

JavaScript (Parte II)

Integrando jQuery

Competencias

- Implementar adecuadamente la librería jQuery en un proyecto web, utilizando recursos locales y externos.

Introducción

Hasta ahora hemos experimentado con JavaScript y pudimos hacer un par de cosas interesantes, como interactuar con el usuario a través de ventanas emergentes y realizar operaciones matemáticas.

Estas competencias son la base para lo que abordaremos en este capítulo, donde nos centraremos en la librería jQuery, que viene a simplificar la tarea de programar en JavaScript y permite agregar interactividad a un sitio web, optimizando el tiempo y la legibilidad del código.

Integrando jQuery

Anteriormente conocimos un poco sobre los conceptos básicos de JavaScript. Desde ahora comenzaremos a trabajar con jQuery.

¿Qué es jQuery?



Imagen 1. Logo jQuery.

Fuente: [jQuery Brand Guidelines](#).

jQuery es una biblioteca de JavaScript que nos permite escribir menos código y obtener los mismos resultados que con JavaScript puro.

También hace más sencilla la captura y manipulación de los diferentes elementos de un sitio. Esto lo realiza encapsulando el código JavaScript en métodos más cortos y simples.

Además, jQuery es software libre y de código abierto, bajo una licencia MIT y GNU que permite su uso privado y comercial.

Características

La biblioteca de jQuery tiene las siguientes características:

- Manipulación del DOM/HTML.
- Manipulación de CSS.
- Métodos de eventos para HTML.
- Efectos y animaciones.
- Entre otras.

Formas de integrar jQuery

Para integrar jQuery a un proyecto tenemos dos opciones:

1. Descargando el archivo:

- **Paso 1:** Bajar la biblioteca desde [jQuery](#), para esto descargamos el archivo jQuery, haciendo clic con el botón derecho sobre el link de descarga:

Downloading jQuery

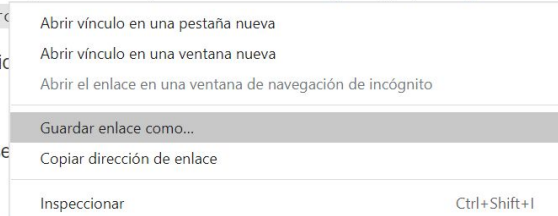
Compressed and uncompressed copies of jQuery files are available. The uncompressed file is best used during development or debugging; the compressed file saves bandwidth and improves performance in production. You can also download a [sourcemap file](#) for use when debugging with a compressed file. The map file is *not* required for users to run jQuery, it just improves the developer's debugger experience. As of jQuery 1.11.0/2.1.0 the `//# source`

To locally download these files, right-click

jQuery

For help when upgrading jQuery, please see the [jQuery Upgrade](#) [plugin](#).

Download the compressed, production-ready jQuery 3.5.1



recommend using the [jQuery Migrate](#)

Imagen 2. Descarga jQuery.

Fuente: [jQuery.com](#).

- **Paso 2:** Añadir el archivo dentro del directorio de nuestro proyecto:

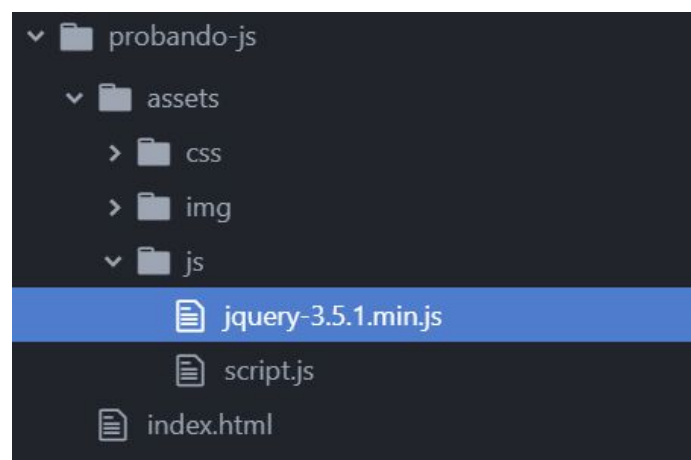


Imagen 3. Descarga jQuery.

Fuente: [jQuery.com](#).

- **Paso 3:** Llamarlo desde la etiqueta `<script>` con el atributo `src`. Tal como llamamos a un archivo `.js`, ya que toda la biblioteca de jQuery está escrita en JavaScript:

```
<!DOCTYPE html>
<html lang="es" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Probando JS</title>
    <script src="assets/js/jquery-3.5.1.min.js"></script>
  </head>
  <body>

  </body>
</html>
```

2. CDN.

La otra forma, y la que utilizaremos más a menudo, es usar un CDN desde [Code jQuery](#).

- **Paso 1:** Visitamos la página de [Code jQuery](#) y escogemos la versión a utilizar:

jQuery CDN – Latest Stable Versions

Powered by  **StackPath**

jQuery Core

Showing the latest stable release in each major branch. [See all versions of jQuery Core](#).

jQuery 3.x

- jQuery Core 3.5.1 - [uncompressed](#), [minified](#), [slim](#), [slim minified](#)

jQuery 2.x

- jQuery Core 2.2.4 - [uncompressed](#), [minified](#)

jQuery 1.x

- jQuery Core 1.12.4 - [uncompressed](#), [minified](#)

Imagen 4. Code jQuery.

Fuente: [jQuery CDN](#).

- **Paso 2:** Se debe agregar la referencia en la etiqueta `<script>` en nuestro HTML:

```
<!DOCTYPE html>
<html lang="es" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Probando JS</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"><
  /script>
</head>
<body>

  </body>
</html>
```

Sintaxis jQuery

La sintaxis de jQuery está hecha para seleccionar elementos HTML y realizar alguna acción en el o los elementos seleccionados.

Su sintaxis básica es la siguiente:

```
$(selector).accion()
```

Es una buena práctica esperar a que el documento esté completamente cargado y listo antes de trabajar con él, es por eso que existe el Document Ready Event. Dentro de esa función debemos escribir todas las acciones que necesitemos ejecutar.

A través de las nuevas versiones esta función se ha simplificado hasta quedar así:

```
$(function(){

  // métodos de jQuery...

});
```

Más adelante profundizaremos más en la sintaxis de jQuery a través de las distintas acciones que podemos hacer.

Selectores

Los selectores jQuery nos permiten seleccionar y manipular elementos HTML.

Los selectores jQuery se utilizan para "buscar" (o seleccionar) elementos HTML según su nombre, id, clases, tipos, atributos, valores de atributos y mucho más. Se basa en los selectores de CSS existentes y, además, tiene algunos selectores personalizados.

Todos los selectores en jQuery comienzan con el signo `$` y siguen con los paréntesis: `$()`.

Tipos de selectores		
• Selector universal.	• Selector de elemento actual.	• Selector por etiqueta.
• Selector por id.	• Selector por clase.	• Selector por atributo.
• Selector por valor de atributo.	• Selector por número de elemento.	• Selector por elemento par.
• Selector por elemento impar.	• Entre muchos otros.	

Tabla 1. Tipos de selectores.
Fuente: Desafío Latam.

Puedes ver el [selector tester de W3Schools](#), donde podrás conocer mejor a los selectores.

