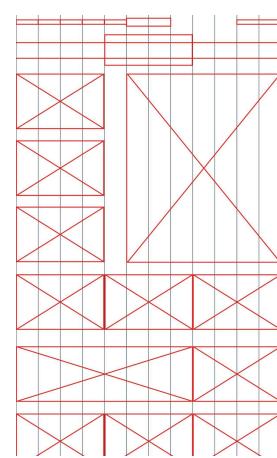
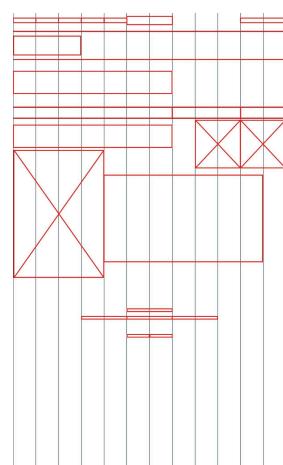
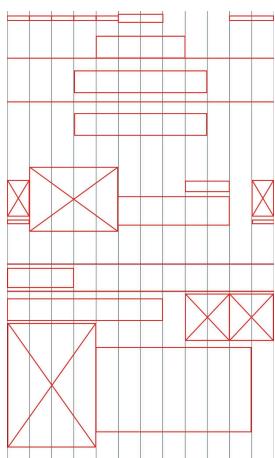
‘they.html’



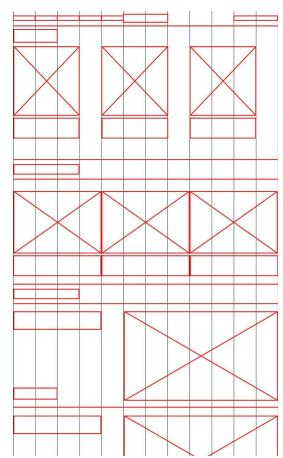
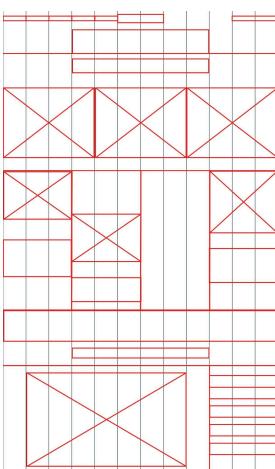
‘they.html’



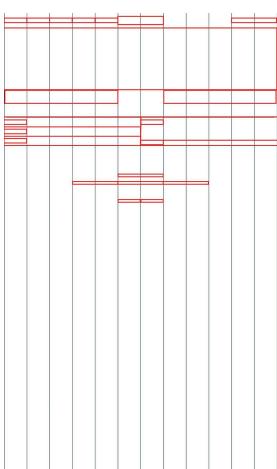
‘they.html’



‘take-them.html’



‘contact.html’



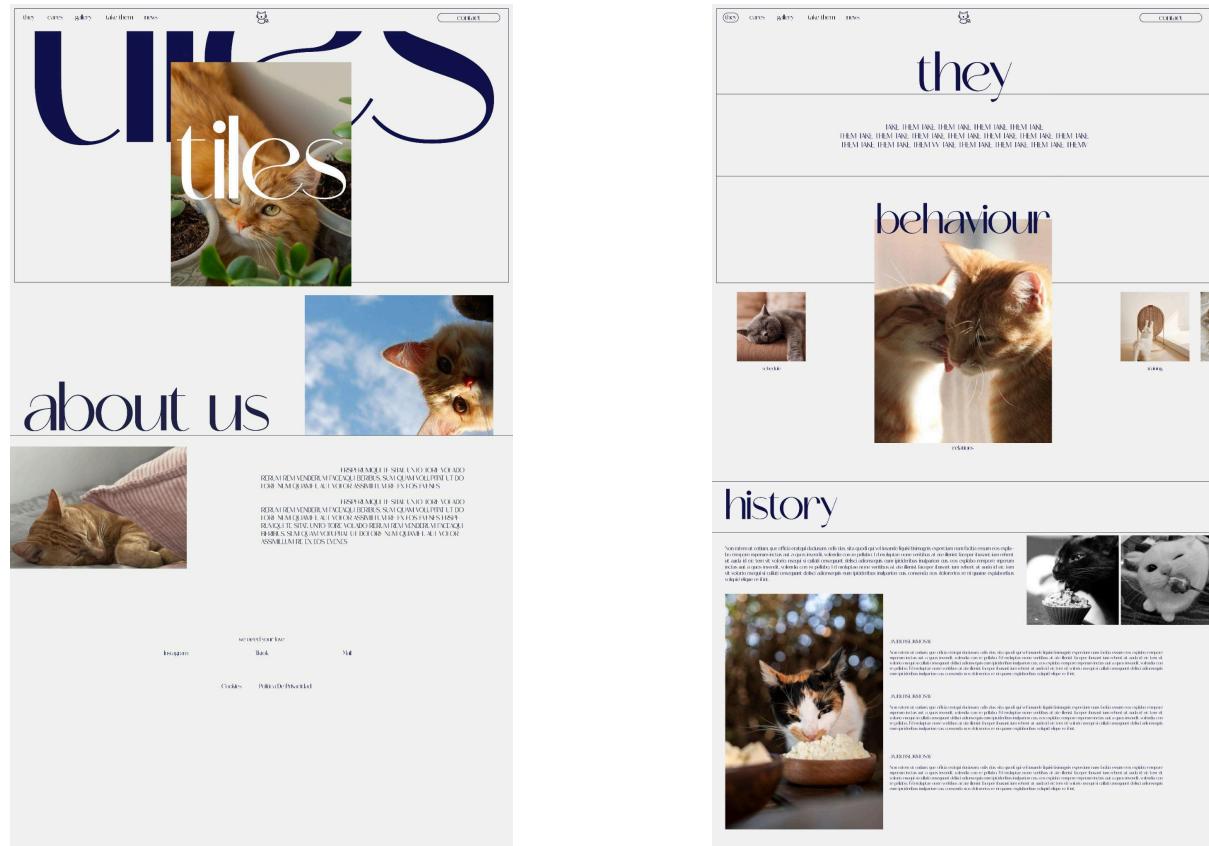
En las imágenes anteriores se indica, encima de los *wireframes* correspondientes, el documento HTML en el que finalmente se introdujo la estructura que se representa en dicho *wireframe*.

