

# Html en soluciones IoT

## Objetivo

El objetivo de esta página es lograr conocer los fundamentos del protocolo HTTP, de HTML, de las herramientas como editores, simuladores y servidores web, sin JavaScript por ahora, pero de manera suficiente para que con ayuda de Chat GPT el estudiante pueda cómodamente superar cualquier dificultad en soluciones para IoT en el lado del cliente. Esto puede ser visto como el preámbulo a las Web Apps donde se trabajarán soluciones cliente-servidor con su Front-End y su Back-End. En una siguiente práctica se incluirá el uso de JavaScript para concluir los conocimientos necesarios para crear soluciones de Front-End.

## Conocimientos previos

### Lenguaje HTML

HTML significa lenguaje de marcado de hipertexto. HTML es un lenguaje utilizado para estructurar un contenido visual amigable a la vista de los humanos. En este sentido es muy similar al lenguaje que usan editores como Word o Latex, pero el HTML tiene una ventaja - es el mundialmente aceptado para los navegadores web. HTML hace que el contenido de una página web pueda ser interpretado como en forma de texto para que sea fácilmente interpretado en un punto final. Es posible ver el HTML como un conjunto de órdenes que se le dan a un navegador para que logre construir una página en una forma deseada.

### Servidor web

El servidor web es una pieza de software que almacena y entrega contenido web. Esto podría ser un servidor físico o una infraestructura en la nube. Cuando se utiliza AJAX en una página web, las solicitudes asíncronas se envían al servidor web para obtener o enviar datos, como información de la base de datos, archivos, actualizaciones en tiempo real, etc.

### Servicio de páginas web

El uso más común del protocolo HTTP consiste en un servidor web en la nube que contiene el código html de una página web. Por solicitud de un navegador, el servidor web envía el código html al navegador, donde el código es interpretado y mostrado como la página deseada. Como se muestra en la siguiente figura, un usuario escribe la URL de una página web en un navegador, el navegador hace una solicitud GET al servidor, el servidor empaqueta el texto de la página, es decir el html en el paquete protocolo HTTP. Una vez

llega el paquete al navegador, en el computador del cliente, este extrae el cuerpo del mensaje que corresponde a la versión en HTML de la página, interpreta ese HTML para traducirlo a una forma más amigable para el ser humano.

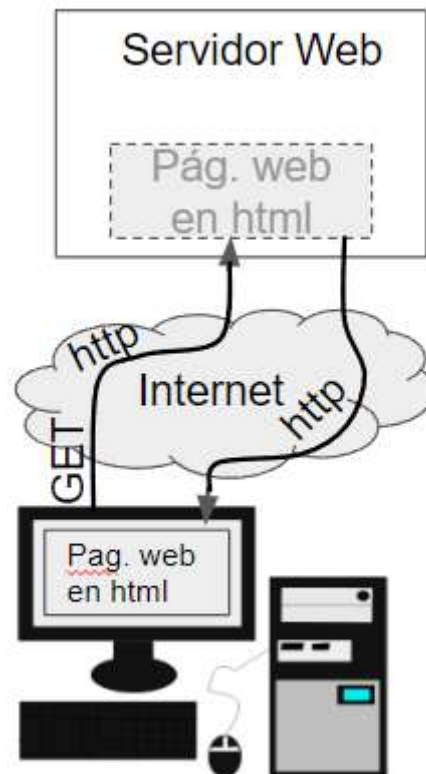


Fig. 2. Funcionamiento de un servicio de página web.

Con el tiempo, los navegadores han evolucionado para interpretar no solo contenido HTML, sino también código JavaScript y contenido CSS en una página web. Esto es un hecho clave en el desarrollo web moderno.

## Un elemento HTML

Un documento HTML está compuesto de elementos. Un elemento está compuesto de una etiqueta de apertura y una de cierre. En medio de las etiquetas va el contenido, como se muestra en la Fig. 3

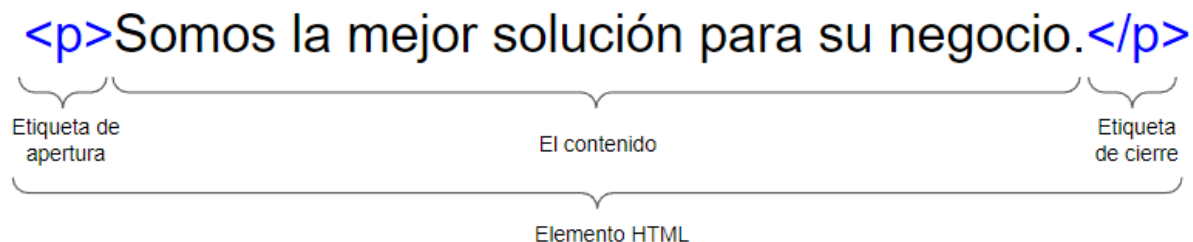


Fig. 3. Cómo es un elemento HTML

- La etiqueta de apertura: consiste en el nombre del elemento (en este caso, p), encerrado por paréntesis angulares (< >) de apertura y cierre. Establece dónde

comienza o empieza a tener efecto el elemento - en este caso, dónde es el comienzo del párrafo

- La etiqueta de cierre: es igual que la etiqueta de apertura, excepto que incluye una barra de cierre (/) antes del nombre de la etiqueta. Establece dónde termina el elemento - en este caso dónde termina el párrafo.
- El contenido: este es el contenido del elemento, que en este caso es sólo texto.

### Los atributos en HTML:

Se trata de información del elemento que no debe ser vista por el cliente (usuario) pero que contiene información sobre el elemento. Es decir, son metadatos. Es usualmente una pareja nombre=valor. Va dentro de la etiqueta de apertura del elemento. En el ejemplo de la siguiente figura, "class" es el nombre del atributo, "editor-note" es el valor del atributo. Se usa mucho para crear marcas que pueden ser usadas por código de JavaScript o por CSS.

`<p class="editor-note">Somos la mejor solución para su negocio.</p>`

Atributo

Fig. 4. Partes de un atributo dentro de un elemento

## El anidamiento en HTML

Es posible anidar un elemento dentro de otro, por ejemplo

```
<p>Mi gato es <strong>muy</strong> gruñón.</p>
```

## Elementos vacíos

Es posible encontrar elementos HTML que no están pensados para visualizar información, de manera que la etiqueta de apertura y la de cierre es diferente. Por ejemplo:

```
<img>
```

## Anatomía de un documento HTML

Los seres humanos tenemos una cabeza, un tronco y extremidades, de igual manera, un documento HTML tiene mínimamente un preámbulo, un encabezado y un cuerpo. Esas partes del documento se definen usando etiquetas, de manera que también podemos hablar de unas etiquetas mínimas en un documento HTML.

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Mi página de prueba</title>
  </head>
  <body>
    <p>Mi gato es <strong>muy</strong> gruñón.</p>
```

```
</body>
</html>
```

Veamos las etiquetas mínimas a usar:

- **<!DOCTYPE html>**: Para indicar el tipo de documento. Anteriormente, cuando HTML era joven (cerca de 1991/2), los tipos de documento actuaban como vínculos a un conjunto de reglas que el código HTML de la página debía seguir para ser considerado bueno, lo que podía significar la verificación automática de errores y algunas otras cosas de utilidad. Sin embargo, hoy día es simplemente un artefacto antiguo que a nadie le importa, pero que debe ser incluido para que todo funcione correctamente.
- **<html></html>** Todo lo que aparece en medio de estas etiquetas es lo que se conoce como el elemento `<html>`, a veces se le conoce como el elemento raíz o root element. Este elemento encierra todo el contenido de la página entera.
- **<head></head>** Todo lo que aparece en medio de esas etiquetas conforma el elemento `<head>`. Este elemento actúa como un contenedor de todo aquello que quieres incluir en la página HTML que no es contenido visible para los visitantes de la página. Incluye cosas como palabras clave (keywords), una descripción de la página que quieres que aparezca en resultados de búsquedas, código CSS para dar estilo al contenido, declaraciones del juego de caracteres, etc. El elemento `<head>` tiene unos elementos mínimos como:
  - **<meta charset="utf-8">**: Es el elemento `<meta>`. Este elemento establece el juego de caracteres que tu documento usará en utf-8, que incluye casi todos los caracteres de todos los idiomas humanos. Básicamente, puede manejar cualquier contenido de texto que puedas incluir. No hay razón para no establecerlo, y puede evitar problemas en el futuro.
  - **<title></title>** - el elemento `<title>` establece el título de tu página, que es el título que aparece en la pestaña o en la barra de título del navegador cuando la página es cargada, y se usa para describir la página cuando es añadida a los marcadores o como favorita.
- **<body></body>** - el elemento `<body>`. Encierra todo el contenido que desees mostrar a los usuarios web que visiten tu página, ya sea texto, imágenes, videos, juegos, pistas de audio reproducibles, y demás.

## Otras etiquetas de marcado de texto

En el tema anterior se vieron etiquetas mínimas en un documento HTML. Ahora podemos hablar de otras que son de uso frecuente

### Etiquetas de encabezados

```
<h1>Mi título principal</h1>
```

```
<h2>Mi título de nivel superior</h2>
```

```
<h3>Mi subtítulo</h3>
```

```
<h4>Mi sub-subtítulo</h4>
```

## Código CSS

Significa Cascading Style Sheets. Se trata de código que puede aparecer en un documento HTML, en el elemento `<head>` para responder a preguntas como: ¿Cómo hago mi texto rojo o negro? ¿Cómo hago que mi contenido se muestre en tal y tal lugar de la pantalla? ¿Cómo decoro mi página web con imágenes de fondo y colores?. Es un *lenguaje de hojas de estilo*, es decir, te permite aplicar estilos de manera selectiva a elementos en documentos HTML.

Muchas veces el código se escribe en un archivo `style.css`. Pero si se hace de esa manera es necesario referenciar ese archivo en el elemento `<head>` del documento HTML.

La anatomía de una orden CSS es como se muestra en la siguiente figura

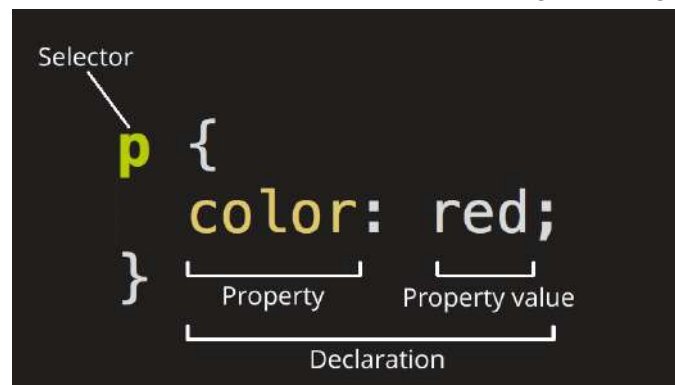


Figura 5. Anatomía del código CSS

**El selector:** determina a qué cosa aplica la orden. Para el ejemplo de la figura anterior, aplica para a los elementos `<p>` que hayan en el cuerpo del documento

**Par clave:valor:** Entre llaves vemos un conjunto de pares clave: valor, tal como se hace en los objetos de datos de JavaScript. Nota: a diferencia de los objetos de JavaScript, donde se usa como (,) para separar una declaración de otra, aquí se debe usar el punto y coma (;). Para el ejemplo de la Figura anterior significa que los elementos `<p>` en el cuerpo aparecerán en color rojo

En realidad un CSS es un elemento `<style>` dentro del elemento `<head>`. Ejemplo de CSS

```
<head>
  <title>Mi título</title>
  <style>
    p {
      color: red;
      width: 500px;
      border: 1px solid black;
    }
  </style>
</head>
```

Los selectores más comunes aparecen en la siguiente tabla

Nombre del selector	Qué selecciona	Ejemplo
Selector de elemento (llamado algunas veces selector de etiqueta o tipo)	Todos los elementos HTML del tipo especificado.	p Selecciona <p>
Selector de identificación (ID)	El elemento en la página con el ID especificado (en una página HTML dada, solo se permite un único elemento por ID).	#mi-id Selecciona <p id="mi-id"> y <a id="mi-id">
Selector de clase	Los elementos en la página con la clase especificada (una clase puede aparecer varias veces en una página).	.mi-clase Selecciona <p class="mi-clase"> y <a class="mi-clase">
Selector de atributo	Los elementos en una página con el atributo especificado.	img[src] Selecciona  pero no <img>
Selector de pseudoclase	Los elementos especificados, pero solo cuando esté en el estado especificado, por ejemplo cuando el puntero esté sobre él.	a:hover Selecciona <a>, pero solo cuando el puntero esté sobre el enlace.

## Frameworks y Bibliotecas:

Muchos desarrolladores utilizan frameworks y bibliotecas como Bootstrap, jQuery, React, Angular, y otros para agilizar el desarrollo y mejorar la eficiencia.

## HTML con procesamiento (procesamiento en el frontend)

Una página que baja de un servidor a un navegador puede incorporar código de procesamiento para que se ejecute en el lado del cliente. Javascript es el lenguaje usado para este tipo de código.

### JavaScript:

JavaScript es un lenguaje de programación que se utiliza para agregar interactividad y funcionalidad a una página web. Puedes utilizar JavaScript para crear efectos dinámicos, validar formularios, interactuar con el usuario y realizar solicitudes a servidores.

## AJAX

AJAX significa Asynchronous JavaScript and XML. Se trata de una solución que incorporan los navegadores para atender solicitudes de código JavaScript que se aloja en páginas Web. Para esto ocurre una comunicación asincrónica entre el JavaScript en cliente-AJAX-servidor web, logrando cambios en la vista de la página que ve el cliente sin necesidad de recargar toda la página del cliente. Como resultado, la página puede cambiar de manera dinámica.



## DOM

(Document Object Model) es un concepto fundamental en la programación web y está intrínsecamente relacionado con JavaScript. El DOM es una representación estructurada y jerárquica de un documento HTML o XML, que permite a los programadores acceder y manipular los elementos de una página web de manera dinámica a través de JavaScript.

El DOM organiza todos los elementos de una página web en una estructura de árbol, donde cada elemento (como etiquetas HTML) es representado como un objeto en ese árbol. Esto

significa que cada parte de una página web, como encabezados, párrafos, enlaces, imágenes, formularios, etc., es representada por un objeto en el DOM. JavaScript puede interactuar con estos objetos para realizar acciones como cambiar el contenido, modificar estilos, agregar o eliminar elementos y más.

En resumen, el DOM es una interfaz que proporciona una representación programática de una página web, permitiendo a JavaScript interactuar con los elementos de la página de manera dinámica. JavaScript puede acceder a los objetos del DOM y manipularlos para crear aplicaciones web interactivas y dinámicas.

El siguiente enlace permite consultar diversos comandos del DOM: <https://developer.mozilla.org/es/docs/Web/API/Document>

## Como se embebe código JavaScript en un documento html

- Usualmente el código JavaScript se introduce como un elemento `<script>`. Por lo tanto aparece así  
`<script>` Aquí va el código del JavaScript `</script>`
- Si el script es un conjunto de funciones, el elemento `<script>` puede ir en principio en cualquier parte dentro del elemento `<html>`. Pero si se trata de un comando que debe correr una vez se cree la vista del usuario, debería estar en la parte final del `<body>`. Si es un comando que afecta el estilo de la página debería estar dentro de `<head>`
- Colocar los scripts al final del `<body>` es una práctica común porque garantiza que los elementos HTML se carguen antes de que se ejecute el script. Esto es especialmente importante para scripts que interactúan con el contenido HTML o que dependen de elementos de la página. Ayuda a evitar problemas con el acceso a elementos que aún no se han creado cuando se ejecuta el script.
- Scripts en el `<head>`: Puedes incluir scripts en la sección `<head>`, pero ten en cuenta que se ejecutarán antes de que se cargue el contenido del `<body>`. Esto significa que si tu script intenta interactuar con elementos del `<body>`, es posible que no los encuentre, lo que podría llevar a errores. Los scripts en el `<head>` se utilizan principalmente para cargas de recursos, definiciones globales o cualquier cosa que no requiera acceso inmediato al contenido de la página. Si un script afecta el estilo de la página (por ejemplo, al cambiar el CSS), generalmente se coloca en el `<head>`. Esto se debe a que esas modificaciones deberían aplicarse antes de que se muestre el contenido en el navegador. Sin embargo, ten en cuenta que, como se mencionó anteriormente, debes considerar la secuencia de carga y asegurarte de que el script no intente modificar elementos que aún no se han creado.

## JavaScript reconoce el DOM del HTML por defecto

- Cuando trabajamos manipulando una hoja de Google Sheet vimos que era necesario crear un objeto que llamamos libro y otro que llamamos hoja. Pues bien, cuando se usa JavaScript embebido en un documento HTML también es necesario ver el HTML como un objeto, solo que no es necesario crearlo por comando sino que se crea por defecto y se llama "document".



Ejemplo 1. Embeber un código JavaScript para que corra en modo de lenguaje interpretado, después de cargar la página

Tenemos el siguiente documento HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>Ejemplo de Encabezado H1</title>
</head>
<body>
  <h1>Este es un encabezado de nivel 1</h1>
  <p>Este es un párrafo de ejemplo que sigue al encabezado.</p>
</body>
</html>
```

Lo que muestra el navegador

*Output*

# Este es un encabezado de nivel 1

Este es un párrafo de ejemplo que sigue al encabezado.

Ahora, queremos que JavaScript produzca un cambio en el elemento <h1> mediante un comando que supone que existe el objeto “document”, de manera que aprovecha funciones asociadas a ese objeto o clase

```
document.querySelector('h1').textContent = "Efecto de mi JavaScript por consola";
```

```
<!DOCTYPE html>
<html>
<head>
  <title>Ejemplo de Encabezado H1</title>
</head>
<body>
  <h1>Este es un encabezado de nivel 1</h1>
  <p>Este es un párrafo de ejemplo que sigue al encabezado.</p>
<script>
  document.querySelector('h1').textContent = "Atención: JavaScript ha cambiado este escrito";
</script>
</body>
</html>
```

Se produce el siguiente efecto en el navegador

<div>Output</div> <div> <h2>Atención: JavaScript ha cambiado este escrito</h2> <p>Este es un párrafo de ejemplo que sigue al encabezado.</p> </div>
Análisis
<ul style="list-style-type: none"> <li>• El navegador carga primero que todos la página correspondiente. Porque es lo primero que encuentra</li> <li>• Ejecuta finalmente el código de JavaScript porque es lo último que encuentra.</li> </ul>

<p>Ejemplo 2. El código JavaScript es un conjunto de funciones, de manera que algo las tiene que consumir o activar para aprovechar ese código. En este ejemplo, cuando el usuario oprime un botón es cuando se consumirá la función en el <code>&lt;script&gt;</code></p>
<pre> &lt;head&gt;   &lt;title&gt;Ejemplo de Encabezado H1&lt;/title&gt; &lt;/head&gt; &lt;body&gt;   &lt;h1&gt;Este es un encabezado de nivel 1&lt;/h1&gt;   &lt;p&gt;Este es un párrafo de ejemplo que sigue al encabezado.&lt;/p&gt;    &lt;button onclick="cambiarTexto()"&gt;Cambiar Texto&lt;/button&gt;    &lt;script&gt;     function cambiarTexto() {       document.querySelector("h1").textContent = "Atención: JavaScript ha cambiado este escrito";     }   &lt;/script&gt; &lt;/body&gt; &lt;/html&gt; </pre>
Lo que muestra el navegador inicialmente
<div>Output</div> <div> <h2>Este es un encabezado de nivel 1</h2> <p>Este es un párrafo de ejemplo que sigue al encabezado.</p> <p>Cambiar Texto</p> </div>
Lo que muestra el navegador al oprimir el botón

<div>Output</div> <div> <h2>Atención: JavaScript ha cambiado este escrito</h2> <p>Este es un párrafo de ejemplo que sigue al encabezado.</p> <div>Cambiar Texto</div> </div>
Análisis
<ul style="list-style-type: none"> <li>El código JavaScript consiste en una función <code>cambiarTexto()</code></li> <li>Fue necesario crear un elemento <code>&lt;button&gt;</code> y configurarlo para que en caso de ser oprimido produzca el cambio deseado.</li> </ul>

<p>Ejemplo 3. Haremos lo mismo del ejemplo 2, pero usando marcas, de manera que el código JavaScript afectará el elemento que tenga la marca correspondiente</p>
<pre> &lt;!DOCTYPE html&gt; &lt;html&gt; &lt;head&gt;   &lt;title&gt;Ejemplo de Encabezado H1&lt;/title&gt; &lt;/head&gt; &lt;body&gt;   &lt;h1 id="marca1"&gt;Este es un encabezado de nivel 1&lt;/h1&gt;   &lt;p&gt;Este es un párrafo de ejemplo que sigue al encabezado.&lt;/p&gt;    &lt;button onclick="cambiarTexto()"&gt;Cambiar Texto&lt;/button&gt;    &lt;script&gt;     function cambiarTexto() {       var elementoAfectado = document.getElementById("marca1");       elementoAfectado.innerHTML = "Atención: JavaScript ha cambiado este escrito";     }   &lt;/script&gt; &lt;/body&gt; &lt;/html&gt; </pre>

## Parte 1. El reto general

Debemos crear un sitio Web elegante en el que se muestren los datos IoT que se encuentran en una Google Sheet (la Google Sheet usada en el último taller). Pero vamos a imaginar que el desarrollador apenas si está aprendiendo HTML y por ello necesita comenzar realizar pequeños sprints para ir conquistando competencias que lo llevarán al producto final. Esos sprints comprenden:

- Crear un documento HTML muy básico, pero suficiente como para comprobar cosas como:
  - Conocer los mínimos que debe contener un documento HTML para que pueda ser publicado como una página Web

- Conocer cómo usar un simulador para comprobar que el documento HTML va a lograr el efecto deseado.
- Lograr publicar el código como una página web y ver el resultado.
- Usar el documento HTML como parte de otra página web ya existente para lograr una solución elegante de manera asombrosamente rápida.
- Conocer cómo usar un editor de Front-End para producir código de manera más profesional y publicarlo

## Tarea 1. Con ayuda de Chat GPT genera el HTML para un documento HTML simple. Uso de un simulador de navegador y tablas HTML.