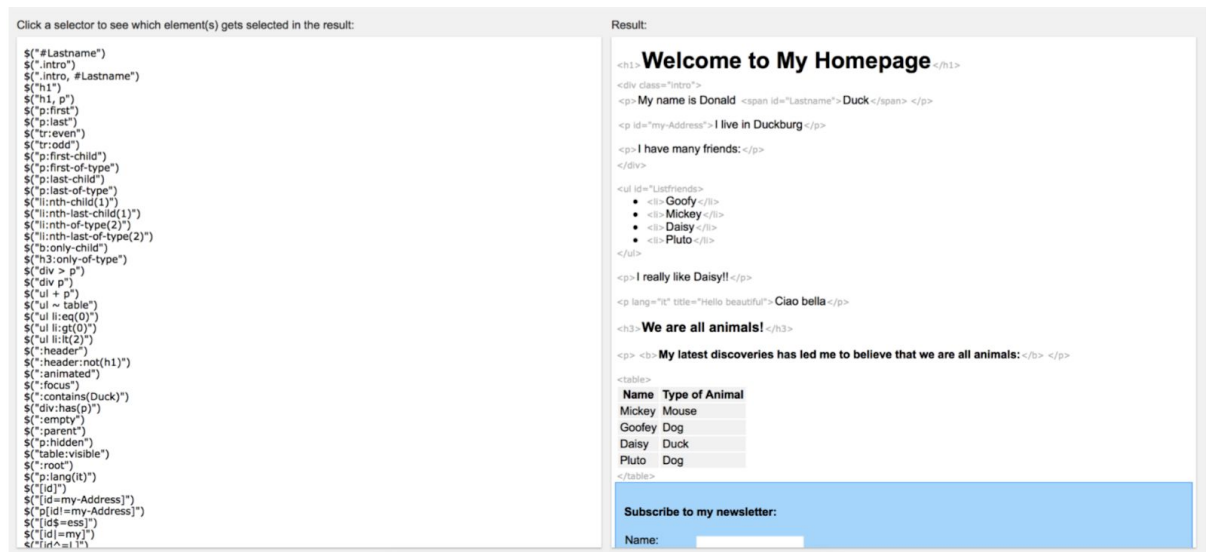


Imagen 5. Sección de Selectores.

Fuente: [Try jQuery Selector](#).

Además de revisar esta [referencia](#), también de W3Schools donde enlistan los distintos selectores.

Consultando las referencias se te pueden hacer un poco más cercanos los selectores. Lo importante es que selecciones bien el elemento al que le quieres dar la acción o atribuirle un método.

Eventos y Funciones

¿Qué son los eventos?

Las diferentes acciones que los visitantes pueden hacer dentro de una página web se denominan eventos y representa el momento preciso en que algo sucede.

Ejemplos:

- Cuando el usuario clickea un elemento.
- Cuando el usuario presiona una tecla.
- Cuando el usuario hace scroll.
- Cuando el usuario cambia el tamaño de la ventana de visualización del navegador.

Aquí te presentamos algunos ejemplos de los eventos más usados:

Mouse	Teclado	Formularios	Documento/Ventana
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload
hover			

Tabla 2. Eventos.
Fuente: Desafío Latam.

Sintaxis de los eventos

La sintaxis es la siguiente:

```
$(selector).evento(function(){  
  // La acción ocurre aquí  
});
```

Como vemos es necesario pasarle una función al evento, que se ejecutará cuando el evento suceda. Ejemplo:

```
$("p").dblclick(function(){  
  $(this).hide();  
});
```

Por ejemplo, en el siguiente código, jQuery hará que los elementos <p> se oculten, si el usuario les da doble click:

```
<!DOCTYPE html>  
<html>  
<head>  
<script  
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"><  
</script>  
<script>  
$(document).ready(function(){  
  $("p").dblclick(function(){  
    $(this).hide();  
  });  
});  
</script>  
</head>  
<body>  
<p>If you double-click on me, I will disappear.</p>  
<p>Click me away!</p>  
<p>Click me too!</p>  
</body>  
</html>
```

Puedes ejecutar este código tú mismo, desde el [editor de W3Schools](#).

Métodos

Los métodos son la acción a la que se ve enfrentada el selector y son diferentes a los eventos.

```
$(selector).accion()
```

Un evento es el resultado de una acción realizada por el usuario, como hacer click, desplazar, arrastrar, redimensionar, etc.

En cambio un método es la acción a la que se ve enfrentado. Éstos métodos pueden ser concatenados y afectar al selector. Muchas veces estos métodos son activados cuando un evento ocurre (como lo que hicimos en el ejercicio anterior).

Existen muchos métodos de jQuery, y podemos dividirlos en los siguientes tipos:

- **Métodos de efectos:** Proporcionan varias técnicas para agregar animación a una página web, incluyendo efectos simples y personalizados.
- **Métodos de manipulación de HTML y CSS:** Manipulan los elementos, cambiando sus atributos, estilos y/o propiedades.

Veamos algunos de ellos:

Efectos

- `animate();`: Realiza una animación personalizada de un conjunto de propiedades CSS.
- `delay();`: Configura un temporizador para retrasar la ejecución de los elementos siguientes en la cola.
- `fadeIn();`: Muestra los elementos coincidentes, atenuándolos a opacos.
- `fadeOut();`: Oculta los elementos coincidentes, atenuándolos a transparentes.
- `fadeTo();`: Ajusta la opacidad de los elementos seleccionados.

- `fadeToggle()`; Muestra u oculta los elementos seleccionados, animando su opacidad
- `finish()`; Detiene la animación que se está ejecutando, elimina todas las animaciones en cola y completa todas las animaciones para los elementos seleccionados.
- `hide()`; Oculta los elementos coincidentes
- `show()`; Muestra los elementos seleccionados.
- `stop()`; Detiene la animación que se está ejecutando actualmente en los elementos coincidentes.
- `toggle()`; Muestra u oculta los elementos coincidentes.

Para ver la lista completa de efectos, puedes consultar la [documentación de jQuery](#):

Manipulación de HTML y CSS

- `addClass()`; Agrega la clase especificada a cada elemento seleccionado.
- `after()`; Inserta contenido, especificado por parámetro, después de cada elemento seleccionado.
- `append()`; Inserta contenido, especificado por parámetro, al final de cada elemento seleccionado.
- `appendTo()`; Inserta todos los elementos seleccionados hasta el final del objetivo definido por parámetro.
- `attr()`; Obtiene el valor de un atributo para el primer elemento seleccionado o establece uno o más atributos para cada elemento coincidente.
- `before()`; Inserta contenido, especificado por parámetro, antes de cada elemento seleccionado.
- `clone()`; Crea una copia profunda del conjunto de elementos coincidentes.

- **css()**;: Obtiene el valor de una propiedad de estilo para el primer elemento seleccionado o establece una o más propiedades CSS para cada elemento seleccionado.
- **detach()**;: Elimina el conjunto de elementos coincidentes del DOM.
- **empty()**;: Elimina todos los nodos secundarios del conjunto de elementos seleccionados del DOM.
- **hasClass()**;: Determine si alguno de los elementos seleccionados posee la clase dada.
- **height()**;: Obtiene la altura calculada actual para el primer elemento del conjunto de elementos seleccionados o establece la altura de cada elemento coincidente.
- **html()**;: Obtiene el contenido HTML del primer elemento del conjunto de elementos seleccionados o establece el contenido HTML de cada elemento coincidente.
- **remove()**;: Elimina el conjunto de elementos seleccionados del DOM.
- **removeAttr()**;: Elimina un atributo de cada elemento del conjunto de elementos seleccionados.
- **removeClass()**;: Elimina una sola clase, varias clases o todas las clases de cada elemento en el conjunto de elementos seleccionados.
- **text()**;: Obtiene el contenido de texto combinado de cada elemento en el conjunto de elementos seleccionados, incluidos sus descendientes, o establece el contenido de texto de los elementos coincidentes.
- **toggleClass()**;: Agrega o elimina una o más clases de cada elemento en el conjunto de elementos seleccionados.
- **val()**;: Obtiene el valor actual del primer elemento en el conjunto de elementos seleccionados o establece el valor de cada elemento coincidente.
- **width()**;: Obtiene el ancho calculado actual para el primer elemento en el conjunto de elementos coincidentes o establezca el ancho de cada elemento coincidente.

Para ver la lista completa de métodos, puedes consultar la [documentación de jQuery](#).

Ejercicio guiado: Ejercitando con jQuery

Contexto

Al igual que hicimos con JavaScript puro, es importante que practiquemos de forma aislada la sintaxis y algunos métodos de jQuery, para entender su alcance y experimentar la utilidad que nos ofrecen.

Para esto, vamos a probar algunos métodos y eventos de jQuery.