La no utilización de algunos de los anteriores *wireframes* se debe, de nuevo, a la falta de tiempo para un desarrollo más extenso.

## Mockups

Son representaciones detalladas y muy cercanas al diseño final del sitio. Incluyen detalles de diseño con colores, tipografías y elementos gráficos. Se crean después de los *wireframes* y se utilizan para mejorar el diseño visual antes de empezar el desarrollo del sitio.

Mostraré únicamente los *mockups* que se hicieron en su momento sobre las páginas que finalmente se han realizado en el proyecto final.



[they](#)

[behaviour](#)

[relations](#)

[history](#)

[physical features](#)

[breeds](#)

[take them](#)

[adoptar o acoger?](#)

[everything you need](#)

[CERCA](#) [cared](#) [glory](#) [take them](#) [ness](#)

[CONTACT](#)

[they](#)

[behaviour](#)

[relations](#)

[history](#)

[physical features](#)

[breeds](#)

[take them](#)

[adoptar o acoger?](#)

[everything you need](#)

[NAME](#) [MESSAGE](#)

[SUBJECT](#) [SEND](#)

[SEARCH](#)

[info@tiles.com](#)

[CERCA](#) [cared](#) [glory](#) [take them](#) [ness](#)

[CONTACT](#)

[UIES](#)

[contact with us](#)

[info@tiles.com](#)

[SEARCH](#)

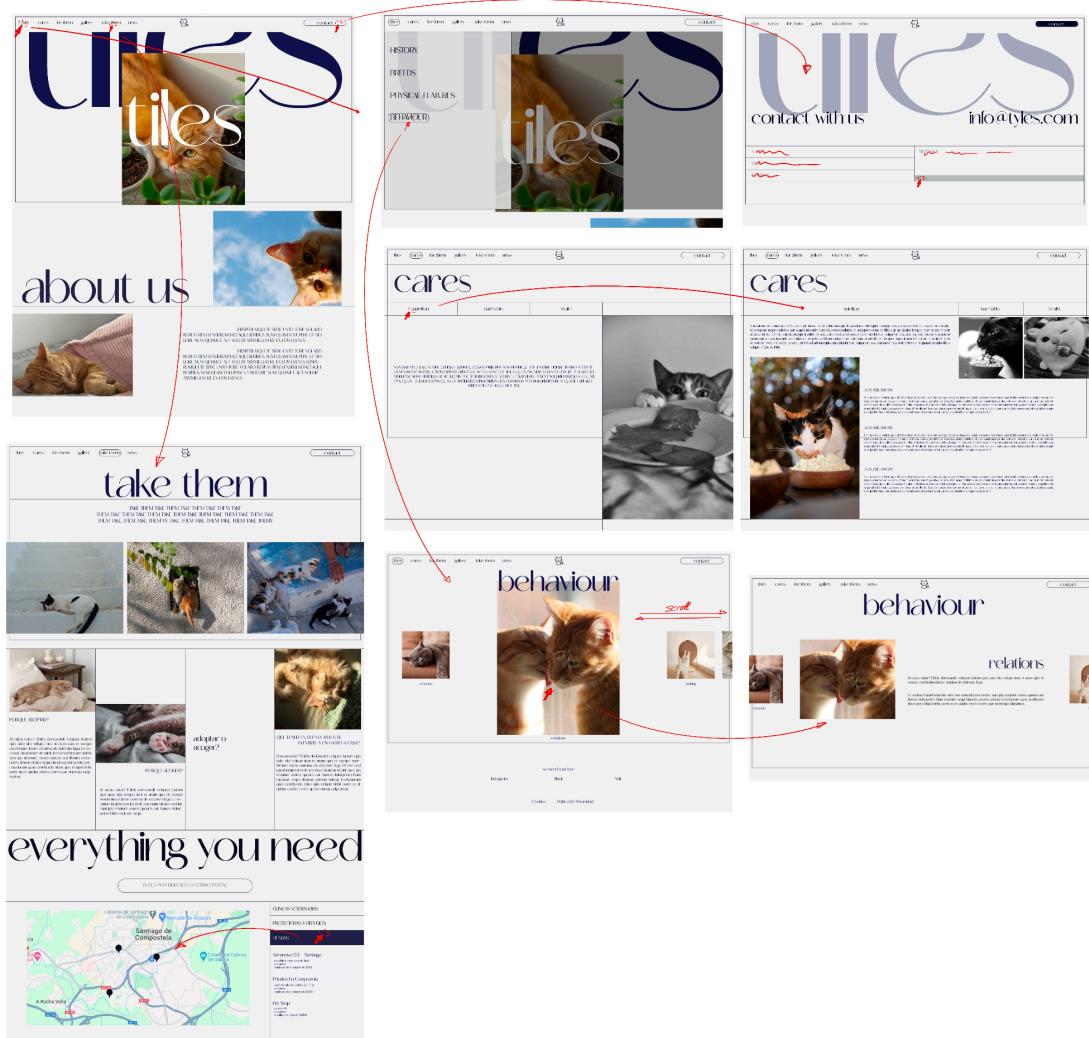
[DISPONIBLE](#) [BOK](#) [SA](#)

[TODAS](#) [TIPOS DE PROYECTOS](#)

Como se puede comprobar observando el resultado final, he podido replicar los *mockups* realizados, con bastante exactitud. Se han hecho algunos cambios en la web debido a las tecnologías que había que utilizar, pero nada que quede muy lejos de lo que se ideó inicialmente.