Para comenzar a relacionar esto con una solución IoT, este es el preámbulo para un ejemplo de servicio requerido por la oficina de despachos de productos para atender solicitudes que provienen de neveras remotas que solicitan surtido de productos. Suscribirse por MQTT o usar una Google Sheet no resulta una solución elegante para la oficina de despachos.

Paso 1. Crear un documento HTML que represente un avance para la oficina de despachos, pero que por ahora incluye solo elementos mínimos de un documento HTML con el fin de comprender/repasar cuales son esos elementos.

Ejemplo de la orden para Chat GPT:

Dame el código HTML para un documento HTML de acuerdo a los siguientes requerimientos:

Título: "Servicios a la oficina de reparto"

Lo que los clientes verán en el navegador es: Órdenes recibidas para el despacho

Escribe aquí el código HTML obtenido. Nota: Si al pegar el código no aparece con estilo elegante usa un embellecedor de código como por ejemplo: <http://hilite.me/>

```
<!DOCTYPE html>
<html>
<head>
  <title>Servicios a la oficina de reparto</title>
</head>
<body>
  <h1>Órdenes recibidas para el despacho</h1>
</body>
</html>
```

Usando el embellecedor el código queda de la siguiente manera:

```
<!DOCTYPE html>
<html>
```

```

<head>
  <title>Servicios a la oficina de reparto</title>
</head>
<body>
  <h1>Órdenes recibidas para el despacho</h1>

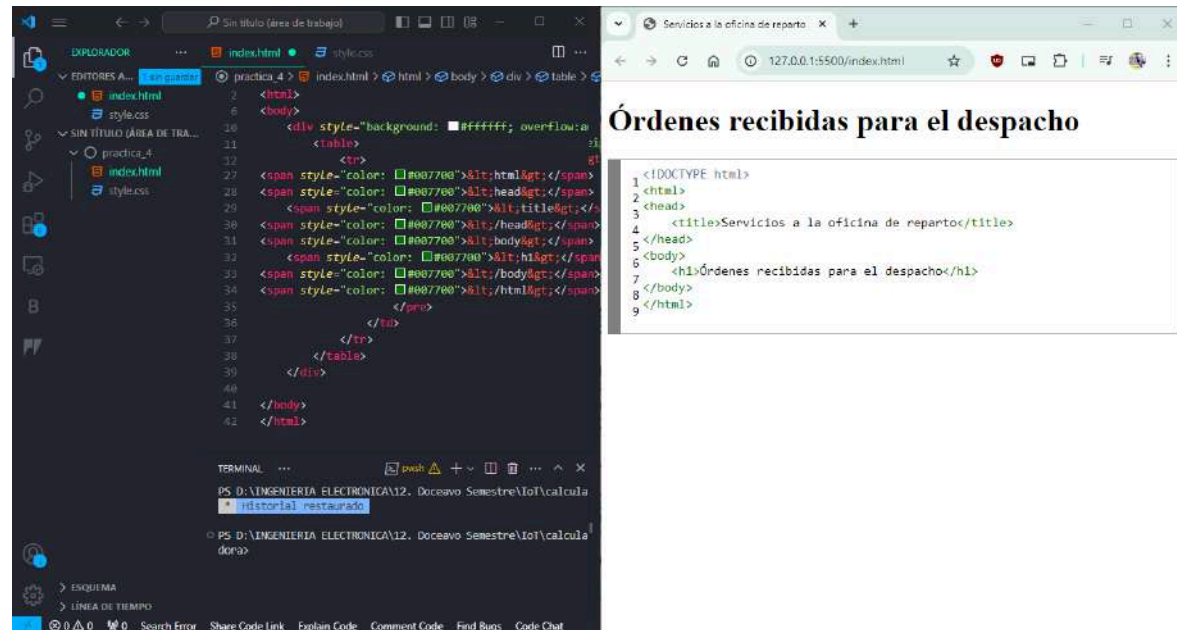
  <!-- HTML generated using hilite.me -->
      <div style="background: #ffffff;
overflow:auto;width:auto;border:solid gray;border-width:.1em .1em
.1em .8em;padding:.2em .6em;">
    <table>
      <tr>
        <td>
          <pre style="margin: 0; line-height: 125%">1
2
3
4
5
6
7
8
9</pre>
          </td>
          <td>
            <pre style="margin: 0; line-height: 125%">
<span style="color: #557799">&lt;!DOCTYPE html&gt;</span>
<span style="color: #007700">&lt;html&gt;</span>
<span style="color: #007700">&lt;head&gt;</span>
  <span style="color: #007700">&lt;title&gt;</span>Servicios a la
oficina de reparto<span style="color: #007700">&lt;/title&gt;</span>
<span style="color: #007700">&lt;/head&gt;</span>
<span style="color: #007700">&lt;body&gt;</span>
  <span style="color: #007700">&lt;h1&gt;</span>Órdenes recibidas
para el despacho<span style="color: #007700">&lt;/h1&gt;</span>
<span style="color: #007700">&lt;/body&gt;</span>
<span style="color: #007700">&lt;/html&gt;</span>
            </pre>
          </td>
        </tr>
      </table>
    </div>

```

```
</body>
</html>
```

Usa un simulador del navegador (HTML viewer) para comprobar el resultado del HTML anterior. Por ejemplo se puede usar: [Code Viewer\(https://codebeautify.org/htmlviewer\)](https://codebeautify.org/htmlviewer). También puedes usar un editor en línea de Front-End como:

Use Visual Studio Code junto con una extensión llamada Live Server para poder observar los cambios del código en tiempo real a medida que se va guardando el código en VS Code:



¿Qué has aprendido en este paso? ¿Qué es lo mínimo que debe tener un documento HTML?

Se ha aprendido la estructura básica de un código HTML, además de cómo ubicar los elementos y cómo es la estructura mínima necesaria para que un archivo HTML sea funcional y válido.

Un documento HTML válido y funcional debe tener los siguientes elementos:

1. **Declaración del tipo de documento (<!DOCTYPE html>):** Indica al navegador que el documento está escrito en HTML5.
2. **Etiqueta <html>:** Es el contenedor raíz de todo el contenido de la página. Todo el contenido HTML debe estar dentro de esta etiqueta.
3. **Etiqueta <head>:** Contiene metadatos sobre el documento, como el título, enlaces a archivos CSS, scripts de JavaScript, metaetiquetas y más.
4. **Etiqueta <body>:** Contiene el contenido visible de la página, como texto, imágenes, enlaces, etc.
5. **Etiqueta <title>:** Especifica el título de la página que aparece en la pestaña del navegador. Se ubica dentro de la etiqueta <head>.

```
<!DOCTYPE html>
```

```
<html>
<head>
  <title>Título de la página</title>
</head>
<body>
  <p>Contenido visible de la página</p>
</body>
</html>
```

Paso 2. Usa Chat GPT para conocer el elemento que debes agregarle al <body> de tu HTML para que aparezca una imagen elegante en la parte superior de la página

Busca la URL de una imagen en internet. Por ejemplo en el buscador de google busca imágenes de neveras inteligentes. Copia la URL (la dirección del vínculo) y escríbela aquí:

<https://img.global.news.samsung.com/cl/wp-content/uploads/2020/05/FAMILYHUB.jpg>

Ejemplo de pregunta para Chat GPT:

Quiero conocer el elemento que puedo agregar al <body> de tu HTML para que aparezca en la parte superior de la página una imagen. No quiero un ejemplo de todo el código HTML, solo el código del elemento cuando la imagen tiene la URL siguiente: [escribe aquí la URL].

```

```

Código completo para el documento HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>Servicios a la oficina de reparto</title>
</head>
<body>
  <h1>Órdenes recibidas para el despacho</h1>

  <!-- HTML generated using hilite.me -->
  <div style="background: #ffffff;
overflow:auto;width:auto;border:solid gray;border-width:.1em .1em
```

```

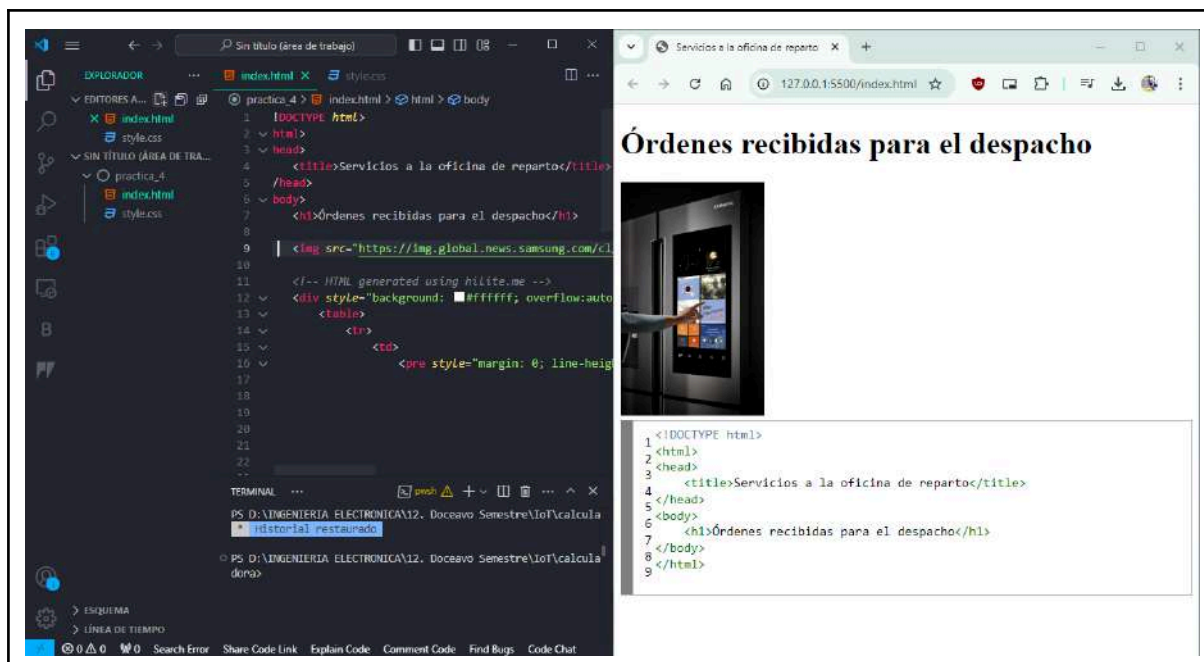
.1em .8em;padding:.2em .6em;">
    <table>
        <tr>
            <td>
                <pre style="margin: 0; line-height: 125%">1
2
3
4
5
6
7
8
9</pre>
                </td>
            <td>
                <pre style="margin: 0; line-height: 125%">
<span style="color: #557799">&lt;!DOCTYPE html&gt;</span>
<span style="color: #007700">&lt;html&gt;</span>
<span style="color: #007700">&lt;head&gt;</span>
    <span style="color: #007700">&lt;title&gt;</span>Servicios a la
oficina de reparto<span style="color: #007700">&lt;/title&gt;</span>
<span style="color: #007700">&lt;/head&gt;</span>
<span style="color: #007700">&lt;body&gt;</span>
    <span style="color: #007700">&lt;h1&gt;</span>Órdenes recibidas
para el despacho<span style="color: #007700">&lt;/h1&gt;</span>
<span style="color: #007700">&lt;/body&gt;</span>
<span style="color: #007700">&lt;/html&gt;</span>
                </pre>
            </td>
        </tr>
    </table>
</div>

</body>
</html>

```

Resultado que entrega el simulador de navegador





Análisis. ¿Qué has aprendido en este paso? ¿Cómo es el elemento HTML para una imagen? ¿dónde debe ir ese elemento?

Se ha aprendido cómo insertar una imagen usando la etiqueta `<img>`, también cómo controlar el tamaño de la imagen usando CSS en la línea 'style', cómo ajustar el ancho de la imagen, finalmente el uso del atributo 'alt' para proporcionar un texto alternativo en el caso de que la imagen no se cargue.

El elemento para una imagen es la etiqueta `<img>`, esta etiqueta no tiene cierre y usa atributos como 'src' para especificar la URL y 'alt' para describir la imagen en caso de que no se muestre.

```



```

El elemento `<img>` debe ir dentro de la etiqueta `<body>` del código HTML, si se quiere que la imagen aparezca en la parte superior de la página, se debe colocar la etiqueta `<img>` al inicio de la etiqueta `<body>`

Paso 3. Con ayuda de Chat GPT agrega código CSS al documento HTML del paso anterior

Ejemplo 1:

Tengo un HTML y quiero que le agregues el CSS necesario para que un escrito aparezca en color rojo, dentro de un marco de 500 píxeles de ancho. El siguiente es el HTML que tengo: [pega aquí tu código HTML]

Ejemplo 2 de una solicitud a Chat GPT:

Necesito que me ayudes agregando CSS para que la página se vea mucho mejor, más elegante y sea más atractiva a la vista del cliente, me gustaría que la página se viera similar a la de la web de Samsung de forma elegante y profesional, agregando lo necesario para obtener la mejor visualización posible.

Escribe aquí el código obtenido

```
<!DOCTYPE html>
<html>
<head>
  <title>Servicios a la oficina de reparto</title>
  <style>
    .texto-en-marco {
      color: red;
      width: 500px;
      border: 2px solid black;
      padding: 10px;
      margin: 20px auto; /* Centrará el marco horizontalmente
*/
    }
  </style>
</head>
<body>
  <h1>Órdenes recibidas para el despacho</h1>

  <div class="texto-en-marco">
    Este es un texto en color rojo dentro de un marco de 500
píxeles de ancho.
  </div>

  <!-- HTML generated using hilite.me -->
  <div style="background: #ffffff;
overflow:auto;width:auto;border:solid gray;border-width:.1em .1em
.1em .8em;padding:.2em .6em;">
    <table>
      <tr>
        <td>
          <pre style="margin: 0; line-height: 125%">1
```

2
3
4
5
6

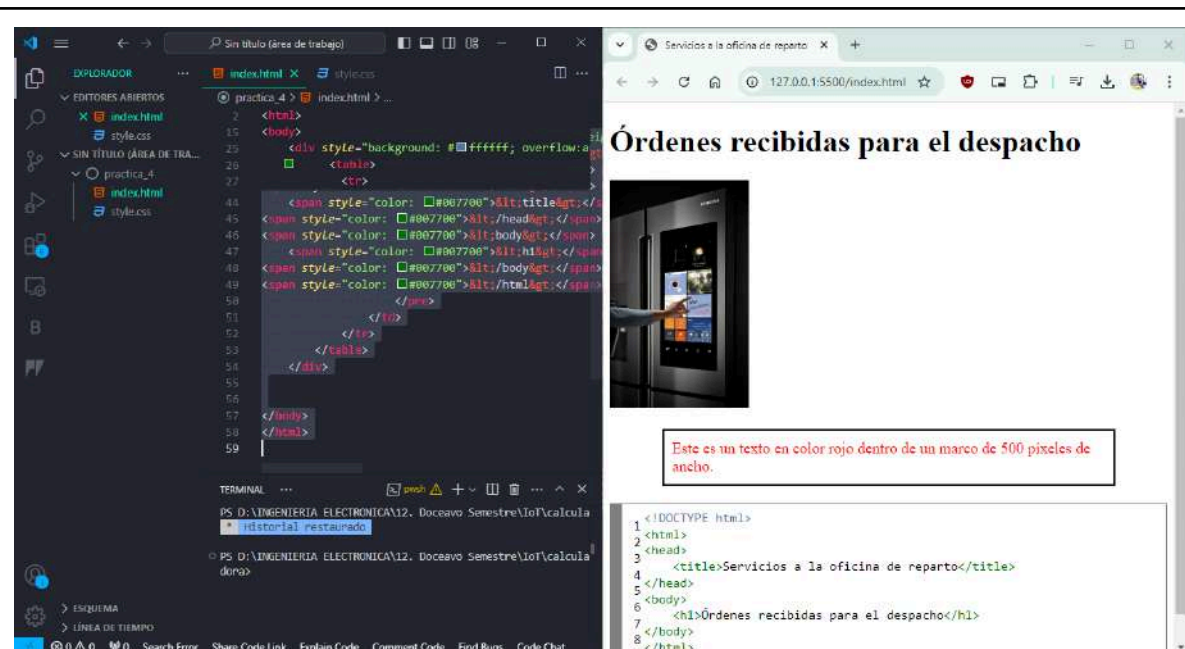
```

7
8
9</pre>
        </td>
        <td>
            <pre style="margin: 0; line-height: 125%">
<span style="color: #557799">&lt;!DOCTYPE html&gt;</span>
<span style="color: #007700">&lt;html&gt;</span>
<span style="color: #007700">&lt;head&gt;</span>
    <span style="color: #007700">&lt;title&gt;</span>Servicios a la
oficina de reparto<span style="color: #007700">&lt;/title&gt;</span>
<span style="color: #007700">&lt;/head&gt;</span>
<span style="color: #007700">&lt;body&gt;</span>
    <span style="color: #007700">&lt;h1&gt;</span>Órdenes recibidas
para el despacho<span style="color: #007700">&lt;/h1&gt;</span>
<span style="color: #007700">&lt;/body&gt;</span>
<span style="color: #007700">&lt;/html&gt;</span>
            </pre>
        </td>
    </tr>
</table>
</div>

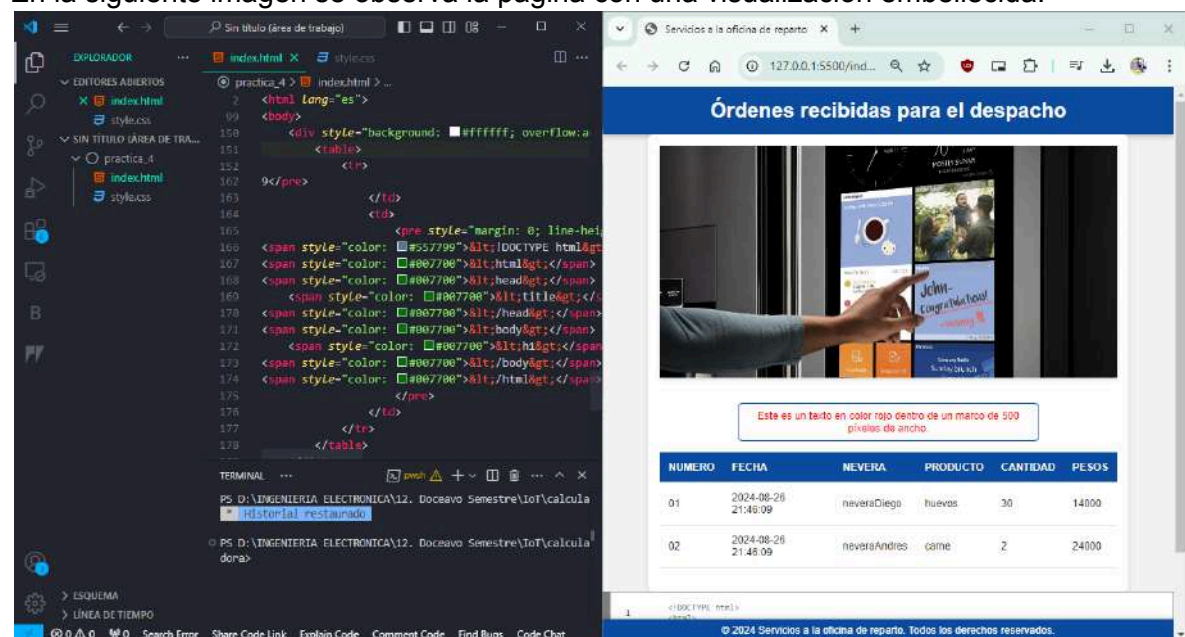
</body>
</html>

```

Agrega aquí captura de pantalla del simulador de navegador



En la siguiente imagen se observa la página con una visualización embellecida:



¿Qué has aprendido en este paso? ¿para qué sirve el CSS? ¿En qué parte de un documento HTML se debe ubicar el CSS?

CSS se usa para controlar la apariencia y el diseño de una página web, donde se puede definir un estilo visual de los elementos del HTML, como lo son colores, fuentes, márgenes, espaciados y la disposición de los diferentes elementos dentro de la página web, haciendo que el usuario tenga una experiencia interactiva con la página web.

El CSS se puede ubicar en tres lugares diferentes dentro de un documento HTML:

- **En el <head> del documento HTML:** Esto se hace utilizando la etiqueta <style>. El CSS incluido en esta sección se aplica a toda la página.

