

History of War

Carpintero Díaz David, Seijo García Pablo

March 31, 2024

Contents

1 History of War: Un camino por la historia	3
2 Inventario de Contenido	4
3 Arquitectura de la Información	5
4 Casos de Uso Típicos	6
5 Mapa de Navegación	8
6 Interfaces	9
6.1 Manual (Boceto Manual)	9
6.2 Wireframes	10
6.3 Mockups	14
7 Storyboards	20
8 Estructura de Ficheros y Carpetas	23
9 HTML	25
9.1 Mapa de etiquetas	25
9.2 Estructura de ficheros actualizada	32
10 CSS	33
10.1 Flexible Grids	33
10.2 CSS Multicol	34

10.3 Flex Container	35
10.4 CSS Grid	36
10.5 Bootstrap	37
11 JavaScript	39
11.1 Funcionalidades	39
11.1.1 Header	39
11.1.2 Mapa	41
11.1.3 Buscador	44
11.1.4 Foro	47
12 Estructura de datos final	54
13 De la idea primigenia a la página web final	55

1 History of War: Un camino por la historia

Nuestra página web está dedicada a la historia bélica del mundo. La página web cuenta con diferentes secciones con información variada. La sección más llamativa es un mapa interactivo que sirve como portal a las páginas de las batallas más significativas de todas las épocas. Este mapa permite explorar visualmente el desarrollo de conflictos, desde las antiguas guerras hasta los enfrentamientos contemporáneos, brindando una perspectiva única y global de la historia militar.

También cuenta con diversos apartados temáticos, donde se encuentra información detallada sobre guerras que han moldeado naciones enteras, batallas que han cambiado el curso de la historia y personajes legendarios cuyas acciones resonaron a lo largo del tiempo. Cada sección está cuidadosamente elaborada para ofrecer datos exhaustivos y análisis profundos, proporcionando un contexto completo para comprender el trasfondo y la importancia de cada acontecimiento.

Además contamos con una biblioteca, que cuenta con una selección de libros recomendados, cada uno abordando distintas épocas de la historia bélica con profundidad. Cada recomendación va acompañada de una breve pero informativa descripción, que te brindará una visión general del contenido y del enfoque particular del libro.

2 Inventario de Contenido

Con el objetivo de establecer una estructura clara y organizada para el desarrollo de nuestra página web, hemos procedido a la elaboración de un inventario de contenido. Dicho inventario actúa como un esquema preliminar que facilitará la implementación final del sitio web.

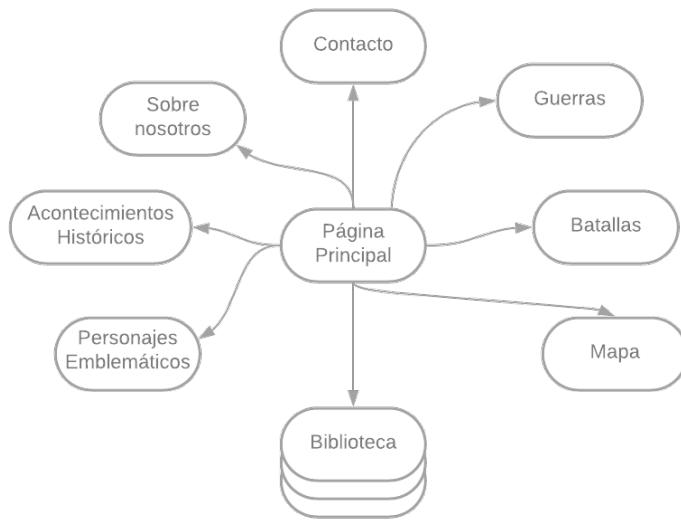


Figure 1: Inventario de Contenido

Nuestra plataforma se articula en diversas secciones, cada una con un propósito específico y contenido diferenciado. De particular relevancia es la sección denominada "Biblioteca", la cual albergará una selección cuidadosamente curada de libros recomendados. Asimismo, destacamos la inclusión de un mapa interactivo, diseñado para guiar a los usuarios a través de diversos episodios históricos, permitiéndoles explorar las distintas batallas que han marcado el curso del tiempo. Adicionalmente, la sección dedicada a las guerras ofrecerá una visión exhaustiva sobre los múltiples conflictos bélicos que han tenido lugar a lo largo de la historia, proporcionando información detallada y contextos relevantes.

3 Arquitectura de la Información

Basándonos en el esquema preliminar previamente delineado, procedemos a desarrollar la arquitectura de la información. Este proceso nos facilitará la organización de las ideas iniciales en una estructura jerárquica, la cual establecerá la profundidad y el alcance de cada sección dentro de nuestra página web.

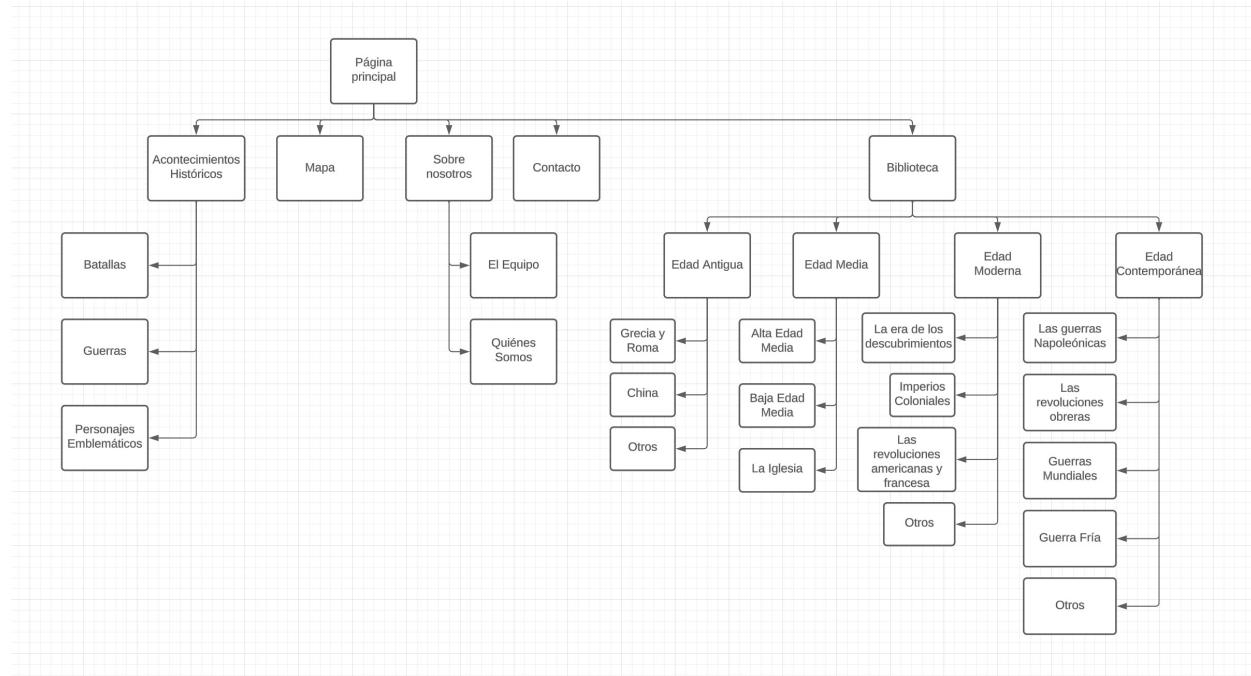


Figure 2: Arquitectura de la Información

Dentro de esta arquitectura, la sección más amplia corresponde a la biblioteca. Esta área está meticulosamente segmentada en diversas subsecciones que reflejan las distintas eras históricas, abarcando desde la Edad Antigua, iniciando en el año 3000 a.C., hasta la contemporaneidad. Esta organización no solo facilita la navegación sino que también enriquece la experiencia educativa del usuario al proporcionar un contexto histórico claro y bien definido.

Adicionalmente, cabe resaltar la sección denominada "Acontecimientos Históricos", la cual funciona como categoría principal para temas relacionados con guerras, batallas y figuras históricas destacadas. Esta categorización permite una exploración temática coherente y detallada de los eventos que han moldeado la historia humana.

En conjunto, la arquitectura de la información diseñada busca optimizar la usabilidad y accesibilidad del sitio web, asegurando que los usuarios puedan navegar de manera intuitiva y obtener información de forma eficiente.

4 Casos de Uso Típicos

Tras definir la estructura de nuestro sitio web, presentaremos los casos de uso comunes, los cuales mostrarán las acciones que los usuarios pueden realizar al visitar nuestra página.

Esta parte es esencial para entender cómo los visitantes interactúan con el sitio, desde explorar las diferentes secciones hasta acceder a contenidos específicos. El objetivo es proporcionar una visión clara de lo que se espera que los usuarios hagan y cómo pueden aprovechar al máximo los recursos disponibles en la web.

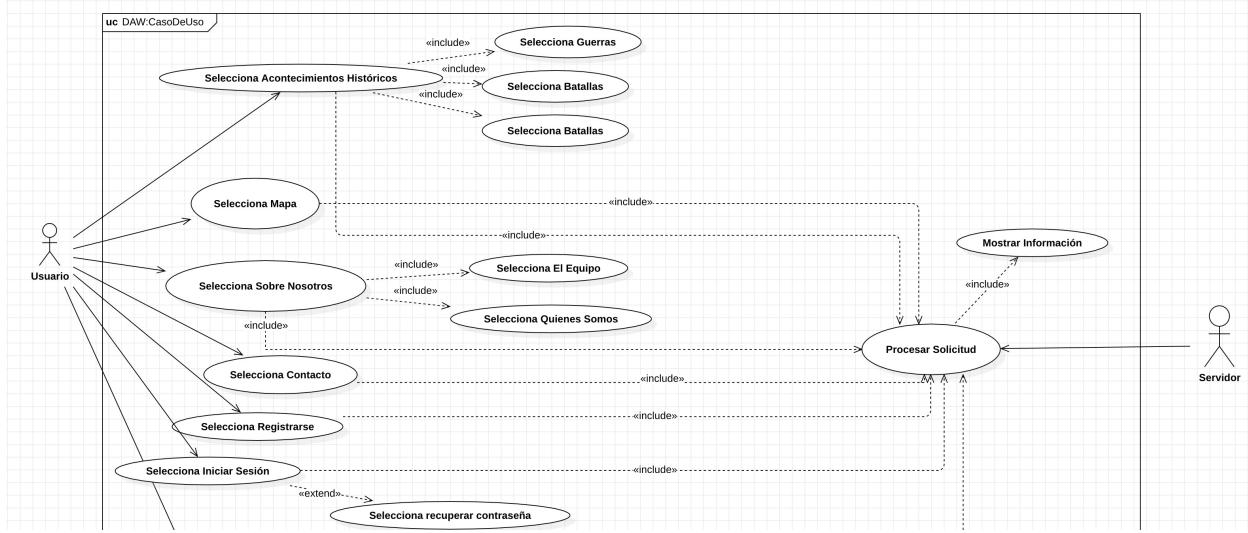


Figure 3: Casos de Uso Típicos (1)

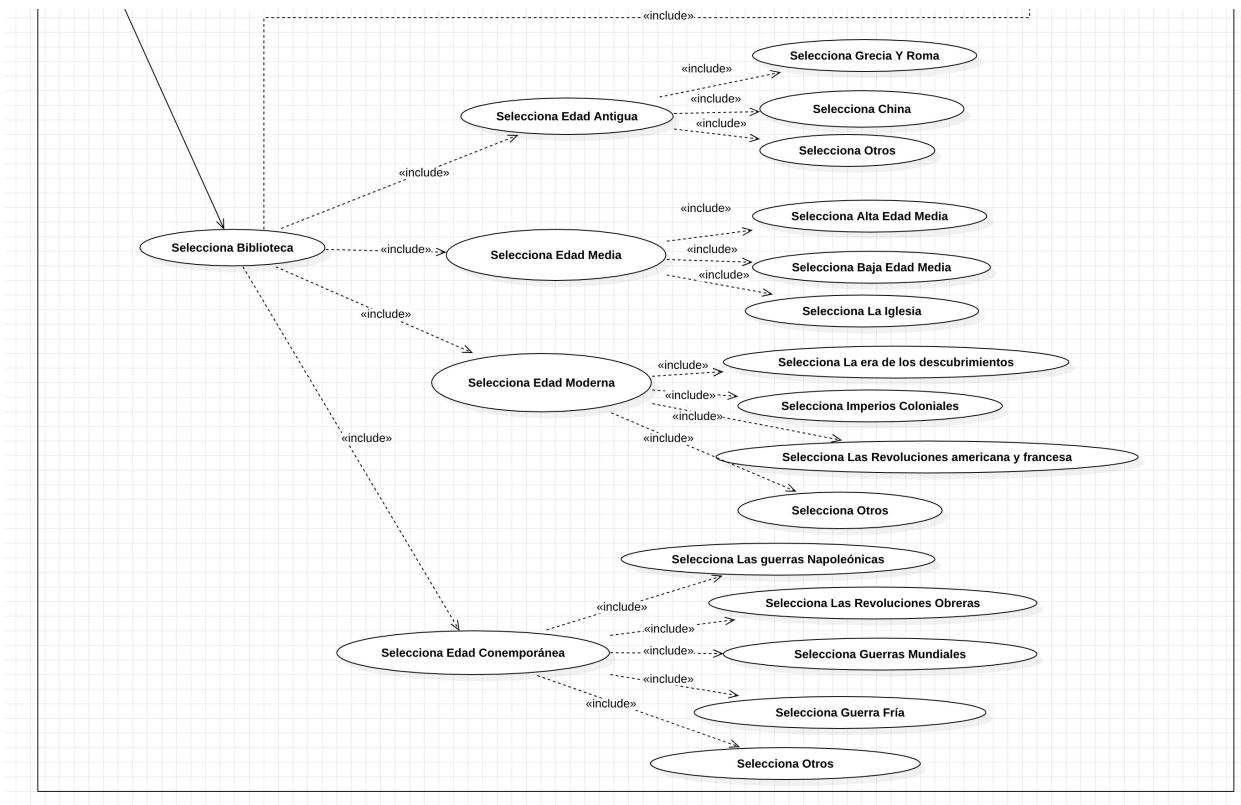


Figure 4: Casos de Uso Típicos (2)

5 Mapa de Navegación

El mapa de navegación es un elemento clave en el diseño de nuestro sitio web, que actúa como una guía visual para estructurar y presentar la forma en que las distintas páginas y secciones están interconectadas. Este mapa facilita la comprensión de la organización del sitio, permitiendo a los usuarios y desarrolladores visualizar la jerarquía de la información y las rutas de acceso disponibles para navegar por el contenido. Al ofrecer una representación clara de cómo se organizan y relacionan los distintos componentes del sitio, el mapa de navegación mejora la usabilidad y la experiencia de usuario, asegurando que la información sea accesible y fácil de encontrar.

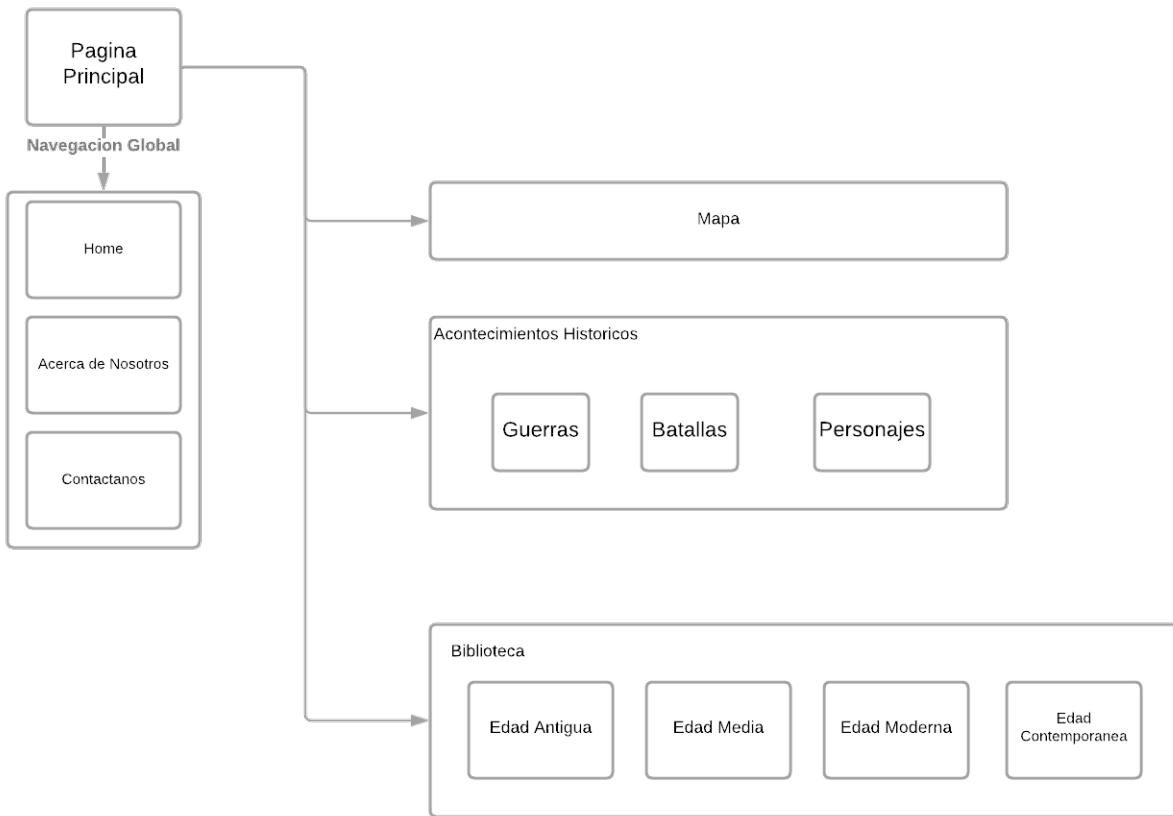


Figure 5: Mapa de Navegación

6 Interfaces

6.1 Manual (Boceto Manual)

Se trata de un boceto manual de como nos imaginamos la página web, de tal manera que basaremos la evolución de la página basandonos en este concepto primigenio de las páginas.