## Storyboard

Un *storyboard* representa gráficamente las acciones del usuario sobre las diferentes páginas de un sitio web. Su principal utilidad es la visualización detallada de la experiencia del usuario, lo cual es crucial para entender cómo los usuarios navegan y utilizan el sitio. Facilita la identificación temprana de problemas de usabilidad, permitiendo ajustes antes de la implementación técnica.

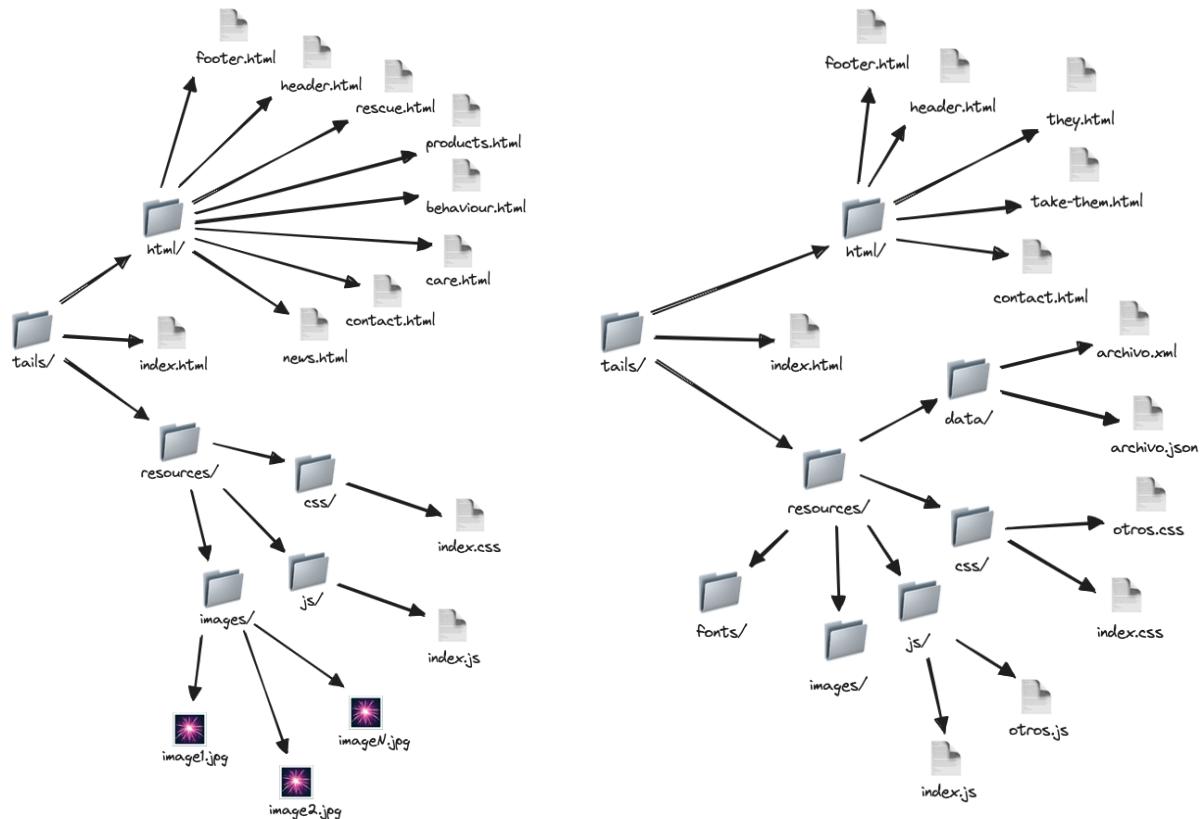


Este es el *storyboard* original, que contenía algunas de las páginas que se habían diseñado. La página de “cares”, en el centro de la imagen, es la única que no se ha implementado finalmente de las que aparecen en el *storyboard*. También se ha optado por un diseño diferente para la barra de navegación cuando nos encontramos en dispositivos móviles.

## Estructura de ficheros

Es muy importante mantener el código organizado y accesible a la hora de realizar un proyecto de un sitio web. Esto permite al desarrollador ser más eficiente y tener una mayor claridad.

Las estructuras de ficheros pueden ser muy flexibles y se pueden adaptar según las necesidades del proyecto. Lo más importante es mantener una organización clara y escalable.



La estructura de directorios de la izquierda es la que se diseñó inicialmente. La de la derecha es la que resultó tras la realización de la web.

Se puede observar que se ha mantenido la estructura inicial y que se ha extendido, añadiendo dos directorios de recursos a mayores: uno para almacenar los datos correspondientes a veterinarios, refugios, etc.; y otro para almacenar las fuentes que se utilizan en la página web, ya que estas no se encuentran entre las que proporciona Google.

## Antes de empezar con el código

Recomendaría, antes de nada, tener los archivos a los que se hace referencia abiertos a medida que se lee este informe. Esto permitirá entender mejor cómo funciona cada una de las partes del código, ya que incluye comentarios explicativos.

Para la visualización de la web, es preferible el uso de un navegador basado en Chromium, como **Google Chrome** o **Brave**, en su última versión, ya que no se asegura que funcionen en otro tipo de navegadores alguna de las implementaciones de estilos CSS que se han realizado. En el caso de que el carrusel de la página de “They” se mueva a saltos, por favor, **actualiza el navegador** para poder tener una mejor experiencia.

Además, recomiendo utilizar en todo momento, o por lo menos la gran mayoría del tiempo, la vista de desarrollador con la posibilidad de redimensionar la ventana. Esto permitirá poder comprobar la responsividad de la página, así como simular una pantalla táctil, lo cual es útil sobre todo en la página de “Take them”.

## HTML

A continuación se muestran sobre los *mockups* las etiquetas que se utilizan en los documentos HTML correspondientes.

