

Hito Grupal

Lenguaje de Marcas



María Fernández
Mauro Ortiz
Alejandro Nicolás
Diego Rodrigues

ÍNDICE

Introducción XML(Diego).....	pág 2.
Introducción JSON(Alejandro).....	pág 4.
Memoria de proyecto (María y Mauro).....	pág 7.
Proceso QA(Todos).....	pág 21.
Conclusión(Todos).....	pág 22.

XML

¿Qué es?:

XML (Extensible Markup Language) es un lenguaje de marcado que se utiliza para almacenar y transmitir datos estructurados en la World Wide Web. XML es un estándar abierto y flexible que permite a los desarrolladores crear sus propios formatos de marcado para describir los datos, en lugar de estar limitados a un conjunto fijo de etiquetas predefinidas. Los documentos XML se pueden leer y utilizar por cualquier aplicación o dispositivo que sea compatible con XML.

XML es una tecnología ampliamente utilizada para la transmisión de datos entre aplicaciones y sistemas. Un documento XML contiene elementos y atributos que describen el contenido de los datos. Estos elementos y atributos se encierran entre etiquetas, que son similares a las etiquetas HTML utilizadas en la World Wide Web, pero más flexibles y potentes.

Además de describir el contenido de los datos, los documentos XML también pueden incluir información sobre su estructura y organización, lo que permite a los sistemas intercambiar información de forma eficiente y fácilmente comprensible.

XML es compatible con una amplia variedad de lenguajes de programación y plataformas, lo que lo hace ideal para la integración de sistemas y la creación de aplicaciones Web. Además, muchos formatos de archivos y protocolos de intercambio de datos utilizan XML como su formato base, incluyendo RSS, Atom, SOAP y muchos otros.

Historia:

El XML (Extensible Markup Language) fue desarrollado por el World Wide Web Consortium (W3C) a principios de los años 90 como una evolución del HTML (HyperText Markup Language), que se utilizaba para describir la estructura y contenido de las páginas Web.

En aquel entonces, la World Wide Web estaba ganando rápidamente popularidad y había una necesidad creciente de una forma más flexible y potente de describir la estructura de los datos en la Web.

Al mismo tiempo, había una demanda de una forma más eficiente de intercambiar datos entre aplicaciones y sistemas, lo que llevó al desarrollo de XML.

El primer borrador de especificación XML fue presentado en febrero de 1996 y se convirtió en un estándar formal en febrero de 1998. Desde entonces, XML se ha utilizado ampliamente para el intercambio de datos en la World Wide Web y en aplicaciones empresariales.

El éxito de XML se debe en parte a su flexibilidad y facilidad de uso. Al contrario de HTML, que tiene un conjunto fijo de etiquetas predefinidas, XML permite a los desarrolladores crear sus propios formatos de marcado personalizados para describir sus datos. Esto ha permitido la creación de muchos formatos de archivos y protocolos de intercambio de datos basados en XML, incluyendo RSS, Atom, SOAP y otros.

Normas de uso y sintaxis:

Algunas de las normas y reglas básicas que se deben seguir al usar XML incluyen:

1. Bien formado: Todos los documentos XML deben ser bien formados, lo que significa que deben tener una estructura clara y lógica.
2. Nombres de etiquetas únicos: Cada etiqueta en un documento XML debe tener un nombre único y no puede haber dos etiquetas con el mismo nombre en el mismo documento.
3. Anidación de etiquetas: Las etiquetas en un documento XML deben estar anidadas de manera correcta, lo que significa que una etiqueta hijo debe estar contenida dentro de su etiqueta padre.
4. Cierre de etiquetas: Todas las etiquetas en un documento XML deben ser cerradas, ya sea explícitamente con una etiqueta de cierre o implícitamente con una etiqueta autocerrable.

En las empresas:

En el ámbito empresarial, XML se utiliza ampliamente para intercambiar información entre sistemas y aplicaciones. Algunos de los usos más comunes incluyen:

1. Integración de sistemas: XML permite a los sistemas intercambiar información de forma clara y estructurada, lo que facilita la

integración de sistemas de diferentes empresas y la automatización de procesos empresariales.

