Introducción a HTML

¡Comencemos!

¿Qué aprenderás?

- Identificar las diferencias entre los roles del desarrollo web para la construcción de aplicaciones y sitios web.
- Conocer la estructura, alcance y características de HTML para la creación de páginas web.
- Implementar la estructura assets del proyecto (imágenes y archivos CSS), utilizando rutas locales, absolutas y/o CDN, de acuerdo a las buenas prácticas de la industria.

Introducción

¡Bienvenid@! En las siguientes páginas, encontrarás un primer acercamiento a los distintos roles del desarrollo web y los conceptos principales de HTML, partiendo por la estructura de carpetas y los assets, para que conozcas cómo organizar los archivos y así los puedas localizar de una forma sencilla. Además, se abordarán cuáles son los requisitos del software que requieres instalar para comenzar a escribir código HTML, el uso del editor de código y algunos atajos.

Aprender HTML te permitirá ingresar en el lenguaje de la programación, ya que con él podrás construir páginas web y expandir tus conocimientos hacia otros lenguajes.

¡Vamos con todo!

¿Qué es el desarrollo web?

Detrás de todo sitio web se encuentran instrucciones escritas con una sintaxis predeterminada que cumplen distintos objetivos. Estas instrucciones están basadas en lenguajes que nos ayudan a construir interfaces gráficas y la lógica que contribuye con el avance tecnológico que hoy en día conocemos como aplicaciones o plataformas web. El acople y estandarización de las distintas tecnologías que ocupamos en el desarrollo web es implementado por la W3C, que por sus siglas **World Wide Web Consortium**, nos garantiza el crecimiento estable de la web a largo plazo.

Cuando hablamos de desarrollo, nos referimos al proceso de construcción de aplicaciones o sitios web, y dentro de este superfluo concepto podemos encontrarnos con distintos roles que conforman el espectro moderno de desarrollo web.

Diferencias entre Front-End, Back-End y Fullstack

Antiguamente no era extraño conocer personas que por sí solas eran capaces de construir los sitios web de los años 90's y 2000's, no obstante con los avances tecnológicos de entonces, el proceso de desarrollo se fue acomplejando hasta llegar al punto en el que una sola persona dejaba de ser suficiente para aprender e implementar las nuevas tecnologías que cumplían con las exigencias de la actualidad, por este motivo se fue dividiendo el antiguo rol conocido como "webmaster" en un especialista del acabado frontal o interfaces gráficas (frontend) y otro para el desarrollo de lo que sucede tras bambalinas (backend).

Frontend developer:

Si entendemos que con "front" nos referimos a lo que puede observar un usuario, el frontend developer es quien se encarga de desarrollar la interfaz gráfica que interactúa con los usuarios de un sitio web.

Backend developer:

Para entender este rol, podemos asociar una analogía con el teatro, en donde los actores que interpretan una obra no están solos, detrás de ellos se encuentran personas que cooperan con que el espectáculo funcione cómo se espera. Volviendo con el desarrollo web, las interfaces gráficas necesitan comunicarse con servidores que puedan recibir los datos recibidos por los usuarios y emitir distintas operaciones que aunque no son visuales o gráficas, son parte fundamental de un sistema web moderno.

Full Stack developer:

Ahora que entendemos que es un Frontend y un Backend developer, ¿En donde encaja un Full Stack Developer?, para entenderlo podemos ocupar otra analogía, en esta ocasión

viajamos al mundo de la construcción en donde tenemos ingenieros, obreros y arquitectos por solo mencionar algunos, cada quien es especialista en su área, sin embargo para lograr un acople sano y deseado, es ideal que exista un perfil que aunque no sea especialista en los distintos roles, tenga conocimientos en todas las áreas y coopere con la comunicación efectiva y técnica de las piezas de rompecabezas, a este perfil le podemos llamar entonces un Full Stack Developer. Es importante entender que con el tiempo este rol puede colaborar e incluso convertirse en especialista en una o más áreas sin dejar de ser un Full Stack, por este motivo ha destacado tanto en los últimos años y ahora es un perfil reconocido a nivel mundial.

¿Qué es HTML?

Sigla que proviene del inglés ***HyperText Markup Language*****,** podemos traducir HTML como un **lenguaje de marcas de hipertexto**, que sirve para la elaboración de páginas web.