**header** **nav** **section** **h1** **h2** **h3** **img** **div**

**header > nav** **header > main** **main** **content**

**main** **h1** **h2** **h3** **img** **div**

**about us** **h2** **h3** **img** **div**

**factor** **h2** **h3** **img** **div**

**we need your love** **h3**

**Instagram** **link** **link**

**Cookies** **Policy** **Privacy**

**header** **nav** **main**

**h1** **h2** **h3** **img** **div**

**behaviour** **h2** **h3** **img** **div**

**schedule** **relations**

**history** **h2** **h3** **img** **div**

**h3** **img** **div**

**h3** **img** **div**

**h3** **img** **div**

**PUBLISHING**

**PUBLISHING**

**PUBLISHING**

**behaviour**

**h3** **img** **div**

**schedule**

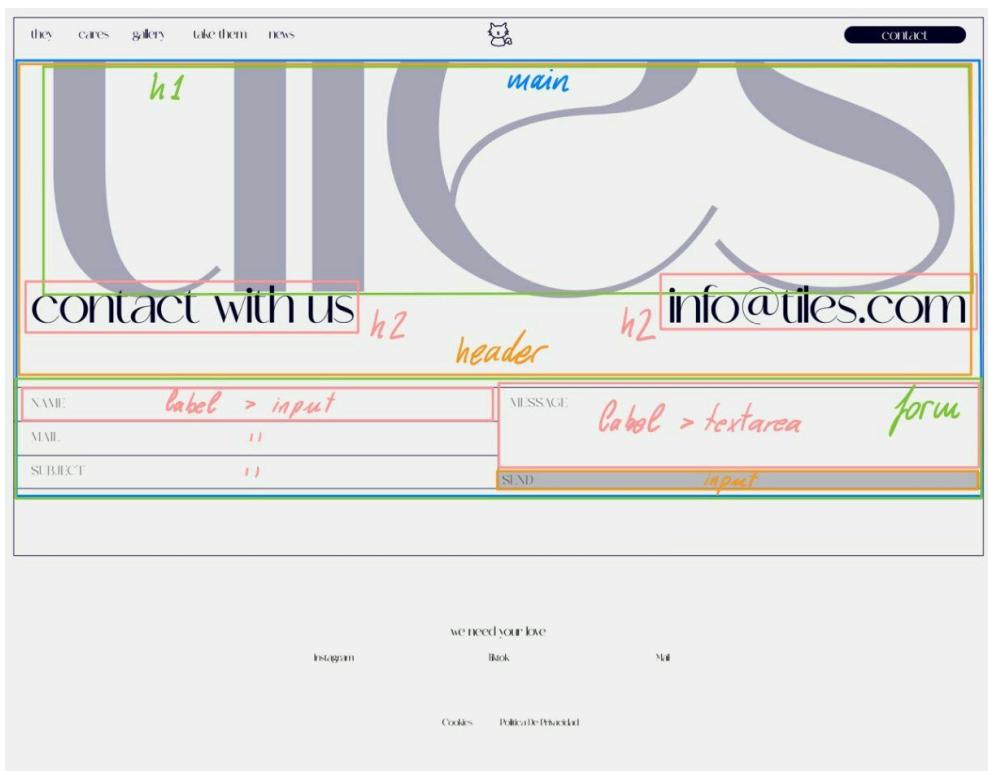
**relations** **h3** **img** **div**

**relations**

This image displays a complex web page layout for a cat adoption organization. The page is organized into several sections:

- Section 1: Physical Features** (h1)
  - Content: Descriptions and images of different cat breeds based on physical features.
  - Visuals: Multiple images of cats, some with handwritten annotations like "figura = manz" and "figura = rata".
- Section 2: Breeds** (h2)
  - Content: Descriptions and images of different cat breeds.
  - Visuals: Multiple images of cats, some with handwritten annotations like "figura = manz" and "figura = rata".
- Section 3: Take Them** (h1)
  - Content: Descriptions and images of cats available for adoption.
  - Visuals: Multiple images of cats being handled by people.
- Section 4: Everything You Need** (h2)
  - Content: A map of Santiago de Compostela and surrounding areas, along with lists of veterinary clinics, breeders, and adoption centers.
  - Visuals: A map showing the city layout and various adoption locations marked with icons.

The page includes a header with navigation links (they, cores, gallery, take them, news), a footer with social media links (Instagram, Facebook, Mail), and a footer with legal links (Cookies, Política De Privacidad).



Las imágenes anteriores se realizaron en las primeras sesiones de clase, cuando el objetivo era crear los documentos HTML que serían la base de la web. Se han mantenido al 90% las etiquetas que se indican, pero he hecho algunos cambios durante el desarrollo de la web.

En cuanto a los cambios en etiquetas, no hay mucho que destacar. Se han incluido algunas etiquetas `div` por la necesidad de agrupar algunas etiquetas para después poder estilar más fácilmente.

Los cambios más significativos están en el mapa de `take-them.html` y el acordeón que se encuentra justo a su lado. El mapa se introdujo mediante el uso de una API llamada [Leafletjs](#), la cual permite trabajar con mapas de una manera muy sencilla. Más adelante se explicará el código JavaScript referente a dicho mapa. Para este mapa solo es necesaria una etiqueta (`div`) con un id concreto, que permita la referenciar en JavaScript e incrustar el mapa en ella. En cuanto al acordeón, el cambio significativo ha sido por el hecho de utilizar [Bootstrap](#) para crearlo. Lo único que se ha añadido ha sido una etiqueta `button` dentro de cada uno de los `h3`, que son los títulos de los apartados de la lista. Ese botón es el que permite que, al ser pulsado, se

muestre el contenido de ese elemento y se oculte el contenido de otro elemento de la lista que estuviera abierto.

## CSS

En este apartado se incluye lo más relevante sobre los estilos que se han aplicado a la página web. Me centro sobre todo en los apartados que se pide que se incluyan en la página, como la creación de un archivo de estilos genéricos a todas las ventanas o el uso de distintas técnicas que permiten una página web responsive.

Las “media queries” se pueden encontrar siempre al final de cada uno de los archivos CSS.