2. Intercambio de datos: Muchas empresas utilizan XML para intercambiar datos con sus clientes, proveedores y otras empresas. Por ejemplo, una empresa puede utilizar XML para transmitir pedidos a un proveedor o recibir información sobre el estado de un pedido.
3. Documentación: XML se utiliza para crear y almacenar documentos en un formato estructurado y fácilmente comprensible. Por ejemplo, se pueden utilizar documentos XML para describir facturas, contratos y otros documentos empresariales.
4. Aplicaciones Web: Muchas aplicaciones Web utilizan XML para transmitir y recibir información de un servidor a un navegador. Por ejemplo, se pueden utilizar documentos XML para transmitir datos a una aplicación de escritorio desde un servidor Web

JSON

Historia:

JSON (JavaScript Object Notation) fue introducido por Douglas Crockford en el año 2000 como una alternativa ligera a XML para intercambiar datos en aplicaciones web. Desde entonces, se ha convertido en uno de los formatos de intercambio de datos más populares en el mundo, especialmente en el desarrollo de aplicaciones web y móviles.

Normas de uso y sintaxis:

JSON es un formato de texto plano que se basa en la sintaxis de objetos de JavaScript. Las normas de uso incluyen la representación de datos como pares clave-valor, donde la clave es una cadena de texto y el valor puede ser un número, una cadena de texto, un booleano, un objeto JSON, una matriz o null. Los objetos y matrices se encierran en corchetes y llaves, respectivamente. Las cadenas de texto se encierran en comillas dobles y los nombres de clave también deben estar en comillas dobles.

Aplicaciones en el ámbito empresarial:

JSON es ampliamente utilizado en el desarrollo de aplicaciones empresariales debido a su capacidad para transmitir datos de manera eficiente y fácil de leer. Algunas de las aplicaciones más comunes incluyen:

- **Integración de aplicaciones:** JSON es compatible con una amplia variedad de lenguajes de programación, lo que lo hace ideal para la integración de aplicaciones empresariales.
- **Intercambio de datos entre aplicaciones:** JSON permite la transmisión de datos entre aplicaciones de manera fácil y eficiente, lo que lo hace útil para la creación de sistemas distribuidos.
- **Almacenamiento de datos:** JSON es un formato de almacenamiento de datos popular, especialmente en aplicaciones web y móviles, debido a su capacidad para transmitir y almacenar datos de manera eficiente.
- **API:** JSON es un formato popular para los servicios web RESTful y las API, ya que permite la transmisión de datos entre aplicaciones de manera fácil y eficiente.

Características de JSON

- JSON es un lenguaje de modelado de datos
- Consiste en pares "clave - valor"
- Los valores pueden ser cadenas, números o booleanos, así como otros objetos JSON, con cualquier nivel de anidación
- Es un formato flexible, ligero y fácilmente transferible a través de las redes

Ventajas de JSON

- La lectura del código resulta de fácil lectura y la información es suficientemente expresiva para poder ser leída por personas, además de máquinas.
- El tamaño de los archivos que se transfieren es ligero.
- El código está basado en el lenguaje Javascript, lo que es ideal para las aplicaciones web.
- Todos los lenguajes disponen de funciones para interpretar cadenas JSON y convertir datos en cadenas JSON válidas.
- Se escribe en archivos de texto plano con codificación UTF8, que es compatible con todos los sistemas.

Ejemplo:

Archivo colores1.json	Archivo colores2.json	Archivo colores3.json
<pre>{ "arrayColores": [{ "nombreColor": "rojo", "valorHexadec": "#f00" }, { "nombreColor": "verde", "valorHexadec": "#0f0" }, { "nombreColor": "azul", "valorHexadec": "#00f" }, { "nombreColor": "cyan", "valorHexadec": "#0ff" }, { "nombreColor": "magenta", "valorHexadec": "#f0f" }, { "nombreColor": "amarillo", "valorHexadec": "#ff0" }, { "nombreColor": "negro", "valorHexadec": "#000" }] }</pre>	<pre>{ "arrayColores": [{ "rojo": "#f00", "verde": "#0f0", "azul": "#00f", "cyan": "#0ff", "magenta": "#f0f", "amarillo": "#ff0", "negro": "#000" }] }</pre>	<pre>{ "rojo": "#f00", "verde": "#0f0", "azul": "#00f", "cyan": "#0ff", "magenta": "#f0f", "amarillo": "#ff0", "negro": "#000" }</pre>

En resumen, JSON es un formato de intercambio de datos popular en el ámbito empresarial debido a su capacidad para transmitir y almacenar datos de manera eficiente, su compatibilidad con una amplia variedad de lenguajes de programación y su uso en aplicaciones web y móviles.