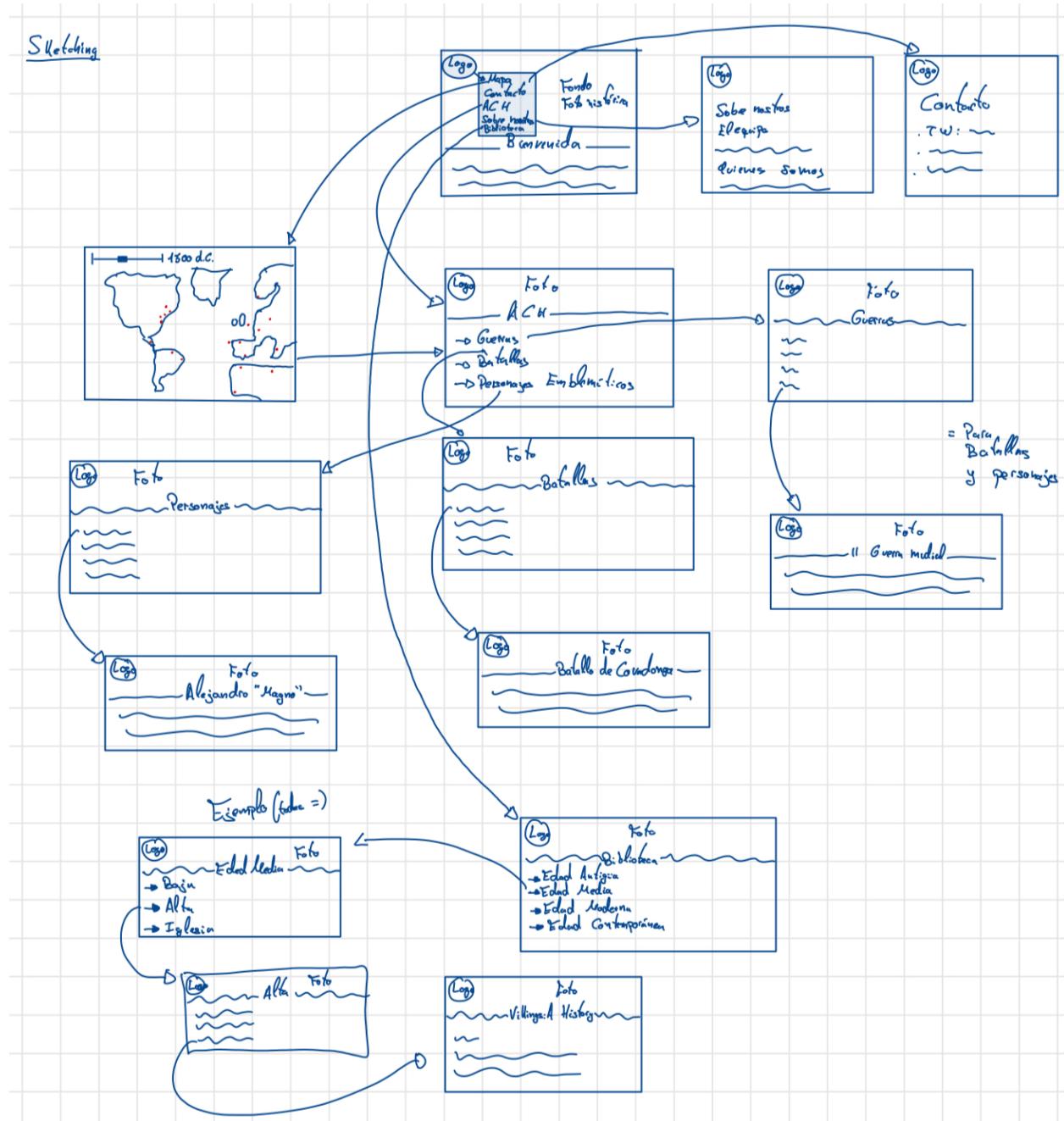


Figure 6: Boceto Manual

6.2 Wireframes

Los wireframes constituyen un conjunto de esquemas visuales que representan de forma preliminar el diseño y la estructura de nuestra página web. Estos bocetos iniciales son fundamentales para conceptualizar la disposición de los elementos en las páginas, sirviendo como base para el desarrollo y refinamiento posterior del sitio. A través de estos wireframes, establecemos un marco de referencia que guiará las fases subsiguientes del diseño, asegurando que la visión original se mantenga coherente a lo largo del proceso de desarrollo.

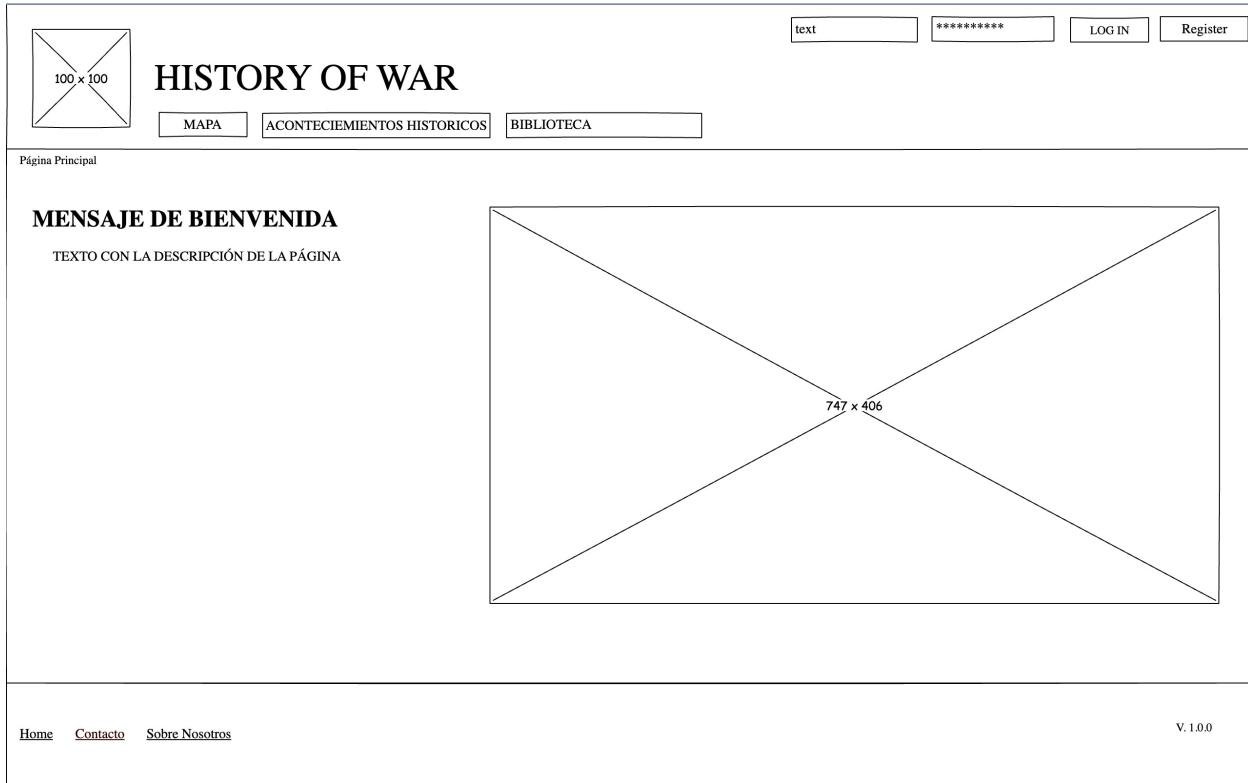


Figure 7: Página Principal

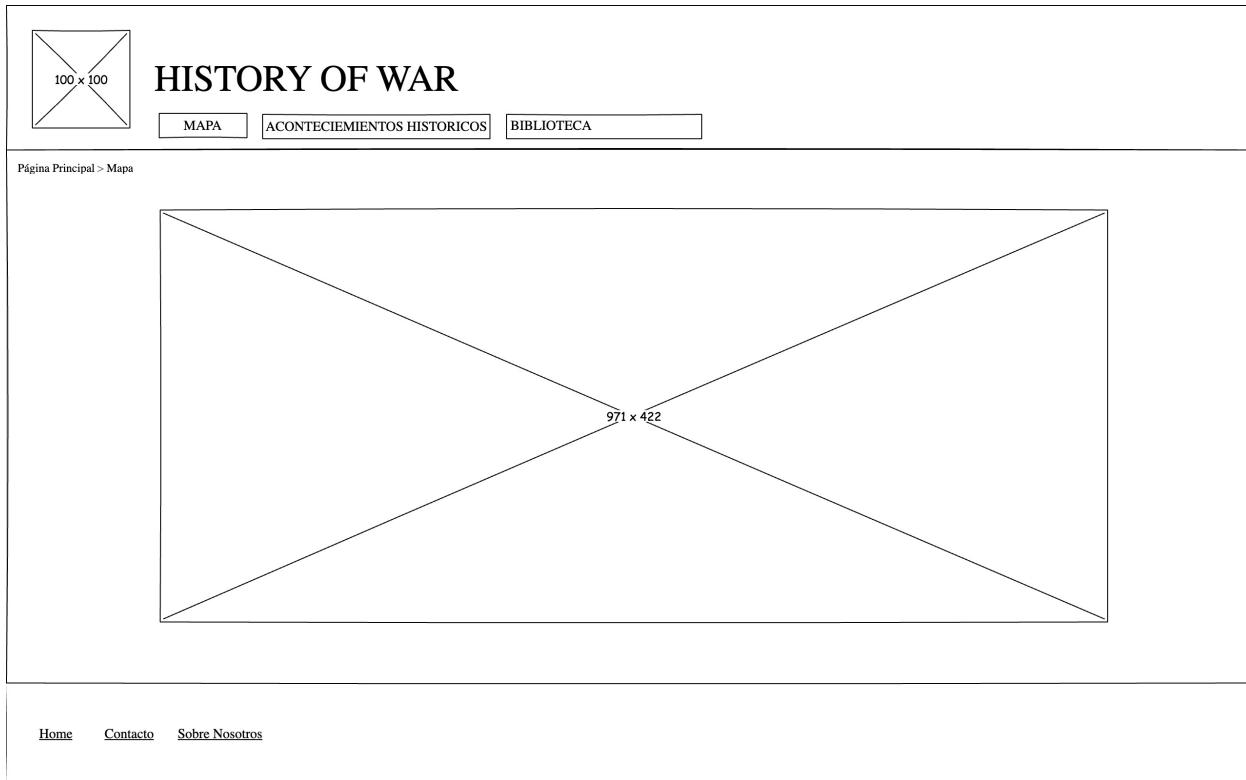


Figure 8: Mapa

The screenshot shows a web page titled "HISTORY OF WAR". In the top left corner is a logo consisting of a square with a diagonal cross and the text "100 x 100". Below the title are three buttons: "MAPA", "ACONTECIMIENTOS HISTORICOS", and "BIBLIOTECA". A breadcrumb navigation bar at the top indicates the page's path: "Página Principal > Acontecimientos Históricos > Batallas > Batalla de Stalingrado". The main content area is titled "BATALLA DE STALINGRADO" and features a large, empty rectangular frame with a diagonal cross, labeled "516 x 255". Below this frame is the text "TEXTO CON LA DESCRIPCIÓN DE LA BATALLA". At the bottom of the page, there are links for "Home", "Contacto", and "Sobre Nosotros" on the left, and "V.1.0.0" on the right.