### Archivos utilizados

En la imagen de la derecha se pueden ver los archivos finales que se han creado durante el desarrollo de la página web.

Existen dos archivos que son los que se utilizan en todas las páginas, que son: `index.css` y `fonts.css`.



El archivo `fonts.css` contiene las normas necesarias para poder utilizar las fuentes que he descargado de Internet, así como la asignación de las distintas fuentes a las etiquetas que me interesan.

El archivo `index.css` contiene los estilos necesarios para la barra de navegación superior, el marco que rodea la página en la que nos encontramos (borde negro) y el pie de página. Estos elementos son utilizados en todas las páginas del sitio web. Además, importa al principio de todo el archivo `fonts.css`, para que este no se tenga que incluir en todos los documentos HTML.

Los demás archivos, son específicos de cada una de las páginas.

Ahora entraré en detalle sobre las diferentes técnicas que se pedía que se utilizaran en proyecto para proporcionar responsividad a la página web.

## Float

La técnica 'float' en CSS se utiliza para posicionar elementos en una página web, permitiendo que los elementos fluyan a su alrededor. Los valores comunes son 'left', 'right' y 'none'. Aunque 'float' puede ser útil para algunos diseños, ha sido en gran medida reemplazado por Flexbox y CSS Grid en el diseño web responsivo. Estos ofrecen mayor control y adaptabilidad para distintos tamaños de pantalla

Float sigue siendo relevante para mantener o actualizar sitios web más antiguos, pero para diseños modernos y responsivos, Flexbox y CSS Grid son generalmente preferidos.

El uso de 'float' en esta web se encuentra en la página principal, en el apartado de "About us". Su uso permite que cada elemento se mantenga pegado a la izquierda o a la derecha del contenedor en el que se encuentran.

Hago hincapié en que en el código se explican con más detalle todas las técnicas utilizadas.



## Multicol

Con `multicol` nos referimos a la propiedad CSS `column-count`, que se utiliza para dividir el contenido de un elemento en múltiples columnas, facilitando la creación de layouts tipo periódico o revista. Por ejemplo, establecer `column-count: 3` en un contenedor dividirá su contenido en tres columnas. Aunque este diseño es ideal para pantallas grandes, puede ser menos efectivo en dispositivos pequeños.

Hay propiedades adicionales como `column-gap` para el espacio entre columnas, `column-rule` para líneas divisorias, y `column-width` para el ancho entre columnas.

En la web, se utiliza la propiedad `column-count` en la página de “Take them”, en la sección de “Physical features”.



El archivo CSS correspondiente, `they.css`, está dividido por secciones, por lo que no debería ser complicado encontrar los estilos referentes a esta sección (“PHYSICAL”). También las “media queries” están divididas por secciones.

En este caso, `column-count` se utiliza para cualquier ancho de pantalla, aplicando diferentes números de columnas para cada rango de píxeles.

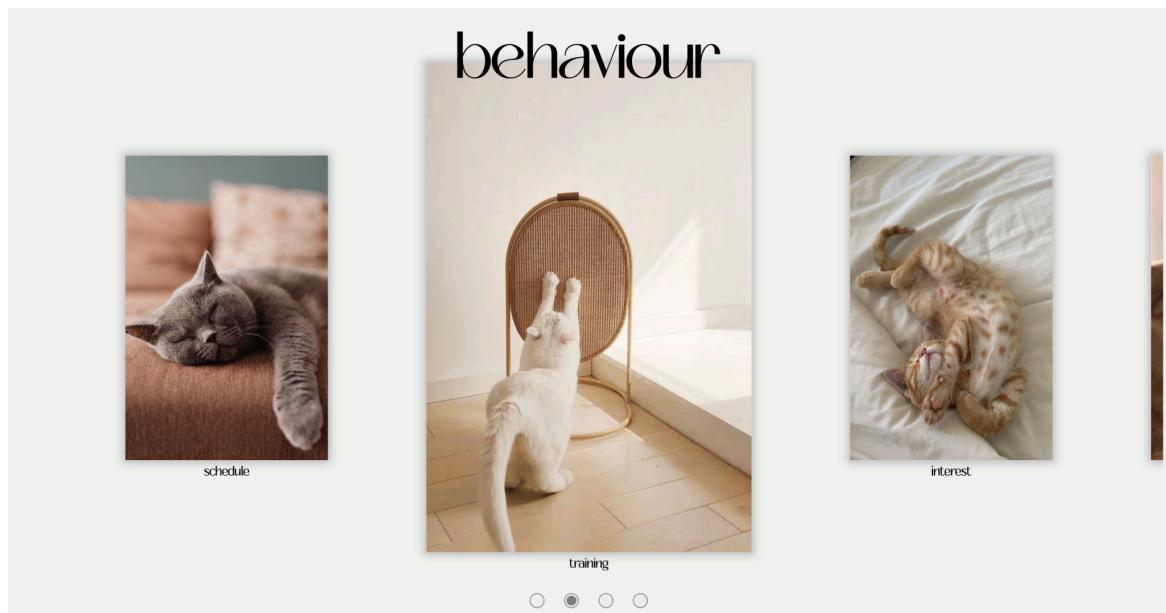
## Flex container

Un elemento “flex” en CSS es un elemento al que se le aplica `display: flex` o `display: inline-flex`, convirtiendo automáticamente a sus hijos directos en “flex items”. Este contenedor controla el diseño de sus elementos hijos a través de propiedades como:

- `flex-direction`: Define la orientación de los “flex items” (horizontal o vertical).
- `justify-content`: Alinea los “flex items” horizontalmente dentro del contenedor.
- `align-items`: Alinea los “flex items” verticalmente.
- `flex-wrap`: Permite que los “flex items” se distribuyan en múltiples líneas.
- `align-content`: Alinea las líneas de “flex items” cuando hay espacio extra.

Flexbox es ideal para diseños responsivos, permitiendo que los elementos se ajusten dinámicamente a diferentes tamaños de pantalla. También simplifica tareas como el centrado de elementos, tanto vertical como horizontalmente, que son más complejas con otras técnicas de CSS.