```
<head>
  <style>
    body {
      font-family: Arial, sans-serif;
    }
    /* Más reglas CSS aquí */
  </style>
</head>
```

- **En un archivo CSS externo:** Se enlaza al documento HTML mediante la etiqueta `<link>` en el `<head>`. Este método es útil para mantener el código CSS separado del HTML y facilita su mantenimiento y reutilización en múltiples páginas.

```
<head>
  <link rel="stylesheet" href="styles.css">
</head>
```

- **En línea dentro de un atributo style en una etiqueta HTML:** Esto aplica el estilo solo a ese elemento específico. Aunque es menos recomendable para grandes cantidades de estilo debido a la falta de centralización y mantenimiento.

```
<p style="color: red; font-size: 20px;">Este es un texto en rojo.</p>
```

Paso 4. Agrega al HTML anterior una tabla de las solicitudes que provienen de la nevera

Ejemplo de la solicitud que puedes hacerle a Chat GPT:

Quiero que ahora el cliente pueda ver debajo del primer aviso, una tabla que tiene estas características: hay 6 columnas y 3 filas; la primera fila contiene los encabezados: número, fecha, nevera, producto, cantidad, pesos; la 2da fila tendrá los valores: 1, 2023-09-27 6:16:38, neveraChachones, huevos, 10, 30000; la fila 1 estará resaltada en color naranja

Escribe aquí el código obtenido

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Servicios a la oficina de reparto</title>
  <style>
    /* Estilos generales */
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 0;
      color: #333;
    }

    header {
      background-color: #0c4da2;
      color: white;
```

```

        padding: 20px 0;
        text-align: center;
        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);
    }

    h1 {
        margin: 0;
        font-size: 2rem;
    }

    /* Imagen principal */
    .main-image {
        width: 100%;
        max-height: 400px;
        object-fit: cover;
        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
        margin-bottom: 20px;
    }

    /* Contenedor general */
    .container {
        width: 80%;
        max-width: 1200px;
        margin: 0 auto;
        background-color: white;
        padding: 20px;
        box-shadow: 0px 4px 12px rgba(0, 0, 0, 0.1);
        border-radius: 8px;
    }

    /* Tabla de datos */
    table {
        width: 100%;
        border-collapse: collapse;
        margin-bottom: 20px;
    }

    th, td {
        padding: 15px;
        text-align: left;
        border-bottom: 1px solid #ddd;
    }

    th {
        background-color: orange;
        color: white;
        text-transform: uppercase;
    }

    tr:hover {
        background-color: #f1f1f1;
    }

    /* Texto dentro de un marco */
    .texto-en-marco {
        color: red;
        width: 500px;

```

```

        border: 2px solid #0c4da2;
        padding: 10px;
        margin: 20px auto;
        text-align: center;
        border-radius: 8px;
        background-color: #fefefe;
    }

    /* Footer */
    footer {
        background-color: #0c4da2;
        color: white;
        text-align: center;
        padding: 10px 0;
        position: fixed;
        bottom: 0;
        width: 100%;
        box-shadow: 0px -2px 8px rgba(0, 0, 0, 0.2);
    }

</style>
</head>
<body>

    <!-- Header -->
    <header>
        <h1>Órdenes recibidas para el despacho</h1>
    </header>

    <!-- Contenedor principal -->
    <div class="container">
        

        <!-- Tabla de datos -->
        <table>
            <tr>
                <th>Numero</th>
                <th>Fecha</th>
                <th>Nevera</th>
                <th>Producto</th>
                <th>Cantidad</th>
                <th>Pesos</th>
            </tr>
            <tr>
                <td>01</td>
                <td>2023-09-27 06:16:38</td>
                <td>neveraChachones</td>
                <td>huevos</td>
                <td>10</td>
                <td>30000</td>
            </tr>
            <tr>
                <td>02</td>
                <td>2024-08-28 22:34:20</td>

```

```

                <td>neveraDiego</td>
                <td>carne</td>
                <td>2</td>
                <td>24000</td>
            </tr>
        </table>
    </div>

    <div class="texto-en-marco">
        Este es un texto en color rojo dentro de un marco de 500
        píxeles de ancho.
    </div>

    <!-- Footer -->
    <footer>
        &copy; 2024 Servicios a la oficina de reparto. Todos los
        derechos reservados.
    </footer>

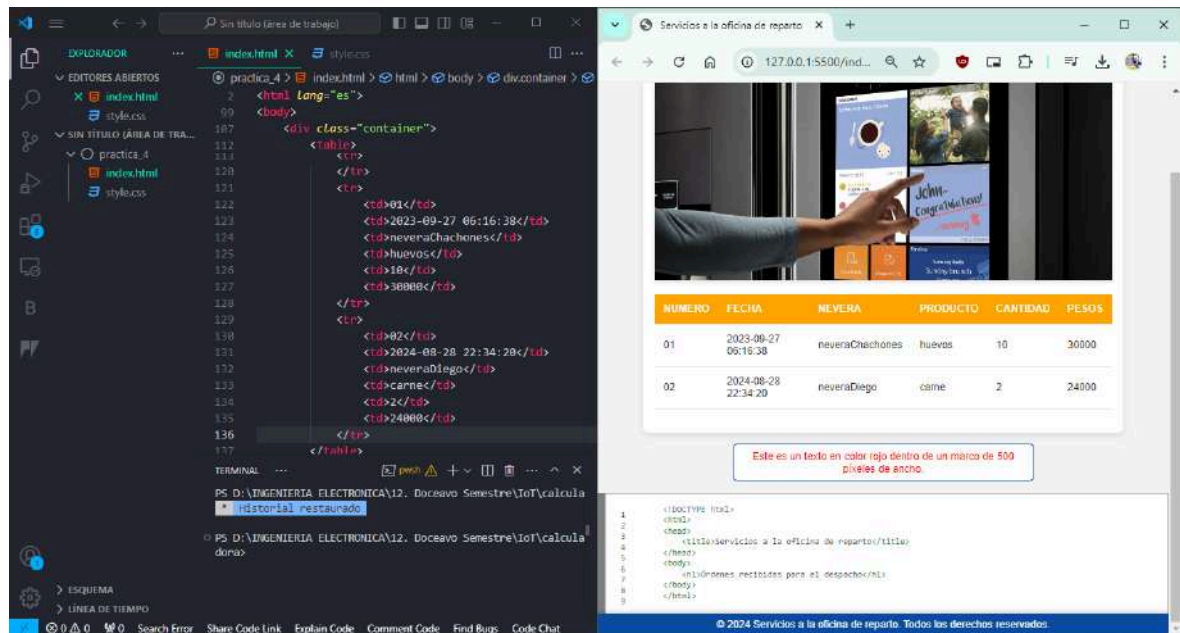
    <!-- Código generado -->
    <div style="background: #ffffff;
overflow:auto;width:auto;border:solid gray;border-width:.1em .1em
.1em .8em;padding:.2em .6em; margin-top: 20px;">
        <table>
            <tr>
                <td>
                    <pre style="margin: 0; line-height: 125%">1
2
3
4
5
6
7
8
9</pre>
                </td>
                <td>
                    <pre style="margin: 0; line-height: 125%">
<span style="color: #557799">&lt;!DOCTYPE html&gt;</span>
<span style="color: #007700">&lt;html&gt;</span>
<span style="color: #007700">&lt;head&gt;</span>
    <span style="color: #007700">&lt;title&gt;</span>Servicios a la
oficina de reparto<span style="color: #007700">&lt;/title&gt;</span>
<span style="color: #007700">&lt;/head&gt;</span>
<span style="color: #007700">&lt;body&gt;</span>
    <span style="color: #007700">&lt;h1&gt;</span>Órdenes recibidas
para el despacho<span style="color: #007700">&lt;/h1&gt;</span>
<span style="color: #007700">&lt;/body&gt;</span>
<span style="color: #007700">&lt;/html&gt;</span>
                    </pre>
                </td>
            </tr>
        </table>
    </div>

</body>
</html>

```



Agrega aquí captura de pantalla del simulador de navegador



Pídele a Chat GPT que mejore el estilo de la tabla usando un servicio externo de estilos como por ejemplo Bootstrap. Ejemplo de la solicitud a Chat GPT:

Tengo un HTML. Deseo que modifiques el CSS para que la tabla sea más elegante usando un servicio de estilos como bootstrap. Mi código HTML es el siguiente: [escribe tu código HTML aquí]

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Servicios a la oficina de reparto</title>
  <!-- Incluyendo Bootstrap CSS -->
  <link
href="https://cdn.jsdelivrivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap
.min.css" rel="stylesheet">

  <style>
    /* Estilos generales */
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 0;
      color: #333;
    }

    header {
      background-color: #0c4da2;
      color: white;
      padding: 20px 0;
      text-align: center;
```

```

        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);
    }

    h1 {
        margin: 0;
        font-size: 2rem;
    }

    /* Imagen principal */
    .main-image {
        width: 100%;
        max-height: 400px;
        object-fit: cover;
        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
        margin-bottom: 20px;
    }

    /* Contenedor general */
    .container {
        width: 80%;
        max-width: 1200px;
        margin: 0 auto;
        background-color: white;
        padding: 20px;
        box-shadow: 0px 4px 12px rgba(0, 0, 0, 0.1);
        border-radius: 8px;
    }

    /* Texto dentro de un marco */
    .texto-en-marco {
        color: red;
        width: 500px;
        border: 2px solid #0c4da2;
        padding: 10px;
        margin: 20px auto;
        text-align: center;
        border-radius: 8px;
        background-color: #fefefe;
    }

    /* Footer */
    footer {
        background-color: #0c4da2;
        color: white;
        text-align: center;
        padding: 10px 0;
        position: fixed;
        bottom: 0;
        width: 100%;
        box-shadow: 0px -2px 8px rgba(0, 0, 0, 0.2);
    }

</style>
</head>
<body>

<!-- Header -->

```

```

<header>
  <h1>Órdenes recibidas para el despacho</h1>
</header>

<!-- Contenedor principal -->
<div class="container">
  

  <!-- Tabla de datos -->
  <table class="table table-striped table-hover">
    <thead class="table-dark">
      <tr>
        <th>Numero</th>
        <th>Fecha</th>
        <th>Nevera</th>
        <th>Producto</th>
        <th>Cantidad</th>
        <th>Pesos</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>01</td>
        <td>2023-09-27 06:16:38</td>
        <td>neveraChachones</td>
        <td>huevos</td>
        <td>10</td>
        <td>30000</td>
      </tr>
      <tr>
        <td>02</td>
        <td>2024-08-28 22:34:20</td>
        <td>neveraDiego</td>
        <td>carne</td>
        <td>2</td>
        <td>24000</td>
      </tr>
    </tbody>
  </table>
</div>

<div class="texto-en-marco">
  Este es un texto en color rojo dentro de un marco de 500
píxeles de ancho.
</div>

<!-- Footer -->
<footer>
  &copy; 2024 Servicios a la oficina de reparto. Todos los
derechos reservados.
</footer>

<!-- Código generado -->
<div style="background: #ffffff;
overflow:auto;width:auto;border:solid gray;border-width:.1em .1em

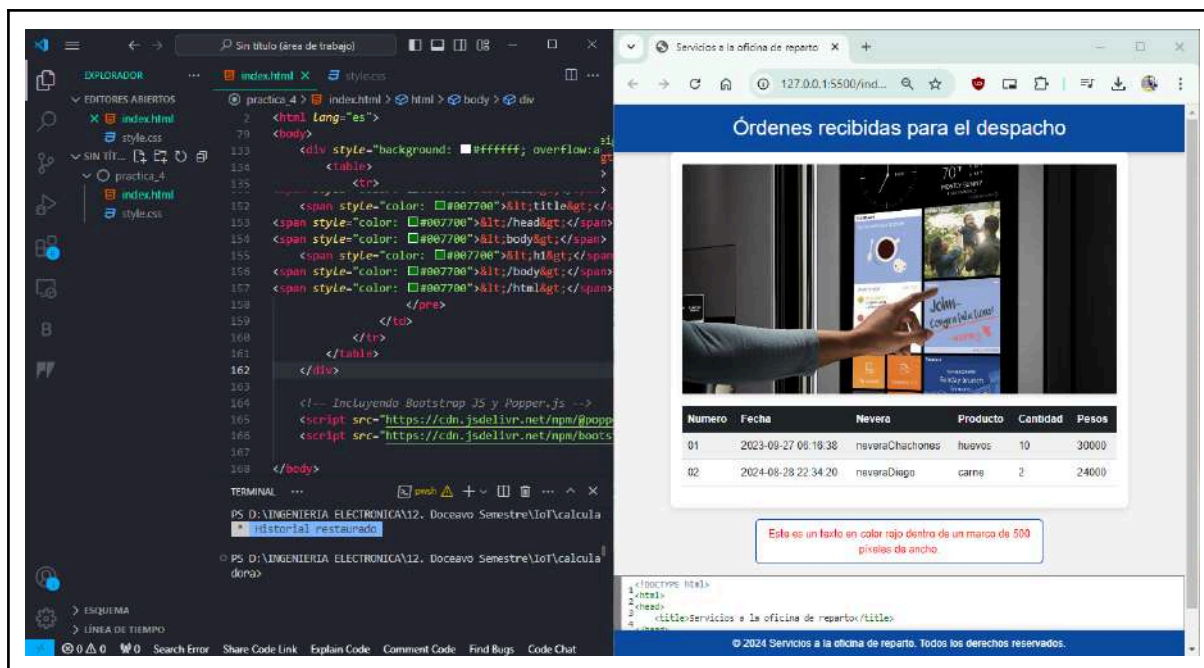
```

```

1 .1em .8em;padding:.2em .6em; margin-top: 20px;">
2   <table>
3     <tr>
4       <td>
5         <pre style="margin: 0; line-height: 125%">1
6
7
8
9</pre>
10
11       </td>
12       <td>
13         <pre style="margin: 0; line-height: 125%">
14 <span style="color: #557799">&lt;!DOCTYPE html&gt;</span>
15 <span style="color: #007700">&lt;html&gt;</span>
16 <span style="color: #007700">&lt;head&gt;</span>
17   <span style="color: #007700">&lt;title&gt;</span>Servicios a la
18 oficina de reparto<span style="color: #007700">&lt;/title&gt;</span>
19 <span style="color: #007700">&lt;/head&gt;</span>
20 <span style="color: #007700">&lt;body&gt;</span>
21   <span style="color: #007700">&lt;h1&gt;</span>Órdenes recibidas
22 para el despacho<span style="color: #007700">&lt;/h1&gt;</span>
23 <span style="color: #007700">&lt;/body&gt;</span>
24 <span style="color: #007700">&lt;/html&gt;</span>
25   </pre>
26   </td>
27 </tr>
28 </table>
29 </div>
30
31 <!-- Incluyendo Bootstrap JS y Popper.js -->
32 <script
33 src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>
34 <script
35 src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.min.js"></script>
36
37 </body>
38 </html>

```

Resultado del simulador aquí.



¿Qué has aprendido en este paso? ¿Cuál es el elemento HTML para crear tablas en un documento HTML? ¿Por qué se requiere código a nivel del CSS?

Se ha aprendido a integrar un framework llamado Bootstrap en HTML para darle un mejor estilo visual a una tabla.

### Elemento HTML para crear tablas:

El elemento HTML básico para crear tablas es `<table>`. Dentro de este, se usan otros elementos:

- `<thead>`: Define la cabecera de la tabla.
- `<tbody>`: Contiene el cuerpo principal de la tabla.
- `<tr>`: Define una fila de la tabla.
- `<th>`: Define una celda de encabezado en una fila (dentro de `<thead>`).
- `<td>`: Define una celda de datos en una fila (dentro de `<tbody>`).

El código CSS se requiere para controlar el estilo visual y la presentación de elementos HTML. Aunque los elementos HTML definen la estructura y contenido de una página, el CSS define su apariencia.

### Paso 5. Uso de hipervínculos

La solución anterior tiene una falla: que no muestra en realidad los datos de Google Sheet que es parte de la solución IoT. Nota: debes usar la Google Sheet del taller anterior. Eso se podría resolver con trucos modernos y JavaScript.

Por ahora recurriremos a una solución poco elegante pero que soluciona el problema usaremos un hipervínculo que apunte a la Google Sheet.

Escribe aquí la URL de la Google Sheet donde son registrados los datos. Procura previamente configurar la Google Sheet con permisos para que cualquier persona la pueda ver. Ejemplo:

<https://docs.google.com/spreadsheets/d/1bBtfmCT5sGC3dHzFXXno6FM2zUipWa2yGj4tdVOfm22>

[https://docs.google.com/spreadsheets/d/1FmE40k\\_cKIGeJ\\_n3YZdiUZD6reeLojNGRtz9nCUIDpo/edit?usp=sharing](https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6reeLojNGRtz9nCUIDpo/edit?usp=sharing)

Hazle a Chat GPT una solicitud de apoyo para agregar el hipervínculo al código HTML obtenido en la tarea anterior.

Ejemplo para solicitar a Chat GPT la solución:

Tengo el HTML de una página y quiero agregar un hipervínculo. Los datos son los siguientes:

Enlace para el hipervínculo: [pega aquí la URL de tu google sheet],

Nombre del hipervínculo: Haz click aquí para ver tabla de datos

El HTML que tengo es el siguiente: [aquí pega el último HTML obtenido]

Respuesta de Chat GPT

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Servicios a la oficina de reparto</title>
  <!-- Incluyendo Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap
.min.css" rel="stylesheet">

  <style>
    /* Estilos generales */
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 0;
      color: #333;
    }

    header {
      background-color: #0c4da2;
      color: white;
      padding: 20px 0;
      text-align: center;
      box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);
    }

    h1 {
      margin: 0;
      font-size: 2rem;
    }

    /* Imagen principal */
    .main-image {
      width: 100%;
      max-height: 400px;
```

```

        object-fit: cover;
        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
        margin-bottom: 20px;
    }

    /* Contenedor general */
    .container {
        width: 80%;
        max-width: 1200px;
        margin: 0 auto;
        background-color: white;
        padding: 20px;
        box-shadow: 0px 4px 12px rgba(0, 0, 0, 0.1);
        border-radius: 8px;
    }

    /* Texto dentro de un marco */
    .texto-en-marco {
        color: red;
        width: 500px;
        border: 2px solid #0c4da2;
        padding: 10px;
        margin: 20px auto;
        text-align: center;
        border-radius: 8px;
        background-color: #fefefe;
    }

    /* Footer */
    footer {
        background-color: #0c4da2;
        color: white;
        text-align: center;
        padding: 10px 0;
        position: fixed;
        bottom: 0;
        width: 100%;
        box-shadow: 0px -2px 8px rgba(0, 0, 0, 0.2);
    }
</style>
</head>
<body>

    <!-- Header -->
    <header>
        <h1>Órdenes recibidas para el despacho</h1>
    </header>

    <!-- Contenedor principal -->
    <div class="container">
        

        <!-- Tabla de datos -->
        <table class="table table-striped table-hover">