Memoria documentada del proyecto Web

La organización en la que nos hemos basado para estructurar la web, ha sido la creación de una web principal o “Home” llamada ‘web1.html’ donde nos encontraremos principalmente la barra de menú horizontal con los apartados ‘XML’, ‘JSON’, ‘CONTACTO’ y el menú desplegable con una información más extensa como links a vídeos y a redes sociales.

Seguidamente nos encontramos con dos archivos llamados ‘xml.html’ y ‘json.html’ en los que hemos creado otras dos webs vinculadas a la principal, siguiendo el mismo formato que la home para conseguir y buscar una mayor armonía y cohesión con la web. Por consiguiente, nos encontramos con el archivo ‘contacto.html’ en el que se presenta un formulario de petición por parte del usuario, para darle veracidad y realismo a la web. De igual forma hemos utilizado el mismo formato para dar una mayor cohesión con la web principal.

Por otra parte, en la carpeta se encuentran 2 archivos css, llamados ‘style1.css’ vinculado a ‘web1.html’, ‘xml.html’ y ‘json.html’ y ‘style2.css’ vinculado a nuestro formulario ‘contacto.html’.

Esta es la estructura de nuestra web, pasaremos a detallar y a explicar el diseño, cómo hemos implementado en nuestra web responsive (QA, system grid, flexbox, bootstrap..)

WEB1.HTML

```
● ● ●  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4      <meta charset="UTF-8">  
5      <meta name="viewport" content="width=device-width, initial-scale=1">  
6      <title>Schoolweb</title>  
7      <link rel="stylesheet" href="style1.css">  
8  </head>  
9  <body>  
10     <p class="centrado">SCHOOLWEB</p>  
11     <header class="header">  
12         <div class="container">  
13             <div class="btn-menu">  
14                 <label for="btn-menu">☰</label>  
15             </div>  
16             <div class="logo">  
17             </div>  
18             <nav class="menu">  
19                 <a href="web1.html">Home</a>  
20                 <a href="xml.html">XML</a>  
21                 <a href="json.html">JSON</a>  
22                 <a href="contacto.html">Contacto</a>  
23             </nav>  
24         </div>  
25     </div>  
26     </header>  
27     <div class="capa"></div>
```

```
● ● ●  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4      <meta charset="UTF-8">  
5      <meta name="viewport" content="width=device-width, initial-scale=1">  
6      <meta name="keywords" content="schoolweb, xml, json">  
7      <meta name="description" content="Es una web educativa que introduce una resumida introducción sobre archivos  
8      XML y JSON con acceso a tutoriales">  
9      <meta name="title" content="schoolweb">  
10     <title>Schoolweb</title>  
11     <link rel="stylesheet" href="style1.css">  
12 </head>
```

Aquí implementamos el proceso QA. Hemos indexado nuestra web mediante: <meta name="keywords"> <meta name="description"> y con <meta name="title">.

Para comenzar, abrimos nuestro archivo ‘web1.html’ y nos encontramos con el código con el que hemos diseñado nuestra web ‘home’ he dividido las explicaciones en dos partes para facilitar el entendimiento y la visión que le hemos querido dar a nuestra web. Utilizamos bootstrap, le ponemos un título a nuestra web, como se puede observar la hemos llamado ‘schoolweb’ y hemos vinculado la hoja de estilos llamada ‘style1.css’ En el cuerpo de la página o <body> hemos añadido una <p class='centrado'> donde incluiremos el título oficial de nuestra página web, así nos resultará más fácil centrarlo en la hoja de estilos. Guardándolo en clases.

En el header hemos incluido una clase llamada ‘header’ y dentro de esta dos <div class> una llamada container para editar después el contenedor de la barra de menú y otra llamada ‘btn-menu’ para incluir el mediante etiquetas <label> icono de las 3 líneas que posteriormente nos abrirá un menú lateral desplegable.

Creamos un <div logo> para los titulares del menú ya que con <nav class="menu"’ creamos el menú principal de la página con los nombres ‘HOME’, ‘XML’, ‘JSON’, ‘CONTACTO’. Vinculamos nuestras otras webs y nuestro formulario a esa barra de navegación.