Esta técnica se utiliza a lo largo de toda la página web, por lo que mostraré a continuación la parte del sitio web en la que la presencia de Flexbox es más significativa. Esa parte de la que hablo es el carrusel de la página de “They”.



# behaviour



## training

Amet accusantium voluptatum numquam harum fugit? Voluptate vitae rerum numquam quis architecto, natus Unde numquam atque rem commodi itaque aperiam. Nostrum iure culpa a beatae officiis Sunt fugit optio molestias velit consequetur. Veniam explicabo quod temporibus minus enim Repellendus totam dolorem architecto praesentium excepturi Aliquid dolor doloremque cupiditate officiis architecto?



perspiciatis veritatis hic rerum adipisci. Tempore fugit repellendus optio nostrum hic Aut accusamus consequatur facilis consequatur deleniti? Necessitatibus repudiandae possimus voluptate voluptatum eligendi Eos.

# behaviour



training



# history

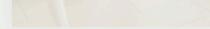


perspiciatis veritatis hic rerum adipisci. Tempore fugit repellendus optio nostrum hic Aut accusamus consequatur facilis consequatur deleniti? Necessitatibus repudiandae possimus voluptate voluptatum eligendi Eos.

# behaviour

## training

Amet accusantium voluptatum numquam harum fugit? Voluptate vitae rerum numquam quis architecto, natus Unde numquam atque rem commodi itaque aperiam. Nostrum iure culpa a beatae officiis Sunt fugit optio molestias velit consequetur. Veniam explicabo quod temporibus minus enim Repellendus totam dolorem architecto praesentium excepturi Aliquid dolor doloremque cupiditate officiis architecto?



# history

Las imágenes anteriores muestran el comportamiento del carrusel al ser clicado y en pantallas más pequeñas.

Reitero, antes de continuar, que se puede dar un comportamiento no deseado en el que el movimiento que se produce al cambiar de elemento durante el *scroll* es a trompicones, dando una muy mala experiencia de usuario. Si esto ocurre, por favor, **actualiza el navegador Google Chrome o Brave** (cualquiera con kernel Chromium) a su última versión. Esto debería solucionar el efecto visual.

Para observar el movimiento de las imágenes al hacer *scroll*, lo mejor es utilizar las herramientas de desarrollador con la opción de cambiar el tamaño del *viewport*. Esto nos permite utilizar el ratón como si fuera una pantalla táctil. Sin embargo, se han incluido botones que nos llevan al elemento que se encuentra en esa posición. También se puede pulsar directamente en los elementos para dirigirnos hacia ellos.

## CSS Grid

CSS Grid es un sistema de diseño bidimensional en CSS que permite crear diseños complejos y versátiles con facilidad. Se activa usando `display: grid` en un contenedor, permitiéndote definir filas y columnas y posicionar elementos dentro de estas estructuras. Sus características clave incluyen:

- `grid-template-columns` y `grid-template-rows`: Para definir el tamaño de las columnas y filas.
- `grid-column` y `grid-row`: Para ubicar y dimensionar elementos en el grid.
- `grid-area`: Asigna elementos a áreas específicas.
- `gap`: Establece el espacio entre filas y columnas.

CSS Grid es ideal para diseños que requieren un control preciso de la disposición y el alineamiento, superando las limitaciones de métodos anteriores como los flotantes. A diferencia de Flexbox, que se enfoca en diseños unidimensionales (una fila o columna), Grid maneja diseños bidimensionales (filas y columnas juntas). Además, facilita la creación de diseños responsivos, ajustándose a diferentes tamaños de pantalla mediante “media queries”.

Al igual que ocurre con Flexbox, he utilizado grid a lo largo de toda la página web, pero el lugar más representativo de su uso, y que encaja perfectamente con lo que he explicado justo antes, es la sección de “History”, dentro de la página “They”, justo debajo del carrusel.

En el apartado correspondiente a esta sección (“HISTORY”) dentro del archivo ‘they.css’ se puede ver cómo se utilizan las propiedades comentadas anteriormente. Además, se emplean “media queries” en las que se reorganizan los elementos para diferentes tamaños de pantalla, siempre utilizando grid para el posicionamiento.





Las imágenes anteriores muestran la disposición de los elementos en el grid con diferentes tamaños de pantalla. De arriba a abajo se pueden ver los cambios que se producen al reducir el tamaño del viewport.

## Bootstrap

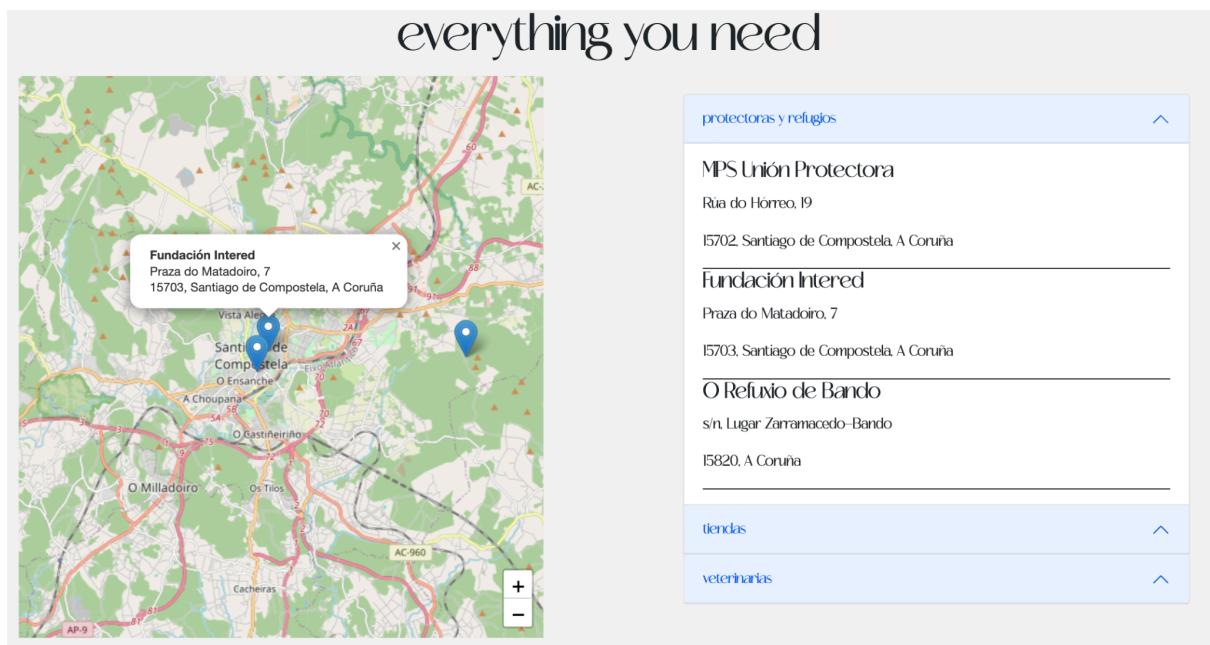
Bootstrap es un framework de front-end para el desarrollo web que proporciona herramientas para crear diseños responsivos y móviles primero. Sus características clave incluyen:

- Sistema de Grid: un sistema basado en 12 columnas para crear layouts responsivos que funcionan bien en dispositivos móviles, tabletas y ordenadores.
- Componentes reutilizables: Amplia gama de componentes de UI, como botones y formularios, que son personalizables y fáciles de integrar.
- Utilidades de CSS: Clases de utilidad para estilos comunes, facilitando la aplicación rápida de estilos.

- Plugins de JavaScript: Varios plugins basados en jQuery que añaden funcionalidades interactivas a los sitios web.
- Personalización: Permite personalizar y sobreescibir estilos predeterminados para adaptarse a necesidades específicas.

Bootstrap es ideal para desarrolladores que buscan construir interfaces web limpias y funcionales rápidamente, y es conocido por su facilidad de uso y compatibilidad entre navegadores. Sin embargo, debido a su popularidad, es importante personalizar los diseños para evitar un aspecto genérico.

En mi página web utilizo Bootstrap para crear el acordeón de “Take them”, que permite visualizar las diferentes marcas en el mapa, y en el que se listan todos los veterinarios, refugios, etc.



Para crear el acordeón he seguido las instrucciones de la propia página de Bootstrap. En este caso, los estilos incluidos en el archivo `take-them.css` no son tan relevantes, sino las clases utilizadas en las propias etiquetas HTML, en el archivo `take-them.html`.

# JavaScript

Para dotar de dinamismo a la página web se utiliza JavaScript junto con un poco de jQuery. Todo el código JavaScript está comentado, de manera que se entiende perfectamente para qué sirve cada una de las partes del código implementado.

A continuación indico dónde se han introducido las características que se piden en los documentos de prácticas con respecto al código JavaScript.

## Acceso al DOM

Se piden cinco accesos diferentes de acceso al DOM. Los que he utilizado son los siguientes:

- `getElementById`: Se utiliza en todos los archivos JavaScript.
- `getElementsByClassName`: Se utiliza en `they.js`, para obtener los elementos de la lista del carrusel.
- `getElementsByTagName`: Se utiliza para obtener los botones que se encuentran debajo del carrusel, también en `they.js`.
- `querySelector`: Se utiliza en los archivos `take-them.js` y `they.js`, por ejemplo para obtener el contenido de cada uno de los elementos de la lista del acordeón.
- `querySelectorAll`: Se utiliza en dos ocasiones en `they.js`: para obtener todos los botones de debajo del carrusel y para obtener y eliminar el texto “Click me!” que aparece antes de pulsar en algún elemento del carrusel.

## Respuesta a eventos

Se pide responder a tres eventos diferentes a lo largo de la página web. Estos son los diferentes eventos a los que respondo para realizar cambios dinámicos. Todos están explicados con detalle en el código mediante comentarios:

- `click`: En el archivo `they.js`, este evento es necesario para cambiar el elemento del carrusel que se centra al pulsar en el botón correspondiente

- `scroll`: También en `they.js`, se utiliza para detectar en todo momento el elemento de la lista que está centrado en la pantalla.
- `submit`: En el archivo `contact.js`, para mostrar un mensaje de alerta con la información introducida en el formulario una vez se pulsa en el botón de enviar.
- `load`: En `take-them.js` y en `they.js` para diferentes inicializaciones según el archivo.

## jQuery

jQuery es una biblioteca de JavaScript que simplifica la manipulación del DOM, el manejo de eventos, las animaciones y las interacciones Ajax. Sus principales características incluyen:

- Manipulación del DOM: Facilita la adición, eliminación y modificación de elementos HTML.
- Manejo de eventos: Simplifica agregar manejadores de eventos a los elementos.
- Animaciones y efectos: Proporciona funciones para aplicar animaciones y efectos visuales.
- Ajax: Facilita la realización de llamadas Ajax para cargar datos asíncronamente.
- Compatibilidad entre navegadores: Resuelve inconsistencias entre distintos navegadores.

Aunque jQuery fue muy popular, su uso ha disminuido con el desarrollo de JavaScript moderno y frameworks como Angular, React y Vue.js. Sin embargo, sigue siendo útil para proyectos que requieren una rápida manipulación del DOM sin la complejidad de un framework completo.

En esta web, jQuery se utiliza para dos acciones principales:

- Para cargar la cabecera y el pie de página en todas las páginas del sitio web. Esto se hace en el `index.js` .

- Para leer el documento JSON que contiene información que luego se muestra tanto en el mapa como en el acordeón en la página de “Take them”. Esto se puede ver en el archivo `take-them.js`, en las funciones `getShops` y `getVets`.

El código está debidamente comentado, igual que anteriormente.