Figure 9: Ejemplo Batalla

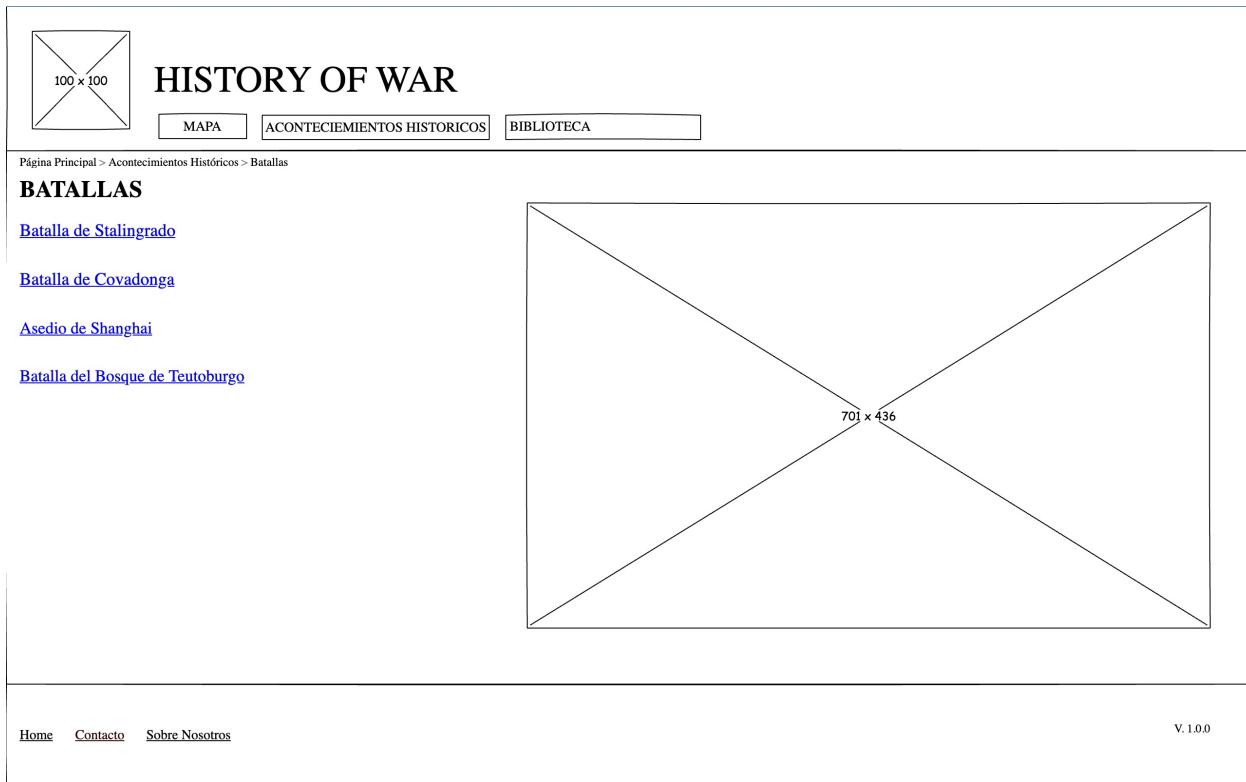


Figure 10: Batallas

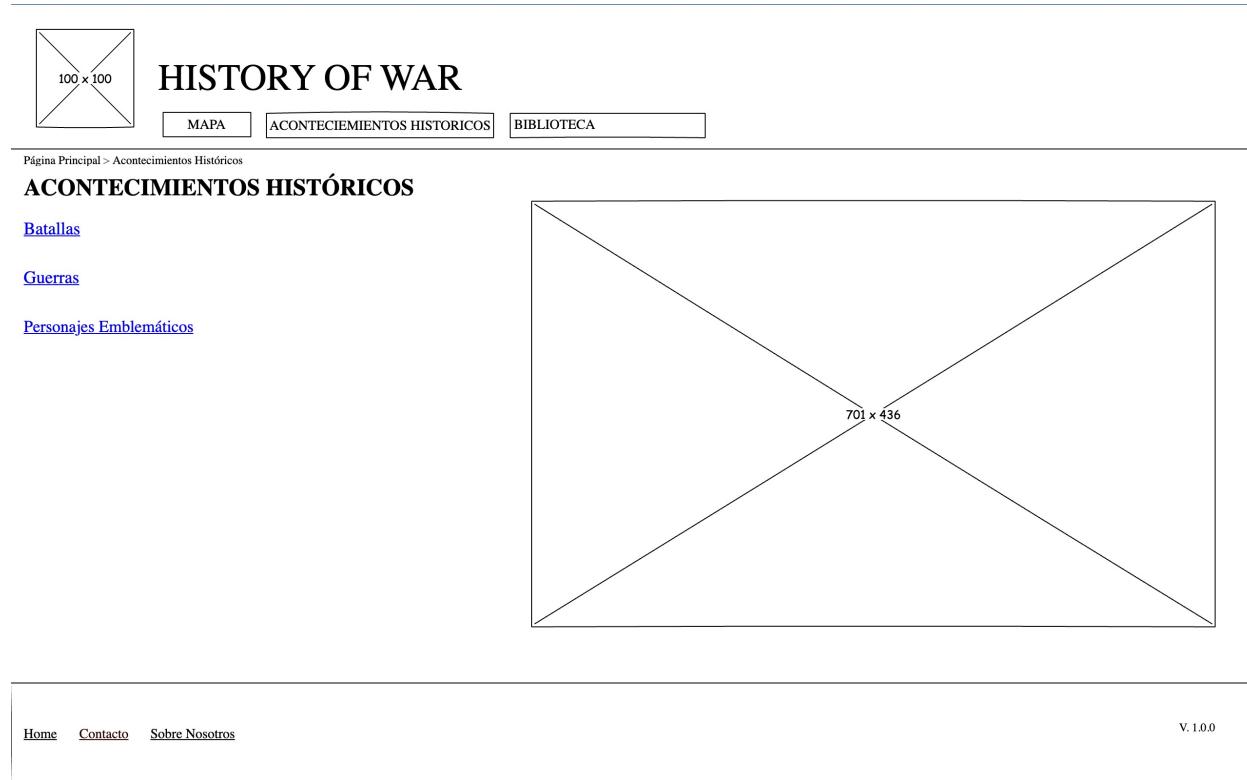


Figure 11: Acontecimientos Históricos

6.3 Mockups

Los mockups son representaciones visuales de alta fidelidad que avanzan sobre la base establecida por los wireframes, ofreciendo una versión más detallada y cercana al diseño final de nuestra página web. Estas maquetas incorporan elementos de diseño gráfico, paletas de colores, tipografías y otros componentes estéticos, proporcionando una vista previa más precisa de cómo se verá y sentirá el sitio web una vez completado. Funcionan como una herramienta esencial en el proceso de diseño, facilitando la evaluación y ajuste del aspecto visual y la experiencia de usuario antes de entrar en las etapas de desarrollo técnico.



Figure 12: Página Principal



Figure 13: Mapa

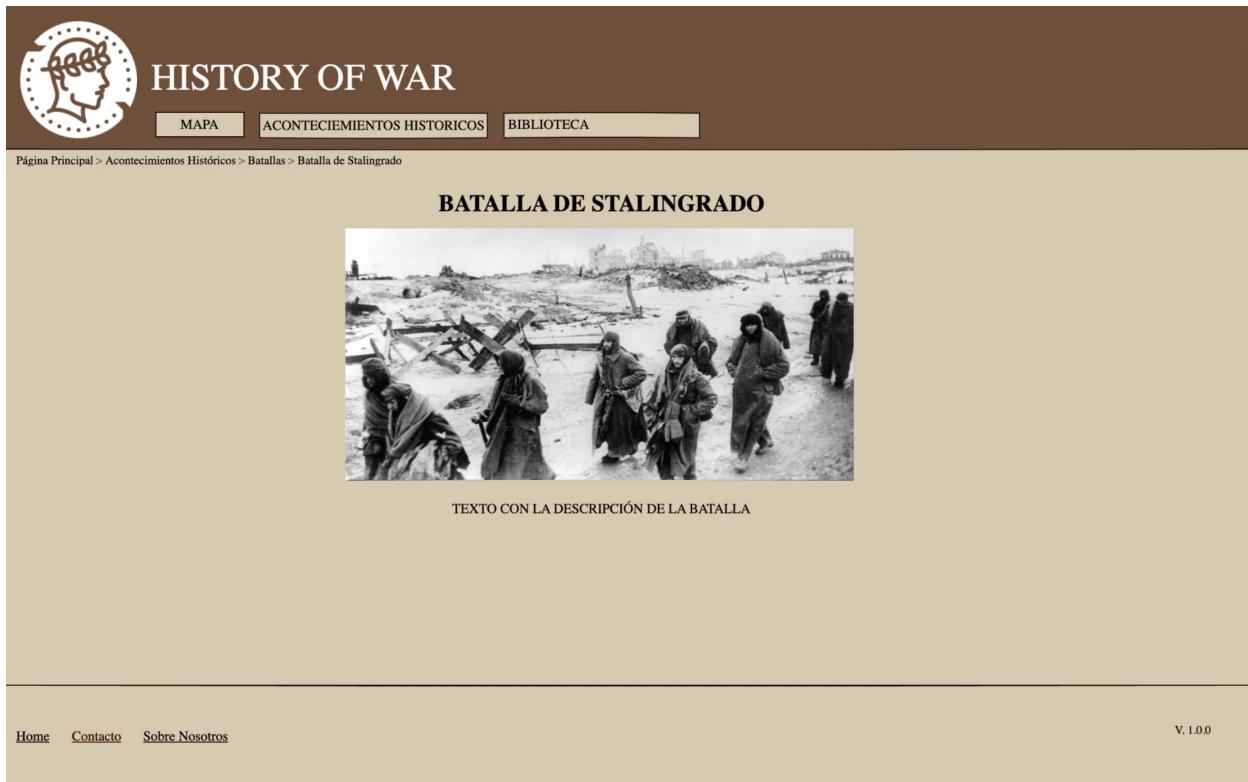


Figure 14: Ejemplo Batalla

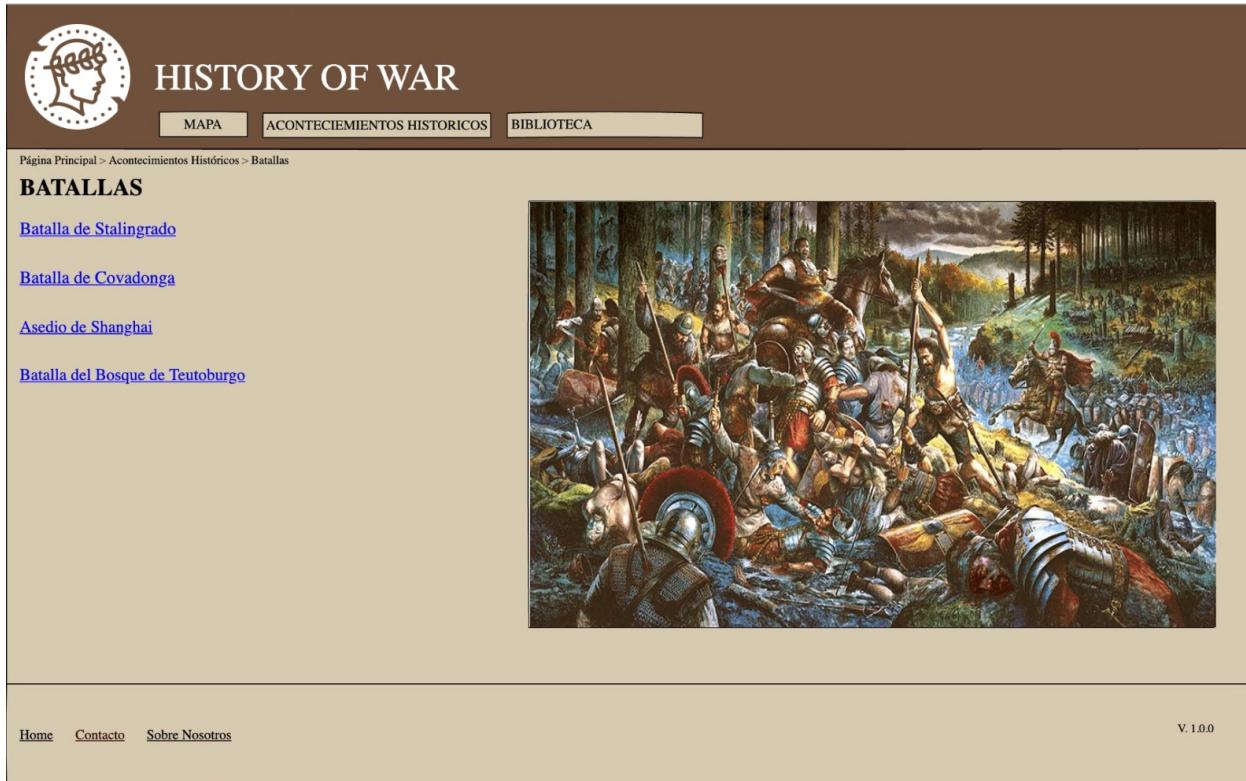


Figure 15: Batallas

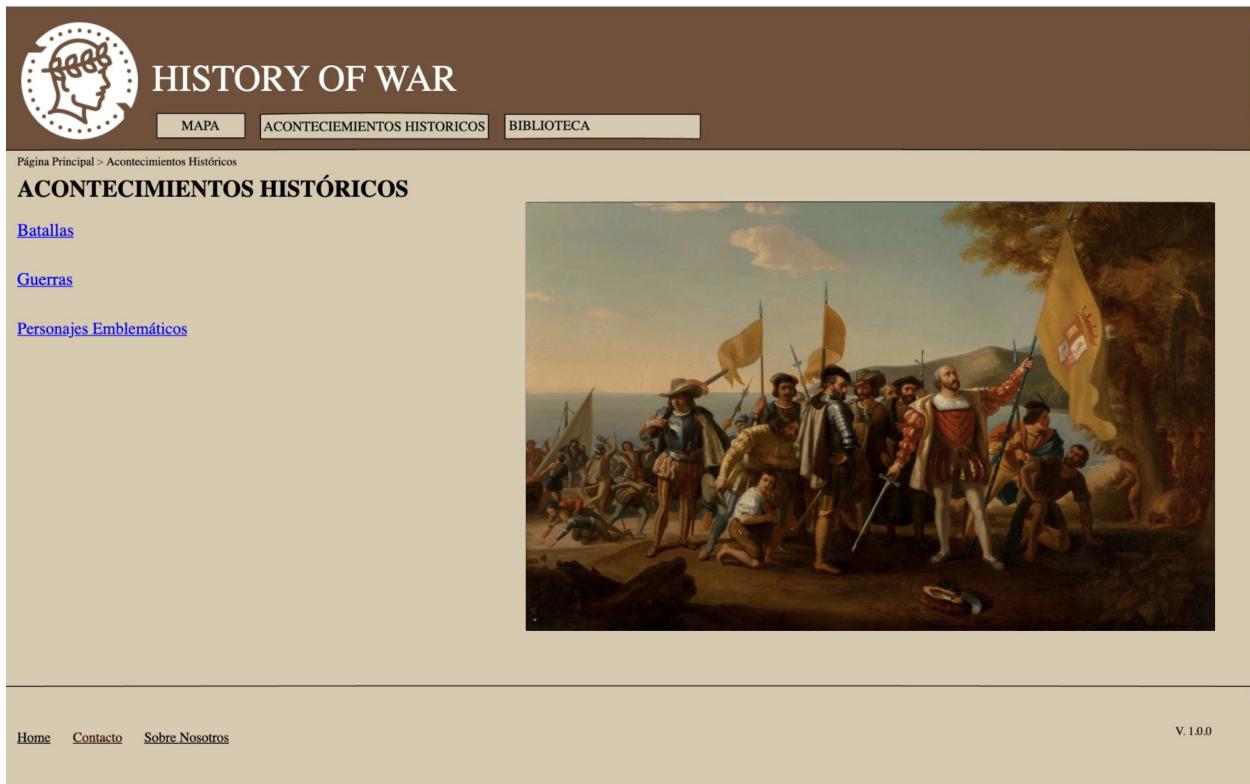


Figure 16: Acontecimientos Históricos

7 Storyboards

Son secuencias narrativas visuales que ilustran el flujo y la experiencia del usuario al interactuar con la página web. Estos diagramas cuentan la "historia" de cómo un usuario típico podría navegar a través del sitio, desde la página de inicio hasta la realización de acciones específicas, como realizar una compra, inscribirse en un servicio o encontrar información relevante.