Hipertexto, *"hace referencia a los enlaces que conectan páginas web entre sí, ya sea dentro de un único sitio web o entre sitios web. Los enlaces son un aspecto fundamental de la Web. Al subir contenido a Internet y vincularlo a las páginas creadas por otras personas, te conviertes en un participante activo en la «World Wide Web» (Red Informática Mundial)"*. (MDN Web docs, 2021)

HTML es un lenguaje de marcado que se utiliza para estructurar y escribir contenido, el cual luego es interpretado por un **navegador web**. Conocer este lenguaje y su alcance, es el primer paso para empezar a comprender cómo crear sitios que se adapten a las necesidades del mercado, con **buenas prácticas de programación** y una visión completa del proceso de desarrollo.

Estructura base de un documento HTML

HTML se organiza en base a **etiquetas**, que son los elementos con los que puedes **dar formato y estructura a un archivo HTML**. Cada una de ellas apunta a diferentes tipos de elementos, además, cada una tiene un formato determinado por defecto (que revisaremos más adelante). A continuación, te mostramos un ejemplo:

```
<p> Hola </p> <!-- esto es una etiqueta! -->
```

Por lo general, la estructura, o **sintaxis** de una etiqueta es la siguiente:

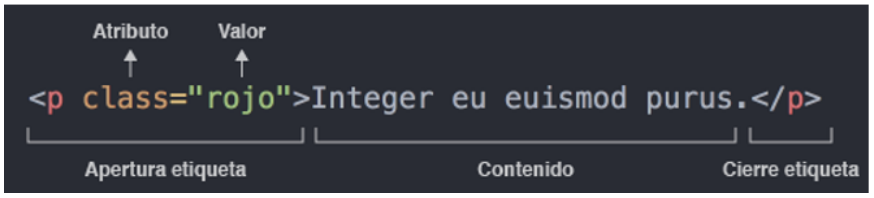

Imagen 1. Sintaxis de etiquetas.
Fuente: Desafío Latam.

Como puedes observar en la imagen, una etiqueta cuenta con una apertura, contenido y cierre.

Cada etiqueta puede contener ciertos **atributos** junto a su valor, por ejemplo, a la etiqueta <p> creada anteriormente, se le da un **atributo** class y a ese atributo le asignaremos su **valor**, el valor puede ser un nombre que definamos o alguna palabra específica que definirá el comportamiento de la etiqueta, en este caso el valor es "rojo".

Otro ejemplos de etiquetas:

```
<div>Hola soy una etiqueta Div </div>
<button> Hola soy una etiqueta button </button>
```

En este caso definiremos un nombre "div-contenedor" para así reconocer este por el nombre de su clase (class):

```
<div class="div-contenedor">Hola soy una etiqueta Div </div>
```

Podemos agregar más atributos como id, type, alt, src, entre otros.

Existen muchas etiquetas y estas pueden estar una dentro de otras, como lo que verás a continuación, ejemplificado en la estructura base de un documento HTML. Para conocer más en detalle la sintaxis de una etiqueta, revisa el siguiente enlace.

La estructura básica de un archivo HTML consiste principalmente en una cabeza (<head>) y un cuerpo (<body>), la cual, conoceremos a partir de la siguiente imagen:

```
<!DOCTYPE html>
<html>
<head>
   <!-- Aquí va la información para el navegador -->
```

```
</head>
<body>
   <!-- Aquí va el contenido para el usuario -->
</body>
</html>
```

La primera **etiqueta** `<!DOCTYPE html>` es la que le indica al navegador cómo debe interpretar el resto del documento (el doctype especificado es de HTML5, el cual es el estándar de hoy).

Luego, el `<head>` contiene toda la información que es para el navegador y el `<body>` contiene todo el contenido que es para el usuario.

Finalmente, la etiqueta `<html>` especifica que todo lo que está dentro de ella deberá ser interpretado como HTML.

Ahora que conoces los conceptos iniciales de HTML, puedes comenzar a trabajar en tu primera página web.

Antes de empezar, ¿Qué herramientas necesitas?

Para desarrollar tu primera página web, necesitas que tu computador cuente con **Un navegador web,** que es un programa diseñado para acceder y navegar por la web y a través de la interpretación de los archivos, visualizar las páginas. En esta oportunidad utilizamos Google Chrome y lo puedes descargar acá.