## ES6

ES6, también conocido como ECMAScript 2015, es una versión significativa de JavaScript que introdujo varias características nuevas y mejoradas, marcando un gran avance en la forma de escribir JavaScript. Entre sus características más destacadas se encuentran:

- `let` y `const`: Nuevas formas de declarar variables, proporcionando alcance de bloque y constantes inmutables.
- Plantillas de cadenas (“template literals”): Para cadenas de texto multi-linea y con interpolación de variables.
- Funciones “arrow”: una sintaxis más corta para las funciones y un manejo diferentes de `this`.
- Clases: Sintaxis de clases para la creación de objetos, facilitando la programación orientada a objetos.
- Módulos: Sintaxis nativa para importar y exportar valores entre diferentes archivos JavaScript.
- Parámetros por defecto y operador de propagación: Mayor flexibilidad en la definición de funciones y manejo de arrays y objetos.
- Promesas. Una forma más eficiente de manejar operaciones asíncronas.
- Desestructuración: Acceso más directo y legible a elementos de arrays y propiedades de objetos.

ES6 ha mejorado significativamente la legibilidad, mantenibilidad y eficiencia del código JavaScript, siendo esencial para el desarrollo moderno de aplicaciones web.

Su uso a lo largo del sitio web es evidente y esencial para dotar de dinamismo a la página.

## Carga de contenido

Se pide cargar contenido para introducir en la página web utilizando dos tipos de archivos diferentes y, a su vez, utilizando dos técnicas distintas de entre las tres explicadas.

El contenido que se ha querido incluir en la página a través de la carga de datos es el relacionado con el mapa, que muestra ubicaciones de lugares como refugios y tiendas para animales. Por lo tanto, se puede encontrar el código correspondiente en el archivo `take-them.js`. Estos archivos podrían representar una base de datos de ubicaciones en las que encontrar este tipo de lugares.

Los archivos que se cargan se encuentran en el directorio de recursos, dentro del directorio de datos.

### XML

XML, que significa eXtensible Markup Language, es un lenguaje de marcado utilizado para codificar documentos de manera que sean tanto legibles por humanos como por máquinas. Fue diseñado para almacenar y transportar datos, enfocándose en lo que los datos son, más que en cómo se muestran. Sus características principales son:

- Estructurado: Utiliza etiquetas para definir elementos y estructurar los datos.
- Auto-descriptivo: Los nombres de las etiquetas en XML son definidos por el usuario, lo que hace que los datos sean comprensibles ya que cada etiqueta describe el tipo de contenido que contiene.
- Jerárquico: Permite representar datos complejos y anidados a través de una estructura jerárquica de elementos.
- Transportable: Al ser un formato basado en texto, es independiente del software y del hardware, lo que facilita el intercambio de datos entre sistemas incompatibles.

XML es comúnmente utilizado en aplicaciones donde la interoperabilidad entre diferentes sistemas es esencial, como en servicios web, configuraciones de software y en la transmisión de datos en Internet. Aunque ha sido parcialmente eclipsado por formatos más modernos como JSON en algunos usos, XML sigue siendo importante

en muchos entornos empresariales y tecnológicos debido a su flexibilidad y soporte extenso.

En esta página web se cargan datos de un archivo XML para obtener los distintos refugios y protectoras que se muestran posteriormente en el mapa. Se utiliza para conseguir esto XMLHttpRequest, que se trata de un objeto que permite a los navegadores realizar solicitudes HTTP a servidores web. Se usa principalmente para intercambiar datos entre un cliente y un servidor sin necesidad de recargar la página completa. Este objeto es clave para la funcionalidad de AJAX (Asynchronous JavaScript And XML), lo que permite la actualización de contenidos de una página web de forma asíncrona.

Las funciones en las que se utiliza dicho objeto, dentro del archivo mencionado antes, son `getProtectorsRefugees` y `processProtectorsRefugees`. Para procesar los elementos XML que se encuentran en el archivo, existe la función `processXMLElement`.

## JSON

JSON, que significa JavaScript Object Notation, es un formato ligero de intercambio de datos. Es fácil de leer y escribir para los humanos y fácil de parsear y generar para las máquinas. JSON se basa en dos estructuras:

- Una colección de pares de clave/valor. En JSON, un objeto se delimita con `{}' y contiene una lista de pares de clave/valor separados por comas. Cada nombre de propiedad (clave) es seguido por dos puntos y el valor asignado a esa propiedad.
- Una lista ordenada de valores. En JSON, un array se representa con corchetes `[]` y contiene una lista de valores separados por comas.

JSON es independiente del lenguaje: aunque se deriva de la notación de objetos de JavaScript, puede ser utilizado con muchos lenguajes de programación modernos. Es comúnmente usado para transmitir datos en aplicaciones web (entre cliente y servidor), debido a su simplicidad, eficiencia y su amplia compatibilidad con sistemas y lenguajes de programación.

Para leer el archivo JSON que contiene los datos de ubicaciones de veterinarios y tiendas, se ha utilizado jQuery, utilizando la función `\$.getJSON`. Esta es una función que facilita la realización de solicitudes AJAX para obtener datos en formato JSON. Es una forma abreviada y conveniente del método `\$.ajax` de jQuery, específicamente ajustado para manejar respuestas JSON. La función toma una URL desde donde cargar los datos JSON y una función de callback que se ejecuta una vez que los datos son recibidos y parseados. Este método es ampliamente utilizado para obtener datos de forma asíncrona sin recargar la página web, lo que permite una interactividad fluida y dinámica en aplicaciones web.

Las funciones en las que se utiliza la función mencionada, también en el archivo `take-them.js`, son `getShops` y `getVets`. A mayores, existen las funciones `processShops` y `processVets` que utilizan la función `processJSONElement` para obtener la información requerida del archivo JSON.

## Nota

Por alguna razón, una vez terminada la página y el informe, me he dado cuenta de que he confundido la palabra “tails” con “tiles”. Por lo que en algunas imágenes no aparece el título como debería estar.

Siento el despiste.