Fin de nuestro <header>.



```
1 <div class="capa"></div>
2 <!-- ----->
3 <input type="checkbox" id="btn-menu">
4 <div class="container-menu">
5   <div class="cont-menu">
6     <nav>
7       <a href="https://www.youtube.com/watch?v=k-wfUkqMm-w">XML</a>
8       <a href="https://www.youtube.com/watch?v=QMTJ_z_EHC8">JSON</a>
9       <a href="https://www.facebook.com/hashtag/xml">Facebook</a>
10      <a href="https://twitter.com/">Twitter</a>
11    </nav>
12    <label for="btn-menu">X</label>
13  </div>
14 </div>
```

Continuamos la web creando un `<div class="capa">` con el que editaremos en el css el contenedor del menú lateral que creamos a continuación.

Para darle realismo al menú lateral, metemos un checkbox con `id="btn-menu"` porque queremos abrir y cerrar tantas veces como queramos por decisión propia ese menú lateral.

Creamos el contenedor de ese menú con `<div class='container-menu'>` que será el genérico y creamos dentro de éste un `<div class='cont.menu'>` donde implementaremos nuestra barra de navegación.

ATENCIÓN: Hemos vinculado tutoriales sobre XML Y JSON para realizar consultas y hacer una práctica de ambos. Después hemos añadido los links de Facebook y twitter.

Al final de este menú incorporamos la etiqueta `<label for="btn-menu">`(icono de cerrar) con el que nos permitirá cerrar el menú lateral desplegable con el checkbox tantas veces como queramos.



```
1 <div class="box">
2   <div><p>El objetivo fundamental de XML es intercambiar datos estructurados entre sistemas de información, fundamentalmente a través de Internet. Se trata de un formato de texto plano, lo que facilita enormemente la transferencia de información, logrando independencia con respecto a las diferentes plataformas.<br><br><br>
3   El uso principal de JSON es el intercambio de datos entre aplicaciones. Es un formato independiente del lenguaje, por lo que hay numerosas librerías que lo
4   soportan y facilitan su utilización en cualquier lenguaje de programación moderno.</p></div>
5
6
7
8
9 <div class="copyright">© <span id="year">2023</span> Connection Soft Service<br>Mauro Fernández<br>Diego Rodríguez<br>Alejandro Nicolás</div>
10
11
12 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-w7oAqPFDkWBOx08JS1gez6pr3xSMlQ1ZAGC+nuZB+EYdgr2giwxhTBkf7CXwN" crossorigin="anonymous"></script>
13 </body>
14 </html>
```

Después, hemos decidido crear un `<div class=box>` donde incluiremos una breve introducción de XML y de JSON.

Al final le hemos añadido nuestros nombres y unos términos de verificación de la página web.

STYLE1.CSS

```

1 @import url('https://fonts.googleapis.com/css?family=Montserrat|Montserrat+Alternates|Poppins&display=swap');
2 {
3     margin: 0;
4     padding: 0;
5     box-sizing: border-box;
6     font-family: 'Montserrat Alternates', sans-serif;
7 }
8 body{
9     background: url(https://4kwallpapers.com/images/walls/thumbs\_2t/5630.jpg);
10    background-size: 100vw 100vh;
11    background-repeat: no-repeat;
12 }

```

En nuestra hoja de estilos llamada 'Style1.css' añadiremos los estilos apropiados a nuestra web.

Hemos importado de google una fuente para darle un toque más contemporáneo, hemos importado un fondo de web de internet con un 'background-size' del 100% de píxeles de la página web.

```

1 /*Estilo en 3D para el título*/
2 .centrado{
3     display: flex;
4     justify-content: center;
5     margin-top: 100px;
6     font-family: 'Montserrat Alternates', sans-serif Black;
7     font-weight: bold;
8     font-size: 50px;
9     color: #fff;
10    text-shadow: 0 1px 0 #ddd, 0 2px 0 #ccc, 0 3px 0 #bbb, 0 4px 0 #aaa, 0 5px 0 #acacac, 0 6px 1px rgba(0,0,0,0.1), 0 8px 5px rgba(0,0,0,0.1), 0 1px 3px rgba(0,0,0,0.3), 0 3px 5px rgba(0,0,0,0.2), 0 5px 10px rgba(0,0,0,0.25), 0 10px 10px rgba(0,0,0,0.2), 0 20px 20px rgba(0,0,0,0.15));
11 }
12 /*Estilo para caja de texto*/
13 .box{
14     background-color:#f1e22357a;
15     position: relative;
16     font-family: 'Montserrat Alternates' sans-serif white;
17     margin-left:80px ;
18     margin-right: 80px;
19     justify-content: space-between;
20     height: 80px;
21     text-align: center;
22     display: flex;
23     flex-direction: column;
24     color: #ccc;
25 }