Además, un **editor de texto,** que es un programa que se instala en el computador y que ocuparemos para escribir el lenguaje que luego interpretará el navegador. Existen muchos editores y los más populares hasta ahora son **Visual Studio Code, Vim, VSC y Sublime Text**. Para este módulo, se utilizará Visual Studio Code, el cual puedes descargar acá. Este es un editor de texto multiplataforma (Windows, Linux y Mac), compatible con diversos lenguajes, por ejemplo: HTML, CSS, Less,C/C++, C#, Java, Python, PHP, Ruby, etc.

Una de las ventajas de utilizar editores de texto orientados al desarrollo, como lo es VSC, es que posee herramientas que facilitan las tareas recurrentes, como por ejemplo escribir una y otra vez la estructura base de un documento HTML. Esta función se llama **autocompletado** y, para utilizarla, primero debes asegurarte de que el tipo de código que estás escribiendo sea HTML, y eso se ve en la esquina inferior derecha del editor.

Imagen 2. Tipo de documento en VSC.
Fuente: Desafío Latam.

Luego, simplemente tienes que escribir html:5 y apretar la tecla **Tab**.

```
index.html > ...
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     <title>Document</title>
8   </head>
9   <body>
10
11  </body>
12  </html>
```

Imagen 3. Estructura HTML generada en VSC.
Fuente: Desafío Latam.

Para profundizar y conocer más acerca de las funciones de VSC, te invitamos a revisar su documentación oficial, a la cual puedes acceder a través del siguiente enlace.

Estructura de Assets

Antes de iniciar el desarrollo de tu primera página web, es necesario conocer cómo estructurar las carpetas y archivos que la componen, esto permite que los encuentres de forma más eficiente.

La estructura depende del alcance del proyecto, pero existe una convención que propone un modelo basado en la organización por tipos de archivos. En este modelo, los archivos HTML se guardarán en el directorio principal, mientras que los recursos adicionales (vídeos, hojas de estilo, imágenes, etc.) se almacenarán en subcarpetas dentro de una carpeta común llamada “assets”, como se grafica en la imagen:

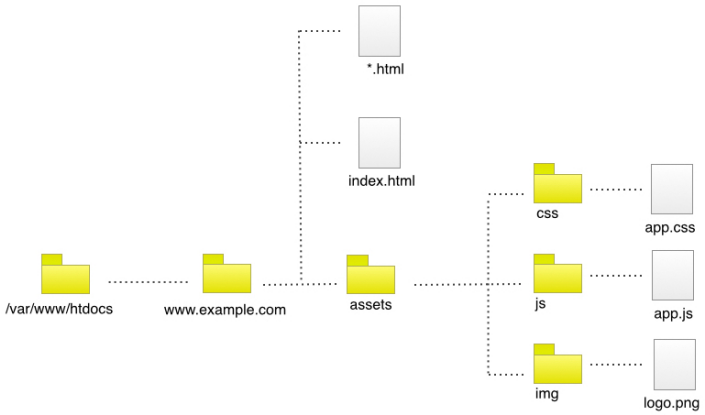

Imagen 4. Estructura de carpetas.
Fuente: Blog EA Mexicano.

La carpeta “assets” corresponde entonces a una convención que indica dónde se deben almacenar los archivos adicionales de nuestro proyecto, como hojas de estilo (css), Javascript (js) o imágenes (img).

Antes de continuar:

¿Tienes instalado Google Chrome o algún otro navegador de internet en tu computador? ¿Instalaste Visual Studio Code?

Si tu respuesta a ambas preguntas es sí, puedes seguir adelante, si alguna es NO, asegúrate de tener ambos elementos instalados.

Ejercicio guiado: Mi primera página web

A continuación, te presentamos los primeros pasos para crear nuestro primer documento HTML, el cual será renderizado por el navegador y representará nuestra primera página web. Sigue cada uno de los pasos que te presentamos. Vuelve atrás las veces que sea necesario.

- ***Paso 1:*** Crea una nueva carpeta en nuestro escritorio, que llamarás "Mi primera página web":

- ***Paso 2:*** Abre la carpeta con el editor de texto. Hay varias formas de lograr esto:

    - La primera y más simple, es arrastrar la nueva carpeta sobre el ícono de VSC.

- Otra forma, sería arrastrar la carpeta directamente sobre una pestaña abierta de VSC.
- También, puedes abrir VSC, ir a "File / Open File" y buscar la carpeta.