- **Paso 1:** Crearemos una carpeta llamada probando-jquery con la estructura que ya conocemos:

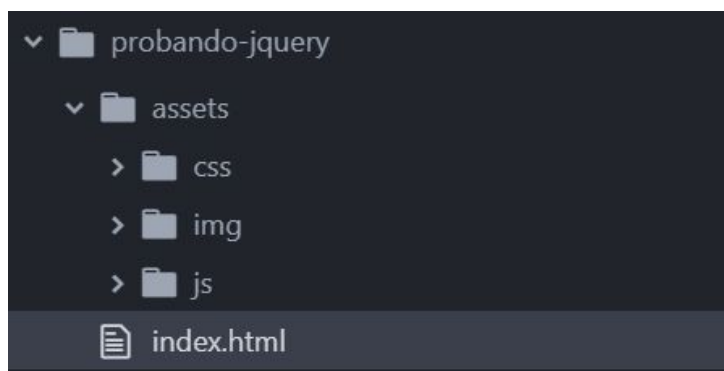


Imagen 6. Probando jQuery.
Fuente: Desafío Latam.

- **Paso 2:** Vamos a integrar jQuery por CDN, para eso, en la estructura del HTML agregaremos un tag `<script>` para agregar al enlace que podemos obtener [aquí](#):

```
<!DOCTYPE html>
<html lang="es" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Probando jQuery</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
    </script>
  </head>
  <body>
  </body>
</html>
```

Esto nos permitirá integrar elementos de jQuery en nuestro archivo. Probemos algunos métodos.

- **Paso 3:** Copiamos el siguiente código en el documento index.html y lo ejecutamos en el navegador:

```
<!DOCTYPE html>
<html lang="es" dir="ltr">
<head>
  <meta charset="utf-8">
  <title>Probando jQuery</title>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"><
/script>
  <script>
$(document).ready(function(){
  $("p").dblclick(function(){
    $(this).hide();
  });
});
</script>
</head>
<body>
  <p>Si me clickeas dos veces, desapareceré.</p>
  <p>¡Yo también!</p>
  <p>¡Yo también!</p>
</body>
</html>
```

- **Paso 4:** Observemos el comportamiento. Si activamos el evento `dblclick()` en cualquiera de los textos, éstos activarán el efecto `hide()` y desaparecerán de manera independiente:



Imagen 7. Primera ejecución con jQuery.
Fuente: Desafío Latam.

- **Paso 5:** Si buscamos la [documentación](#) de `hide()`, observaremos que además puede recibir parámetros para controlar su comportamiento, como la duración, transición, etc. Probemos controlando la duración de la animación, con el parámetro `'slow'`

El script quedará de la siguiente manera:

```
<script>
$(document).ready(function(){
  $("p").dblclick(function(){
    $(this).hide("slow");
  });
});
</script>
```

Guardamos y observamos el resultado.

Otra opción para controlar la duración de la animación es `"fast"`. Comprueba la diferencia.

- **Paso 6:** Ahora probemos algo diferente, modificaremos el CSS a través de jQuery. Haremos que cuando se haga un click sobre el texto, se aumente su tamaño y cambie a color rojo, de la siguiente manera:

```
<script>
$(document).ready(function(){
  $("p").dblclick(function(){
    $(this).hide('slow');
  });

  $("p").click(function(){
    $(this).css({
      "color": "red",
      "font-size": "2em"
    });
  });
});
</script>
```

Guardamos y observamos el resultado:



Imagen 8. Modificando el CSS.

Fuente: Desafío Latam.

- **Paso 7:** Además del formato, podemos cambiar el contenido del HTML. Por ejemplo, haremos que al pasar el mouse sobre cualquier párrafo, éste se transforme en un link y cambie su contenido:

```
<script>
$(document).ready(function(){
  $("p").dblclick(function(){
    $(this).hide('slow');
  });

  $("p").click(function(){
    $(this).css({
      "color": "red",
      "font-size": "2em"
    });
  });

  $("p").hover(function(){
    $(this).html("<a href='#'>Me convertí</a>");
  });
});
</script>
```


Guardamos y validamos el resultado:

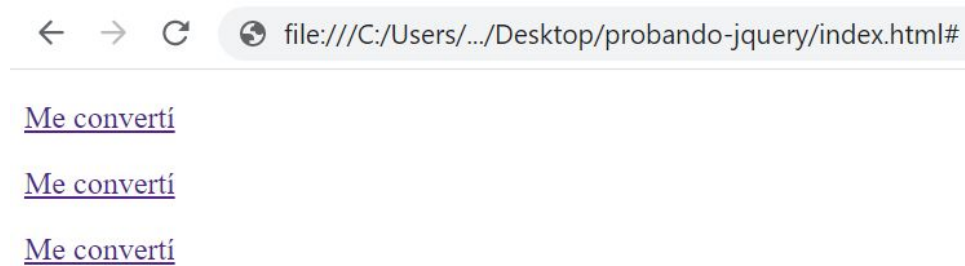


Imagen 9. Modificando el contenido.
Fuente: Desafío Latam.

Ejercicio propuesto (1)

Contexto

Aprender cómo utilizar los métodos, eventos y funciones a clave para utilizar jQuery correctamente y crear páginas web que reflejen el estilo que queremos imprimir en ellas.

A continuación te invitamos a responder cómo sería nuestro código de requerir lo siguiente:

1. Seleccionar las etiquetas "img".
2. Seleccionar los elementos con id="imagen".
3. Seleccionar los elementos con clase="centrado".
4. Seleccionar los elementos con id="imagen" con clase="centrado" y que son etiqueta "img".
5. Seleccionar las etiquetas "a" con clase="encabezado" dentro de una etiqueta "p".
6. Seleccionar los elementos con id="imagen" o clase="centrado".
7. Seleccionar todos los elementos.
8. Seleccionar los elementos con el atributo href.
9. Seleccionar todos los input de tipo text.

10. Un evento que al hacer click en un párrafo, imprime por consola "click".

11. Modifica el precio del texto para que imprima \$100.000:

```
<li class="vacaciones">  
  <h2>La Serena, Chile</h2>  
  <span class="valor">$120.000</span>  
</li>
```

12. Al código anterior, ponle color amarillo de fondo a todos los elementos de la clase "vacaciones".

Bootstrap JS

Competencias

- Implementar componentes JavaScript de Bootstrap, para agregar elementos visuales a un sitio web.

Introducción

En este capítulo retomaremos el framework Bootstrap, como herramienta para construir sitios web de forma más ágil y con recursos adicionales, como vimos anteriormente. Esta vez, añadiremos componentes que utilizan jQuery y nos permitirán añadir dinamismo y efectos interesantes a nuestros proyectos.

Es importante mencionar que a partir de la versión 5 de Bootstrap se elimina la dependencia con jQuery y se agrega VanillaJS.

Smooth Scroll

Se le denomina Smooth Scroll, al efecto de transición entre diferentes secciones de una misma página sin tener que recargar, lo que mejora la experiencia del usuario y el aspecto visual de nuestro sitio.

Este efecto se puede lograr tanto con CSS, como con jQuery. En el caso de CSS, existe una propiedad llamada **scroll-behavior**.

```
html {  
  scroll-behavior: smooth;  
}
```

Pero existen aún navegadores que no soportan esta propiedad, por lo que se sugiere utilizar otras alternativas, como JavaScript puro o su biblioteca jQuery:

CSSOM Scroll-behavior WD

Usage % of all users
Global 76.27%

Method of specifying the scrolling behavior for a scrolling box, when scrolling happens due to navigation or CSSOM scrolling APIs.

Current aligned Usage relative Date relative Filtered All

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Opera Mobile *	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet
			4-40		10-27								
	12-18	2-35	41-60		28-47								4-7.4
6-10	79-86	36-83	61-86	3.1-13.1	48-71	3.2-13.7		2.1-4.4.4	12-12.1				8.2-12.0
11	87	84	87	14	72	14.2	all	81	59	87	83	12.12	13.0
		85-86	88-90	TP									

Imagen 10. Compatibilidad scroll-behavior.