```

Decidimos resaltar el texto buscando efectos en 3D o que dieran ese efecto visual resaltado respecto del resto de la web.

Así que para nuestra clase '.centrado' donde se encontraba nuestro título hemos añadido display flex ya que es una propiedad resumida que indica la capacidad de un elemento flexible indicar la capacidad de un elemento flexible para alterar sus dimensiones y llenar el espacio disponible, <justify center> para alinearla al centro, un ancho un alto y un <text-shadow> que encontramos por internet que da ese toque efecto 3D.

Para nuestra caja de texto de la web principal con la introducción de XML y de JSON, en nuestra clase '.box' le aplicamos un fondo de caja, una posición relativa, un margin-left y margin-right y en justify-content un 'space between' para organizar la información ya que no queremos mezclar el texto de xml con el texto de json. Añadimos display:flex, y un flex column para que se ordene un texto debajo de otro, así como si fuera una columna. A la fuente le añadimos un tipo y un color.

```
1  /* Estilo para texto en web principal */
2  .box p1{
3      display: table-cell;
4      padding: 0 1rem;
5      font-size: 1.2rem;
6      margin: auto;
7      vertical-align: middle;
8      position: absolute;
9      top: 100px;
10     justify-content: center;
11     margin-top: 70px;
12     margin-left: 80px;
13     margin-right: 80px;
14
15 }
16 .capa{
17     position: fixed;
18     width: 100%;
19     height: 100vh;
20     background: rgba(0,0,0,0.6);
21     z-index: -1;
22     top: 0; left: 0;
23 }
```

Con nuestra clase creada '.box p1{}' le damos estilo para el texto en la web principal, con un display table-cell ordenado por columnas anteriormente. Un padding y en font-size en rem porque si estás haciendo un estilo que pretendes utilizar en muchos elementos distintos, por ejemplo una class de CSS que pretende usar en varias

partes, es mejor usar rem. Añadimos unos márgenes y un justify-center centrado.

```
● ● ●

1  /*Estilos para el encabezado*/
2  .header{
3      width: 100%;
4      height: 100px;
5      position: fixed;
6      top: 0; left: 0;
7  }
8  .container{
9      width: 90%;
10     max-width: 1200px;
11     margin: auto;
12 }
13 .container .btn-menu, .logo{
14     float: left;
15     line-height: 100px;
16 }
17 .container .btn-menu label{
18     color: #fff;
19     font-size: 25px;
20     cursor: pointer;
21 }
22 .logo h1{
23     color: #fff;
24     font-weight: 400;
25     font-size: 22px;
26     margin-left: 10px;
27 }
28 .container .menu{
29     float: right;
30     line-height: 100px;
31 }
32 .container .menu a{
33     display: inline-block;
34     padding: 15px;
35     line-height: normal;
36     text-decoration: none;
37     color: #fff;
38     transition: all 0.3s ease;
39     border-bottom: 2px solid transparent;
40     font-size: 15px;
41     margin-right: 5px;
42 }
43 .container .menu a:hover{
44     border-bottom: 2px solid #c7c7c7;
45     padding-bottom: 5px;
46 }
47 /*Fin de Estilos para el encabezado*/
```

Los estilos para el encabezado los hemos distribuido por orden. en '.header' he añadido un tamaño de ancho y de alto y la posición he elegido 'fixed' que hace que un elemento se coloque respecto al viewport, Le dices dónde posicionarse al elemento, y se queda en la posición que le damos mientras el usuario hace scroll a la web.

En nuestra clase '.container' le hemos aportado un margen automático y le hemos aplicado un tamaño máximo.