```

```

        <thead class="table-dark">
            <tr>
                <th>Numero</th>
                <th>Fecha</th>
                <th>Nevera</th>
                <th>Producto</th>
                <th>Cantidad</th>
                <th>Pesos</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td>01</td>
                <td>2023-09-27 06:16:38</td>
                <td>neveraChachones</td>
                <td>huevos</td>
                <td>10</td>
                <td>30000</td>
            </tr>
            <tr>
                <td>02</td>
                <td>2024-08-28 22:34:20</td>
                <td>neveraDiego</td>
                <td>carne</td>
                <td>2</td>
                <td>24000</td>
            </tr>
        </tbody>
    </table>

    <!-- Hipervínculo -->
    <p class="mt-4">
        <a
href="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD
6reeLojNGRtz9nCUIDpo/edit?usp=sharing" class="btn btn-primary">
        Haz click aquí para ver tabla de datos
    </a>
    </p>
</div>

<div class="texto-en-marco">
    Este es un texto en color rojo dentro de un marco de 500
píxeles de ancho.
</div>

<!-- Footer -->
<footer>
    &copy; 2024 Servicios a la oficina de reparto. Todos los
derechos reservados.
</footer>

<!-- Incluyendo Bootstrap JS y Popper.js -->
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popp
er.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.m

```



```
in.js"></script>

</body>
</html>
```

Captura de pantalla del resultado que entrega el simulador

The screenshot shows a web application interface with a table titled "Órdenes recibidas para el despacho". The table has columns: Numero, Fecha, Nevera, Producto, Cantidad, and Precio. Below the table, there is a button "Haz click aquí para ver tabla de datos". To the right, a Google Sheet is open, showing the same data in a table format.

numero	fecha	nevera	producto	cantidad	precio
01	2023-09-27 06:15:38	neveraChachones	huevos	10	30000
02	2024-06-28 22:34:20	neveraZiajo	carne	2	24000

¿Qué has aprendido? ¿Cómo se llama el elemento HTML que permite crear hipervínculos? ¿Cómo se usa?

Se ha aprendido que para agregar un hipervínculo en un documento HTML se utiliza el elemento `<a>`, que significa "anchor" (ancla en inglés). Este elemento se usa para crear enlaces a otras páginas web, archivos, ubicaciones dentro del mismo documento, o direcciones de correo electrónico.

### ¿Cómo se usa el elemento `<a>`?

1. **Atributo href:** El atributo más importante del elemento `<a>` es `href`, que contiene el URL al cual el hipervínculo dirigirá cuando se haga clic.
2. **Texto del enlace:** El contenido entre las etiquetas de apertura `<a>` y de cierre `</a>` es el texto que los usuarios verán como el enlace en la página web.

## Tarea 2. Embeber un código html en otro HTML.

Paso 1. Embeber código de una página web externa. Uso de frames

El problema de la solución anterior es que los datos no aparecen embebidos en la página. Además, al hacer click en el hipervínculo se abre una nueva ventana del navegador lo cual hace que el cliente se pueda alejar de nuestra página.

Para solucionarlo, usaremos un truco que incluyen las hojas de Google Sheet - la de ser vista como una página web. Es decir, Google Sheet puede ser vista como una cierta página Web y esa página Web puede ser embebida en nuestra página como un frame.

Copia el código HTML resultante, el cual incluye la URL pero dentro de un frame

Convierte a la google sheet a una solución web.

El proceso es así:

abres tu google sheet > Menu > Archivo > Compartir > publicar en la Web > Insertar > Fíjate en la configuración para que esa publicación sea abierta al público.

Copia abajo el Código HTML obtenido para embeber la versión web de la Google Sheet

```
<iframe
src="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6
reeLojNGRtz9nCUIDpo/pubhtml?gid=1200278807&single=true"></iframe>
```

Usualmente el código que ofrece la solución anterior es un recuadro (frame) que no es suficientemente ancho para ver toda la tabla. Pídele a Chat GPT que lo modifique para que el frame sea 2,5 veces más ancho. Escribe abajo el código final obtenido

Ejemplo de solicitud a Chat GPT:

Quiero que mejores este código del siguiente frame para que el cliente lo vea 2,5 veces más ancho [escribe aquí el código obtenido en el paso 1]

```
<iframe
src="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6
reeLojNGRtz9nCUIDpo/pubhtml?gid=1200278807&single=true" width="100%"
height="600" style="border:none;"></iframe>
```

Pídele ayuda a Chat GPT para embeber ese código a tu HTML. Escribe abajo el código final obtenido.

Ejemplo de la solicitud a Chat GPT:

Tengo un HTML quiero embeberle en la parte final el frame obtenido en la pregunta anterior. Mi HTML es el siguiente: [escribe aquí el código de tu documento HTML]

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Servicios a la oficina de reparto</title>
  <!-- Incluyendo Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap
.min.css" rel="stylesheet">

  <style>
    /* Estilos generales */
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 0;
      color: #333;
    }

    header {
      background-color: #0c4da2;
```

```

        color: white;
        padding: 20px 0;
        text-align: center;
        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);
    }

    h1 {
        margin: 0;
        font-size: 2rem;
    }

    /* Imagen principal */
    .main-image {
        width: 100%;
        max-height: 400px;
        object-fit: cover;
        box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
        margin-bottom: 20px;
    }

    /* Contenedor general */
    .container {
        width: 80%;
        max-width: 1200px;
        margin: 0 auto;
        background-color: white;
        padding: 20px;
        box-shadow: 0px 4px 12px rgba(0, 0, 0, 0.1);
        border-radius: 8px;
    }

    /* Texto dentro de un marco */
    .texto-en-marco {
        color: red;
        width: 500px;
        border: 2px solid #0c4da2;
        padding: 10px;
        margin: 20px auto;
        text-align: center;
        border-radius: 8px;
        background-color: #fefefe;
    }

    /* Footer */
    footer {
        background-color: #0c4da2;
        color: white;
        text-align: center;
        padding: 10px 0;
        position: fixed;
        bottom: 0;
        width: 100%;
        box-shadow: 0px -2px 8px rgba(0, 0, 0, 0.2);
    }

</style>
</head>

```

```

<body>

  <!-- Header -->
  <header>
    <h1>Órdenes recibidas para el despacho</h1>
  </header>

  <!-- Contenedor principal -->
  <div class="container">
    

    <!-- Tabla de datos -->
    <table class="table table-striped table-hover">
      <thead class="table-dark">
        <tr>
          <th>Numero</th>
          <th>Fecha</th>
          <th>Nevera</th>
          <th>Producto</th>
          <th>Cantidad</th>
          <th>Pesos</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <td>01</td>
          <td>2023-09-27 06:16:38</td>
          <td>neveraChachones</td>
          <td>huevos</td>
          <td>10</td>
          <td>30000</td>
        </tr>
        <tr>
          <td>02</td>
          <td>2024-08-28 22:34:20</td>
          <td>neveraDiego</td>
          <td>carne</td>
          <td>2</td>
          <td>24000</td>
        </tr>
      </tbody>
    </table>

    <!-- Agregando el iframe después de la tabla -->
    <iframe
src="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6reeLojNGRtz9nCUIDpo/pubhtml?gid=1200278807&single=true" width="100%"
height="600" style="border:none;"></iframe>

    <!-- Hipervínculo -->
    <p class="mt-4">
      <a
href="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6reeLojNGRtz9nCUIDpo/edit?usp=sharing" class="btn btn-primary">

```

```

        Haz click aquí para ver tabla de datos
    </a>
</p>
</div>

<div class="texto-en-marco">
    Este es un texto en color rojo dentro de un marco de 500
    píxeles de ancho.
</div>

<!-- Footer -->
<footer>
    &copy; 2024 Servicios a la oficina de reparto. Todos los
    derechos reservados.
</footer>

<!-- Incluyendo Bootstrap JS y Popper.js -->
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.min.js"></script>
</body>
</html>

```

Captura de pantalla del simulador de navegador.

The screenshot shows a web browser simulator displaying a web application. The application has a blue header with the text "Órdenes recibidas para el despacho". Below the header, there is a large image of a hand pointing at a screen displaying a list of orders. Below the image, there is a table with the following data:

Numero	Fecha	Nevera	Producto	Cantidad	Pesos
01	2023-05-27 06:18:38	neveraChachones	huevos	10	30000
02	2024-08-28 22:34:20	neveraDiego	carne	2	24000

Below the table, there is a small table with the following data:

numero	fecha	nevera	producto	cantidad	pesos
1	20/05/2024	neveraMafe	huevos	30	10000
2	28/09/2024	neveraDiego	carne	2	24000
3	28/09/2024	neveraAndres	pollo	3	18000
4	20/09/2024	neveraDiego	mora	3	8000
5	28/09/2024				

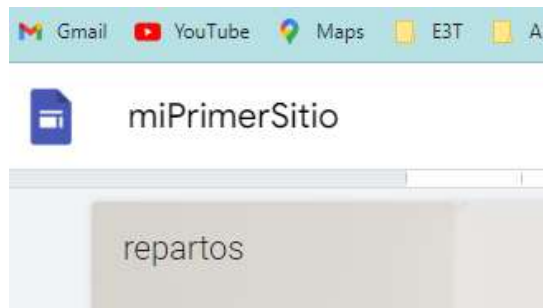
The footer of the application reads: "© 2024 Servicios a la oficina de reparto. Todos los derechos reservados."

In the background, a code editor is visible, showing the HTML code for the page. The code includes a link to a Google Spreadsheet and a link to a table of data.

## Tarea 3. Obtener una solución elegante y rápida usando una página web moderna.

Paso 5. Crea una página Web usando Google Sites y allí incrusta el código HTML desarrollado para comprobar que funciona como parte de otra página en línea:

Parte 1: Te logeas en gmail > <https://sites.google.com/> > Crear un sitio > seleccione una plantilla si puede, si no puede que sea una hoja en blanco > esquina superior izquierda> dele un nombre al archivo > mas abajo, en logotipo dele en nombre de la futura página, por ejemplo “repartos”, como se muestra en la siguiente figura



Parte 2: póngale un título que verá el cliente, por ejemplo “Departamento de repartos” >



Parte 3: inserte su código html así: Menú panel derecho > Insertar > “<>Insertar” > Insertar código > pegue allí su HTML > Siguiente > Insertar > borre todo aquello que ofrece la plantilla pero que tu no deseas para tu página

Parte 4. Note que lo que tienes hasta ahora es simplemente un archivo, no es aún una página web. Puedes ver el archivo en Google Drive > recientes > miPrimerSitio. Lo que falta ahora es generar una página web a partir de ese archivo. Este es el proceso:

Menú superior derecho > Publicar > Publicar. Ahora puedes ver el sitio web publicado así:

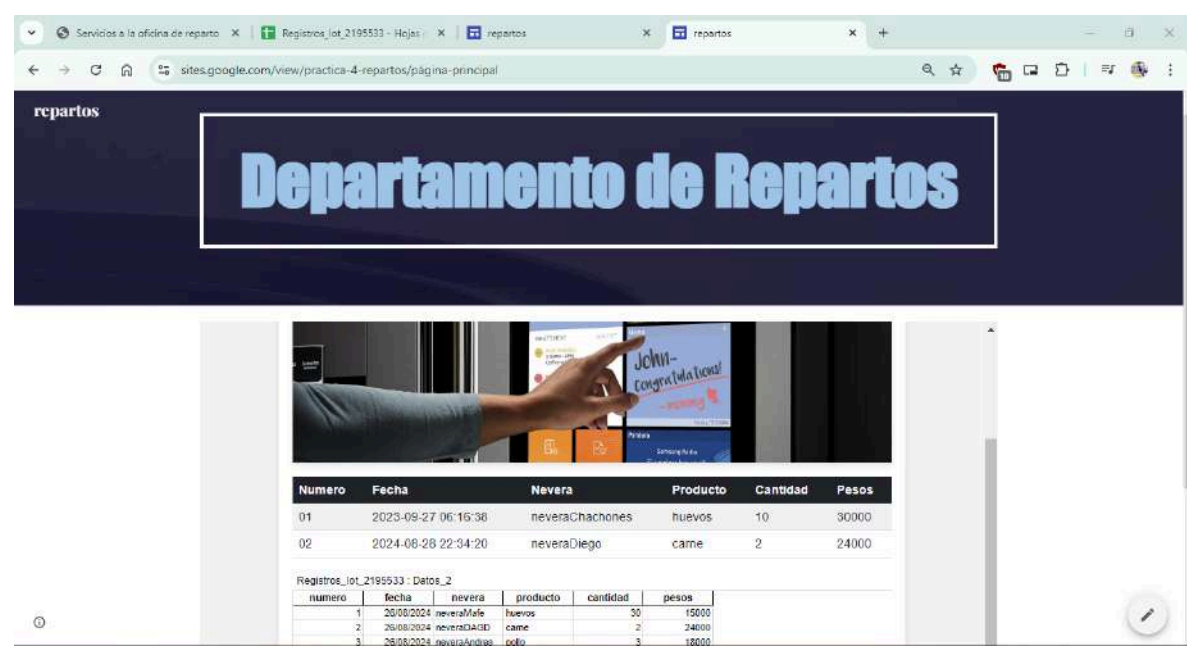


Ahora verás en la barra superior del navegador que tu sitio web tiene una URL

Escribe aquí la URL de tu sitio

<https://sites.google.com/view/practica-4-repartos/p%C3%A1gina-principal>

Captura la pantalla de tu sitio de manera que muestre el resultado visual de tu código HTML



## Paso 6. Análisis

Análisis. Analice el código HTML final obtenido. Explique de qué manera es posible introducir tablas usando HTML.

### Elemento HTML para crear tablas:

El elemento HTML básico para crear tablas es `<table>`. Dentro de este, se usan otros elementos:



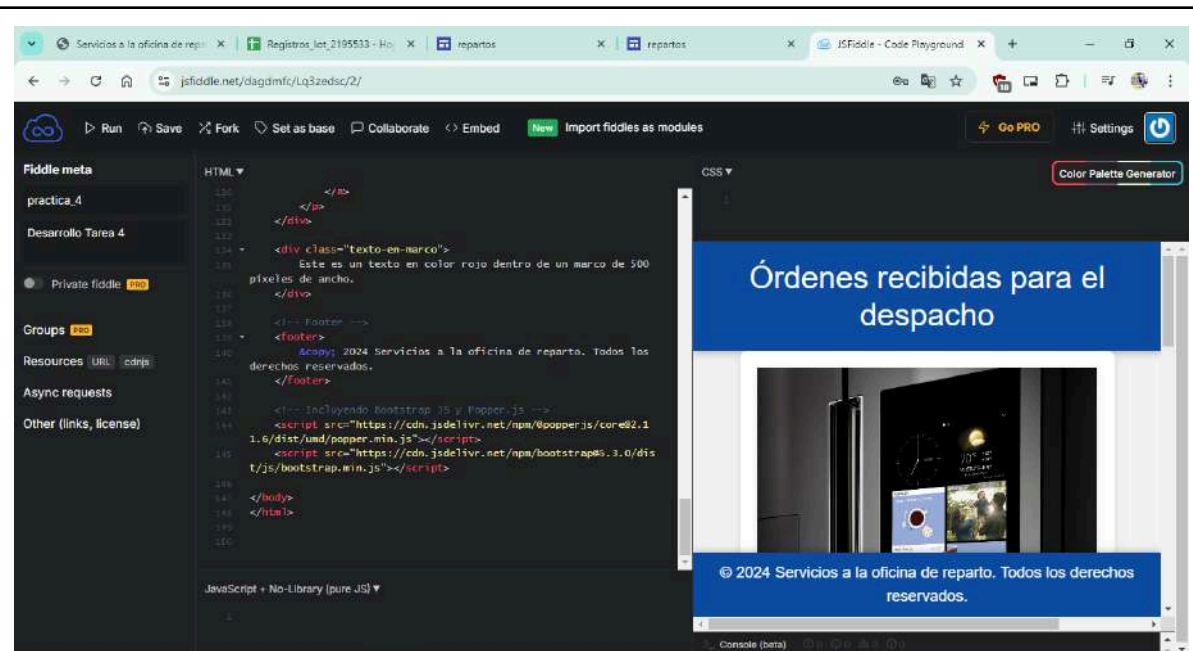
- **<thead>**: Define la cabecera de la tabla.
- **<tbody>**: Contiene el cuerpo principal de la tabla.
- **<tr>**: Define una fila de la tabla.
- **<th>**: Define una celda de encabezado en una fila (dentro de **<thead>**).
- **<td>**: Define una celda de datos en una fila (dentro de **<tbody>**).

## Tarea 4. Uso de un editor profesional de páginas (entorno web de desarrollo)


**Paso 1.** Suscribirse a jsfiddle - un editor en línea de páginas HTML: <https://jsfiddle.net/>. El objetivo es actuar de manera más profesional y conocer la estrategia que usan los editores para separar el HTML del CSS pero que a la postre tienen el mismo efecto de un documento HTML que contiene todo. Nota: podemos decir que jsfiddle es un editor de soluciones Front-End (un Entorno de Desarrollo Web) porque incluye JavaScript. Pero lo usaremos como editor de páginas HTML.

**Paso 2.** Uso de jsfiddle para comprobar que se logra lo mismo que con el HTML Viewer.

Pega el código obtenido en la tarea anterior en la ventana HTML de JSFiddle, córralo con “Run” y observa el resultado. Pega abajo el resultado.



Guarda el código anterior en to jsfiddle para tu colección.

Con  consulta como se guarda tu código (tus jsfiddles)



**Paso 3.** Uso más profesional de un editor de HTML. Separa el código CSS del documento HTML para usarlo de manera más natural en jsfiddle.

Escribe aquí el código HTML sin CSS

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Servicios a la oficina de reparto</title>
  <!-- Incluyendo Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap
.min.css" rel="stylesheet">
  <link rel="stylesheet" href="styles.css"> <!-- Archivo CSS
separado -->
</head>
<body>

  <!-- Header -->
  <header>
    <h1>Órdenes recibidas para el despacho</h1>
  </header>

  <!-- Contenedor principal -->
  <div class="container">
    

    <!-- Tabla de datos -->
    <table class="table table-striped table-hover">
      <thead class="table-dark">
        <tr>
          <th>Numero</th>
          <th>Fecha</th>
          <th>Nevera</th>
          <th>Producto</th>
          <th>Cantidad</th>
          <th>Pesos</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <td>01</td>
          <td>2023-09-27 06:16:38</td>
          <td>neveraChachones</td>
          <td>huevos</td>
          <td>10</td>
          <td>30000</td>
        </tr>
        <tr>
          <td>02</td>
          <td>2024-08-28 22:34:20</td>
```

```

        <td>neveraDiego</td>
        <td>carne</td>
        <td>2</td>
        <td>24000</td>
    </tr>
</tbody>
</table>

<!-- Agregando el iframe después de la tabla -->
<iframe
src="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6
reeLojNGRtz9nCUIDpo/pubhtml?gid=1200278807&single=true" width="100%"
height="600" style="border:none;"></iframe>

<!-- Hipervínculo -->
<p class="mt-4">
    <a
href="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD
6reeLojNGRtz9nCUIDpo/edit?usp=sharing" class="btn btn-primary">
        Haz click aquí para ver tabla de datos
    </a>
</p>
</div>

<div class="texto-en-marco">
    Este es un texto en color rojo dentro de un marco de 500
    píxeles de ancho.
</div>

<!-- Footer -->
<footer>
    &copy; 2024 Servicios a la oficina de reparto. Todos los
    derechos reservados.
</footer>

<!-- Incluyendo Bootstrap JS y Popper.js -->
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popp
er.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.m
in.js"></script>
</body>
</html>