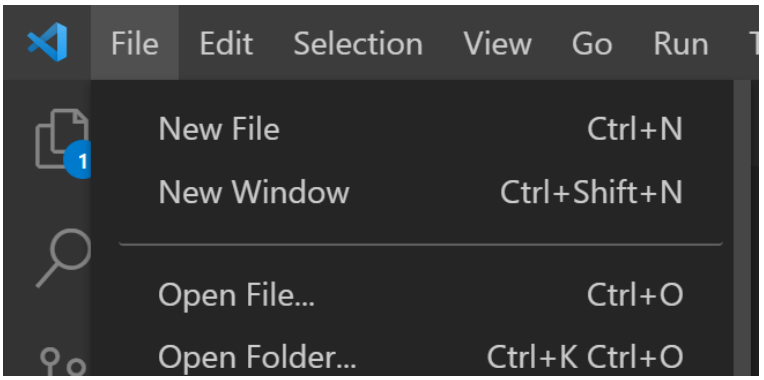

Imagen 5. Abriendo un archivo en VSC.
Fuente: Desafío Latam.

Sabrás que lo has logrado si puedes ver la carpeta al costado izquierdo del editor, como se observa en la siguiente imagen:

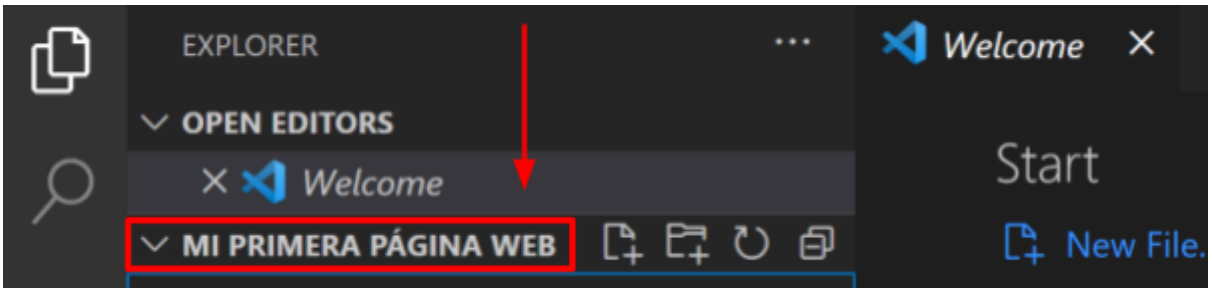

Imagen 6. Selección de carpeta en VSC.
Fuente: Desafío Latam.

- ***Paso 3:*** Ahora puedes generar tu primer archivo en VSC. Para ello, acerca el mouse al nombre de la carpeta dentro del editor y selecciona el icono **new file**, con esto se abrirá un input donde podrás escribir el nombre del archivo, al cual llamarás **index.html.** Una vez creado el archivo, aparecerá al costado izquierdo dentro de la carpeta del proyecto.

La página principal del sitio se debe llamar **index.html** porque es una convención que subentiende que es el archivo índice que inicia un sitio web. El que sea .html, dice que es un archivo que se interpretará como HTML, por lo tanto, podrá ser leído por el navegador.

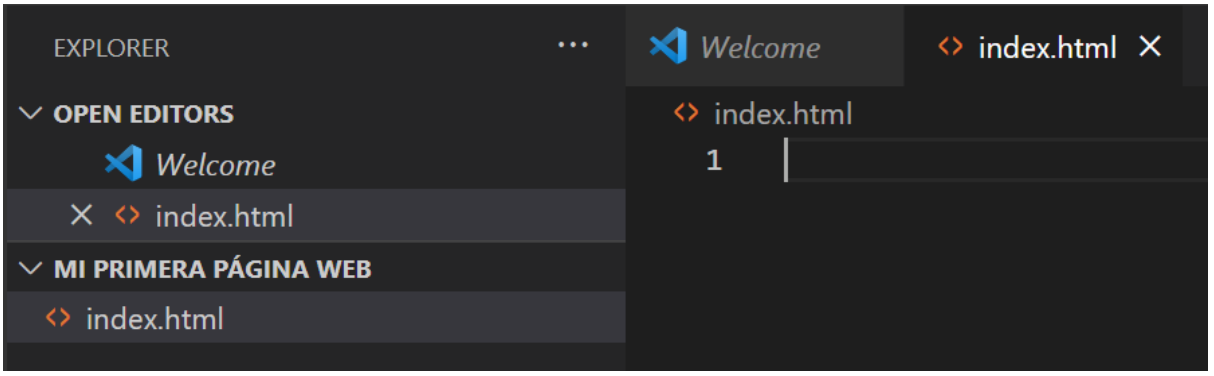

Imagen 7. Creación del archivo index.html.
Fuente: Desafío Latam.