Para nuestra clase hija '.container .btn-menu logo y label{}' le hemos aportado unas características como 'cursor:pointer' para que cuando el cursor se posicione encima de esos elementos cambie la forma predeterminada.

Como observamos para la clase padre '.container' vinculado a las clases hijas les hemos aplicado para el '.menú .a' un 'display:inline block' que es una combinación entre los dos tipos de elementos (block e inline), los elementos con el valor inline-block admiten dimensiones pero todavía son elementos de línea, es decir estarán colocados uno al lado del otro. Y algo que resultó interesante de buscar fue la transición medida en 0.3ms.

También añadimos ':hover' al menú para añadir efecto cuando el cursor del ratón se posicione encima del elemento al que va vinculado en el código.

```
1  /*Menú lateral*/
2  .capa{
3      position: fixed;
4      width: 100%;
5      height: 100vh;
6      background: rgba(0,0,0,0.6);
7      z-index: -1;
8      top: 0; left: 0;
9  }
10 #btn-menu{
11     display: none;
12 }
13 .container-menu{
14     position: absolute;
15     background: rgba(0,0,0,0.5);
16     width: 100%;
17     height: 100vh;
18     top: 0; left: 0;
19     transition: all 500ms ease;
20     opacity: 0;
21     visibility: hidden;
22 }
23 #btn-menu:checked ~ .container-menu{
24     opacity: 1;
25     visibility: visible;
26 }
27 .cont-menu{
28     width: 100%;
29     max-width: 250px;
30     background: black;
31     height: 100vh;
32     position: relative;
33     transition: all 500ms ease;
34     transform: translateX(-100%);
35 }
36 #btn-menu:checked ~ .container-menu .cont-menu{
37     transform: translateX(0%);
38 }
39 .cont-menu nav{
40     transform: translateY(15%);
41 }
42 .cont-menu nav a{
43     display: block;
44     text-decoration: none;
45     padding: 20px;
46     color: #c7c7c7;
47     border-left: 5px solid transparent;
48     transition: all 400ms ease;
49 }
50 .cont-menu nav a:hover{
51     border-left: 5px solid #c7c7c7;
52     background: #1f1f1f;
53 }
54 .cont-menu label{
55     position: absolute;
56     right: 5px;
57     top: 10px;
58     color: #fff;
59     cursor: pointer;
60     font-size: 18px;
61 }
62 .copyright{
63     opacity: .8; /*opacidad*/
64 }
```

Continuamos ajustando la posición del menú lateral con la clase capa, donde metemos los siguientes parámetros de ancho y altura, también le metemos fixed para fijar la barra cuando el usuario haga scroll en la web, además ajustamos los márgenes para colocar la barra a nuestro modo.

En la clase .btn-menu quitamos la decoración.

Después procedemos a editar el estilo de la clase '.container-menu', que es el contenedor base del menú lateral, introducimos la posición absoluta que hace que un elemento se coloque respecto al contenedor posicionado más cercano manteniendo los mismos parámetros de largo y ancho que hemos puesto anteriormente, y hemos seguido ajustando para hacer la barra lateral.

Después hemos seguido editando con clases hija de la clase padre '.container menú' donde hemos editado el color, posicionamiento y forma para finalizar la barra del menú lateral.

Para los archivos html llamados 'xml.html' y 'json.html' lo que hemos decidido hacer es mantener el formato original de la Home con las mismas clases, el mismo menú de navegación horizontal y el menú desplegable vertical, lo único que hemos variado ha sido la caja '.box' el texto de cada una de las páginas. Información de XML recogida del documento escrito e información de JSON recogida del mismo documento.

Pasamos a documentar el formulario llamado 'contacto.html'