```

Escribe aquí el código CSS sin HTML:

```

/* Estilos generales */
body {
    font-family: 'Arial', sans-serif;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
    color: #333;
}

```

```

header {
  background-color: #0c4da2;
  color: white;
  padding: 20px 0;
  text-align: center;
  box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);
}

h1 {
  margin: 0;
  font-size: 2rem;
}


/* Imagen principal */
.main-image {
  width: 100%;
  max-height: 400px;
  object-fit: cover;
  box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
  margin-bottom: 20px;
}

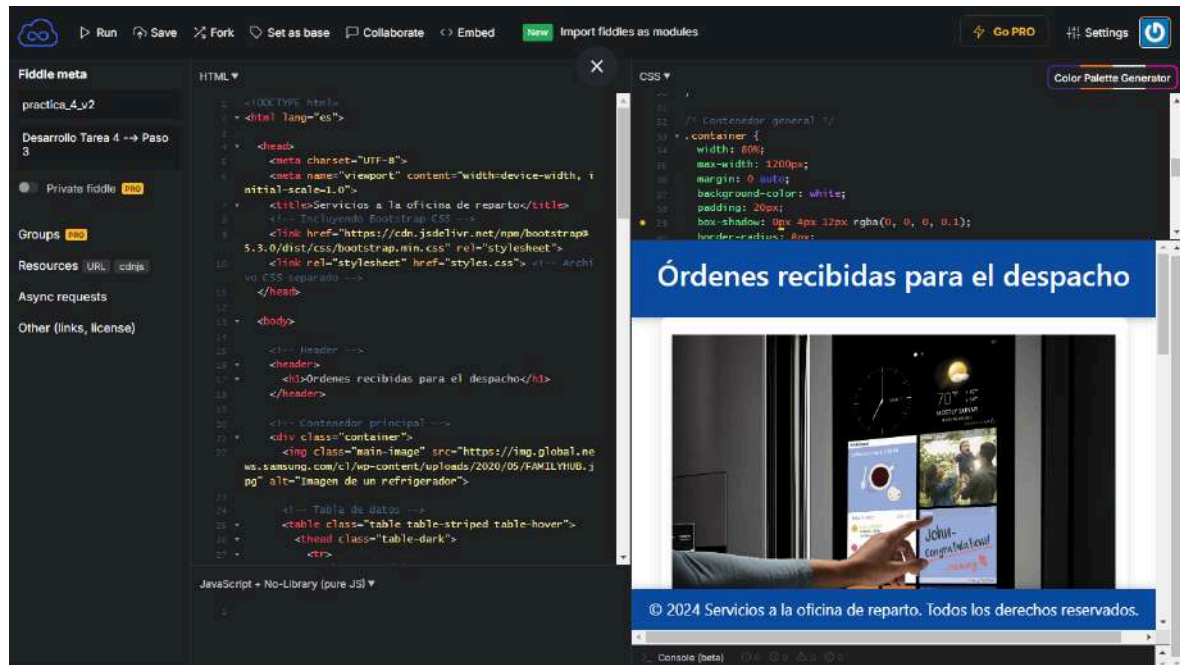
/* Contenedor general */
.container {
  width: 80%;
  max-width: 1200px;
  margin: 0 auto;
  background-color: white;
  padding: 20px;
  box-shadow: 0px 4px 12px rgba(0, 0, 0, 0.1);
  border-radius: 8px;
}

/* Texto dentro de un marco */
.texto-en-marco {
  color: red;
  width: 500px;
  border: 2px solid #0c4da2;
  padding: 10px;
  margin: 20px auto;
  text-align: center;
  border-radius: 8px;
  background-color: #fefefe;
}

/* Footer */
footer {
  background-color: #0c4da2;
  color: white;
  text-align: center;
  padding: 10px 0;
  position: fixed;
  bottom: 0;
  width: 100%;
  box-shadow: 0px -2px 8px rgba(0, 0, 0, 0.2);
}

```

Crea un nuevo fiddle para esta prueba así:  > Your fiddles > New fiddle. Pega el código HTML en la ventana HTML y el código CSS en la ventana CSS. Corre la solución con “Run”, pega un pantallazo del resultado (ventana Result)



Conclusión 1: Usa Chat GPT para aclarar las siguientes inquietudes ¿podríamos decir que un editor permite trabajar HTML y CSS parados cada uno en un archivo? ¿Cómo hace el navegador para operar con los varios archivos de una página si solo tiene una URL? ¿Se supone que esa URL no es de ninguno de esos archivos sino de un producto que resulta de la conversión a página web?. Lee lo que dice Chat GPT y resume la conclusión con tus propias palabras.

Un editor permite trabajar HTML y CSS en archivos separados. El navegador puede manejar múltiples archivos de una página web a través de un solo archivo HTML principal que enlaza con los demás archivos, como CSS y JavaScript. Aunque solo se accede a una URL, dicha URL carga el archivo HTML, que a su vez contiene referencias a otros recursos necesarios como CSS y JavaScript para construir la página completa. La URL apunta al archivo HTML principal que, a través de enlaces, coordina la carga y visualización de los demás archivos (CSS, JavaScript, imágenes) necesarios para construir la página web.

Conclusión 2: Está claro que un entorno de desarrollo como jsfiddle entrega varios archivos para una sola solución, que no hay problema para un servidor web trabajar con varios archivos, pero ¿si tengo varios archivos de una solución web obtenidos con jsfiddle y quiero embeber la solución en una página web, como es el caso de Google Sites, cómo obtengo el código HTML a embeber a partir de los varios archivos de la solución web?

Para embeber una solución web con varios archivos (como HTML, CSS, y JavaScript) en una página web de Google Sites, donde solo puedes insertar un fragmento de código HTML, puedes seguir estos pasos:

1. **Combina el Código:** Agrupa todo el contenido necesario en un solo archivo

HTML. Aquí está el proceso general:

- **HTML:** Copia el contenido del archivo HTML que has obtenido.
- **CSS:** Incorpora el CSS directamente dentro de una etiqueta `<style>` en el archivo HTML.
- **JavaScript:** Incorpora el JavaScript directamente dentro de una etiqueta `<script>` en el archivo HTML.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Mi Solución Web</title>
  <style>
    /* Aquí va el CSS de tus archivos */
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
    }
    /* Otros estilos */
  </style>
</head>
<body>
  <!-- Aquí va el contenido HTML de tus archivos -->
  <h1>Mi Página Web</h1>
  <p>Contenido de la página</p>
  <!-- Otros elementos HTML -->

  <script>
    // Aquí va el JavaScript de tus archivos
    console.log('JavaScript cargado');
    // Otros scripts
  </script>
</body>
</html>
```

2. **Sube Recursos Externos (si es necesario):** Si tu solución web depende de archivos externos como imágenes o bibliotecas JavaScript/CSS que no puedes incluir directamente en el archivo HTML, asegúrate de que estos archivos estén accesibles en la web y usa enlaces absolutos en tu HTML.
3. **Obtén el Código Final:** Una vez que hayas combinado el HTML, CSS y JavaScript en un solo archivo, puedes copiar este código.
4. **Embeber en Google Sites:**
  - En Google Sites, selecciona el lugar donde quieres insertar el código.
  - Usa el widget de "Insertar" y selecciona "Incrustar código" o "HTML" (dependiendo de la versión de Google Sites).
  - Pega el código HTML combinado en el campo correspondiente.

Guarda el código resultante en to jsfiddle para tu colección.

## Tarea 5. Publicación de página html. Creación de una página web

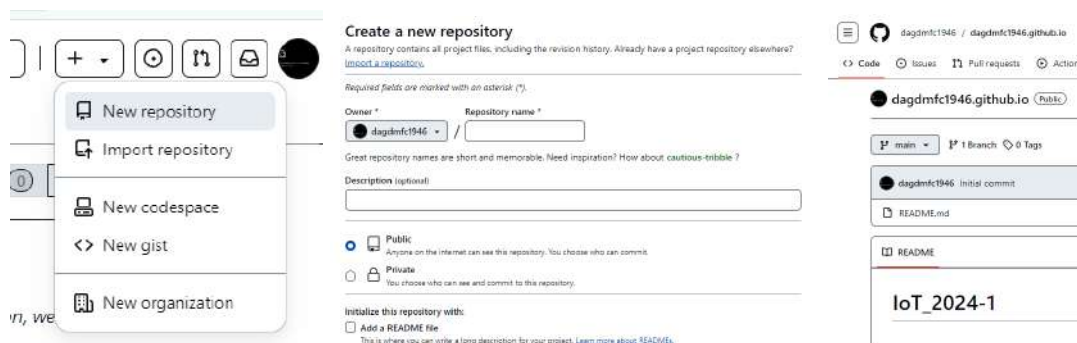
Estás cansado de usar un simulador de navegador. Entonces es hora de crear un sitio web para poder verlo en un navegador real. Esta también puede ser una solución a la pregunta que aparece en la conclusión 2 de la tarea anterior, ya que el resultado de una publicación en Google Sites para embeber una solución no solo está la posibilidad de usar código HTML sino también una URL.

Paso 1. Regístrate en GitHub. Tener una cuenta en GitHub (o en un sistema similar) es esencial para cualquier desarrollador. GitHub brinda el servicio de repositorio de software, lo cual permite guardar y compartir soluciones, pero sobre todo trabajar en equipo.

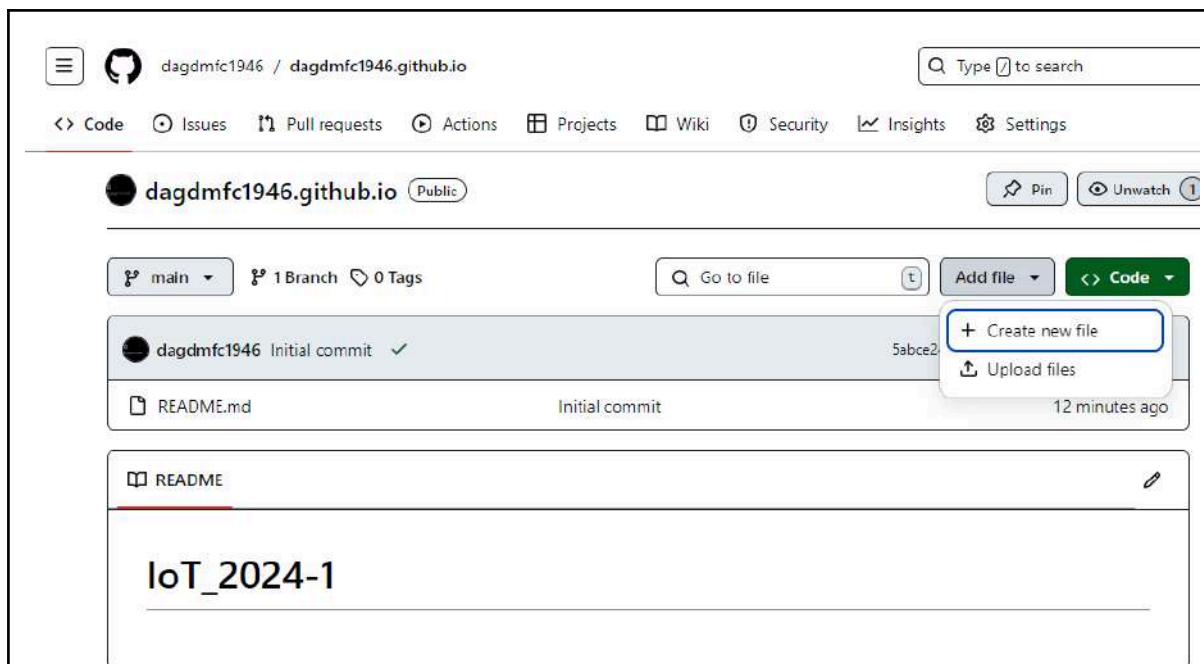
Paso 2. Aprender a manejar GitHub como repositorio requiere un taller solo dedicado a eso. Por ahora, lo que nos interesa es otra cosa: que GitHub tiene un servicio llamado GitHub pages a donde podemos subir los archivos de un desarrollo web y los publica como una página web. Este servicio puede ser considerado como Serverless, ya que de otra suerte tendríamos que crear un servidor web propio para este fin.

El profesor, aunque conoce esta posibilidad, nunca la ha experimentado. Se debe investigar cómo hacerlo y explicarle al profesor el paso a paso. Muestra pantallazos de tu solución

Inicialmente se debe tener una cuenta creada en GitHub, posteriormente se debe crear un repositorio nuevo:



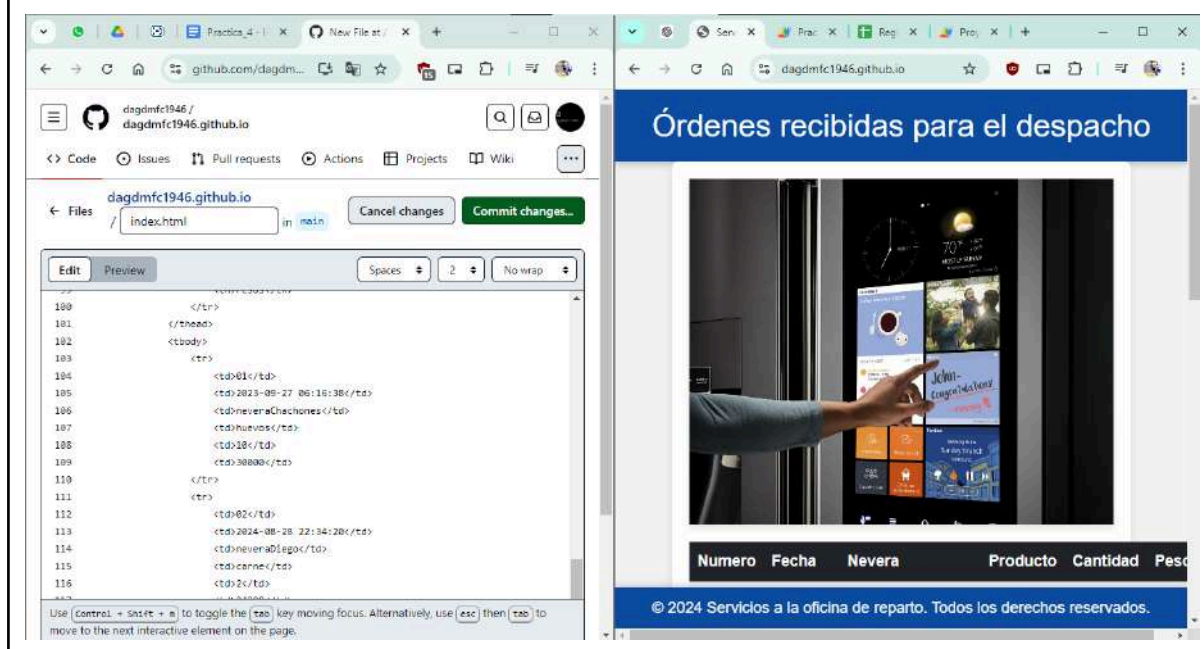
Se sugiere nombrar al repositorio con el nombre de usuario agregando la extensión **.github.io** → **dagdmfc1946.github.io**, esta extensión se usará para crear la “GitHub Page”, sin dicha extensión no se podría crear la la GitHub Page. Enseguida se deben subir los archivos:



En mi caso seleccione “Create new file” y pegue el código de HTML y luego click en “Commit changes...”, se puede agregar una descripción y luego nuevamente click en “Commit changes”:

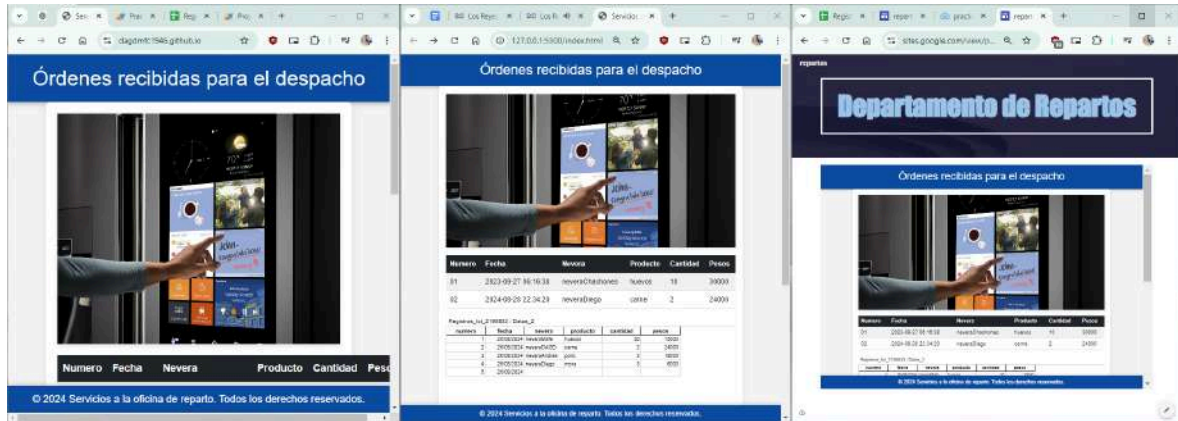
Name	Last commit message	Last commit date
README.md	Initial commit	22 minutes ago
index.html	Create index.html	5 minutes ago

Una vez con el archivo agregado al repositorio, se abre el enlace de la GitHub Page <https://dagdmfc1946.github.io/> y se observa lo siguiente:





Finalmente se muestra las tres formas de poder observar la página web:



Escribe aquí la URL de tu página web

<https://dagdmfc1946.github.io/>

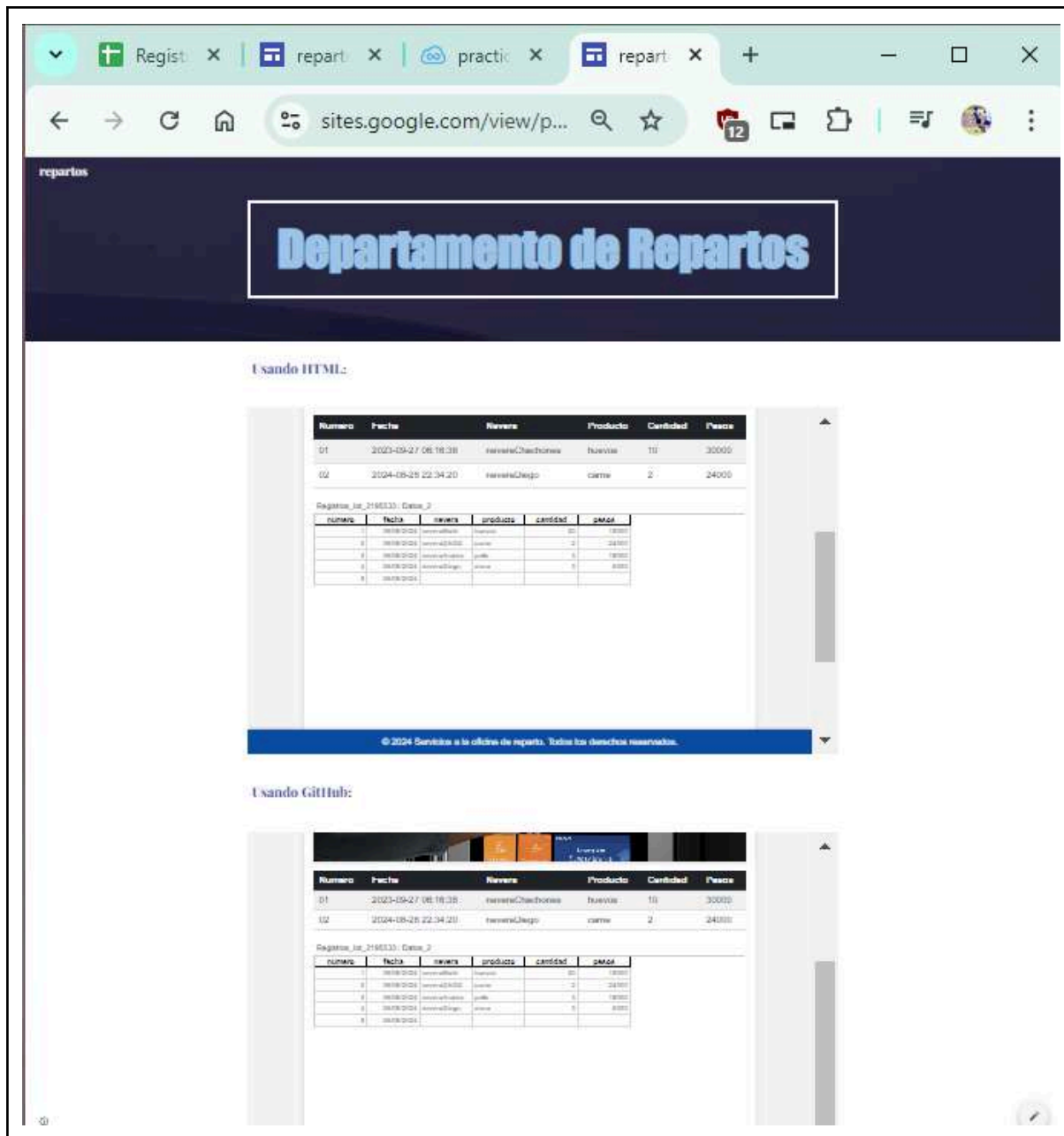
Escribe aquí el paso a paso para usar GitHub Pages como servidor Web

Se explicó en el paso anterior.

Paso 4. Embebe tu página web, en Google Sites usando la URL de tu página web

Toma un pantallazo de la solución





## Parte 2. El reto general para HTML más inteligente

Gracias a las posibilidades que brinda Google Sheet, el desarrollo web realizado en la práctica anterior puede ser visto como una solución final. Pero qué pasaría si no se usa Google Sheet sino otra base de datos u otra forma de almacenamiento de la información. Pero un problema muy claro está en que los datos que se visualizan en la solución de la práctica anterior no se refrescan automáticamente cuando ocurren cambios en la google sheet, sino después de minutos. Podemos agregarle a la página un botón para que, cuando el cliente lo oprima, ocurra el refresco de la página y por consiguiente de los nuevos datos. Pero ¿qué inteligencia se ocupará de vigilar si el botón es oprimido y de ordenar que la página se refresque?. Solucionar algo tan sencillo como esto ya requiere el uso de scripts y

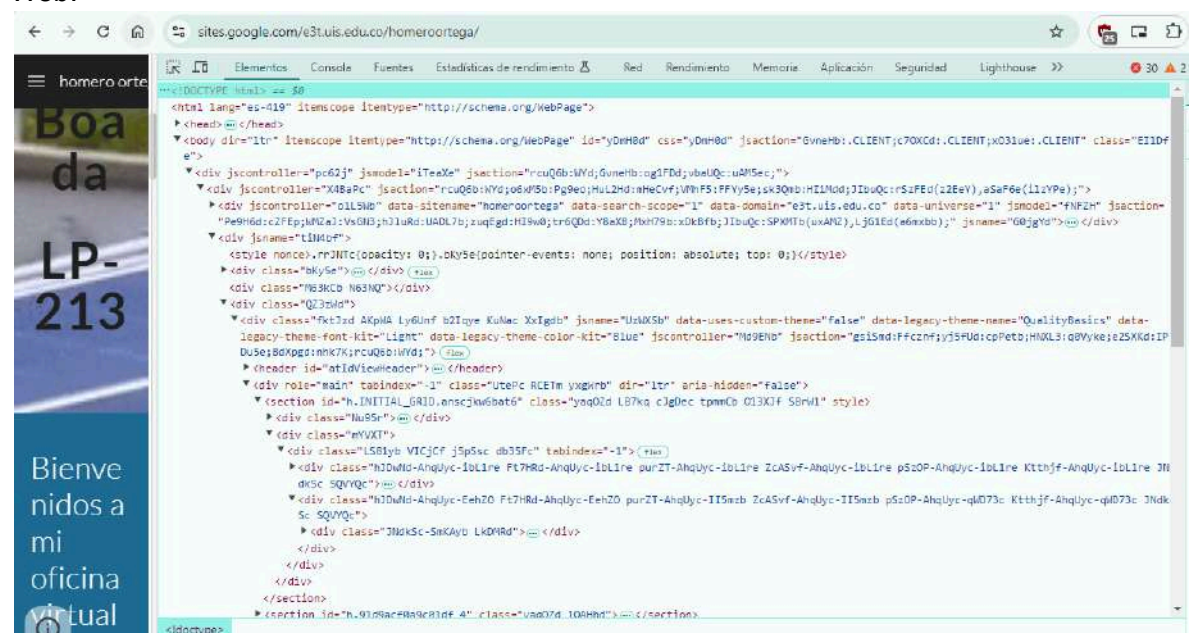
los scripts en html se escriben casi siempre con JavaScript. Ese es el reto, usar JavaScript para que la solución atienda mejor las necesidades del cliente. Pero como es la primera vez que usamos JavaScript, seguiremos pasos más detallados de lo normal para ir conquistando los trucos esenciales para usar scripts en soluciones Web a nivel de Front-End.

## Tarea 1. Conquista del Navegador en modo desarrollador para pruebas de JavaScript.

**Paso 1.** Abrir el navegador en modo desarrollador y explorar lo que allí se encuentra.  
abre tu navegador > visita cualquier página, por ejemplo  
<https://sites.google.com/e3t.uis.edu.co/homeroortega/> > click derecho > Inspeccionar.

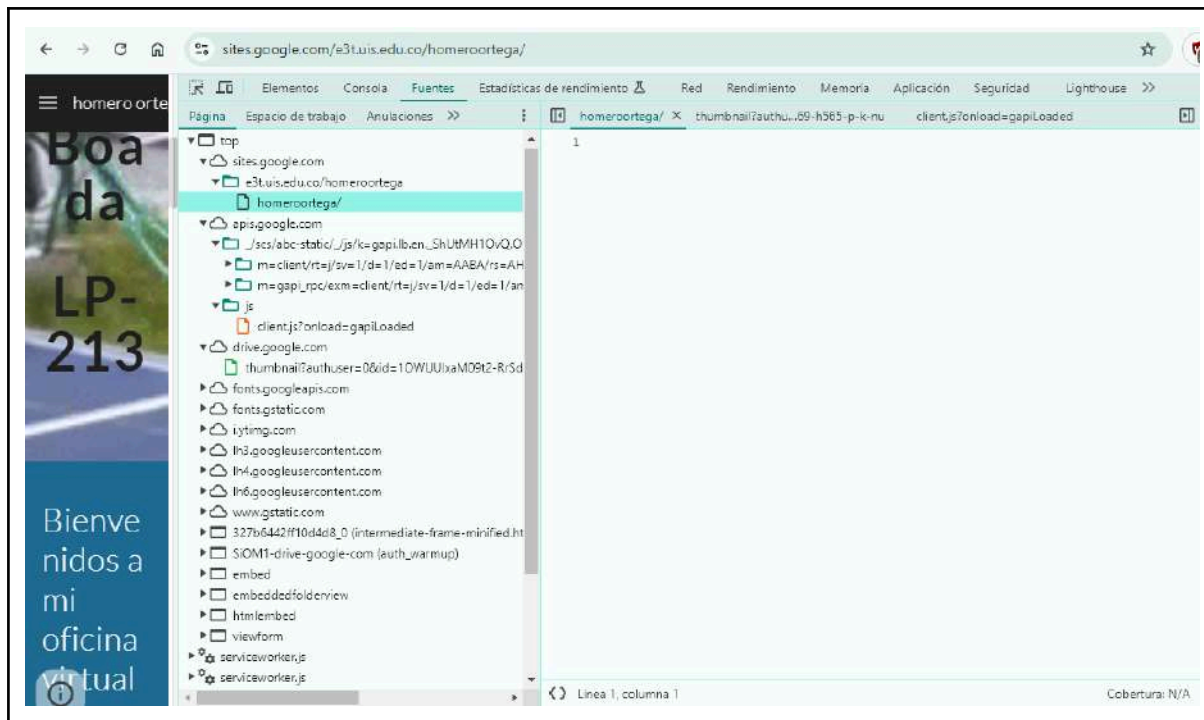
### ¿Qué ves en Elements?

Se observa el código HTML que contiene la estructura y cuerpo de la página Web:



### ¿Qué ves en Sources?

Se ven todos los archivos que componen la página Web, como lo es el HTML, CSS y JavaScript.

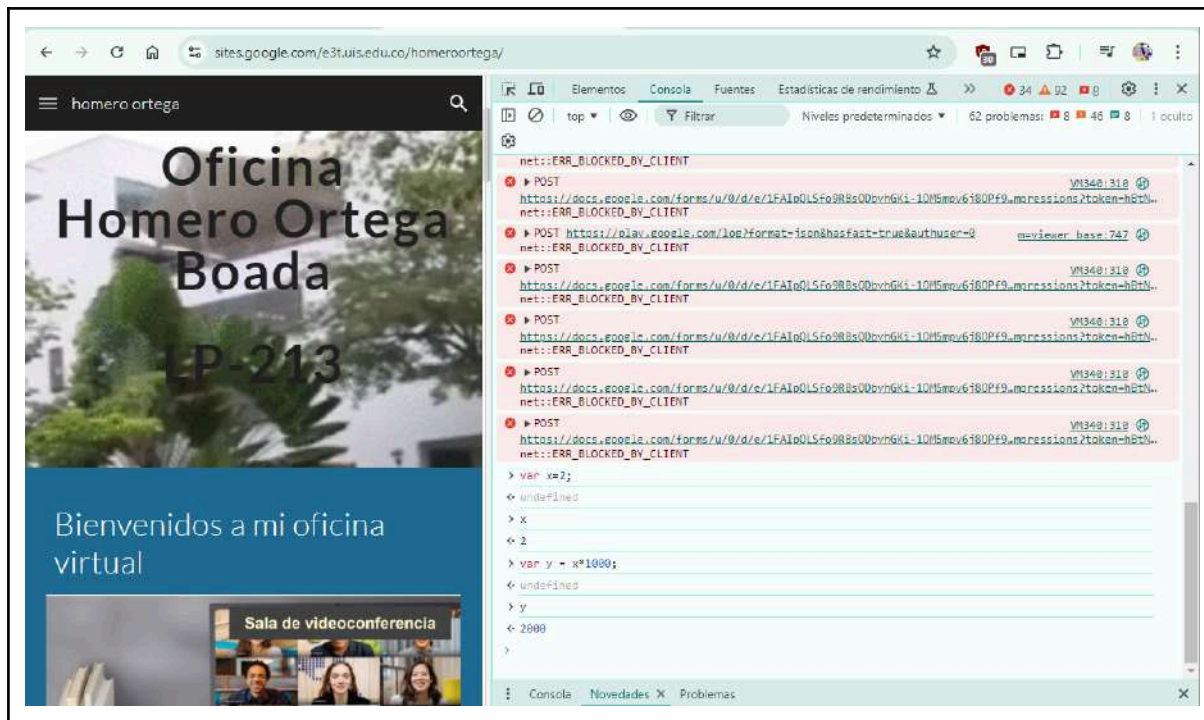


**Paso 2.** Pruebas de JavaScript en consola. Comprobar que la consola sirve para probar código JavaScript en modo interpretado, como se hace con Python o Matlab.

Después del paso anterior, pasa a la pestaña “Console”. Escribe código JavaScript en modo de programación interpretada. Por ejemplo:

```
> var x=2;
< undefined
> x
< 2
> var y = x*1000;
< undefined
> y
< 2000
>
```

Escribe aquí tus pruebas de programación por consola en javascript



**Paso 3.** Aprende a usar JavaScript en un HTML de manera oculta, es decir, que la vista para el usuario no se modifica. Eso significa que JavaScript no realiza cambios al DOM, usa una función propia `alert()` para producir una visualización pero no está ligada al DOM. Solo enviamos un aviso de que hay código en ejecución pero el código no presenta ninguna señal para el usuario, pero sí puede presentar cosas en la consola.

Crea en tu computador, un archivo de texto que se llame "mi\_html.html". Pégame el código y guárdalo. Nota: el explorador debe ser configurado para que muestre la extensión del archivo, para garantizar que la extensión sea html, no txt. Ahora abre el archivo `mi_html.html` usando un navegador de internet: click derecho sobre el archivo > abrir con Google Chrome.

Comienza por correr el siguiente código (simplemente lo abres con el navegador, el navegador lo ejecutará y te invitará a oprimir un botón, al oprimir el botón y aceptar se ejecuta el script) y revisa en modo desarrollador que es posible ver el código JavaScript, pero también los resultados del cómputo en la consola.

```
<!DOCTYPE html>
<html>
<head>
  <title>Ejemplo de HTML con JavaScript sin interacciones con el DOM</title>
</head>
<body>
  <h1>Página para comprobar el uso de JavaScript sin interacciones con el DOM</h1>
  <button onclick="asustador()">Oprime el botón</button>
  <script>
    function asustador() {
      alert('¡Cuidado!. Tan pronto aceptes o hagas algo, comenzaré a ejecutar mi código de
manera oculta para tí. Acepta si no te da miedo esto. Pero te comparto un secreto: puedes ver mi
código en modo desarrollador y el resultado de mi código en la consola');
      var largo = 10;
      console.log('largo = ' + largo);
    }
  </script>
</body>
</html>
```