- ***Paso 4:*** Si necesitas cambiarle el nombre, puedes hacer click con el botón secundario (clic derecho) sobre el archivo y seleccionar **rename** en la barra de navegación o bien presionar F2, luego podremos escribir el nuevo nombre directamente sobre el archivo para renombrarlo, tal como se observa en la siguiente imagen:

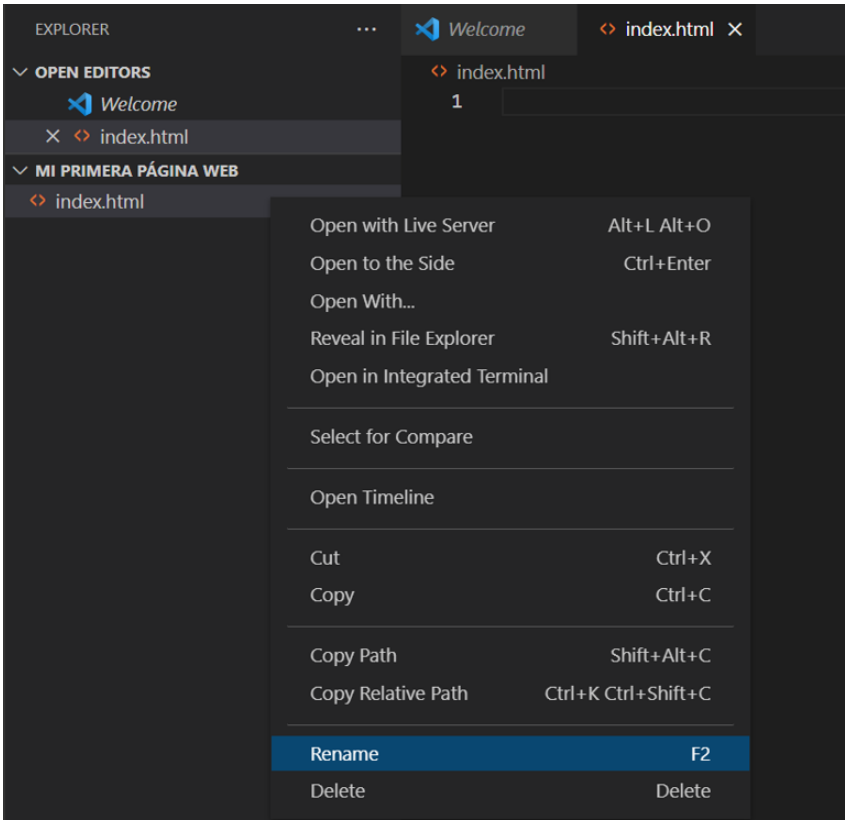

Imagen 8. Renombrar archivo.
Fuente: Desafío Latam.

- ***Paso 5:*** Cada vez que realices un cambio en el archivo, el editor VSC, te avisará dejando un punto de color al lado del nombre del archivo, el cual será el *"punto de la vergüenza"* indicándonos que los cambios no están guardados.

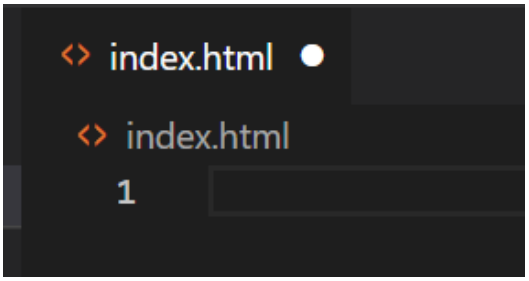

Imagen 9. Archivo sin editar.
Fuente: Desafío Latam.

Para guardar, presiona `cmd + s` en Mac y `ctrl + s` en Windows o Linux.

También puedes hacerlo sin utilizar atajos dentro del menú, haciendo clic sobre file y luego sobre guardar (o save).

Estarás guardando constantemente, así que es importante recordar este atajo. Una vez que esté guardado el archivo, ya podrás abrirlo en el navegador.