CONTACTO.HTML

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <link rel="stylesheet" href="style2.css">
8   <link rel="stylesheet" href="web1.html">
9   <title>Contáctanos</title>
10 </head>
11 <body>
12   <div class="content">
13     <h1 class="logo">Contáctanos</h1>
14
15   <div class="contactopadre">
16     <div class="contact-form">
17       <h3>Contáctanos</h3>
18       <form action="web1.html">
19         <p>
20           <label>Nombre</label>
21           <input type="text" name="fullname">
22         </p>
23         <p>
24           <label>Email</label>
25           <input type="email" name="email">
26         </p>
27         <p>
28           <label>Teléfono</label>
29           <input type="tel" name="phone">
30         </p>
31         <p class="block">
32           <label>Escribe tu consulta</label>
33           <textarea name="message" rows="3"></textarea>
34         </p>
35         <p class="block">
36           <button>
37             Enviar
38           </button>
39         </p>
40       </form>
41     </div>
42   </div>
43
44   </div>
45
46 </div>
47
48   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-w76AqdFDKMBDXo30j51Sgez6pr3xSMlQ1ZAC+nuZB+EYdgRZgiwxnTB1KF7CXvN" crossorigin="anonymous"></script>
49 </body>
50 </html>
51 </body>
52 </html>

```

Este es nuestro código del archivo 'contacto.html' donde hemos creado un formulario con la etiqueta `<form>` en el action podríamos importar un archivo .php que guarde o almacene las peticiones que se realicen tras llenar completamente el formulario, mediante etiquetas `<label>` importamos varios inputs para que el usuario pueda llenar los campos y hemos creado una clase block en mensajes para editarlos con css más detalladamente. Aplicamos un botón de enviar con `<button>` final. Hemos vinculado este archivo a la hoja de estilos llamada 'style2.css'.

STYLE.CSS

```
1 @import url('https://fonts.googleapis.com/css?family=Montserrat|Montserrat+Alternates|Poppins&display=swap');
2
3 * {
4     box-sizing: border-box; /*para aplicar un borde a la caja*/
5 }
6 body {
7
8     color: #fff;
9     line-height: 1.6;
10    padding: 0 1.5em;
11    background: url(https://4kwallpapers.com/images/walls/thumbs\_2t/5630.jpg);
12    background-size: 100vw 100vh;
13    background-repeat: no-repeat;
14 }
15
```

Para este archivo hemos importado de google la misma tipografía que en el resto de archivos para mantener la armonía de la web, agregamos a toda la página un 'border-box' y para el cuerpo hemos añadido un fondo de pantalla con unas medidas de 100vw y 100vh.

```
1 .content {
2     max-width: 1170px;
3     margin-left: auto;
4     margin-right: auto;
5 }
6 ul {
7     list-style: none;
8     padding: 0;
9 }
10 .logo {
11     text-align: center;
12     font-size: 30px;
13     font-family: 'Montserrat Alternates', sans-serif;
14 }
15
16 .logo span {
17     color: #073170cc;
18 }
19 }
20 .contactopadre {
21     box-shadow: 0 0 20px 0 rgba(255, 255, 255, .3); /*efecto sombreado del formulario*/
22 }
```

Hemos ido dando estilo a las clases creadas anteriormente y para resaltar hemos añadido a clase ‘.contactopadre’ un efecto sombreado que da mucha estética visual y moderna.



The screenshot shows a code editor window with a dark theme. At the top left, there are three colored circular icons: red, yellow, and green. The main area contains approximately 49 lines of CSS code. The code is organized into several sections, each starting with a class selector like '.contact-form' or '.contact-form h3'. The styling includes background colors, font families ('Montserrat Alternates', sans-serif), grid layouts, and specific styling for input fields and buttons. A notable section at the bottom defines styles for the ':hover' and ':focus' pseudo-selectors of a button, indicating a color transition effect.