Fuente: [Can I Use](https://caniuse.com/scroll-behavior).

Nota: La página [Can I Use](https://caniuse.com/scroll-behavior) es muy utilizada para ver compatibilidad de navegadores con funcionalidades que queramos incluir en nuestros proyectos. Posee un navegador en la parte superior donde pondremos el nombre de la funcionalidad y nos mostrará una imagen como la anterior, con el detalle del navegador-versión-compatibilidad.

En el caso de jQuery, se deben tener en cuenta un par de consideraciones en el HTML. Veamos un ejemplo en [este link](#).

Si miramos el ejemplo de W3Schools, podemos identificar el código. Este se compone de dos secciones:

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<h1>Smooth Scroll</h1>
<div class="main" id="section1">
  <h2>Section 1</h2>
  <p>Click on the link to see the "smooth" scrolling effect.</p>
  <a href="#section2">Click Me to Smooth Scroll to Section 2 Below</a>
  <p>Note: Remove the scroll-behavior property to remove smooth
scrolling.</p>
</div>

<div class="main" id="section2">
  <h2>Section 2</h2>
  <a href="#section1">Click Me to Smooth Scroll to Section 1 Above</a>
</div>

</body>
</html>
```

Cada sección, posee un id:

```
<div class="main" id="section1">
<div class="main" id="section2">
```

Y cada uno de los links, refieren al id de la sección a la que queremos ir una vez que hagamos click sobre ellos:

```
<a href="#section2">
<a href="#section1">
```

En el CSS, sólo para graficar, se le está asignando un color de fondo y un alto definido:

```
<style>
#section1 {
  height: 600px;
  background-color: pink;
}

#section2 {
  height: 600px;
  background-color: yellow;
}
</style>
```

La magia se produce en el script de jQuery:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"><
/script>
<script>
$(document).ready(function(){
  // Add smooth scrolling to all links
  $("a").on('click', function(event) {

    // Make sure this.hash has a value before overriding default
    behavior
    if (this.hash !== "") {
      // Prevent default anchor click behavior
      event.preventDefault();

      // Store hash
      var hash = this.hash;

      // Using jQuery's animate() method to add smooth page scroll
      // The optional number (800) specifies the number of milliseconds
      it takes to scroll to the specified area
      $('html, body').animate({
        scrollTop: $(hash).offset().top
      }, 800, function(){
```

```
        // Add hash (#) to URL when done scrolling (default click
behavior)
        window.location.hash = hash;
    });
} // End if
});
});
</script>
```

- Al hacer click en el elemento `<a>` se valida que el valor del ancla (#) no sea vacía.
- Se ejecuta el método [preventDefault\(\)](#), que evita el comportamiento por defecto del elemento al hacer click sobre él.
- Asigna el valor del ancla a la variable hash.
- Se activa el método [animate\(\)](#), con la propiedad `scrolltop`, que obtiene el valor del borde superior del elemento a través del offset del mismo y una duración de la animación de 800 milisegundos.
- Finalmente, realiza la ejecución del scroll hacia la zona definida.

Bootstrap JS: Popover, Modal, Tooltip y Dropdown

El Framework Bootstrap, contiene muchos elementos visuales que funcionan con JavaScript, específicamente necesitan jQuery, Popper JS y su propio archivo .js (Bootstrap JS) para funcionar.

Cuando elegimos algún elemento de Bootstrap lo primero que tenemos que hacer es ir a su [documentación](#). Una vez dentro de ella debemos verificar que la versión sea la que utilizáramos, esto lo podemos verificar en la zona superior.

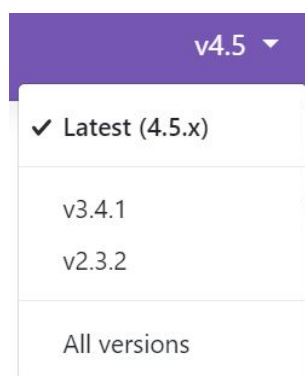


Imagen 11. Trabajando con bootstrap.

Fuente: [Bootstrap 4.5](#).

Siempre es recomendable copiar el código que aparece en la documentación ya que este se actualiza de forma constante, en base a la versión seleccionada.

Ahora conoceremos algunos elementos de Bootstrap.

Popover

Un popover es un elemento visual que se utiliza para mostrar un mensaje emergente, con una referencia específica a un tema.

Podremos verlo en acción en el siguiente enlace [documentación](#).

Si queremos añadirlo en nuestros proyectos solo debemos seguir las instrucciones.

```
$(function () {  
  $('[data-toggle="popover"]').popover()  
})
```


Esto añadirá un tooltip a cada elemento que tenga el atributo `data-toggle="popover"`.

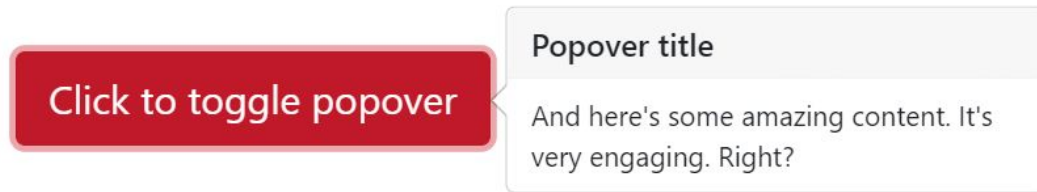


Imagen 12. Ejemplo tooltip.

Fuente: [Bootstrap](#).

Modal

Un modal es otro elemento importante que podemos utilizar gracias a las características de Bootstrap. Sirve para añadir diálogos o mostrar notificaciones.

Podemos ver su efecto en la siguiente [documentación](#).

Podremos observar su funcionamiento en la documentación de Bootstrap, en el botón Launch demo modal.

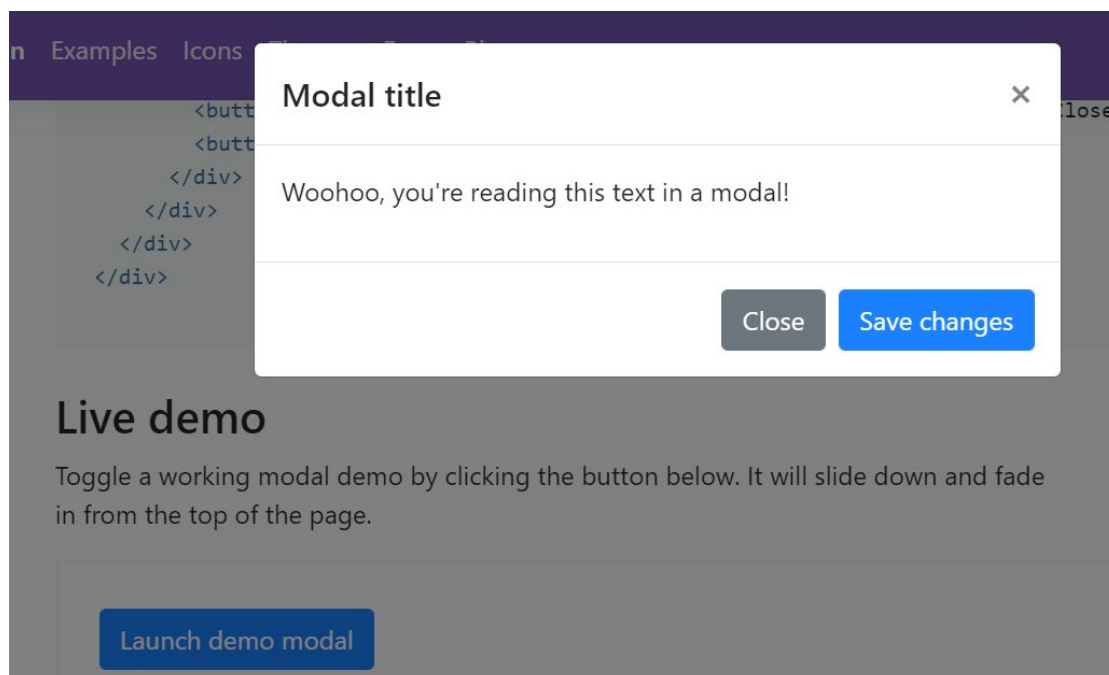


Imagen 13. Funcionamiento de Modal.