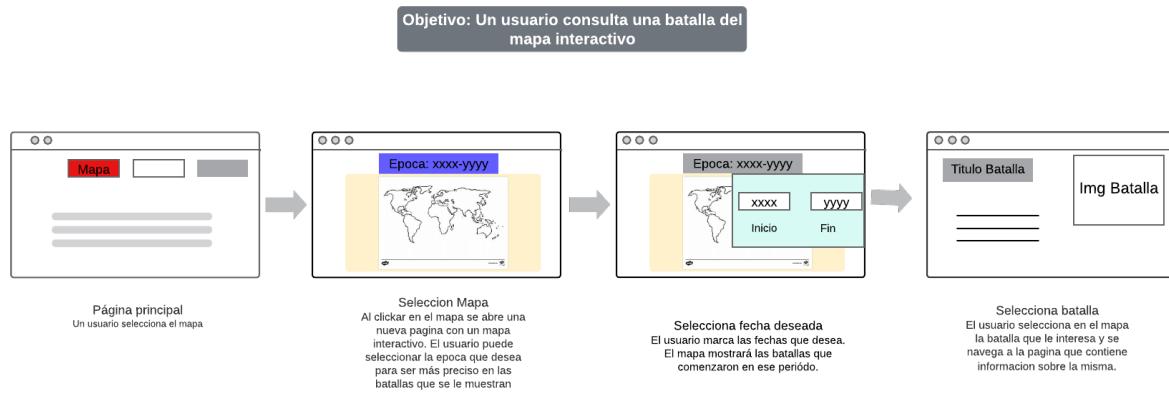


Figure 17: Selección de Batalla

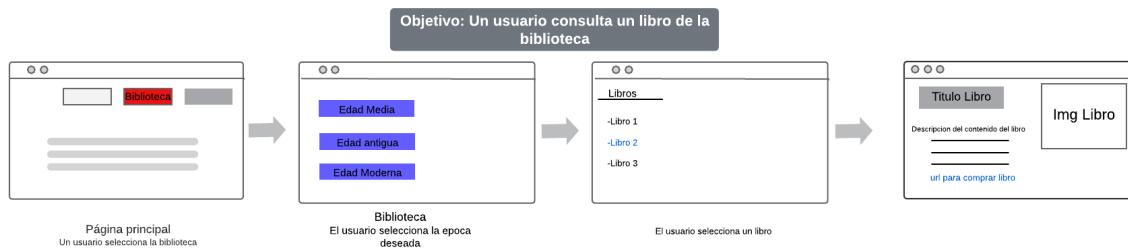


Figure 18: Selección de Libro

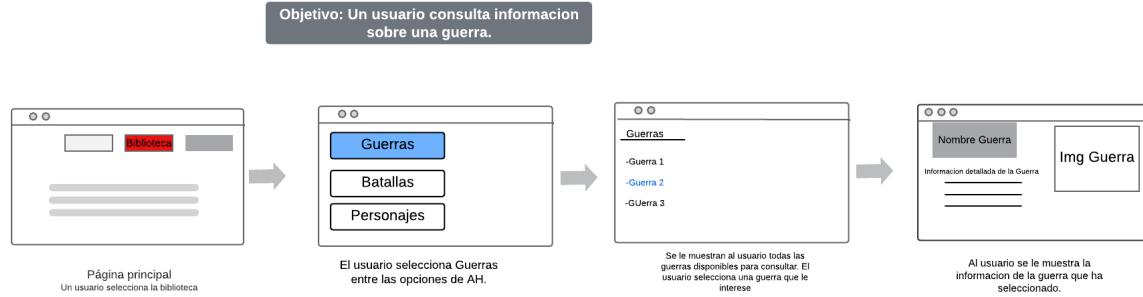


Figure 19: Selección de Guerra

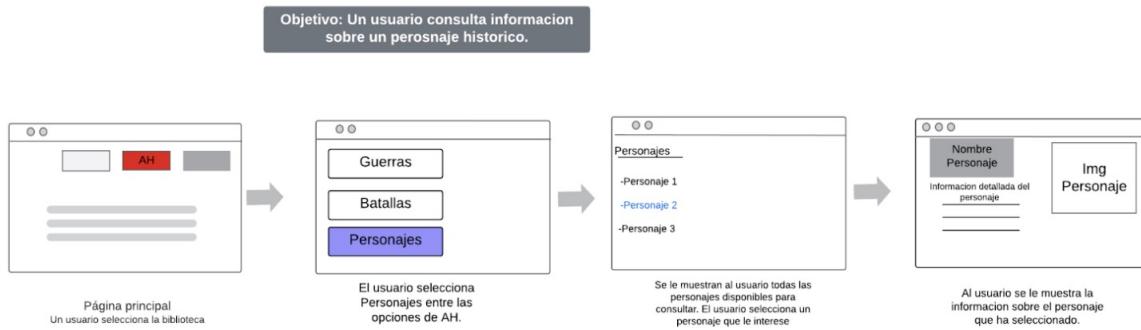


Figure 20: Selección de Personaje

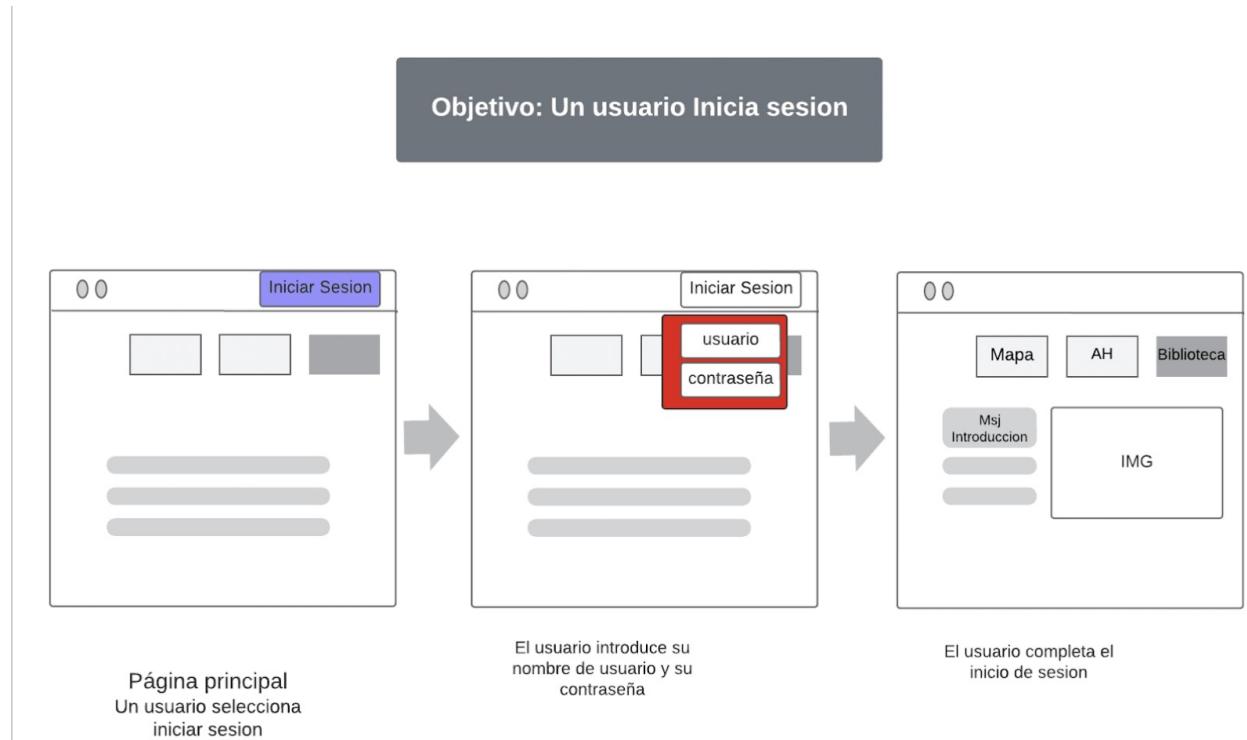


Figure 21: Inicio de Sesión

8 Estructura de Ficheros y Carpetas

La estructura de ficheros y carpetas en un proyecto web es esencial para mantener el orden y facilitar el desarrollo y mantenimiento del sitio. Esta organización comprende la disposición sistemática de archivos HTML, hojas de estilo CSS, scripts de JavaScript, imágenes, fuentes y otros recursos digitales.

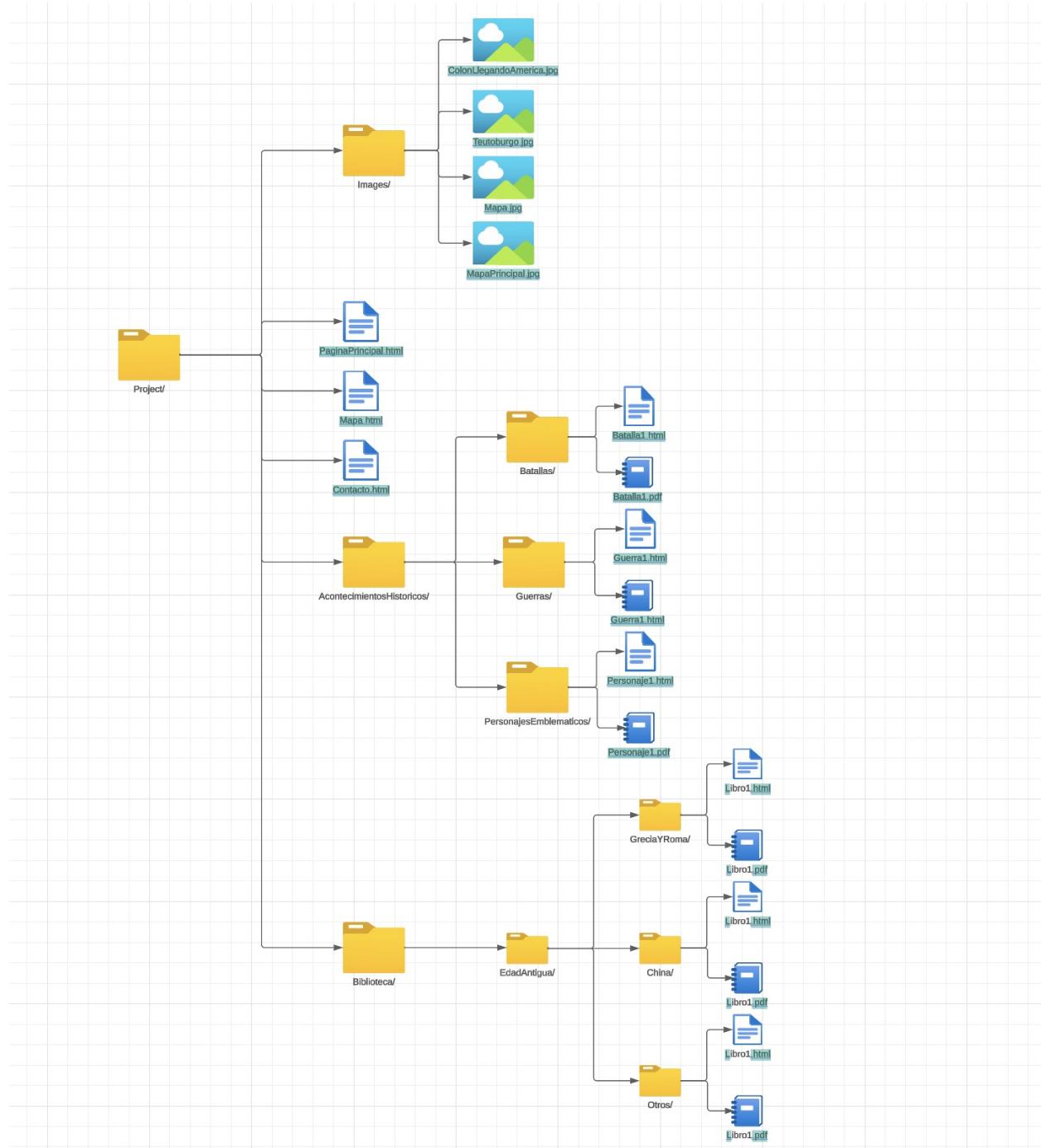


Figure 22: Estructura de Ficheros y Carpetas

Este esquema nos proporciona una vista general de como será la estructura que compondrá la página web, siendo extendida en la parte de la biblioteca, donde solo hemos metido una de las carpetas para ilustrar como será para todas ellas.

9 HTML

Con la evolución del proyecto, las páginas modeladas al comienzo de este también fueron sujetas a cambios, por lo que en esta sección presentamos las páginas HTML finales, de las cinco partes más significativas de nuestra web.

9.1 Mapa de etiquetas

index.html

Esta es la página principal de nuestra web, donde se muestra un mensaje de bienvenida y se da acceso a las distintas secciones de la página.

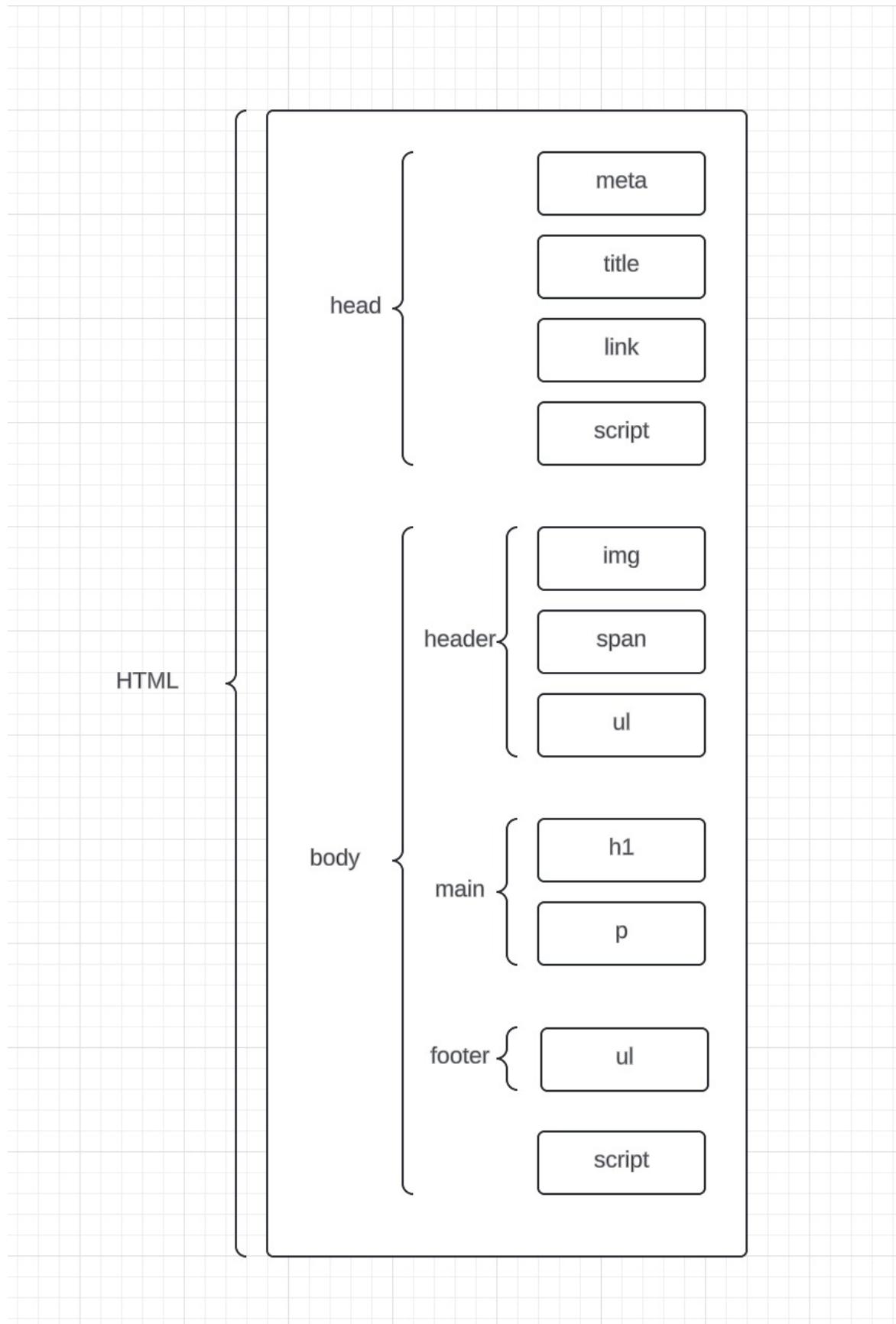


Figure 23: Interfaz de index.html



Figure 24: Prototipo de interfaz de index.html

Figure 25: a) Interfaz de index.html y b) Prototipo de interfaz de index.html



mapa.html

En esta página se muestra un mapa interactivo que permite al usuario acceder a las distintas batallas de la historia. Para simplificar la visualización, se omite la parte del header y del footer, pues es la misma para todas las páginas.

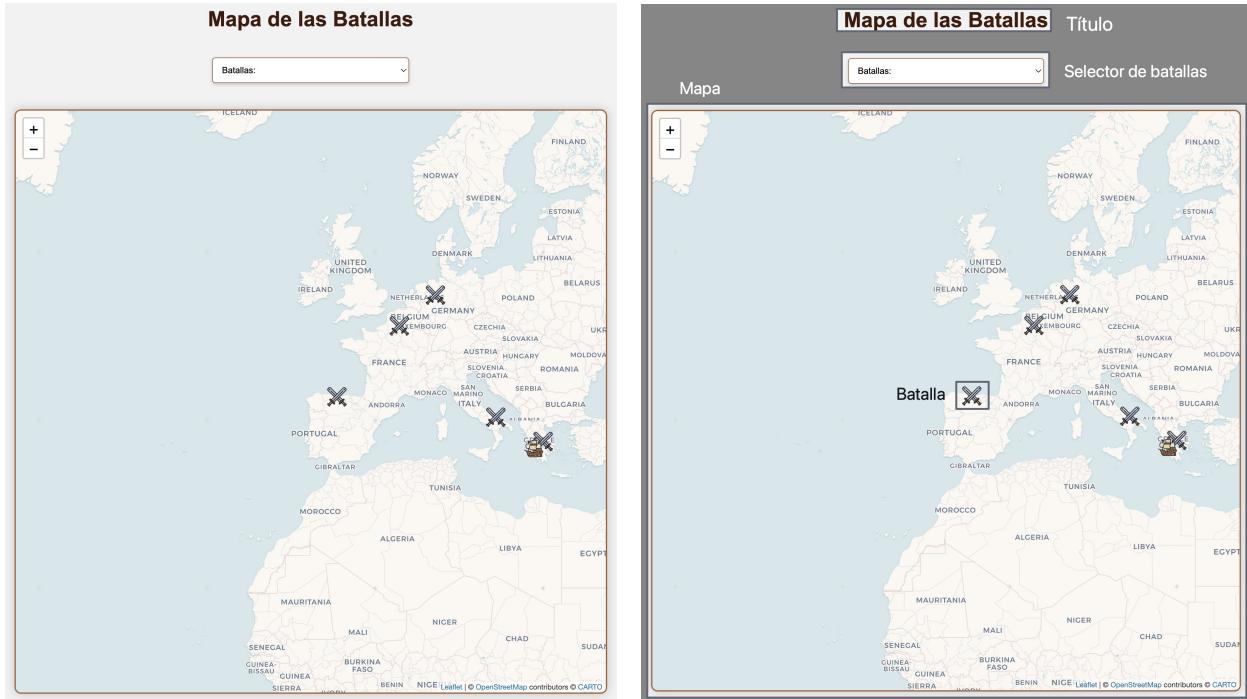


Figure 27: a) Interfaz de AH.html b) Prototipo de interfaz de AH.html c) Mapa de etiquetas de AH.html

AH.html, batallas.html, guerras.html y personajesEmblematicos.html

Para ver como se descomponen estas secciones de nuestra web, tomaremos como ejemplo la página de Acontecimientos Históricos, aunque las otras páginas siguen un esquema similar. Como en el caso anterior, se omite la parte del header y del footer, pues es la misma para todas las páginas.