```
1 .contact-form {  
2     background: #12123bbd;  
3 }  
4 .contact-form h3{  
5     font-family: 'Montserrat Alternates', sans-serif;  
6 }  
7 .contact-form form {  
8     display: grid;  
9     grid-template-columns: auto;  
10 }  
11 .contact-form form label {  
12     display: block;  
13 }  
14 .contact-form form p{ /*Separación de cajas de inputs*/  
15     font-family: 'Montserrat Alternates', sans-serif;  
16     margin: 0;  
17     padding: 1em;  
18 }  
19 .contact-form form .block {  
20     grid-column: 1 / 3;  
21 }  
22  
23 /* estilos de linea de separación entre cajas de inputs boton de enviar y areatext*/  
24 .contact-form form button,  
25 .contact-form form input,  
26 .contact-form form textarea {  
27     width: 100%;  
28     padding: .7em;  
29     border: none;  
30     background: none;  
31     outline: 0;  
32     border-bottom: 1px solid #1a33a1;  
33 }  
34  
35 .contact-form form button {  
36     background: #3141d3d0;  
37     border: 0;  
38     text-transform: uppercase;  
39     padding: 1em;  
40 }  
41  
42 .contact-form form button:hover, /*Hover para dar ese efecto resaltado*/  
43 .contact-form form button:focus {  
44     background: #0e1099c5;  
45     color: #fff;  
46     transition: background-color 1s ease-out;  
47     outline: 0;  
48 }  
49 }
```

Hemos querido desglosar la clase hija ‘.contact-form’ poco a poco para ir dando un estilo más definido ya que si añadimos los estilos directamente en un solo párrafo podría dar lugar a error de organización. Le hemos aportado estilo a la separación entre los inputs incluidos los botones de ‘Enviar’ y el ‘areatexto’ que es la caja donde el usuario puede introducir texto como peticiones y preguntas.

Hemos elegido poner ‘text-transform: uppercase’ en el botón de enviar, ya que al introducir el mensaje se especifica el cambio entre mayúsculas y minúsculas del texto de un elemento.

Utilizamos ‘hover’ para dar ese efecto de resaltado en el botón de ‘enviar’.

Utilizamos ‘focus’ ya que representa un elemento que ha recibido el foco, y se activa cuando el usuario da clic.

PROCESO QA:

El proceso de QA (Quality Assurance) en HTML se refiere a la serie de pruebas y controles de calidad que se llevan a cabo en un sitio web o aplicación web para garantizar su correcto funcionamiento, accesibilidad, rendimiento, seguridad y usabilidad.

A continuación se describe de manera general el proceso de QA en HTML:

Verificación de la sintaxis: Se comprueba que el código HTML sea válido y cumpla con los estándares establecidos por el W3C (World Wide Web Consortium). Para ello se pueden utilizar herramientas como el validador de HTML del W3C.

Pruebas de compatibilidad: Se realizan pruebas en diferentes navegadores (Chrome, Firefox, Safari, Edge, etc.) y en diferentes dispositivos (ordenadores de escritorio, móviles, tablets, etc.) para comprobar que el sitio web se visualiza correctamente en todos ellos.

Pruebas de accesibilidad: Se llevan a cabo pruebas para comprobar que el sitio web cumple con las pautas de accesibilidad establecidas por el WAI (Web Accessibility Initiative), que garantizan que todas las personas, incluyendo aquellas con discapacidades, puedan acceder al contenido de la web.

Pruebas de rendimiento: Se miden y analizan los tiempos de carga de la página, la velocidad de respuesta y otros parámetros que afectan al rendimiento del sitio web. Para ello se pueden utilizar herramientas como Google PageSpeed Insights.

Pruebas de seguridad: Se realizan pruebas para detectar posibles vulnerabilidades en el sitio web que puedan ser explotadas por atacantes. Se pueden utilizar herramientas como OWASP ZAP para llevar a cabo pruebas de seguridad.

Pruebas de usabilidad: Se llevan a cabo pruebas para comprobar que la navegación y la interacción con el sitio web son intuitivas y sencillas para los usuarios. Se pueden utilizar herramientas como UserTesting para realizar pruebas de usabilidad.

Una vez que se han completado todas estas pruebas, se deben corregir los errores y fallos detectados y volver a realizar las pruebas necesarias para asegurarse de que todo funciona correctamente. De esta manera, se garantiza que el sitio web cumpla con los estándares de calidad y proporcione una buena experiencia de usuario.

En nuestra web como explicamos anteriormente implementamos este proceso con:



```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1">
6      <meta name="keywords" content="schoolweb, xml, json">
7      <meta name="description" content="Es una web educativa que introduce una resumida introducción sobre archivos
8      XML y JSON con acceso a tutoriales">
9      <meta name="title" content="schoolweb">
10     <title>Schoolweb</title>
11     <link rel="stylesheet" href="style1.css">
12  </head>
```

CONCLUSIÓN

Ha resultado muy didáctico la capacidad de buscar el realismo y ese toque estético más moderno a nuestra web, hemos recurrido a páginas web donde te explican cómo implementarlo y qué efectos divertidos y llamativos implementar para nuestra web. Así como el ícono de cierre y de despliegue de menú, efectos en 3D, hover, focus, etc.

Hemos implementado método responsive a nuestra página, proceso QA, media query y grid system, lo que nos ha permitido aún más generar esa visión de web realista y contemporánea.