```

    var ancho = 2;
    console.log('ancho = ' + largo);
    var alto = 3;
    console.log('alto = ' + largo);
    var volumen = largo * ancho * alto;
    console.log('El volumen = ' + volumen);
  }
</script>
</body>
</html>

```

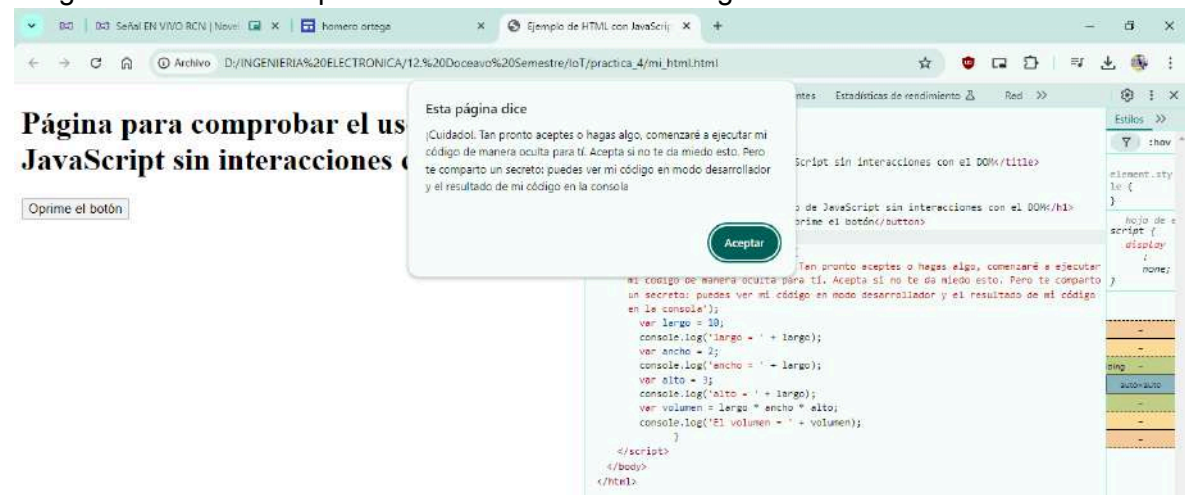
Agrega aquí capturas de pantalla en modo desarrollador que demuestren que lo anterior es correcto

Al abrir el archivo usando un navegador se obtiene lo siguiente:



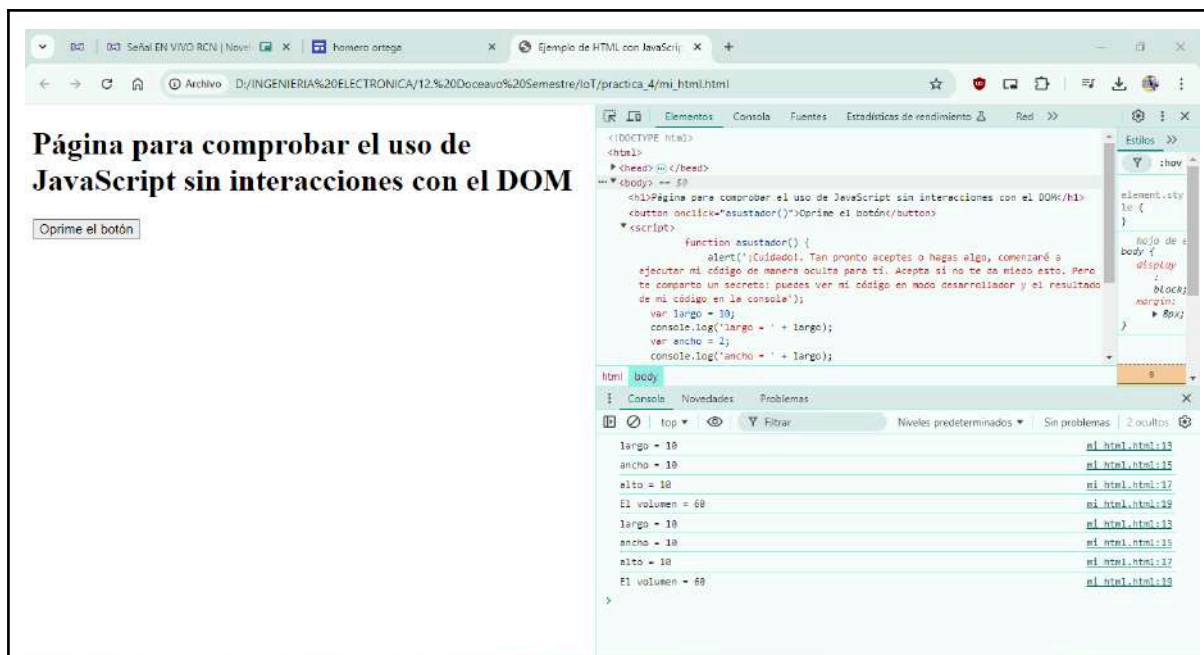
Se observa en la parte izquierda el contenido de archivo .html, en la parte derecha se está en modo desarrollador en la parte de 'Elementos' se puede observar el script HTML

Luego se da click en "Oprime el botón" se obtiene lo siguiente:



Se ejecuta el código y se revisa en la parte de 'Consola' si se ejecutó correctamente el script:





En este caso ejecuté el código dos veces.

**Conclusión 1.** Con lo aprendido en este paso, explica qué normas mínimas hay que seguir para poder insertar código de JavaScript en HTML. Nota: tu explicación debe ser tan buena que servirá para consultas futuras.

Para insertar y ejecutar código JavaScript en un archivo HTML, es importante seguir ciertas normas básicas:

1. **Colocación del código JavaScript:**
  - El código JavaScript puede ser insertado directamente en la sección `<script>` dentro del archivo HTML. Este script puede estar en el `<head>` o en el `<body>`, pero comúnmente se coloca al final del `<body>` para asegurar que todos los elementos del DOM estén cargados antes de que se ejecute el script.
2. **Uso del atributo onclick u otros eventos:**
  - Para ejecutar una función JavaScript en respuesta a una acción del usuario, como hacer clic en un botón, se puede usar el atributo onclick u otros atributos de eventos en elementos HTML. Esto vincula el evento al código JavaScript.
3. **No modificación del DOM:**
  - Es posible tener código JavaScript que no modifique el DOM. Esto significa que el código puede ejecutarse en segundo plano, sin afectar la estructura o contenido visible de la página web. En este caso, se utilizan funciones como `console.log()` para ver los resultados del código en la consola del navegador sin afectar la interfaz del usuario.
4. **Uso de alert() para interacciones mínimas:**
  - Aunque no se modifique el DOM, se puede usar `alert()` para notificaciones simples sin alterar el contenido de la página. Este tipo de notificaciones aparecen como diálogos emergentes que no interfieren directamente con el diseño del HTML.
5. **Depuración mediante la consola del navegador:**
  - Los navegadores modernos ofrecen herramientas de desarrollo que permiten inspeccionar el código HTML, CSS y JavaScript. El código JavaScript que no interactúa con el DOM puede generar resultados en la consola, lo cual facilita la depuración y el análisis del comportamiento del script sin que el usuario vea cambios en la interfaz.

Estos conceptos son fundamentales para el desarrollo de scripts en HTML que no requieren interacción con el DOM, permitiendo el procesamiento en segundo plano y el uso de la consola del

navegador para mostrar los resultados.

**Conclusión 2:** Puede decirse que esta solución usa cómputo en la nube?

Esta solución no usa cómputo en la nube. El código JavaScript que se ejecuta en este caso corre localmente en el navegador del usuario, no en un servidor remoto. El cómputo en la nube implica que la ejecución del código o el procesamiento de datos se realiza en servidores remotos y no en el dispositivo local del usuario.

**Paso 4.** Aprende a programar una solución que combina varios archivos. En este caso un archivo HTML y uno de JavaScript

Crea en tu computador un archivo de texto que se llame “micodigo.js”. Pégale el siguiente código y guárdalo. Nota: el explorador debe ser configurado para que muestre la extensión del archivo, para garantizar que la extensión sea js, no txt

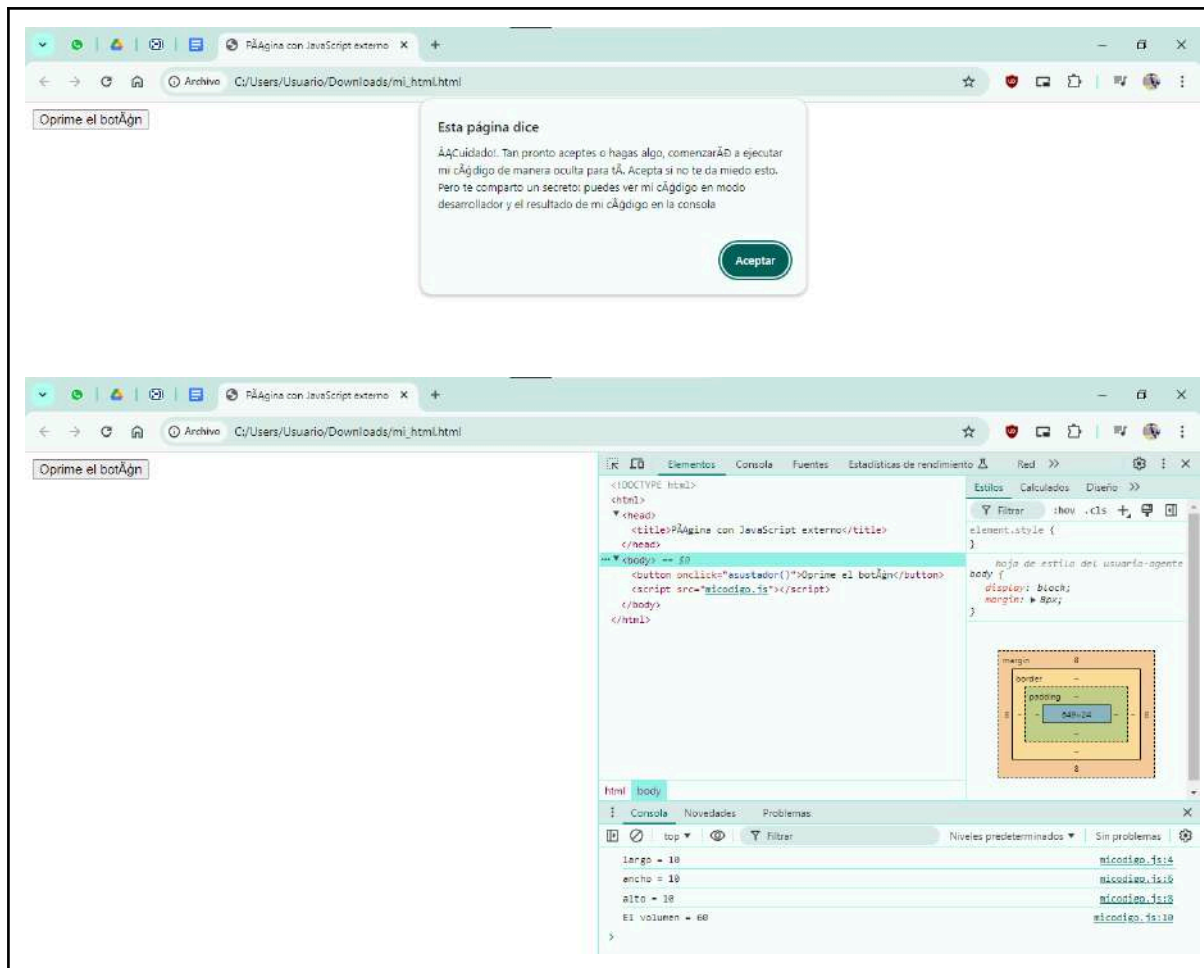
```
function asustador() {  
  alert('¡Cuidado!. Tan pronto aceptes o hagas algo, comenzaré a ejecutar mi código de  
manera oculta para tí. Acepta si no te da miedo esto. Pero te comparto un secreto: puedes ver mi  
código en modo desarrollador y el resultado de mi código en la consola');  
  var largo = 10;  
  console.log('largo = ' + largo);  
  var ancho = 2;  
  console.log('ancho = ' + largo);  
  var alto = 3;  
  console.log('alto = ' + largo);  
  var volumen = largo * ancho * alto;  
  console.log('El volumen = ' + volumen);  
}
```

Crea en tu computador, en el mismo lugar del caso anterior, un archivo de texto que se llame “mi\_html.html”. Pégale el siguiente código y guárdalo. Nota: el explorador debe ser configurado para que muestre la extensión del archivo, para garantizar que la extensión sea html, no txt.

```
<!DOCTYPE html>  
<html>  
<head>  
  <title>Página con JavaScript externo</title>  
</head>  
<body>  
  <button onclick="asustador()">Oprime el botón</button>  
  <script src="micodigo.js"></script>  
</body>  
</html>
```

Abre el archivo mi\_html.html usando un navegador de internet: click derecho sobre el archivo > abrir con Google Chrome

Pega pantallazos como evidencia del trabajo hecho



**Comprueba qué cambios hay que hacer si se usa un ambiente de desarrollo como JSFiddle para lograr el mismo efecto de una solución con un archivo HTML y uno en JavaScript.**

Para esto simplemente se deben ingresar los códigos HTML y JavaScript, en la parte donde corresponde, en JSFiddle no es necesario manejar archivos externos, ya que se hace internamente en los paneles asignados para cada uno de los lenguajes de programación que se usan. Para verificar el funcionamiento del código, simplemente se da click en **"Run"**, donde se hace uso de la consola del navegador para poder ver los mensajes generados por `"console.log()"`.

**Conclusión.** Con lo aprendido en este paso, explica qué normas mínimas hay que seguir para poder crear una solución cuando hay varios archivos que forman parte de ella. Nota: tu explicación debe ser tan buena que servirá para consultas futuras

Cuando se desarrollan aplicaciones web que constan de varios archivos, como HTML, CSS y JavaScript, es esencial seguir ciertas normas para asegurar una correcta organización y funcionamiento. A continuación, se describen las pautas mínimas que se deben seguir para crear una solución modular y escalable.

## 1. Estructura de Archivos y Directorios

- **Organización de Carpetas:**
  - Se deben organizar los archivos en carpetas que faciliten su mantenimiento. Un esquema típico incluye:



- index.html: el archivo HTML principal.
- /css/: carpeta para hojas de estilo CSS.
- /js/: carpeta para archivos JavaScript.
- /img/: carpeta para imágenes.
- /assets/: carpeta opcional para otros recursos (fuentes, iconos, etc.).
- **Uso de Rutas Relativas:**
  - Es importante usar rutas relativas para conectar los archivos. Por ejemplo, desde index.html se puede referenciar un archivo CSS en /css/estilos.css usando la ruta relativa <link rel="stylesheet" href="css/estilos.css">.
  - Del mismo modo, se debe referenciar el JavaScript en el HTML con rutas relativas, por ejemplo, <script src="js/funciones.js"></script>.

## 2. División de Responsabilidades

- **HTML para la Estructura:**
  - El archivo HTML debe ocuparse de definir la estructura básica del contenido de la página. Las etiquetas de HTML organizan el layout y las secciones, pero no deben contener código CSS o JavaScript directamente (excepto scripts específicos o pequeñas adiciones).
- **CSS para el Estilo:**
  - Todo el código de estilo debe ser separado en archivos CSS dedicados. El HTML debe vincularse a estos archivos usando la etiqueta <link> en el <head>. Evitar el uso de estilos en línea para mantener la limpieza del código y facilitar la reutilización y el mantenimiento de los estilos.
- **JavaScript para la Lógica:**
  - El código JavaScript debe estar en archivos .js separados. Este código se puede vincular en el HTML mediante la etiqueta <script> justo antes del cierre de la etiqueta </body>. Esto asegura que el HTML esté completamente cargado antes de ejecutar cualquier código JavaScript que pueda interactuar con el DOM.

## 3. Modularidad del Código

- **Separación del Código por Funcionalidad:**
  - En soluciones más grandes, es buena práctica dividir el código JavaScript en varios archivos según la funcionalidad. Por ejemplo, puedes tener un archivo validacion.js para validaciones de formularios, y otro interacciones.js para manejar eventos en la interfaz de usuario.
- **Uso de Módulos (ES6):**
  - En proyectos más avanzados, es recomendable utilizar la sintaxis de módulos de ECMAScript 6 (ES6). Esto permite importar y exportar funciones o variables entre archivos JavaScript de manera más estructurada, facilitando la escalabilidad del proyecto.

## 4. Integración de Archivos

- **Conexión entre HTML, CSS y JavaScript:**
  - Asegúrate de que el HTML tenga las referencias correctas a los archivos CSS y JavaScript mediante las etiquetas <link> y <script>.
  - Los archivos CSS se colocan en el <head> del documento, mientras que los archivos JavaScript se suelen colocar antes de la etiqueta </body> para

optimizar la carga de la página.

## 5. Consideraciones de Rendimiento

- **Minimización de Archivos:**
  - Es buena práctica minimizar (minify) los archivos CSS y JavaScript en producción para mejorar los tiempos de carga. Esto implica reducir el tamaño de los archivos eliminando espacios, comentarios y otras partes innecesarias del código.
- **Agrupación y Carga Diferida:**
  - En algunos casos, agrupar varios archivos CSS o JavaScript en uno solo reduce la cantidad de solicitudes HTTP al servidor, lo que mejora el rendimiento.
  - También puedes usar técnicas como la carga diferida (defer o async en <script>) para evitar bloqueos en la carga de la página.

## 6. Mantenimiento y Escalabilidad

- **Comentarios y Documentación:**
  - Comentar el código de manera clara y útil es fundamental, especialmente en proyectos que incluyen varios archivos y equipos de desarrollo. Utiliza comentarios para describir la funcionalidad de bloques de código importantes, así como las dependencias entre archivos.
- **Control de Versiones:**
  - Implementar un sistema de control de versiones, como Git, es esencial para gestionar los cambios en un proyecto con varios archivos. Esto te permitirá rastrear el historial de cambios, colaborar con otros desarrolladores y revertir cambios en caso de errores.

## 7. Pruebas y Depuración

- **Uso de Consola y Herramientas de Desarrollo:**
  - Utiliza la consola del navegador y las herramientas de desarrollo para depurar y probar tu código. Al trabajar con varios archivos, es importante identificar problemas en la integración o ejecución del código desde el principio.
- **Pruebas Unitarias:**
  - En proyectos grandes, considera la implementación de pruebas unitarias para asegurar que las diferentes partes de tu código funcione de manera correcta y autónoma.

**Paso 5.** Prueba usar JavaScript en consola para afectar una página web.

Puedes hacerle a Chat GPT esta consulta:

Mi intención es solo comprender si usando JavaScript puedo cambiar la vista de una página. Por ejemplo

- Tengo abierta, en un navegador, mi página web personal con la URL: <https://sites.google.com/e3t.uis.edu.co/homeroortega/>
- Estoy en la consola del navegador.

Necesito los comandos en JavaScript para cambiar el encabezado que ve el usuario. Seguramente obtendrás un código como este:

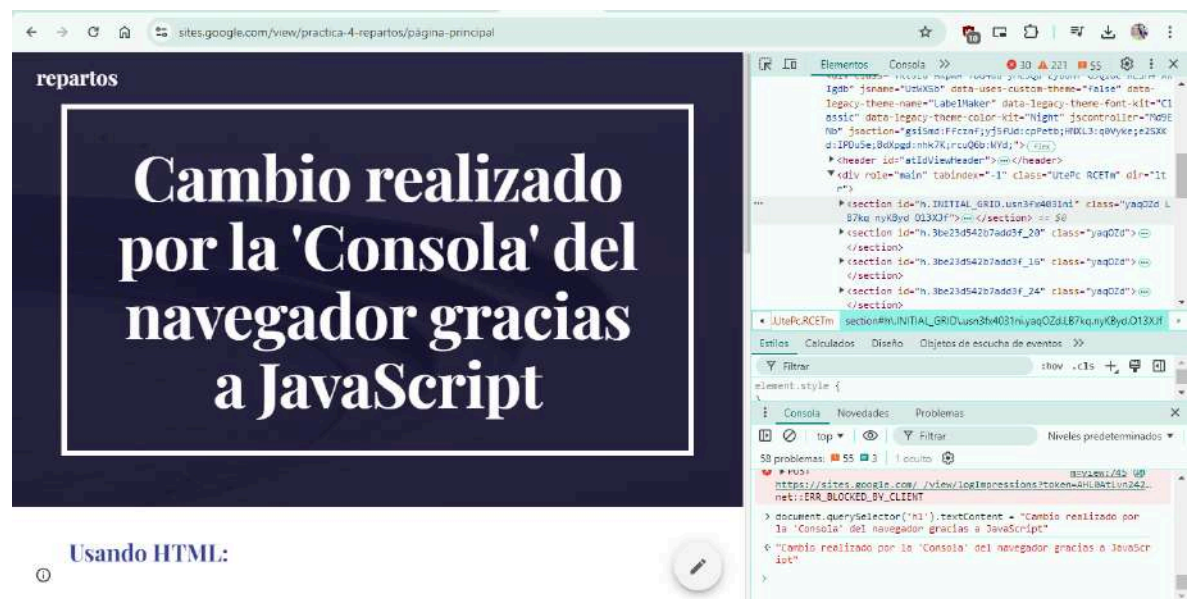
```
document.querySelector('h1').textContent = "Efecto de mi JavaScript por consola";
```

Escríbelo en la consola y observa el efecto en la página.

Escribe abajo el comando usado

```
document.querySelector('h1').textContent = "Cambio realizado por la  
'Consola' del navegador gracias a JavaScript"
```

Mediante una captura de pantalla muestra el efecto logrado en la página web



Explica cómo funciona esto en la interacción cliente-AJAX-Servidor. Hazle la consulta a Chat GPT. Por ejemplo así:

Puedo decir que al lanzar este código en la consola ocurre lo siguiente: 1) la orden que lanza el script llega a AJAX; AJAX solicita al servidor web cambiar el encabezado; el servidor devuelve al navegador el nuevo encabezado?

No, lanzar ese código en la consola del navegador **no** implica una solicitud AJAX ni una interacción directa con el servidor web. En realidad, lo que ocurre es lo siguiente:

### Explicación detallada:

1. **Ejecutar el código en la consola:** Cuando ejecutas un script en la consola del navegador, este código es interpretado y ejecutado **localmente** por el navegador en la página actual que estás viendo.
2. **Manipulación del DOM:** El código como `document.querySelector('h1').textContent = 'Nuevo encabezado'`; modifica directamente el **DOM** (Document Object Model) de la página en el navegador. Esto cambia el contenido visual de la página que se está mostrando al usuario. **No** se envía ninguna solicitud al servidor, ni hay una interacción con AJAX.
3. **Sin comunicación con el servidor:** El servidor web **no** está involucrado en este proceso. No se hace ninguna solicitud al servidor para cambiar el encabezado, ni el servidor devuelve ninguna respuesta. El cambio ocurre solo en la copia de la

página que está cargada en el navegador, y **no** es permanente. Si recargas la página, el contenido volverá a su estado original, porque la modificación solo afecta el estado temporal de la página en el navegador.

### Resumen de lo que sucede:

- **Local:** La modificación ocurre en el lado del cliente (el navegador) y solo afecta la copia actual del DOM.
- **No AJAX:** No se envía ninguna solicitud AJAX al servidor.
- **Temporal:** El cambio es temporal y se perderá al recargar la página, ya que no se hace ningún cambio en el servidor ni en los archivos originales de la página.

Si deseas que los cambios sean permanentes y que se guarden en el servidor, tendrías que hacer una solicitud al servidor (por ejemplo, usando AJAX) y actualizar la página en el servidor con la nueva información.

## Tarea 2. Uso de JavaScript para programar las acciones a realizar cuando el cliente oprima un botón.

Buscaremos mejorar la solución de la práctica 10 para que los datos se actualicen (refresquen) al oprimir un botón. El problema de la solución de la práctica 10 es que para refrescar los datos, el cliente debe refrescar todo el sitio web en el navegador, lo cual visualmente no es bueno, pero además se consumen muchos recursos. Esta acción requiere una interacción de mayor nivel que la que ofrece HTML puro, se requiere un script en JavaScript.

**Paso 1.** Mejora del código final de la Tarea 4 para que incluya un botón de refresco.

Consulta a Chat GPT como solucionar el problema.

Ejemplo de pregunta para Chat GPT:

Tengo un código HTML que contiene un frame. Quiero complementarlo para que contenga un botón llamado "Refrescar". Quiero que, cuando el cliente oprima el botón, se actualice el frame. El código HTML que tengo es el siguiente: [escribe aquí el código obtenido en la Tarea 4]

Escriba aquí el código obtenido

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Servicios a la oficina de reparto</title>
  <!-- Incluyendo Bootstrap CSS -->
  <link
```

```
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
```

```
<style>
/* Estilos generales */
body {
    font-family: 'Arial', sans-serif;
    background-color: #f4f4f4;
    margin: 0;
    padding: 0;
    color: #333;
}

header {
    background-color: #0c4da2;
    color: white;
    padding: 20px 0;
    text-align: center;
    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);
}

h1 {
    margin: 0;
    font-size: 2rem;
}

/* Imagen principal */
.main-image {
    width: 100%;
    max-height: 400px;
    object-fit: cover;
    box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
    margin-bottom: 20px;
}

/* Contenedor general */
.container {
    width: 80%;
    max-width: 1200px;
    margin: 0 auto;
    background-color: white;
    padding: 20px;
    box-shadow: 0px 4px 12px rgba(0, 0, 0, 0.1);
    border-radius: 8px;
}

/* Texto dentro de un marco */
.texto-en-marco {
    color: red;
    width: 500px;
    border: 2px solid #0c4da2;
    padding: 10px;
}
```

```

        margin: 20px auto;
        text-align: center;
        border-radius: 8px;
        background-color: #fefefe;
    }

    /* Footer */
    footer {
        background-color: #0c4da2;
        color: white;
        text-align: center;
        padding: 10px 0;
        position: fixed;
        bottom: 0;
        width: 100%;
        box-shadow: 0px -2px 8px rgba(0, 0, 0, 0.2);
    }
</style>
</head>
<body>

    <!-- Header -->
    <header>
        <h1>Órdenes recibidas para el despacho</h1>
    </header>

    <!-- Contenedor principal -->
    <div class="container">
        

        <!-- Tabla de datos -->
        <table class="table table-striped table-hover">
            <thead class="table-dark">
                <tr>
                    <th>Numero</th>
                    <th>Fecha</th>
                    <th>Nevera</th>
                    <th>Producto</th>
                    <th>Cantidad</th>
                    <th>Pesos</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>01</td>
                    <td>2023-09-27 06:16:38</td>
                    <td>neveraChachones</td>
                    <td>huevos</td>
                    <td>10</td>
                    <td>30000</td>
                </tr>
            </tbody>
        </table>
    </div>