ACONTECIMIENTOS HISTÓRICOS



El milagro de Empel
En diciembre de 1585, sitiado el tercio en la Isla de Bommel, un soldado español encuentra una tabla flamenca con la virgen. Justo esa noche se congela el río que los sitiaba y derrotan a los holandeses.

[Guerras](#)
[Batallas](#)
[Personajes Emblemáticos](#)

ACONTECIMIENTOS HISTÓRICOS

Título

Carousel



Descripción
El milagro de Empel
En diciembre de 1585, sitiado el tercio en la Isla de Bommel, un soldado español encuentra una tabla flamenca con la virgen. Justo esa noche se congela el río que los sitiaba y derrotan a los holandeses.

Lista de navegación

Guerras	Opciones de navegación
Batallas	
Personajes Emblemáticos	

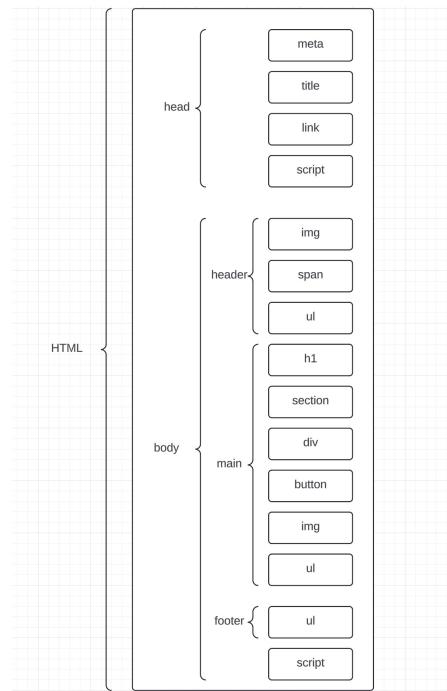


Figure 28: a) Interfaz de mapa.html b) Prototipo de interfaz de mapa.html c) Mapa de etiquetas de mapa.html

LibrosEA.html, LibrosEC.html, LibrosEM.html, LibrosEMed.html

Estas páginas son las que contienen la información de los libros recomendados para cada época histórica. A continuación se muestra la estructura de la página de libros de la Edad moderna, aunque las demás páginas siguen un esquema similar. Nos abstendremos, de nuevo, de mostrar la parte del header y del footer.

LIBROS EDAD MODERNA

EL SIGLO MALDITO

GEOFFREY PARKER

CLIMA, GUERRAS Y CATÁSTROFES EN EL SIGLO XVI

Resumen: Revueltas, sequías, hambrunas, invasiones, guerras, regicidios... Los desastres que se sucedieron en la segunda mitad del siglo XVII no sólo no tenían precedentes, sino que se propagaron por el globo de una forma atroz.

[Comprar Libro](#)

LA CONSPIRACIÓN DE LA POLVORA

ANTONIA FRASER

T

Resumen: En 1605, en Inglaterra, se desarrolló una trama terrorista católica para volar el Parlamento y derrocar al rey protestante. Fraser utiliza una narrativa intrigante que convierte a los personajes históricos en vidas vivas. Investigando archivos europeos, revela las complejas motivaciones políticas y religiosas detrás de la conspiración y la represión católica posterior.

[Comprar Libro](#)

Un tiempo de guerras: Una historia alternativa de Europa 1450-1700

LAURO MARTINES

Martínes cuenta la visión de las guerras de gran parte de la historia moderna desde el punto de vista del tercero de abajo. Es decir, del Tercer Estado: soldados, campesinos, burgueses, etc.

[Comprar Libro](#)

Pioneros del capitalismo "Los Países Bajos 1000-1800

MAARTEN PRAK

JAN LUIJEN VAN ZANDEN

LOS PAÍSES BAJOS 1000-1800

Resumen: Martínes cuenta la visión de las guerras de gran parte de la historia moderna desde el punto de vista del tercero de abajo. Es decir, del Tercer Estado: soldados, campesinos, burgueses, etc.

[Comprar Libro](#)

LIBROS EDAD MODERNA

Contenedor de libro	Título
El siglo maldito Título Libro	La conspiración de la polvora
<small>Revueltas, sequías, hambrunas, invasiones, guerras, regicidios... Los desastres que se sucedieron en la segunda mitad del siglo XVII no sólo no tenían precedentes, sino que se propagaron por el globo de una forma atroz.</small>	<small>En 1605, en Inglaterra, se desarrolló una trama terrorista católica para volar el Parlamento y derrocar al rey protestante. Fraser utiliza una narrativa intrigante que convierte a los personajes históricos en vidas vivas. Investigando archivos europeos, revela las complejas motivaciones políticas y religiosas detrás de la conspiración y la represión católica posterior.</small>
Portada Libro	Comprar Libro
Portada Libro	Comprar Libro
Descripción libro	Enlace de compra
Comprar Libro	Comprar Libro

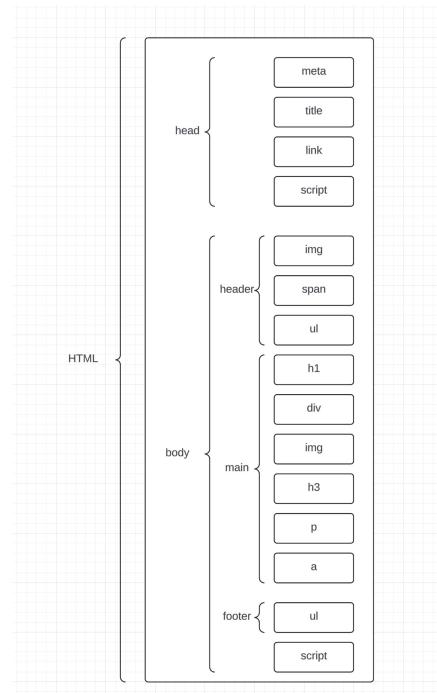


Figure 29: a) Interfaz de libros.html b) Prototipo de interfaz de libros.html c) Mapa de etiquetas de libros.html

Foro.html

Esta página es la que contiene el foro de nuestra web, donde los usuarios pueden discutir sobre distintos temas. Los headers y footers son los mismos que en las otras páginas, por lo que no se mostrarán.

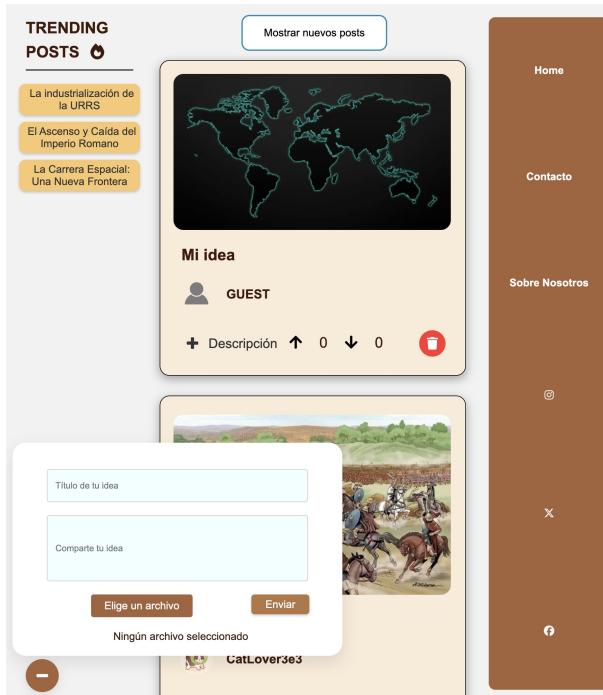


Figure 30: Interfaz de foro.html

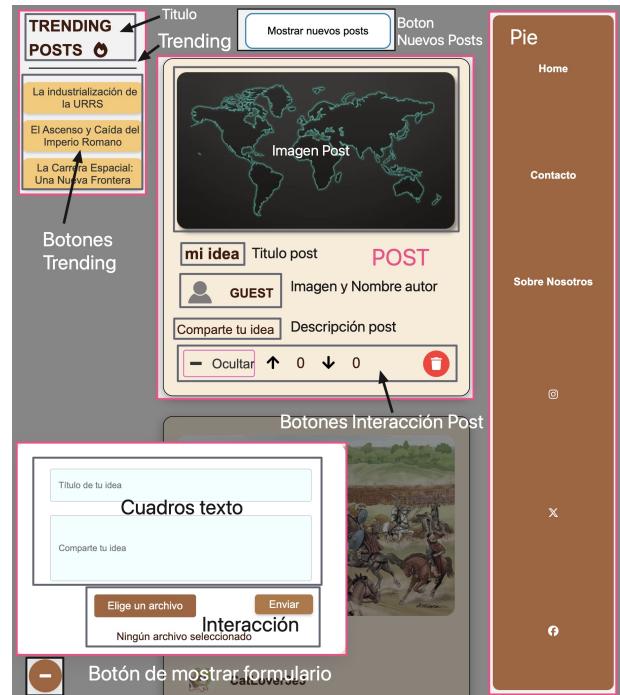


Figure 31: Prototipo de interfaz de foro.html

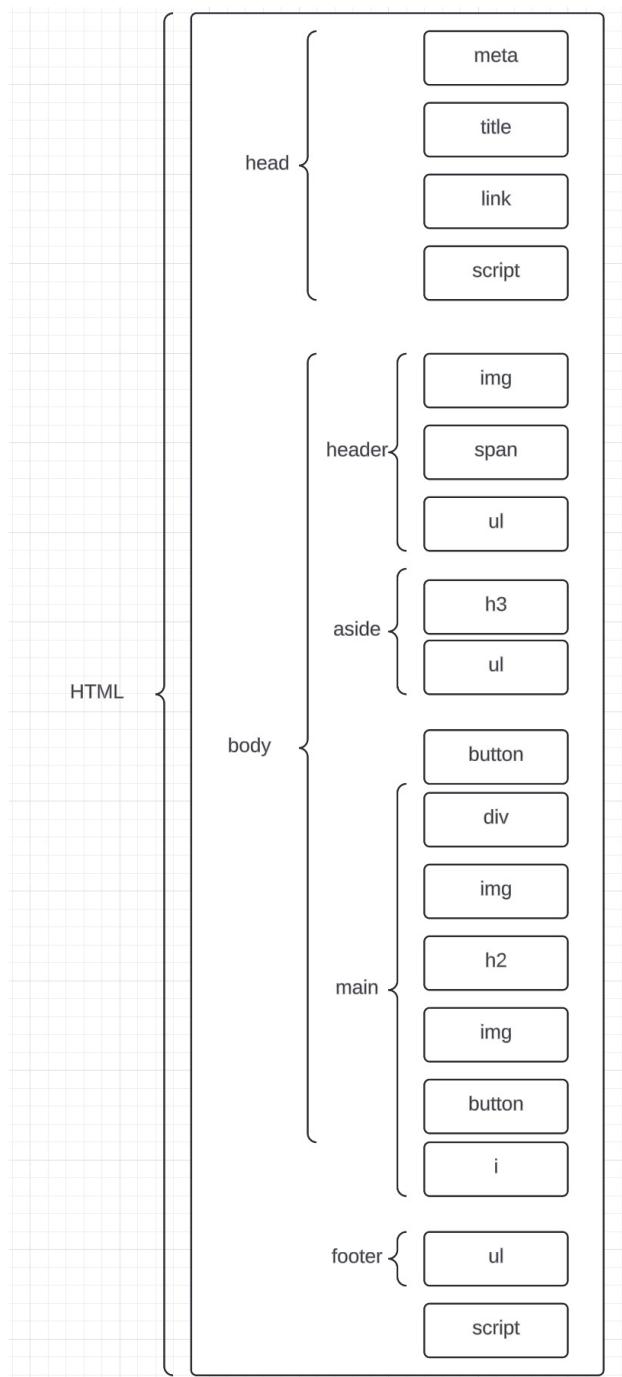


Figure 32: Mapa de etiquetas de foro.html

9.2 Estructura de ficheros actualizada

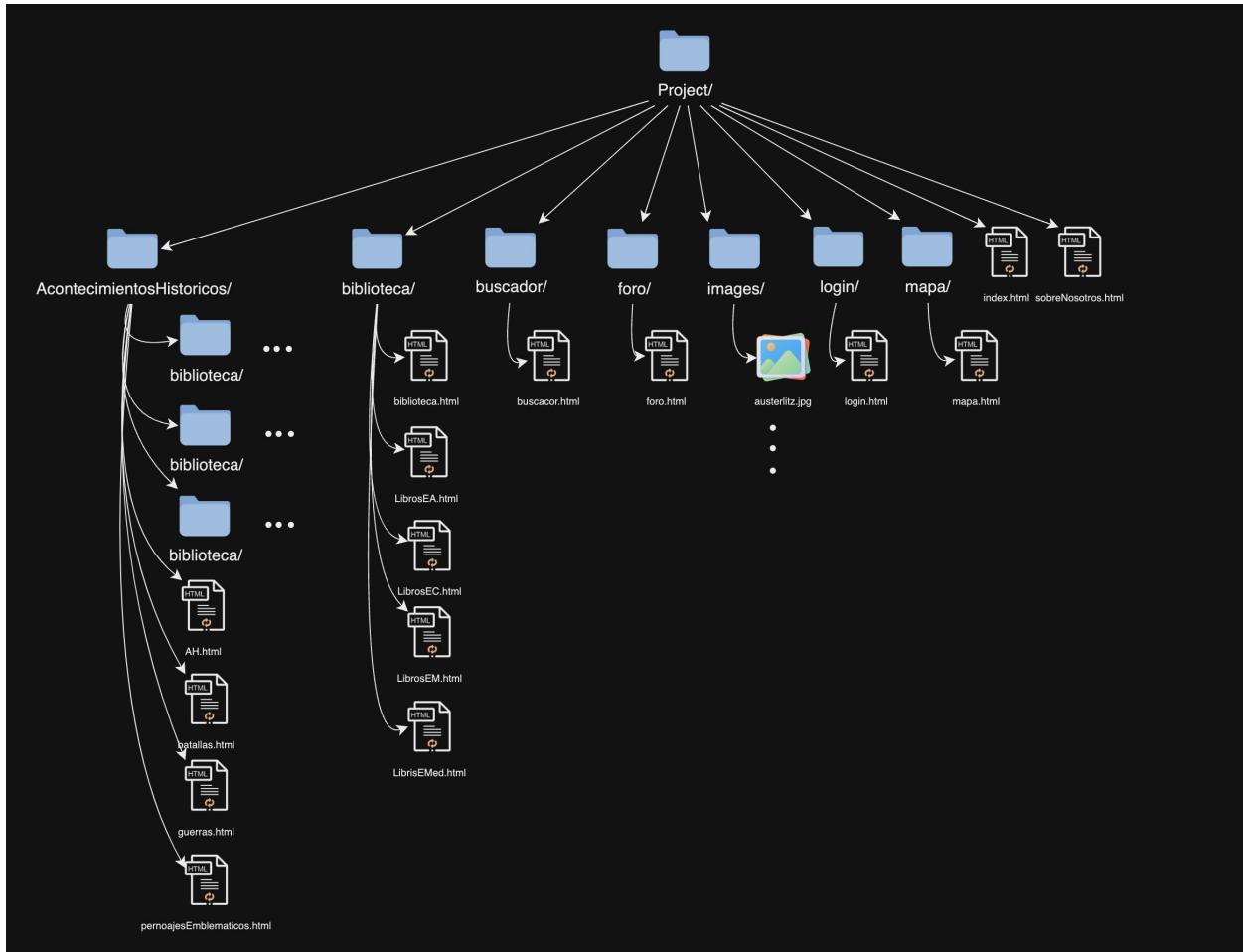


Figure 33: Estructura de ficheros actualizada

10 CSS

Durante esta fase del desarrollo, incluimos en nuestra página web hojas de estilo CSS para mejorar la presentación y el diseño de las páginas. Deberemos implementar 5 técnicas de diseño CSS responsivo:

- **Flexible Grids:** aplicado en diversas páginas, como en biblioteca.html
- **CSS Multicol:** En la página multicol.css aplicada en LibrosEM.html
- **Flex Container:** En la página flexcontainer.css aplicada en LibrosEC.html
- **CSS Grid:** En la página CSSgrid.css aplicada en LibrosEMed.html
- **Bootstrap:** En la página bootstrap.css y bootstrap.min.css (así como la inclusión de diversas librerías de Bootstrap) aplicada en LibrosEA.html, así como en las páginas de inicio de acontecimientos históricos, batallas y personajes emblemáticos.