Fuente: [Bootstrap](#).

Tooltip

Un tooltip es una herramienta aparentemente similar al popover pero que tiene un concepto semántico diferente. El tooltip solo mostrará una ayuda cuando pasemos el puntero sobre la palabra o botón donde se haya añadido. Como su nombre lo dice, es una pista respecto al funcionamiento del elemento en particular.

Si accedes a su [documentación](#) podrás ver la forma de implementarlo en tu proyecto.

```
$(function () {  
  $('[data-toggle="tooltip"]').tooltip()  
})
```

Este código añade un tooltip a cada elemento que tenga el atributo `data-toggle="tooltip"`, entonces si lo añadimos dentro de algún elemento, por ejemplo un botón.

```
<button type="button" class="btn btn-secondary" data-toggle="tooltip"  
data-placement="top" title="Tooltip on top">  
  Tooltip on top  
</button>
```

Podremos ver el funcionamiento de esto en el ejemplo de la documentación, si presionamos el botón Click to toggle popover.

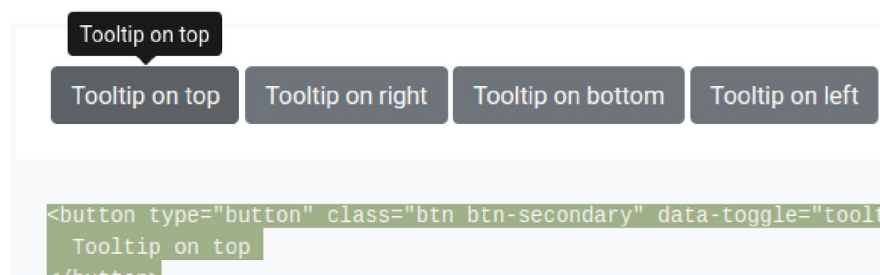


Imagen 14. Funcionamiento de tooltip.

Fuente: [Bootstrap](#).

Dropdown

De acuerdo a la [documentación](#), un dropdown servirá para mostrar un listado desplegable con opciones o links. Existen varias formas de utilizarlo. La más simple es la siguiente:

```
<div class="dropdown">
  <button class="btn btn-secondary dropdown-toggle" type="button"
  id="dropdownMenuButton" data-toggle="dropdown" aria-haspopup="true"
  aria-expanded="false">
    Dropdown button
  </button>
  <div class="dropdown-menu" aria-labelledby="dropdownMenuButton">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <a class="dropdown-item" href="#">Something else here</a>
  </div>
</div>
```

Lo que incluirá un dropdown button que mostrará los ítems (Action, Another action, Something else here).

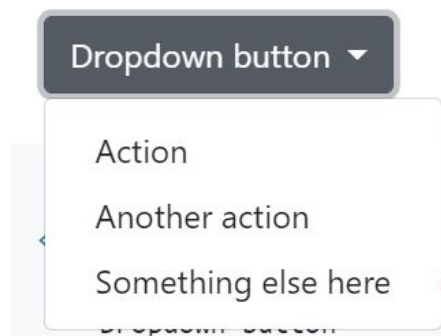


Imagen 15. Funcionamiento de dropdown.

Fuente: [Bootstrap](#).

Basta solo implementar este código y nuestro Dropdown, comenzará a funcionar, ya que nuestro Bootstrap, utiliza un llamado vía Data attributes. De todas formas, también podemos llamarlo desde JS añadiendo lo siguiente en nuestro archivo de configuración.

```
$('.dropdown-toggle').dropdown()
```

Lo que asigna, la función dropdown con la convención data-toogle. Si lo añadimos de esta forma, tendremos más control sobre su funcionamiento.

Carousel de Bootstrap

Un carousel es un componente de presentación de diapositivas para recorrer los elementos (imágenes o diapositivas de texto). Para hacerlo iremos a la [documentación](#).

En el siguiente ejemplo, se muestra un carousel de 3 diapositivas:

```
<div id="carouselExampleSlidesOnly" class="carousel slide"
data-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      
    </div>
    <div class="carousel-item">
      
    </div>
    <div class="carousel-item">
      
    </div>
  </div>
</div>
```

Lo que podemos [observar](#), es una transición de cada una de ellas.

Además es posible agregarle controles (previo, siguiente) como en este ejemplo:

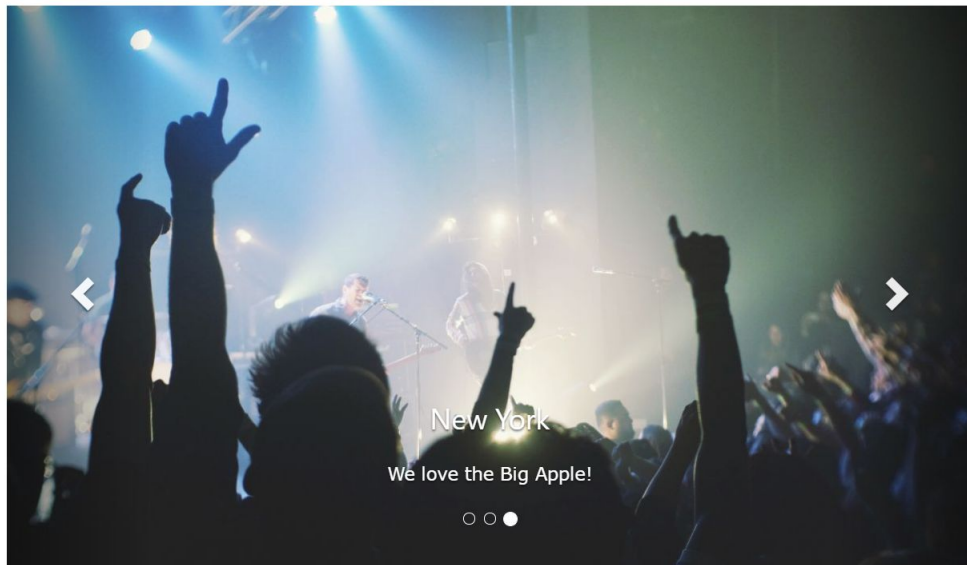


Imagen 16. Carousel Bootstrap.

Fuente: W3Schools.

Conociendo Typeform

Formularios con Typeform (Parte I)

Regularmente las páginas o landing pages buscan mantener una forma de contacto con el usuario. Una de las formas más eficientes de lograrlo es a través de un formulario.

Normalmente estos formularios, envían un email al administrador de la página con los datos que los usuarios ingresen. La lógica detrás de enviar un email de forma automática regularmente es manejada por un servidor, el cual es configurado e integrado con nuestra página. A pesar de que no es una tarea en extremo compleja, se sale de los alcances de este curso. Para suplir esta necesidad utilizaremos una herramienta de terceros, que integra formularios y envío de correo electrónico.

Este es el caso de Typeform, un servicio proporcionado por una empresa que maneja la información que llenarán los usuarios, al mismo tiempo que nos entrega mejoras visuales para estos formularios, y nos enviará un correo.

Si ingresamos a su [página](#), podremos ver en detalle lo que la empresa ofrece. En resumen, muchos ejemplos de formularios para nuestras páginas.