- ***Paso 6:*** Si no lo guardas, solo verás un archivo vacío dentro del navegador. Prueba ingresando texto en tu archivo y guardándolo.

```
Hola esta es mi primera página web
```

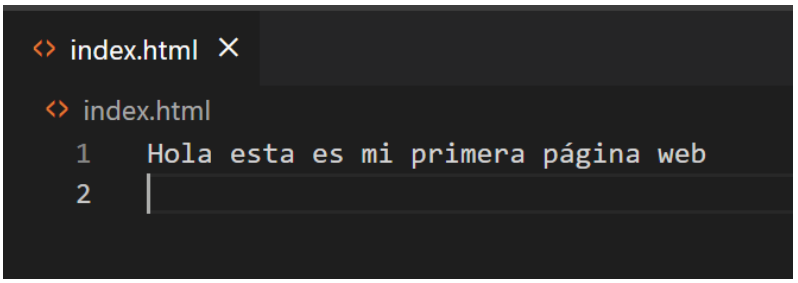

Imagen 10. Agregando texto al archivo HTML.
Fuente: Desafío Latam.

- ***Paso 7:*** Ahora prueba cómo se ve en el navegador. Para eso, abre el archivo con Chrome y podrás ver el texto que escribiste. El archivo lo puedes encontrar en la carpeta que creaste inicialmente en el escritorio, como se observa en la imagen:

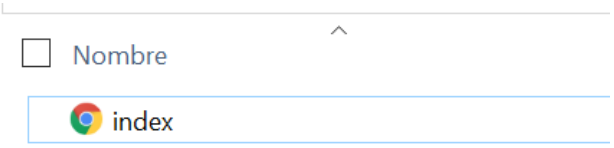

Imagen 11. Archivo creado.
Fuente: Desafío Latam.

Si no guardas el archivo, al recargar el navegador no verás los cambios realizados. De igual forma para recargar la página del navegador, puedes hacer clic en el ícono de recargar o utilizando el atajo `cmd + r` en Mac y `ctrl + r` en Linux o Windows. Para ver los cambios debemos actualizar la página.

Imagen 12. Ver archivo en el navegador.
Fuente: Desafío Latam.

- ***Paso 8:*** Finalmente, crearemos la estructura base en la cual trabajaremos el resto del proyecto, la cual debe configurarse de la siguiente manera:

    - `/assets/img` para las imágenes.
    - `/assets/css` para el archivo CSS.

Puedes hacerlo directamente desde la carpeta "Mi primera página web" o desde VSC, presionando con el botón derecho en la carpeta, de la siguiente manera:

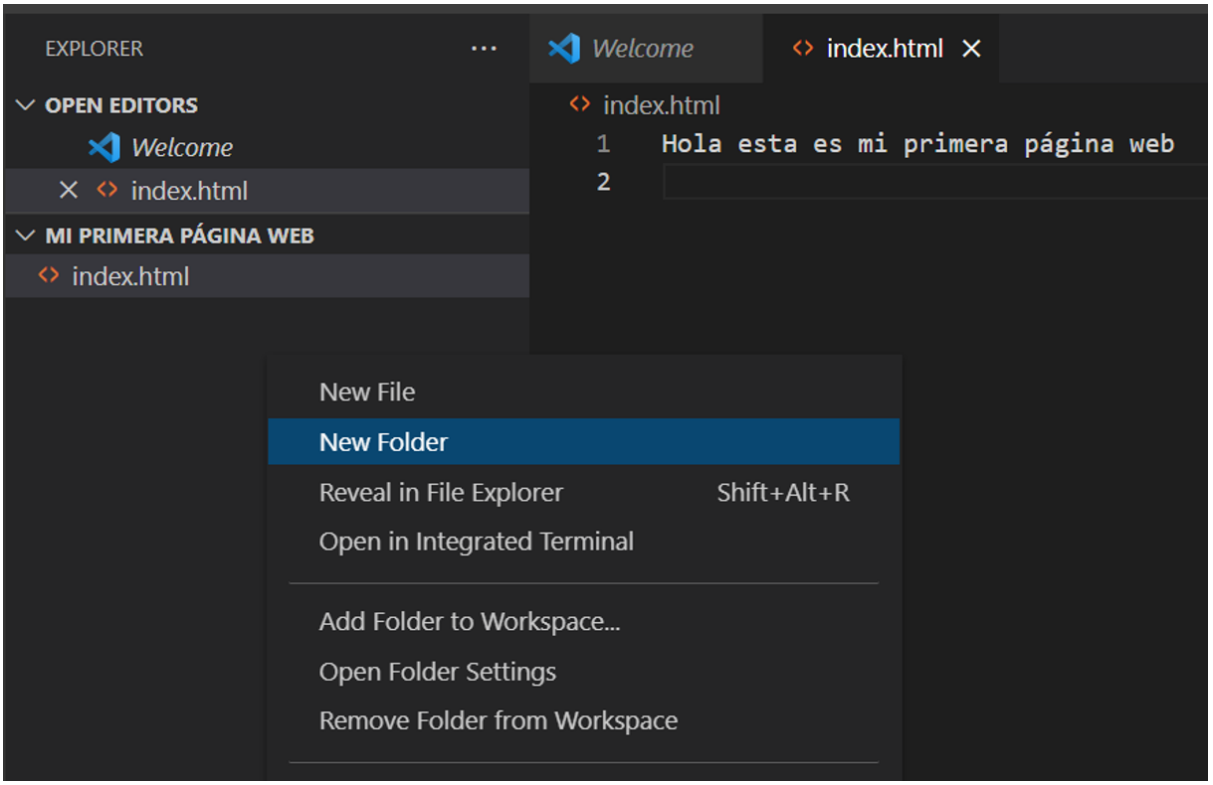

Imagen 13. Crear la estructura de carpetas.
Fuente: Desafío Latam.

Debe quedar la siguiente estructura en nuestro proyecto:

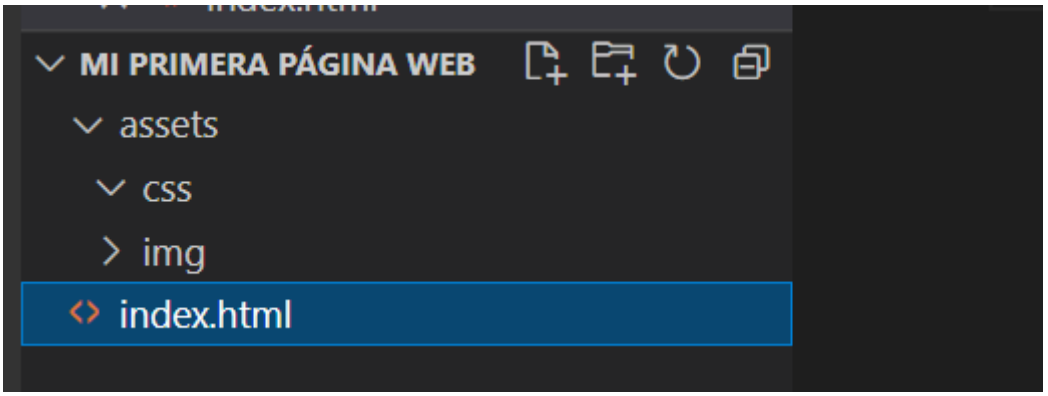

Imagen 14. Estructura de carpetas.
Fuente: Desafío Latam.

Como buena práctica, no dejaremos las imágenes y otros archivos dentro de la raíz de la carpeta, sino crearemos subdirectorios. ¿Por qué? Porque a medida que el proyecto va creciendo puede llegar a ser difícil mantener el orden, especialmente si trabajamos todos los archivos en la raíz del proyecto, por lo tanto, es importante clasificarlos.

¡Felicitaciones! Acabamos de crear nuestro primer proyecto web. En las próximas lecturas iremos agregando código HTML para darle estructura a nuestra página y agregar elementos de texto e imágenes.

Resumen

- El **editor de texto,** que es un programa que se instala en el computador y que ocuparemos para escribir el lenguaje que luego interpretará el navegador.

- **HTML** es un lenguaje de marcado que se utiliza para estructurar y escribir contenido, el cual luego es interpretado por un **navegador web**.

- La estructura básica de un archivo HTML consiste en una cabeza (`<head>`) y un cuerpo (`<body>`).
- El `<head>` contiene toda la información que es para el navegador y el `<body>` contiene todo el contenido que es para el usuario.

- La carpeta Assets indica dónde se deben almacenar los archivos adicionales de nuestro proyecto, como hojas de estilo (css), Javascript (js) o imágenes (img).