10.1 Flexible Grids

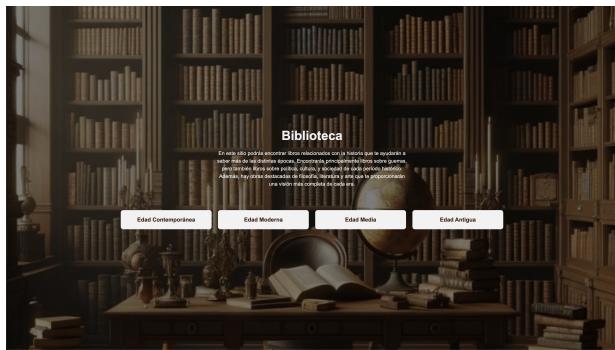


Figure 34: Interfaz de biblioteca.html

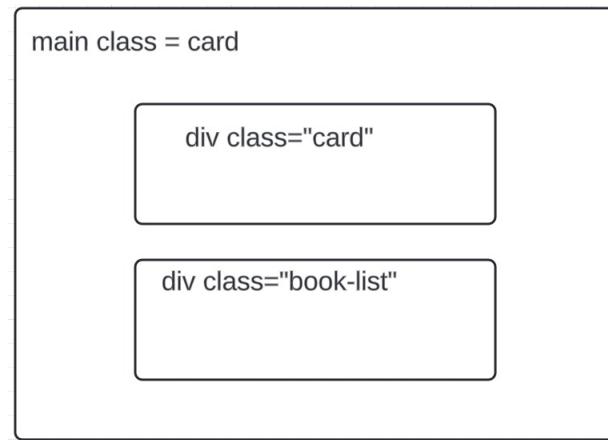


Figure 35: Prototipo de interfaz de biblioteca.html

10.2 CSS Multicol

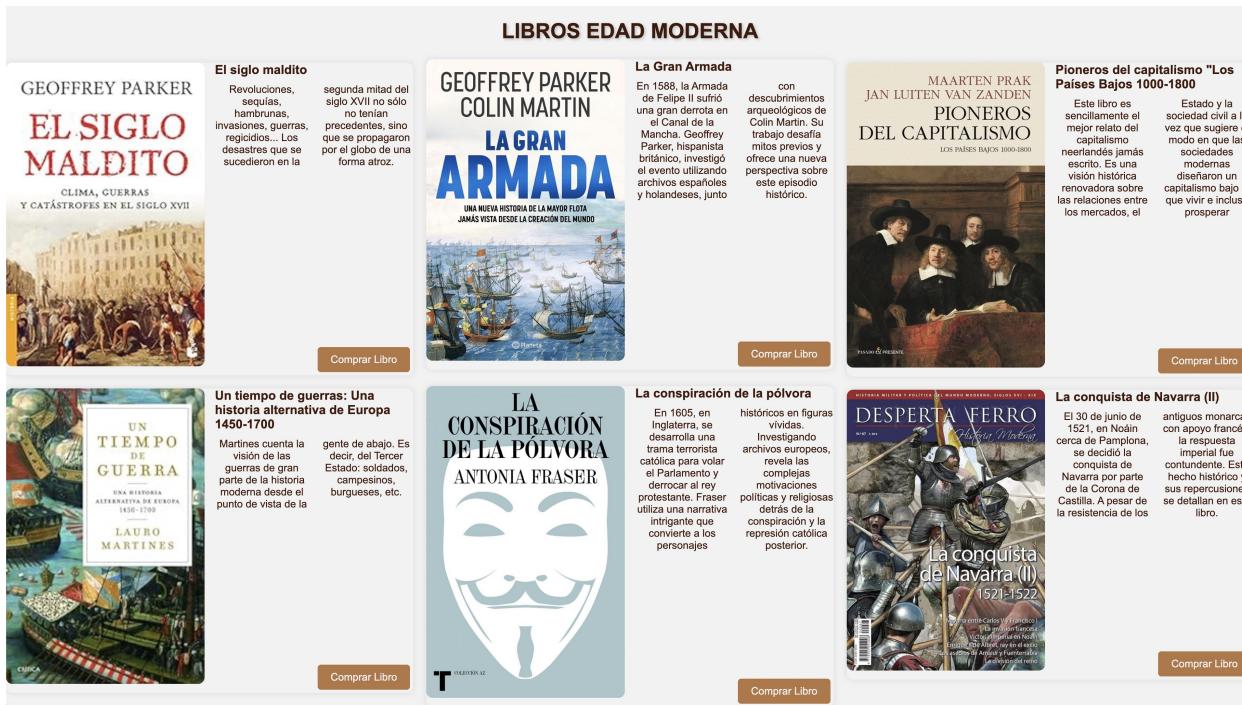


Figure 36: Interfaz de LibrosEM.html

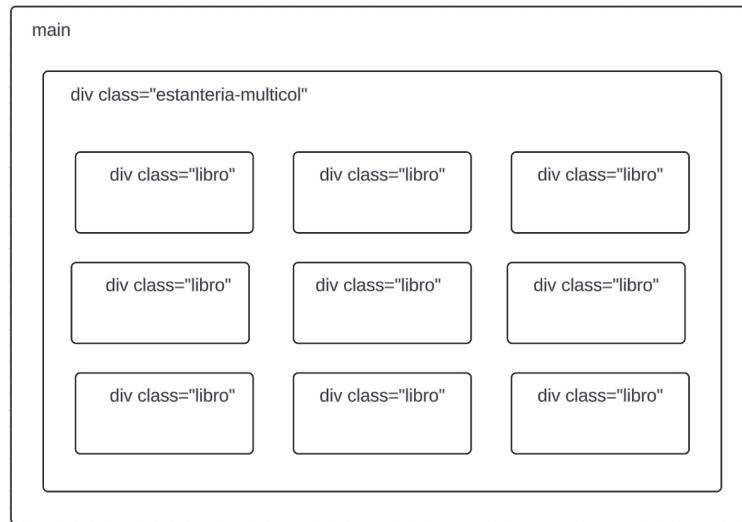


Figure 37: Prototipo de interfaz de LibrosEM.html

10.3 Flex Container



Figure 38: Interfaz de LibrosEC.html

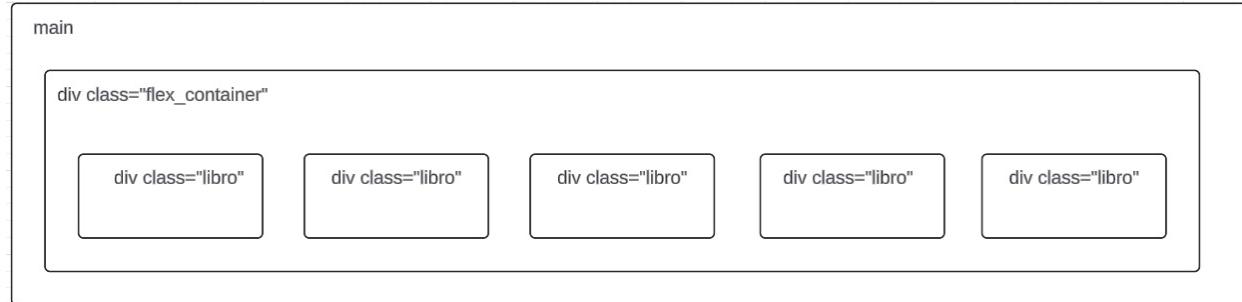


Figure 39: Prototipo de interfaz de LibrosEC.html

10.4 CSS Grid



Figure 40: Interfaz de LibrosEMed.html

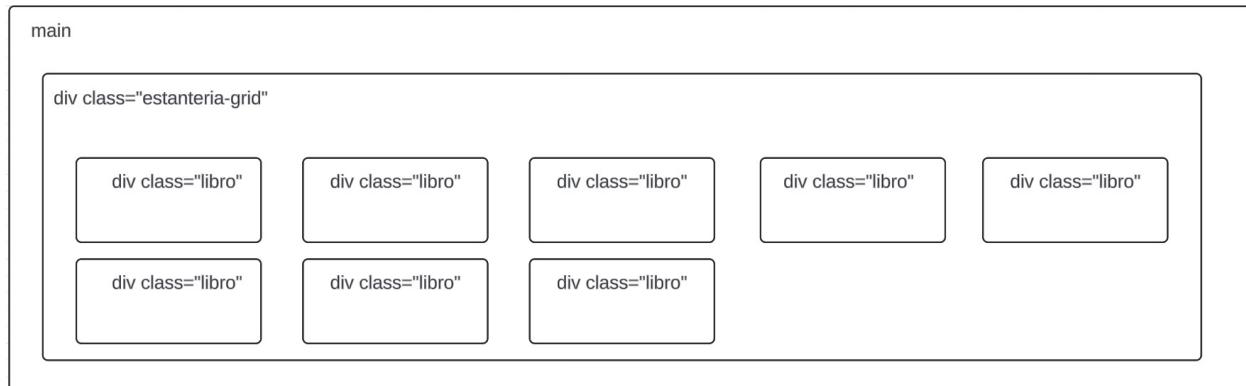


Figure 41: Prototipo de interfaz de LibrosEMed.html

10.5 Bootstrap

ACONTECIMIENTOS HISTÓRICOS



Guerras

Batallas

Personajes Emblemáticos

Figure 42: Interfaz de AH.html

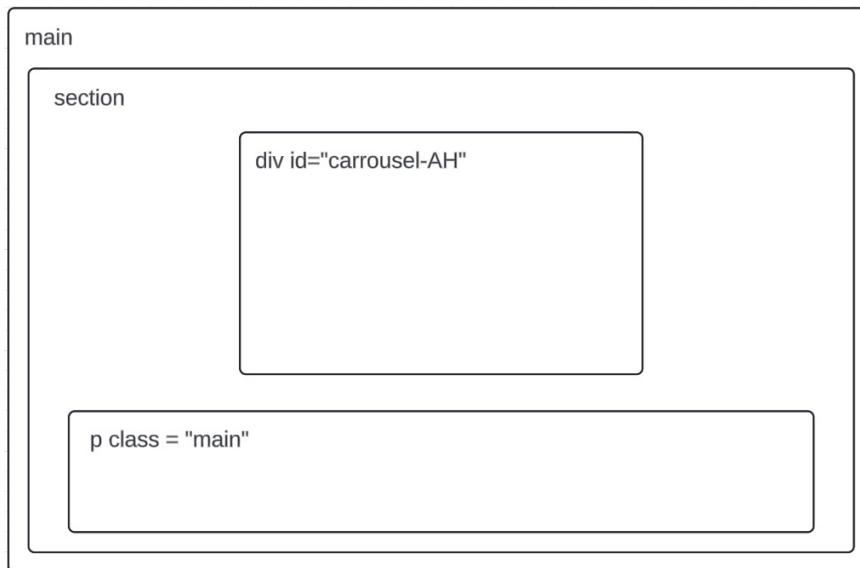


Figure 43: Prototipo de interfaz de AH.html

También podremos ver como se ha aplicado esta librería en la página de LibrosEA.html.



Figure 44: Interfaz de LibrosEA.html

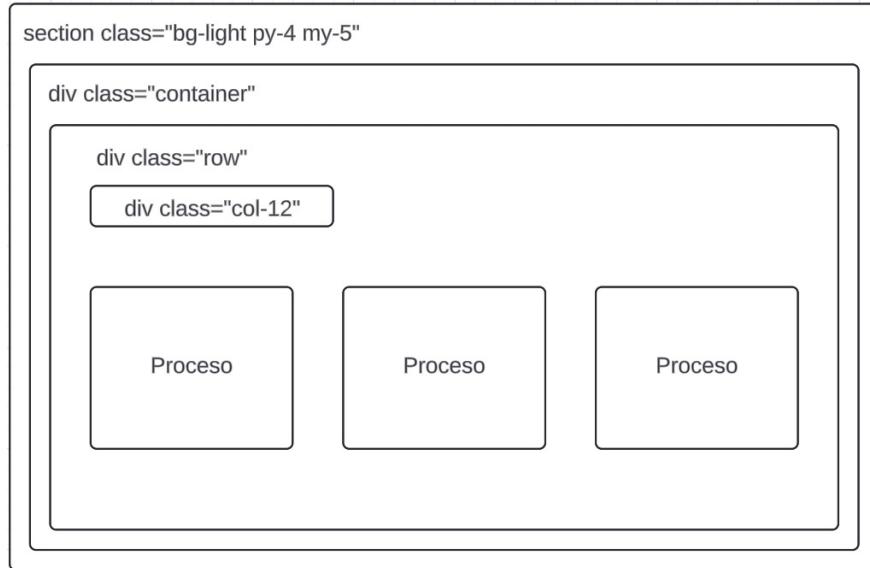


Figure 45: Prototipo de interfaz de LibrosEA.html

11 JavaScript

En esta sección se presentan las funcionalidades de JavaScript implementadas en nuestra página web. Estas funciones permiten una interacción dinámica y enriquecedora con el contenido, mejorando la experiencia del usuario y facilitando la navegación a través de las distintas secciones del sitio.

11.1 Funcionalidades

En primer lugar mostraremos las funcionalidades de la aplicación, con sus correspondientes efectos en ella mediante capturas de pantalla.

11.1.1 Header

En primer lugar, cabe destacar el header, que se encuentra en todas las páginas de la web. En el header incorporamos una funcionalidad tal que, cuando la página web pasa uno de los puntos de ruptura (768 px), el menú de navegación se convierte en un menú desplegable, que se puede abrir y cerrar pulsando el botón de menú.

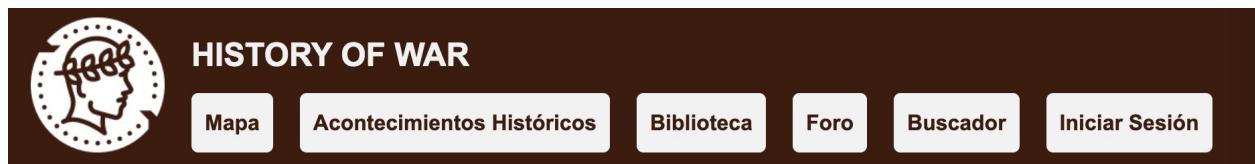


Figure 46: Menú normal



Figure 47: Menú desplegable

Para ello, hemos utilizado el siguiente código JavaScript:

```

1  /**
2   * Función para mostrar y ocultar el menú de navegación en dispositivos
3   * móviles
4   */
5
6  function myFunction() {
7      var x = document.getElementsByClassName("header-list")[0];
8      if (window.innerWidth < 768) {
9          if (x.style.display === "block") {
10              x.style.display = "none";
11          } else {
12              x.style.display = "block";
13          }
14      }
15
16     // Evento para manejar el cambio de tamaño de la ventana
17     window.addEventListener('resize', function() {
18         var x = document.getElementsByClassName("header-list")[0];
19         if (window.innerWidth >= 768) {
20             // Elimina el estilo en línea para permitir que los estilos CSS se
21             // apliquen
22             x.style.display = "";
23         }
24     });

```

Listing 1: Código de Header.js

Por una parte, La función **myFunction()**:

- Obtiene el primer elemento del documento con la clase header-list.
- Si el ancho de la ventana es menor a 768 píxeles (lo que sugiere que está en un dispositivo móvil):
 - Si el elemento está visible (display establecido en "block"), lo oculta cambiando su display a "none".
 - Si el elemento está oculto (display no está en "block"), lo muestra cambiando su display a "block"

Por otra parte, el evento **resize**:

- Se añade un controlador de eventos que escucha por el evento resize en la ventana, lo que ocurre cuando se cambia el tamaño de la ventana del navegador.
- Cada vez que se cambia el tamaño de la ventana, se obtiene nuevamente el primer elemento con la clase header-list.
- Si el ancho de la ventana es igual o superior a 768 píxeles:
 - Se elimina el estilo en línea del elemento, permitiendo que los estilos CSS previamente definidos en una hoja de estilo se apliquen de nuevo.

En estas dos partes del código, podemos observar que se utilizan dos métodos de acceso al DOM, como son **getElementsByClassName()** y **addEventListener()**, y por otra parte un evento de JavaScript, como es **resize**.

11.1.2 Mapa

El mapa es una de las partes más destacadas a nivel de funcionalidad de nuestra página web, en donde ofrecemos una experiencia mejorada al usuario, permitiéndole acceder a las batallas que describimos en nuestra página web con un simple click en los iconos (situados en las coordenadas de las batallas). Todo esto pudimos implementarlo gracias a la librería de JavaScript Leaflet.

Otra cuestión destacada es el evento mouseover que implementamos, el cual muestra un 'pop-up' con el nombre de la batalla con la que se está interactuando.