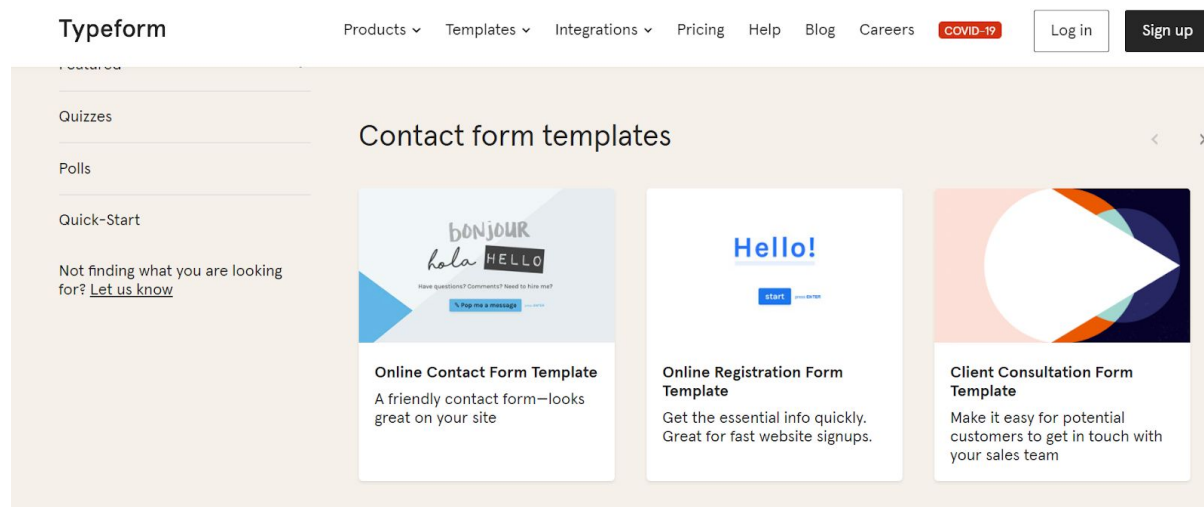



Imagen 17. Templates Typeform.

Fuente: [Typeform](#).

Typeform nos servirá para hacer un formulario llamativo y personalizado sin requerir mucho código. La cuenta gratis nos servirá hasta para 100 mensajes. Comencemos por registrarnos:

- **Paso 1:** Ingresamos al [link de registro](#), ahí nos pedirá nuestros datos para crear una cuenta gratuita:



Spanish

¿Ya tienes una cuenta?

Consigue mejores datos con formularios que convierten, así como encuestas, concursos y más

Email
bruce@wayne.com

Contraseña
Al menos 8 caracteres

☐ Acepto las [Terms of Service](#) de Typeform

☐ Acepto el uso de mis datos por parte de Typeform para el servicio y todo lo descrito en la [Privacy Policy](#)

[Ver opciones](#)

Crear mi cuenta gratuita

Imagen 18. Registro Typeform.

Fuente: [Typeform](#).

- **Paso 2:** Una vez que hayas llenado los datos para la cuenta gratuita e ingresado tu correo y confirmado podemos [loguearnos](#) para realizar un formulario.
- **Paso 3:** Para crear un nuevo formulario seleccionaremos “nuevo typeform”, una vez iniciada la sesión:

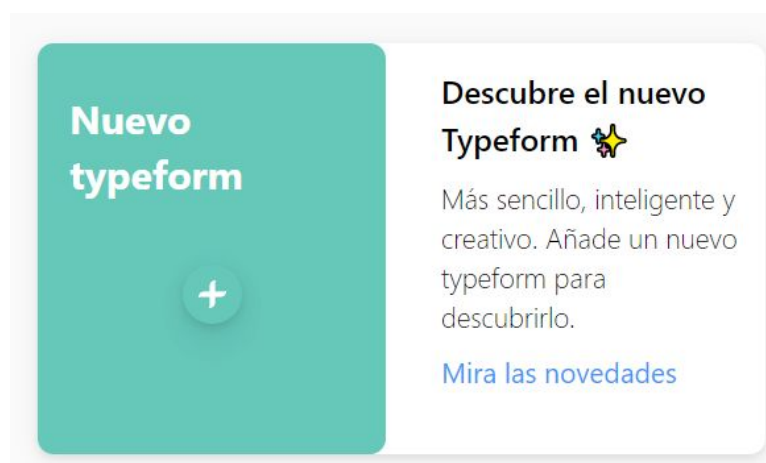


Imagen 19. Nuevo Typeform.

Fuente: [Typeform](#).

- **Paso 4:** Una vez ahí, nos encontraremos con el panel de Typeform, donde podremos utilizar un template para nuestro formulario o crear uno a nuestra medida. Haremos esto último, seleccionando “empezar desde cero”:

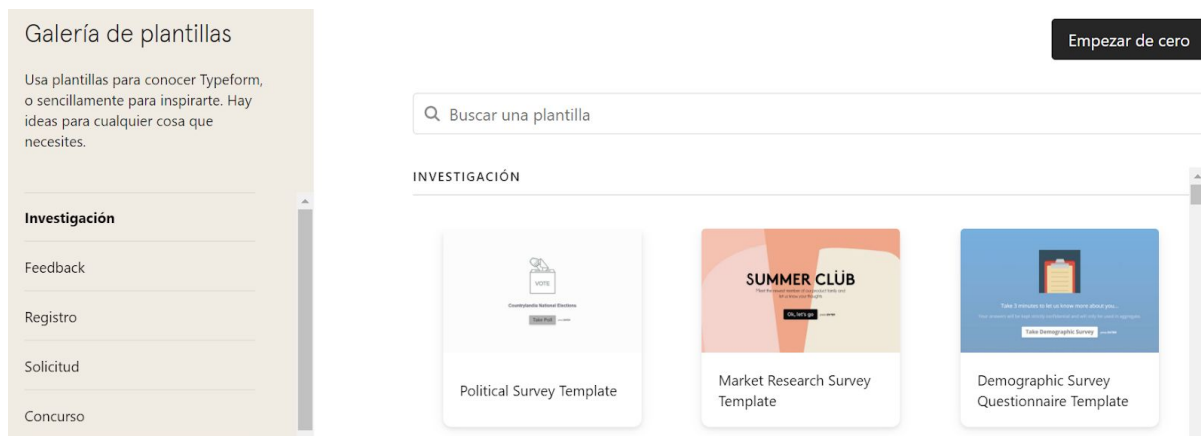


Imagen 20. Creación de template, parte 1.

Fuente: [Typeform](https://typeform.com).

- **Paso 5:** Una vez dentro, nos pide datos básicos de nuestro proyecto, para poder identificarlo:

Imagen 21. Datos del formulario.

Fuente: [Typeform](https://typeform.com).

- **Paso 6:** Una vez en el formulario, tendremos distintas opciones de creación:



Imagen 22. Crear nuevo formulario.

Fuente: [Typeform](https://typeform.com).

A la izquierda, tenemos el editor de contenido, desde donde podremos escoger distintos tipos de bloque.

- **Paso 7:** Presionemos en el + que está en el editor y seleccionamos "Pantalla de Bienvenida".

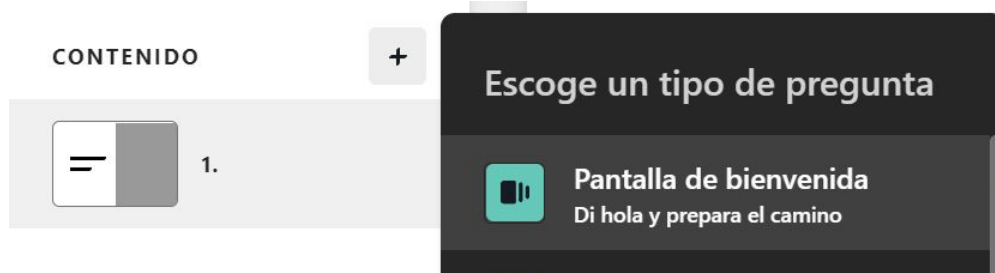


Imagen 23. Pantalla de Bienvenida.

Fuente: [Typeform](https://typeform.com).