```

```

        </tr>
        <tr>
            <td>02</td>
            <td>2024-08-28 22:34:20</td>
            <td>neveraDiego</td>
            <td>carne</td>
            <td>2</td>
            <td>24000</td>
        </tr>
    </tbody>
</table>

<!-- Agregando el iframe después de la tabla -->
<iframe id="miIframe"
src="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6reeL
ojNGRtz9nCUIDpo/pubhtml?gid=1200278807&single=true" width="100%"
height="600" style="border:none;"></iframe>

<!-- Botón para refrescar el iframe -->
<div class="text-center mt-4">
    <button onclick="refrescarIframe()" class="btn
btn-warning">Refrescar</button>
</div>

<!-- Hipervínculo -->
<p class="mt-4">
    <a
href="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6ree
LojNGRtz9nCUIDpo/edit?usp=sharing" class="btn btn-primary">
        Haz click aquí para ver tabla de datos
    </a>
</p>
</div>

<div class="texto-en-marco">
    Este es un texto en color rojo dentro de un marco de 500 píxeles
de ancho.
</div>

<!-- Footer -->
<footer>
    &copy; 2024 Servicios a la oficina de reparto. Todos los derechos
reservados.
</footer>

<!-- Incluyendo Bootstrap JS y Popper.js -->
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.m
in.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.min.j
s"></script>

```

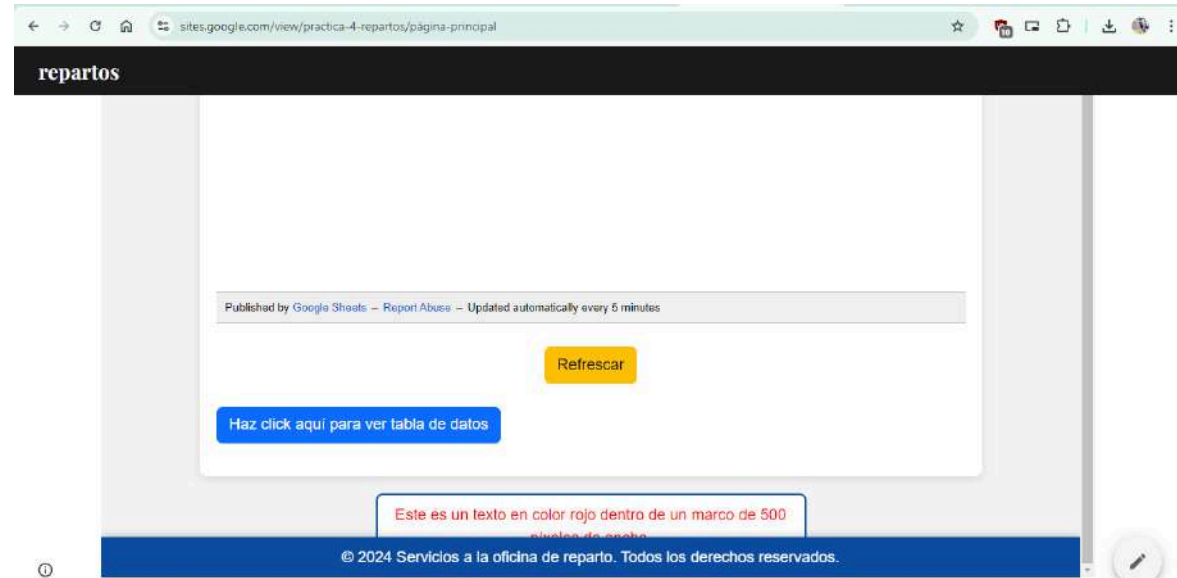
```

<!-- Script para refrescar el iframe -->
<script>
    function refrescarIframe() {
        document.getElementById('miIframe').src += '';
    }
</script>

</body>
</html>

```

En la página Web en versión edición crea una nueva versión de visualización de los datos. Embebe allí el código obtenido en el paso 1, de la misma manera en como se explicó en la tarea 4. Publica la página web nuevamente y pasa a la versión publicada. Captura la pantalla de la página resultante con los cambios



## Paso 2.

**Análisis.** Analice el código del paso 1. Explique dónde y cómo ha sido declarada la función JavaScript que realiza la tarea de refresco. Como ejemplo, puedes también explicar el código usado para invocar esa función. Nota: tu explicación debe ser tan buena que una persona la puede consultar para saber los pasos a seguir.

### 1. Declaración de la Función JavaScript `refrescarIframe()`

La función JavaScript que realiza la tarea de refresco ha sido declarada en la parte final del documento HTML, dentro de una etiqueta `<script>`. A continuación se presenta el bloque relevante del código:

```

<script>
    function refrescarIframe() {
        document.getElementById('miIframe').src += '';
    }
</script>

```



- **Ubicación de la declaración:**
  - La función está declarada dentro de una etiqueta `<script>`, que se encuentra justo antes del cierre de la etiqueta `</body>`. Esta posición asegura que todo el contenido HTML se haya cargado antes de que se ejecute el código JavaScript.
- **Cómo está estructurada:**
  - **Nombre de la función:** `refrescarIframe`.
  - **Contenido de la función:**
    - La función accede al elemento HTML con el atributo `id="milframe"` usando el método `document.getElementById('milframe')`.
    - Luego, modifica la propiedad `src` de ese elemento. La propiedad `src` contiene la URL del contenido que se carga dentro del `iframe`.
    - El código `document.getElementById('milframe').src += "`; básicamente reasigna el mismo valor de `src`, lo que provoca que el navegador recargue el contenido del `iframe`. Este es un truco común en JavaScript para refrescar un `iframe` sin cambiar la URL.

## 2. Invocación de la Función desde el HTML

La función `refrescarIframe()` es invocada cuando el usuario hace clic en un botón. A continuación se muestra cómo se ha implementado la invocación en el código HTML:

```
<div class="text-center mt-4">
  <button onclick="refrescarIframe()" class="btn
btn-warning">Refrescar</button>
</div>
```

- **Descripción del código:**
  - **Elemento HTML:** El botón está representado por una etiqueta `<button>`.
  - **Atributo onclick:** El atributo `onclick` es un atributo HTML que indica qué función JavaScript debe ejecutarse cuando se hace clic en el botón. En este caso, está asignado a la función `refrescarIframe()`, lo que significa que cuando el usuario haga clic en el botón, la función se ejecutará.
  - **Texto del botón:** El botón muestra el texto "Refrescar" en la interfaz.

## 3. Explicación Detallada de los Pasos:

### Paso 1: Crear la Función en el Bloque de Script

- Para crear una función en JavaScript, debes usar la palabra clave `function`, seguida del nombre de la función, paréntesis `()` para los parámetros (en este caso, no hay parámetros), y luego llaves `{}` que contienen el código que se ejecutará cuando se llame a la función.
- La función `refrescarIframe()` está diseñada para recargar un `iframe` en la página web, reasignando el mismo valor `src` del `iframe`.

### Paso 2: Insertar el Script en la Parte Inferior del HTML

- El script que contiene la función se coloca justo antes de cerrar la etiqueta `</body>`, para asegurarse de que todos los elementos del DOM se hayan cargado

antes de que el script sea ejecutado.

### Paso 3: Crear el Botón y Asociarlo con la Función

- Un botón HTML se define con la etiqueta <button>.
- El atributo onclick es utilizado para definir una acción a realizarse cuando el botón es clicado. En este caso, se asocia la función refrescarIframe() al evento onclick, asegurando que al presionar el botón, el iframe se actualice.

### Ejemplo Completo

A continuación se ilustra cómo interactúan los dos elementos (el script con la función y el botón que lo invoca):

#### 1. Declaración de la Función:

```
<script>
  function refrescarIframe() {
    document.getElementById('miIframe').src += '';
  }
</script>
```

#### 2. Invocación de la Función desde el Botón:

```
<div class="text-center mt-4">
  <button onclick="refrescarIframe()" class="btn
  btn-warning">Refrescar</button>
</div>
```

## Tarea 3. Mejorar la solución anterior para que los datos se actualicen (refresque) automáticamente cada 10 segundos

Es posible que el cliente no desee estar oprimiendo un botón para que los datos en la página se refresque. Pídele a Chat GPT que te ofrezca una modificación al código que ya tienes para lograr este efecto

Escribe abajo cómo le harás la pregunta a Chat GPT

Para el siguiente código HTML: [Insertar el código HTML modificado con la función para refrescar]

No necesito que la persona o cliente deba estar oprimiendo el botón de refresh cada que quiera actualizar los datos, quiero que estos se refresquen de forma automática cada 10 segundos, dame el código modificado para que se cumpla esta instrucción.

Nuevo código HTML obtenido

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Servicios a la oficina de reparto</title>
  <!-- Incluyendo Bootstrap CSS -->
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min
.css" rel="stylesheet">

  <style>
    /* Estilos generales */
    body {
      font-family: 'Arial', sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 0;
      color: #333;
    }

    header {
      background-color: #0c4da2;
      color: white;
      padding: 20px 0;
      text-align: center;
      box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.2);
    }

    h1 {
      margin: 0;
      font-size: 2rem;
    }

    /* Imagen principal */
    .main-image {
      width: 100%;
      max-height: 400px;
      object-fit: cover;
      box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
      margin-bottom: 20px;
    }

    /* Contenedor general */
    .container {
      width: 80%;
      max-width: 1200px;
      margin: 0 auto;
      background-color: white;
      padding: 20px;
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="main-image">
      <img alt="Placeholder for the main image" data-bbox="124 93 870 905"/>
    </div>
  </div>
</body>
</html>
```

```

        box-shadow: 0px 4px 12px rgba(0, 0, 0, 0.1);
        border-radius: 8px;
    }

    /* Texto dentro de un marco */
    .texto-en-marco {
        color: red;
        width: 500px;
        border: 2px solid #0c4da2;
        padding: 10px;
        margin: 20px auto;
        text-align: center;
        border-radius: 8px;
        background-color: #fefefe;
    }

    /* Footer */
    footer {
        background-color: #0c4da2;
        color: white;
        text-align: center;
        padding: 10px 0;
        position: fixed;
        bottom: 0;
        width: 100%;
        box-shadow: 0px -2px 8px rgba(0, 0, 0, 0.2);
    }
</style>
</head>
<body>

    <!-- Header -->
    <header>
        <h1>Órdenes recibidas para el despacho</h1>
    </header>

    <!-- Contenedor principal -->
    <div class="container">
        

        <!-- Tabla de datos -->
        <table class="table table-striped table-hover">
            <thead class="table-dark">
                <tr>
                    <th>Numero</th>
                    <th>Fecha</th>
                    <th>Nevera</th>
                    <th>Producto</th>
                    <th>Cantidad</th>
                    <th>Pesos</th>

```

```

        </tr>
    </thead>
    <tbody>
        <tr>
            <td>01</td>
            <td>2023-09-27 06:16:38</td>
            <td>neveraChachones</td>
            <td>huevos</td>
            <td>10</td>
            <td>30000</td>
        </tr>
        <tr>
            <td>02</td>
            <td>2024-08-28 22:34:20</td>
            <td>neveraDiego</td>
            <td>carne</td>
            <td>2</td>
            <td>24000</td>
        </tr>
    </tbody>
</table>

<!-- Agregando el iframe después de la tabla -->
<iframe id="miIframe"
src="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6reeL
ojNGRtz9nCUIDpo/pubhtml?gid=1200278807&single=true" width="100%"
height="600" style="border:none;"></iframe>

<!-- Botón para refrescar el iframe -->
<div class="text-center mt-4">
    <button onclick="refrescarIframe()" class="btn
btn-warning">Refrescar</button>
</div>

<!-- Hipervínculo -->
<p class="mt-4">
    <a
href="https://docs.google.com/spreadsheets/d/1FmE40k_cKIGeJ_n3YZdiUZD6ree
LojNGRtz9nCUIDpo/edit?usp=sharing" class="btn btn-primary">
        Haz click aquí para ver tabla de datos
    </a>
</p>
</div>

<div class="texto-en-marco">
    Este es un texto en color rojo dentro de un marco de 500 píxeles
de ancho.
</div>

<!-- Footer -->
<footer>
    &copy; 2024 Servicios a la oficina de reparto. Todos los derechos

```

```

reservados.
</footer>

<!-- Incluyendo Bootstrap JS y Popper.js -->
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.m
in.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.min.j
s"></script>

<!-- Script para refrescar el iframe -->
<script>
function refrescarIframe() {
    document.getElementById('miIframe').src += '';
}

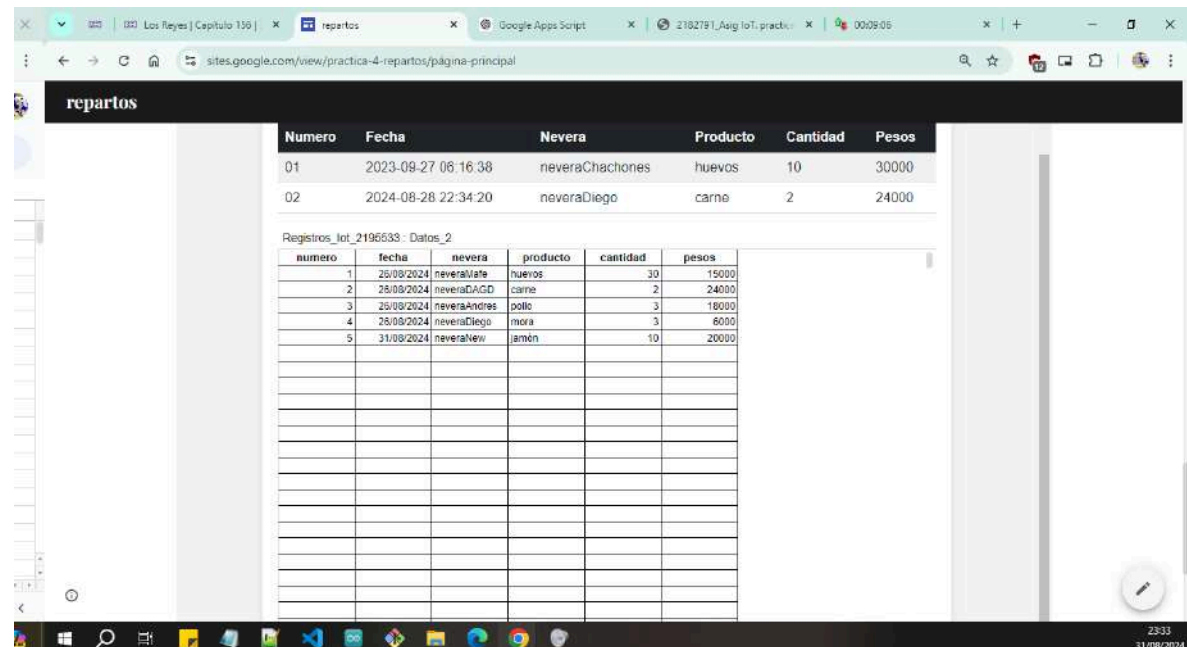
// Refrescar automáticamente cada 10 segundos (10000
milisegundos)
setInterval(refrescarIframe, 10000);
</script>

</body>
</html>

```

Evidencias en la página web, una vez sea re-publicada sobre esta tarea

Inicialmente tiene estos datos:



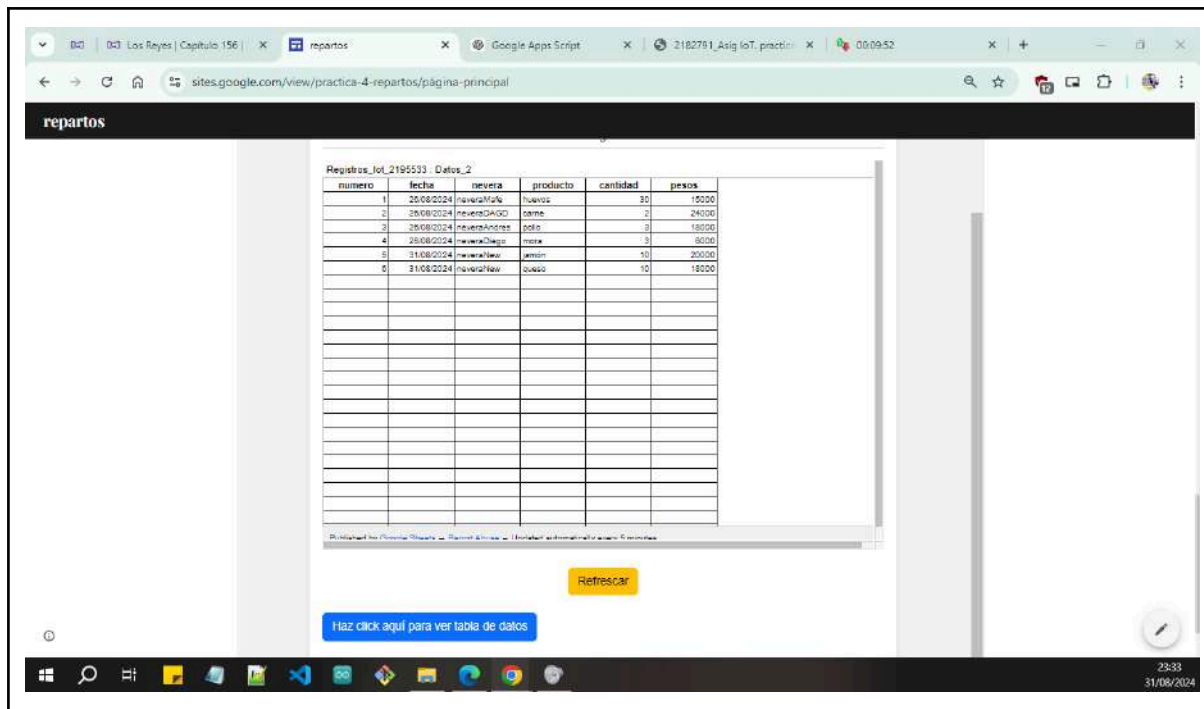
Numero	Fecha	Nevera	Producto	Cantidad	Pesos
01	2023-09-27 06:16:38	neveraChachones	huevos	10	30000
02	2024-08-28 22:34:20	neveraDiego	carne	2	24000

Registros\_lot\_2195633: Datos\_2

numero	fecha	nevera	producto	cantidad	pesos
1	26/09/2024	neveraMate	huevos	30	15000
2	26/09/2024	neveraDAGD	carne	2	24000
3	26/09/2024	neveraAndres	pollo	3	18000
4	26/09/2024	neveraDiego	mora	3	6000
5	31/09/2024	neveraNew	jamon	10	20000

Cuando se actualiza queda de la siguiente forma:



Tarea 4. Crea una solución de HTML con JavaScript que resulte más sorprendente.



## Tarea 5. Crea una solución integral para el departamento de despachos con JavaScript

- Un web service leerá los datos del google sheet que contiene los datos. Los convertirá a un JSON y los entregará a través de una función doGet()
- Una página Web con JavaScript tendrá un código mínimo en html y un javascript que consultará el webservice anterior para conocer el tamaño de los datos en la google sheet. Determinará el tamaño de los datos y basado en ellos creará una tabla apropiada para visualizarlos. Refrescará la lectura de los datos cada medio segundo.