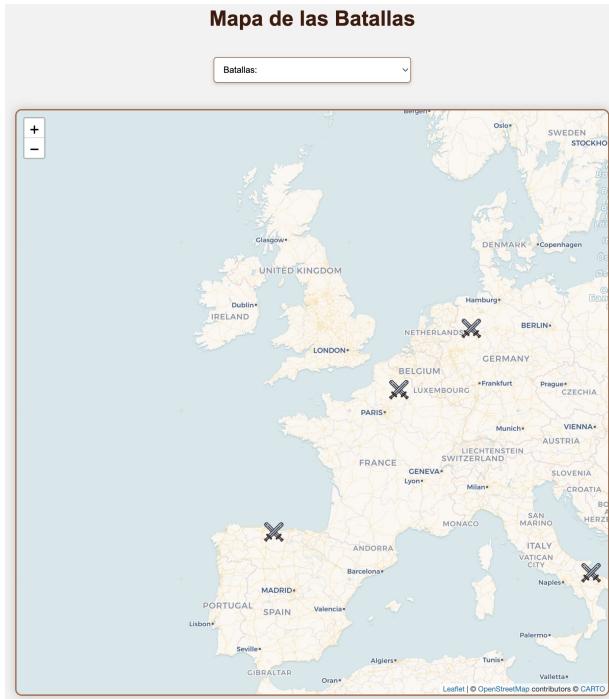


Figure 48: Mapa de batallas

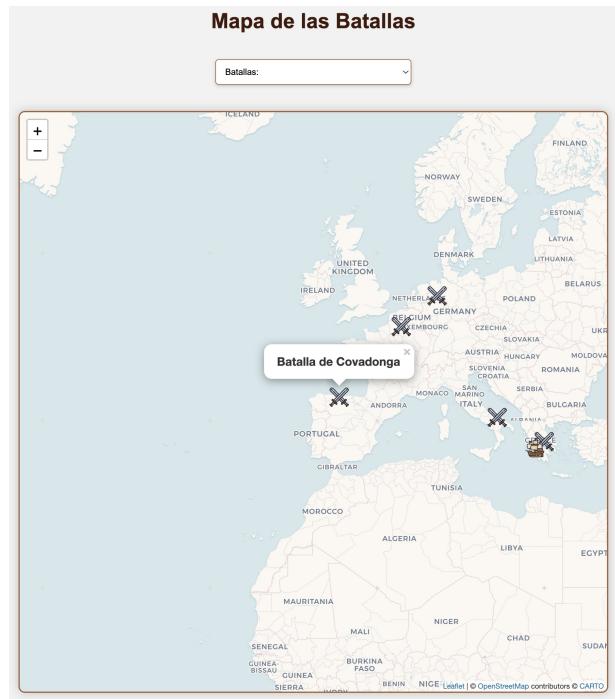


Figure 49: Evento mouseover

Para ello, hemos utilizado el siguiente código JavaScript:

```

1 // Agregar tileLayer mapa base de CartoDB estilo Positron (estilo minimalista)
2 L.tileLayer('https://[s].basemaps.cartocdn.com/rastertiles/voyager/{z}/{x}/{y}
3             ){r}.png', {
4     attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">
5     OpenStreetMap</a> contributors &copy; <a href="https://carto.com/attribution">
6     CARTO</a>',
7     subdomains: 'abcd',
8     maxZoom: 19
9   }).addTo(map);

```

Listing 2: Código para cargar el mapa

En este fragmento de código, se añade un mapa base de CartoDB estilo Positron (un estilo minimalista) al mapa. Se utiliza la función **L.tileLayer()** de la librería Leaflet para cargar el mapa base, especificando la URL del mapa base y sus atributos, como la atribución y el zoom máximo.

```

1 // Marcador de la Batalla de Covadonga
2 L.marker([43.312085, -5.058803], {icon: swordIcon}).addTo(map)
3   .bindPopup('<h2>Batalla de Covadonga</h2>')
4   .on('mouseover', function(e) {
5     this.openPopup();
6   })
7   .on('mouseout', function(e) {
8     this.closePopup();
9   })
10  .on('click', function(e) {
11    window.location.href = ".../AcontecimientosHistoricos/Batallas/
12      batallaCovadonga.html";
13  });

```

Listing 3: Código para añadir los marcadores al mapa

En este fragmento, se muestra un ejemplo de como es un evento de los que hay en el mapa. En este caso, se añade un marcador en las coordenadas de la Batalla de Covadonga, se le añade un pop-up con el nombre de la batalla, y se añaden eventos para que, al pasar el ratón por encima del marcador, se muestre el pop-up, y al hacer click en el marcador, se redirija a la página de la batalla.

Otra funcionalidad clave del mapa es la que nos permite ir a la batalla desde una lista de batallas, que se encuentra en la parte superior de la página.

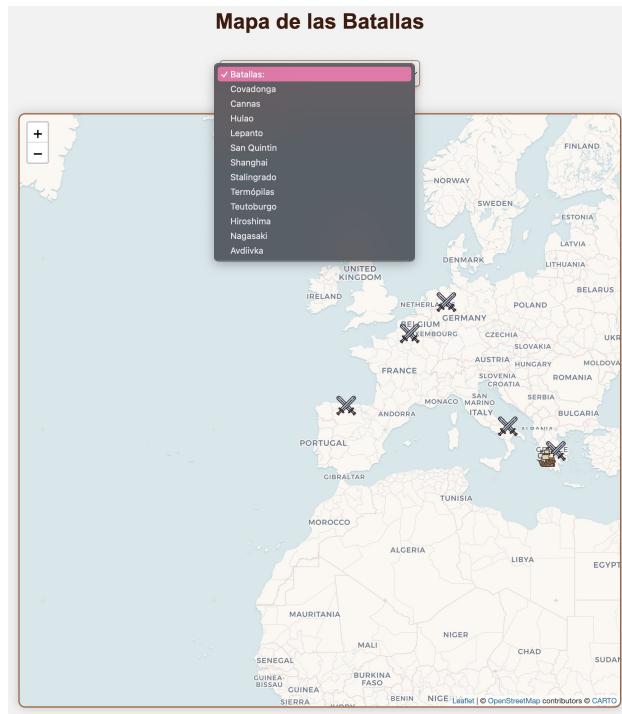


Figure 50: Lista de batallas

Y lo hemos implementado con el siguiente código JavaScript:

```

1 //----- Barra de Navegaci n -----
2 // Agregar barra de navegaci n que nos permitira movernos por el mapa
3 document.getElementById('select-location').addEventListener('change', function
4 (e) {
5     let coords = e.target.value.split(",");
6     map.flyTo(coords, 10);
7 });

```

Listing 4: Código para ir a la batalla desde la lista

En estas funciones podemos ver que utilizamos métodos de la librería Leaflet, como **L.marker()**, **.bindPopup()**, **.on()**, y **.addTo()**, así como eventos de JavaScript, como **mouseover**, **mouseout**, y **click**. Podemos destacar un método de acceso al DOM, como es **getElementById()**.

11.1.3 Buscador

Otra funcionalidad de la página web es el buscador, que permite al usuario buscar por palabras clave en la página web. Por ejemplo, inserte usted en el buscador la palabra "Covadonga" y le redirigirá a la página de la batalla de Covadonga.

Debido a la complejidad del proyecto, no se puede implementar mediante una base de datos, por lo que tuvimos que hacerlo de una manera ciertamente rudimentaria, pero que al final cumple su función. Para hacer esto utilizamos una página JSON en la cual definimos el título de la página y la URL a la que redirigir.

La primera funcionalidad es la siguiente:



Figure 51: Buscador



Figure 52: Búsqueda no encontrada

Para ello, hemos utilizado el siguiente código JavaScript:

```

1  document.getElementById('searchForm').onsubmit = function(event) {
2      document.getElementById('noResults').style.display = 'none';
3
4      event.preventDefault(); // Prevenir la recarga de la página
5      let searchTerm = document.getElementById('searchInput').value.toLowerCase()
6      ();
7
8      // Filtrar las páginas que coincidan con el término de búsqueda
9      let searchResults = pages.filter(function(page) {
10         return page.title.toLowerCase().indexOf(searchTerm) > -1;
11     });
12
13     // Si hay resultados, redirige al primero. Podrás mejorar esto para
14     // mostrar una lista.
15     if (searchResults.length > 0) {
16         window.location.href = searchResults[0].url;
17     } else {
18         document.getElementById('noResults').style.display = 'block';
19     }

```

Listing 5: Código para el buscador

Este fragmento de código JavaScript configura un controlador de eventos para el formulario de búsqueda identificado por searchForm, que intercepta el evento de envío para prevenir la recarga de la página y en su lugar realiza una búsqueda. Extrae el término de búsqueda ingresado en el campo searchInput, lo convierte a minúsculas y lo compara con un conjunto de páginas (almacenado en la variable pages) para encontrar coincidencias basadas en el título de la página.

Si encuentra al menos un resultado, redirige al navegador a la URL del primer resultado encontrado. Si no hay coincidencias, muestra un elemento del DOM, identificado como noResults, para indicar al usuario que no se encontraron resultados para su búsqueda.

Por otra parte, tenemos la carga de las páginas en el JSON:

```

1  [
2  {
3      "title": "mapa",
4      "url": "../Mapa/mapa.html"
5  },
6  {
7      "title": "acontecimientos históricos",
8      "url": "../AcontecimientosHistoricos/AH.html"
9  },
10 {
11     "title": "biblioteca",
12     "url": "../biblioteca/biblioteca.html"
13 },
14 {
15     "title": "iniciar sesión",
16     "url": "../login/login.html"
17 },
18 {
19     "title": "batallas",
20     "url": "../AcontecimientosHistoricos/batallas.html"
21 }

```

```
22     [...]
23 ]
```

Listing 6: Carga de las páginas en el JSON

Que se carga en la variable **pages**, de la siguiente manera:

```
1  let pages = [];
2
3  // Función para cargar las páginas desde el archivo JSON usando
4  XMLHttpRequest
5  function loadPages() {
6      // Creamos un objeto XMLHttpRequest, que nos permite hacer peticiones HTTP
7      // asíncronas
8      var xhr = new XMLHttpRequest();
9
10     // Configuramos la petición
11     xhr.open('GET', './buscador.json', true);
12
13     xhr.onload = function() {
14         // Si la respuesta del servidor es satisfactoria
15         if (this.status >= 200 && this.status < 300) {
16             // Parsea y carga los datos en la variable 'pages'
17             pages = JSON.parse(this.responseText);
18         } else {
19             // Maneja errores de respuesta del servidor
20             console.error('Error al cargar el archivo JSON:', this.statusText)
21         }
22     };
23
24     xhr.onerror = function() {
25         // Maneja errores de red
26         console.error('Error de red.');
27     };
28     // Envía la petición
29     xhr.send();
30
31     // Cargar las páginas al iniciar
32     loadPages();
}
```

Listing 7: Carga de las páginas en el JSON

11.1.4 Foro

Esta sección es la joya de la corona de nuestra página web, en la que los usuarios pueden interactuar entre ellos, comentar y discutir sobre distintos temas. Debido a la rudimentariedad del proyecto (al no incluir una base de datos para los usuarios por ejemplo), no se puede implementar un sistema de usuarios, por lo que hemos incluido comentarios de prueba hechos por nosotros mismos.

El usuario de la página web puede añadir un comentario, y este se añadirá a la lista. De la misma manera, tiene la capacidad de borrar su propio comentario.

Otra funcionalidad para el usuario, es la de darle "like" y "dislike" a los comentarios, y ver cuantos tiene cada uno. Así como la inclusión de una columna de trending topics, que muestra los comentarios más populares.

También tenemos un botón para mostrar los nuevos posts.

Primero mostraremos los comentarios base, y como se cargan en la página web mediante un fichero XML:

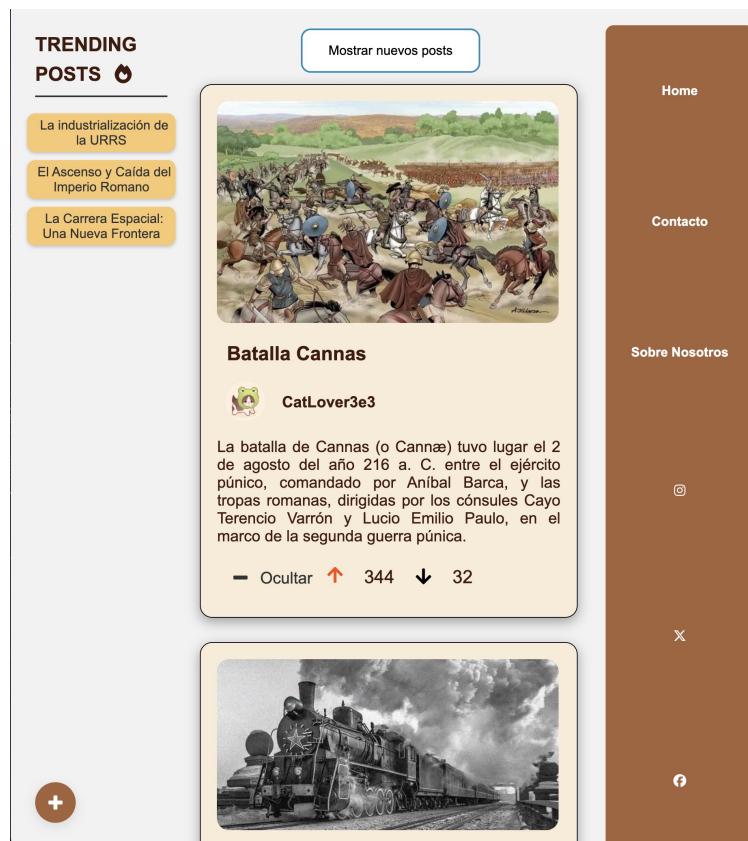


Figure 53: Comentarios

Lo implementamos con el siguiente código JavaScript:

```

1 // Cargar los posts desde el archivo XML
2 $("#btn_mostrar").click(function () {
3     $(this).hide(); // Oculta el botón
4
5     $.ajax({

```

```

6         type: "GET",
7         url: "datos.xml", // Ajusta esta ruta
8         dataType: "xml",
9         success: function (xml) {
10            $('#fill_data').empty();
11            $(xml).find('post').each(function () {
12                let postId = $(this).attr('id');
13                let imagenSrc = $(this).find('imagen').attr('src');
14                let titulo = $(this).find('titulo').text();
15                let autor = $(this).find('autor').text();
16                let pfpSrc = $(this).find('pfp').attr('src') || '../Images/pfp
17 .png';
18                let likes = $(this).find('likes').text() || "0";
19                let dislikes = $(this).find('dislikes').text() || "0";
20                let texto = $(this).find('texto').text();
21
22                let postDiv = $('<div class="post" id=' + postId + '></div>');
23
24                if (imagenSrc) {
25                    postDiv.append($('<img class="img_post" src=' + imagenSrc
26 + '" alt="Imagen del post">'));
27
28                    postDiv.append($('<h2 class="titulo">' + titulo + '</h2>'));
29                    let infoDiv = $('<div class="info"></div>').append(
30                        $('<img class="pfp" src=' + pfpSrc + '" width="50px"
31 height="50px" alt="Imagen de perfil">'),
32                        $('<p class="autor">' + autor + '</p>')
33                    );
34
35                    postDiv.append(infoDiv);
36                    postDiv.append($('<p class="texto_oculto">' + texto + '</p>'))
37
38                    postDiv.append($('<button class="mostrar-descripcion"><i class
39 ="fas fa-plus"></i> Descripción</button>'));
40
41
42
43                    $('#fill_data').append(postDiv);
44
45                },
46                error: function (xhr, status, error) {
47                    console.error("Error al cargar el archivo XML:", error);
48
49            });
50        });
51
52 // Manejador de clic para mostrar/ocultar la descripción
53 $(document).on('click', '.mostrar-descripcion', function() {
54     $(this).prev('.texto').toggleClass('oculto');
55     // Cambia el cono y el texto del botón dependiendo del estado