Aquí escribiremos el mensaje de bienvenida, por ejemplo “Hola, gracias por contactarnos” y en la esquina superior derecha de la pantalla, seleccionamos editar y desactivamos la opción “tiempo para completar”:

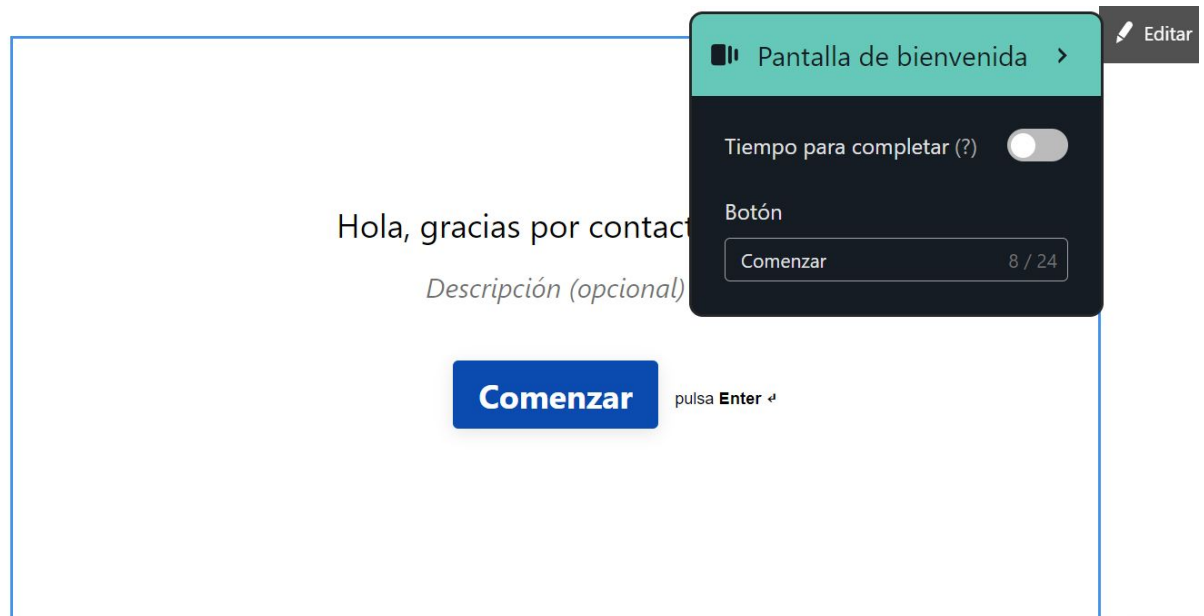


Imagen 24. Editar pantalla de bienvenida.

Fuente: [Typeform](https://typeform.com).

- **Paso 8:** Escogeremos un nuevo contenido de tipo “respuesta corta” y lo editaremos para preguntar “¿Cuál es tu nombre?”.



Imagen 25. Editar pantalla de nombre.

Fuente: [Typeform](https://typeform.com).

- **Paso 9:** Escogeremos un nuevo contenido de tipo “email” y lo editaremos para preguntar “¿Cuál es tu correo?”.
- **Paso 10:** Escogeremos un nuevo contenido de tipo “respuesta larga” y lo editaremos para preguntar “Ingresa tu mensaje”.
- **Paso 11:** Iremos a la sección de final y editaremos el mensaje de “gracias por contactarnos”. Para quitar los íconos de las redes sociales, debemos presionar en “editar”, en la esquina superior izquierda:



Imagen 26. Final.
Fuente: [Typeform](https://www.typeform.com/).

- **Paso 12:** Para ver cómo quedó nuestro formulario, podemos hacer click en el ícono de ‘ver’ que está en la parte superior de la pantalla:



Imagen 27. Ver formulario.
Fuente: [Typeform](https://www.typeform.com/).

- **Paso 13:** Para poder agregar nuestro formulario a la página web, debemos presionar en “compartir” en la parte superior. Una vez ahí nos da la opción de obtener el código para incrustarlo en varias opciones de visualización: pantalla completa, popup, slider, panel lateral, etc.

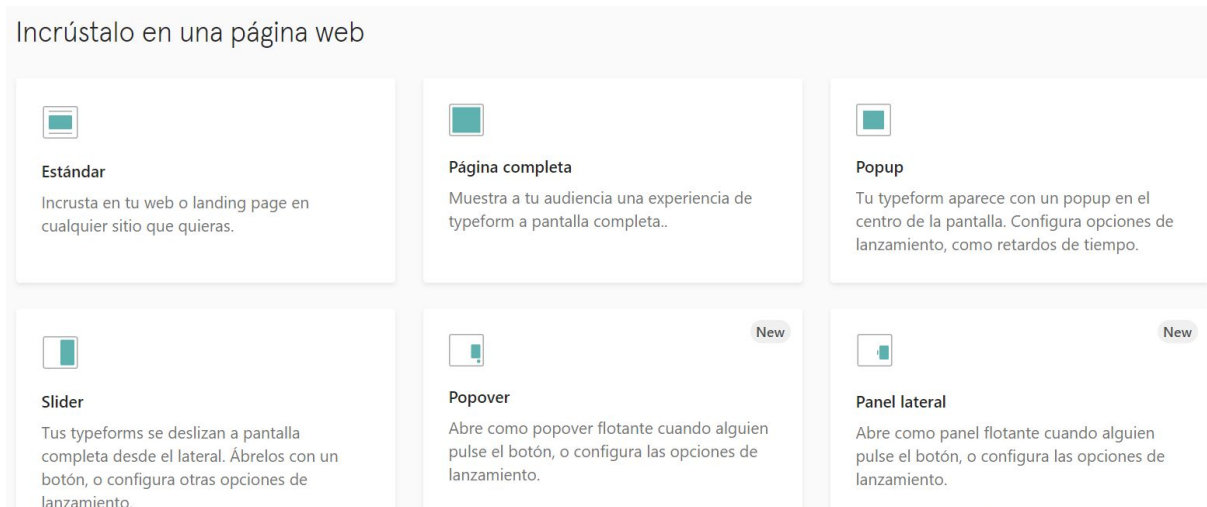


Imagen 28. Compartir formulario.

Fuente: [Typeform](https://typeform.com).

- **Paso 14:** Seleccionaremos el modo estándar. Una vez que nos redirija a la pantalla, podremos realizar unos últimos ajustes y obtener el código:



Imagen 29. Obtener código.
Fuente: [Typeform](https://typeform.com).

Conclusiones

En esta unidad conocimos los conceptos básicos de JavaScript. Luego conocimos sobre la biblioteca jQuery. Utilizamos los componentes JS de Bootstrap como las modales, los carruseles y los popovers y finalmente implementamos un formulario con Typeform.

Recuerda seguir practicando y no olvides leer la documentación de los elementos mencionados.

Tomar el ritmo al desarrollo es cuestión de tiempo y práctica, así que sigue aprendiendo a través de los desafíos.

Ejercicio guiado: Agregando JavaScript al sitio Olivia Ros

Contexto

Como hemos visto, JavaScript, a través de la biblioteca jQuery y Bootstrap, nos permite integrar distintas funcionalidades adicionales a los sitios web, para hacerlos más interesantes, dinámicos y mejorando la experiencia de navegación de los visitantes.