```

```

56     let icon = $(this).find('i');
57     if (icon.hasClass('fa-plus')) {
58         icon.removeClass('fa-plus').addClass('fa-minus');
59         // Cambiar solo el texto, manteniendo el cono
60         $(this).contents().filter(function(){
61             return this.nodeType == 3;
62         }).remove();
63         $(this).append(' Ocultar');
64     } else {
65         icon.removeClass('fa-minus').addClass('fa-plus');
66         // Cambiar solo el texto, manteniendo el cono
67         $(this).contents().filter(function(){
68             return this.nodeType == 3;
69         }).remove();
70         $(this).append(' Descripción');
71     }
72 });
73
74
75 // Manejador de eventos para los botones de like y dislike
76 $(document).on('click', '.like-button, .dislike-button', function() {
77     let isLikeButton = $(this).hasClass('like-button');
78     let likesCount = $(this).siblings('.likes-count');
79     let dislikesCount = $(this).siblings('.dislikes-count');
80
81     // Toggle active class
82     $(this).toggleClass('active');
83
84     if ($(this).hasClass('active')) {
85         // Incrementa o decrementa el contador adecuadamente
86         if (isLikeButton) {
87             likesCount.text(parseInt(likesCount.text()) + 1);
88             // Si el botón opuesto estuvo activo, ajusta su contador también
89             if ($(this).siblings('.dislike-button').hasClass('active')) {
90                 $(this).siblings('.dislike-button').removeClass('active');
91                 dislikesCount.text(parseInt(dislikesCount.text()) - 1);
92             }
93         } else {
94             dislikesCount.text(parseInt(dislikesCount.text()) + 1);
95             if ($(this).siblings('.like-button').hasClass('active')) {
96                 $(this).siblings('.like-button').removeClass('active');
97                 likesCount.text(parseInt(likesCount.text()) - 1);
98             }
99         }
100    } else {
101        if (isLikeButton) {
102            likesCount.text(parseInt(likesCount.text()) - 1);
103        } else {
104            dislikesCount.text(parseInt(dislikesCount.text()) - 1);
105        }
106    }
107 });
108 // Evento de clic para mostrar la imagen ampliada
109     // lo ponemos con on para que tambien se aplique a las posts nuevos que
110     // introducimos y no solo a los que estan por defecto.
111 $(document).on('click', '.img_post, .pfp', function () {
112     let src = $(this).attr('src');
113     if (src == null) console.log("source = NULL");
114     $('#imagenAmpliadaContenido').attr('src', src);

```

```

114     $('#imagenAmpliada').fadeIn();
115 });
116
117 // Evento de clic para ocultar la imagen ampliada
118 $('#imagenAmpliada').click(function () {
119     $(this).fadeOut();
120 });
121
122 $('.lista_nav li a').mouseover(function () {
123     $(this).find('i').css('visibility', 'visible');
124 });
125
126 $('.lista_nav li a').mouseout(function () {
127     $(this).find('i').css('visibility', 'hidden');
128 });
129
130 $("#mostrarFormBtn").click(function() {
131     var formVisible = $("#formContainer").is(":visible");
132     $("#formContainer").slideToggle('fast', function() {
133         if (!formVisible) {
134             // Restablece el formulario solo si se est cerrando
135             $('#miForm')[0].reset();
136             // Restablece tambi n el texto de archivo seleccionado
137             $('#file-chosen').text('Ning n archivo seleccionado');
138         }
139     });
140
141     // Cambiar el cono del bot n dependiendo del estado del formulario
142     $(this).html(formVisible ? '<i class="fas fa-plus"></i>' : '<i class="fas fa-minus"></i>');
143 });

```

Listing 8: Carga de los comentarios

Este extenso fragmento de código JavaScript, integrado con jQuery, implementa diversas funcionalidades interactivas para una página web, principalmente orientadas a la gestión y visualización de contenido dinámico, como posts o artículos, desde un archivo XML. A continuación, se describe cada segmento de funcionalidad:

- **Cargar y Mostrar Posts:** Al hacer clic en un botón identificado por btnmostrar, se oculta dicho botón y se realiza una petición AJAX para cargar datos desde un archivo datos.xml. Los posts contenidos en el archivo XML se procesan individualmente, extrayendo detalles como identificador, imagen, título, autor, imagen de perfil (pfp), contador de likes y dislikes, y texto. Para cada post, se crea un elemento div con esta información y se añade a un contenedor en la página. Si hay una imagen asociada al post, se incluye al principio del div. Además, se añaden botones dinámicos para mostrar u ocultar la descripción del post y contadores de votos (likes/dislikes).
- **Manejadores de Eventos para Likes/Dislikes:** Se utilizan eventos jQuery on para gestionar clics en los botones de like y dislike, ajustando los contadores correspondientes. Si un usuario da like a un post que previamente había dado dislike (o viceversa), el contador del voto opuesto se ajusta también. Los botones alternan una clase active para reflejar el estado actual del voto del usuario.
- **Visualización Ampliada de Imágenes:** Al hacer clic en imágenes de posts o de perfil, se muestra una versión ampliada de la imagen en un modal o un contenedor específico para ello. Al hacer clic fuera de la imagen ampliada, esta se oculta.
- **Interacción con Elementos de Navegación:** Se añaden efectos visuales a elementos de navegación (listanav li a), como hacer visibles los íconos al pasar el ratón por encima y ocultarlos al retirarlo.

- **Mostrar/Ocultar Formulario:** Un botón permite alternar la visibilidad de un formulario (formContainer), deslizándolo hacia arriba o hacia abajo. Al ocultar el formulario, se resetea su contenido y se actualiza el texto del botón para reflejar el estado del formulario (mostrado u oculto), cambiando el ícono del botón adecuadamente.

Este código utiliza técnicas de jQuery para simplificar la selección y manipulación del DOM, el manejo de eventos, y las transiciones visuales.

La otra funcionalidad viene implementada de la siguiente manera:

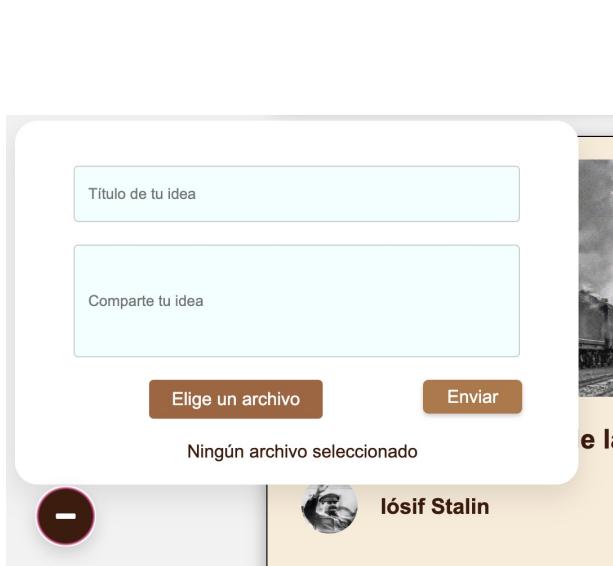


Figure 54: Añadir comentario

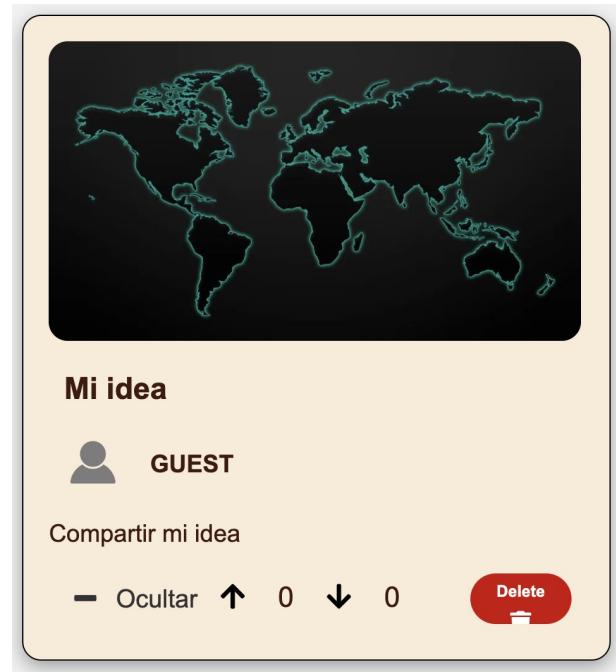


Figure 55: Borrar comentario

Con el consecuente código JavaScript:

```

1 // Manejo del envío del formulario para crear un nuevo post
2 $('#miForm').submit(function (event) {
3     event.preventDefault(); // Prevenir el comportamiento predeterminado
4
5     // Recopilar la información del formulario
6     let titulo = $('#input_titulo').val();
7     let texto = $('#input_txt').val();
8     let archivo = $('#myfile')[0].files[0];
9
10    // Crear el nuevo post, incluyendo el botón de eliminar
11    let nuevoPostHTML = '<div class="post">' +
12        (archivo ? '<img class="img_post" src=' + URL.
13        createObjectURL(archivo) + '>' : '') +
14            '<h2 class="titulo">' + titulo + '</h2>' +
15            '<div class="info"><p class="autor">GUEST</p></div>' +
            '<p class="texto_oculto">' + texto + '</p>' +

```

```

16             '<button class="mostrar-descripcion"><i class="fas fa-plus"></i> Descripción</button>' +
17                     '<div class="voting-buttons">' +
18                         '<i class="fa fa-arrow-up like-button" aria-
19                           hidden="true"></i><span class="likes-count">0</span>' +
20                             '<i class="fa fa-arrow-down dislike-button" aria-
21                               hidden="true"></i><span class="dislikes-count">0</span>' +
22                               '</div>' +
23                                 '<button class="boton-eliminar"> <span class="texto-eliminar animate__animated">Delete</span><i class="fas fa-trash animate__animated"></i></button>' +
24                                     '</div>';
25
26
27     // Aadir el nuevo post al principio de la sección principal
28     $('#fill_data').prepend(nuevoPostHTML);
29
30     // Limpiar el formulario y ocultarlo
31     $('#miForm')[0].reset();
32     $('#formContainer').slideUp();
33     // Restablecer el cono del botón flotante
34     $('#mostrarFormBtn').html('<i class="fas fa-plus"></i>');
35
36     // Evento de clic para eliminar un post
37     $(document).on('click', '.boton-eliminar', function() {
38         $(this).closest('.post').remove(); // Elimina el post más cercano al
39         // botón clickeado
40     });
41
42     // Actualizar el texto al elegir un archivo
43     $('#myfile').change(function () {
44         var fileName = $(this).val().split('\\').pop(); // Obtiene el nombre del
45         // archivo
46         $('#file-chosen').text(fileName ? fileName : 'Ningún archivo seleccionado
47         '); // Actualiza el texto
48     });

```

Listing 9: Nuevo post

Este bloque de código JavaScript maneja varias funcionalidades interactivas en una página web, principalmente enfocadas en la creación y gestión de publicaciones o "posts" a través de un formulario, así como la interacción con un sistema de votos y la visualización de archivos seleccionados. Aquí tienes un resumen de sus operaciones:

- **Creación de un Nuevo Post:**

- Se intercepta el evento de envío (submit) del formulario identificado por miForm para evitar que se recargue la página (comportamiento predeterminado de los formularios).
- Recopila los valores ingresados por el usuario en los campos del formulario: título, texto y un archivo seleccionado.
- Crea un nuevo post en formato HTML, utilizando la información proporcionada. Si se ha seleccionado un archivo, se muestra como parte del post utilizando URL.createObjectURL(archivo) para generar una URL temporal para el archivo seleccionado.
- Añade dinámicamente el nuevo post al principio de la sección de publicaciones (filldata) en la página.
- Limpia los campos del formulario y oculta el contenedor del formulario (formContainer), y restablece el ícono del botón utilizado para mostrar el formulario.

- **Eliminación de un Post:**

- Implementa la funcionalidad para eliminar un post. Cuando se hace clic en el botón de eliminar de un post específico (identificado por .boton-eliminar), el post correspondiente se elimina del DOM. Esto se logra buscando el contenedor .post más cercano al botón clickeado y eliminándolo de la página.

- **Visualización del Nombre del Archivo Seleccionado:**

- Actualiza dinámicamente el texto de un elemento (identificado por file-chosen) para mostrar el nombre del archivo seleccionado por el usuario. Si se selecciona un archivo a través del campo de entrada de archivo (myfile), se extrae su nombre y se muestra. Si no se selecciona ningún archivo, se muestra un mensaje predeterminado ("Ningún archivo seleccionado").

Este conjunto de funcionalidades facilita la interacción del usuario con la página, permitiéndole añadir contenido de manera intuitiva y visual, y proporcionando retroalimentación inmediata, como la visualización del nombre del archivo seleccionado y la posibilidad de eliminar publicaciones de forma directa.

12 Estructura de datos final

La estructura de datos final con todos nuestros archivos de la página web es la siguiente. Cabe destacar que por la simplicidad del diagrama (debido a que hay una cantidad enorme de archivos), no se incluirán todas los archivos de la web en el diagrama, sino que se incluirán representaciones de como es cada uno.

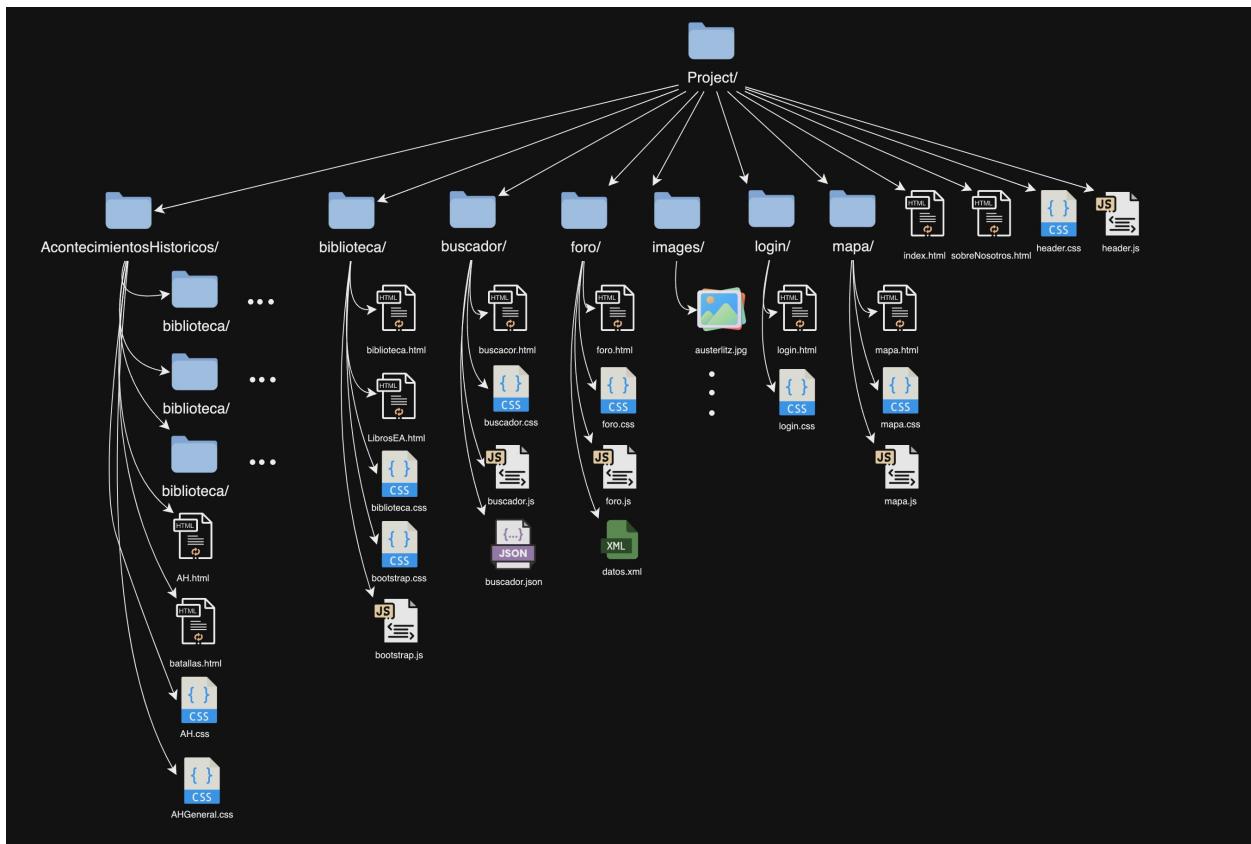


Figure 56: Diagrama de estructura de datos final

13 De la idea primigenia a la página web final

En un principio mostramos una idea de una página web con una estructura muy básica, con un menú de navegación, un mapa, la pagina de acontecimientos históricos y una biblioteca.

A lo largo del mes de desarrollo hemos cambiado el diseño de las principales páginas web, pero manteniendo la esencia de la idea original. El header y el footer se han mantenido con ligeros cambios, como puede ser la transformación del login en una página propia, o la de centrar el footer en la página.

Se han añadido la página de buscador y del foro, que no estaban en la idea original, y se han mejorado las páginas de acontecimientos históricos y la biblioteca.

Se han implementado todos los efectos requeridos así como los ficheros de JavaScript necesarios para el correcto modelaje de la página web.