En el siguiente ejercicio integraremos algunas de las funcionalidades que hemos aprendido a nuestro proyecto de Olivia Ros:

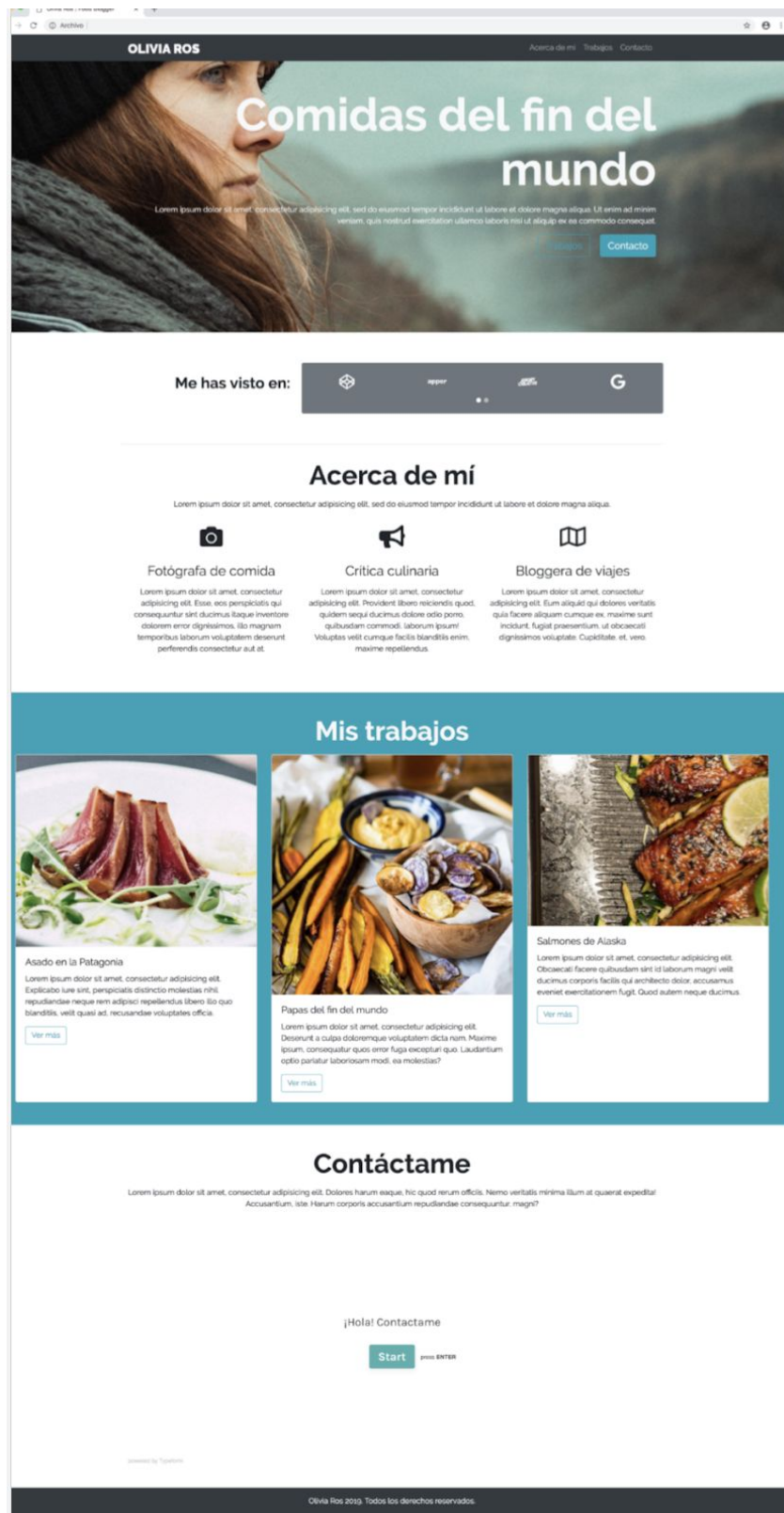


Imagen 30. Vista Final del Proyecto.
Fuente: Desafío Latam.

1. Haremos que al clickear los ítems del menú se haga un scroll suave.
2. Le incorporaremos ventanas modales.

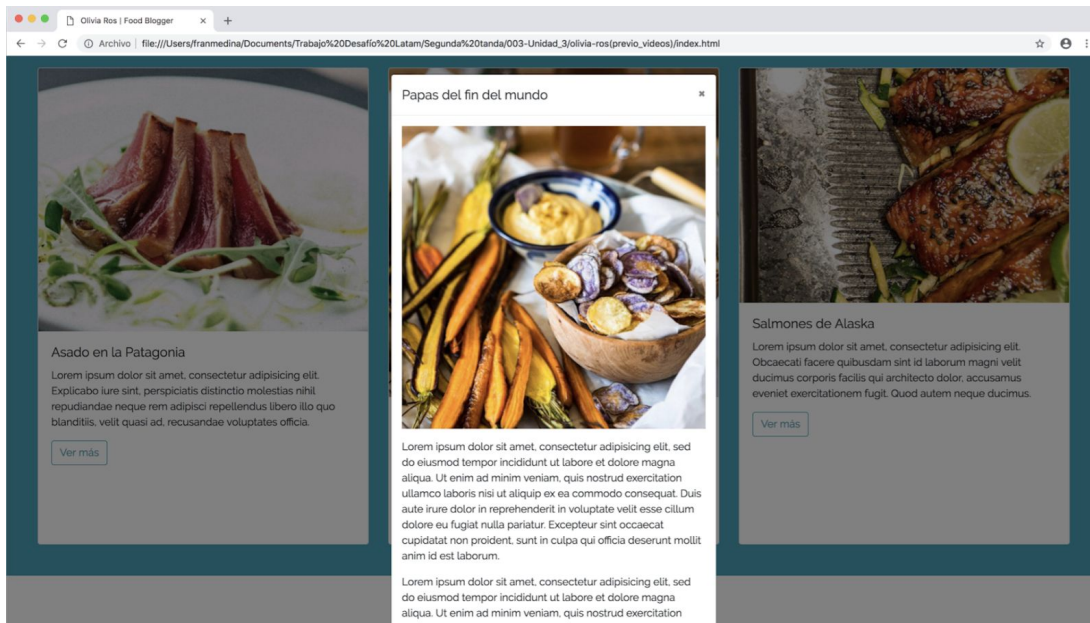


Imagen 31. Vista final del Proyecto - Ventanas Modales.
Fuente: Desafío Latam.

3. También un carrusel de íconos.

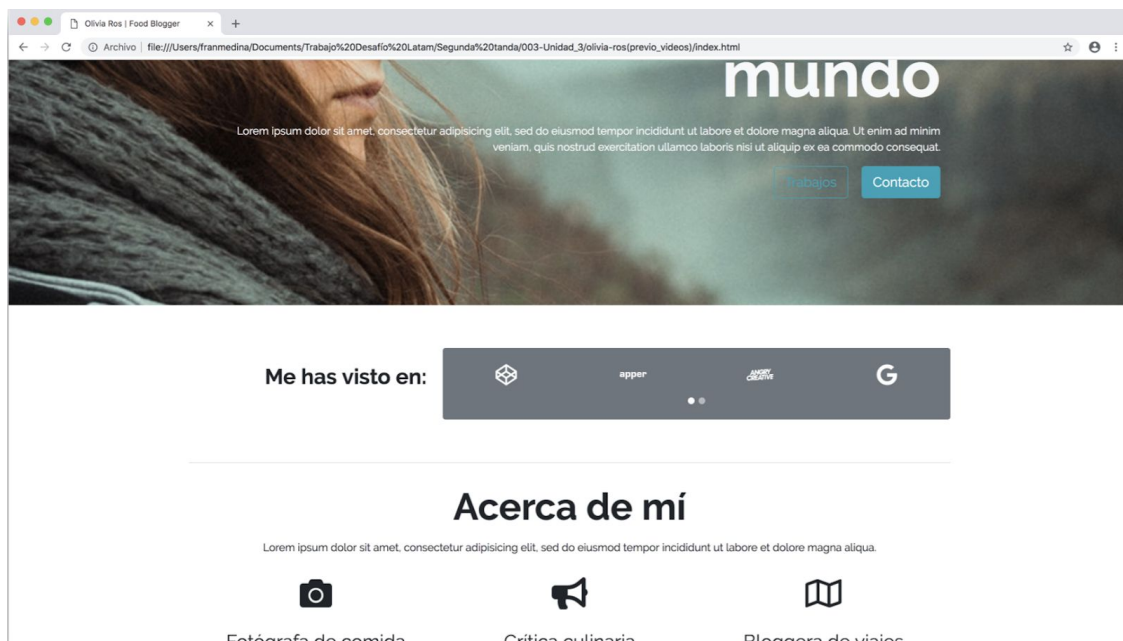


Imagen 32. Vista final del Proyecto - Carrusel de íconos.
Fuente: Desafío Latam.

4. A esos íconos le implementaremos popovers.

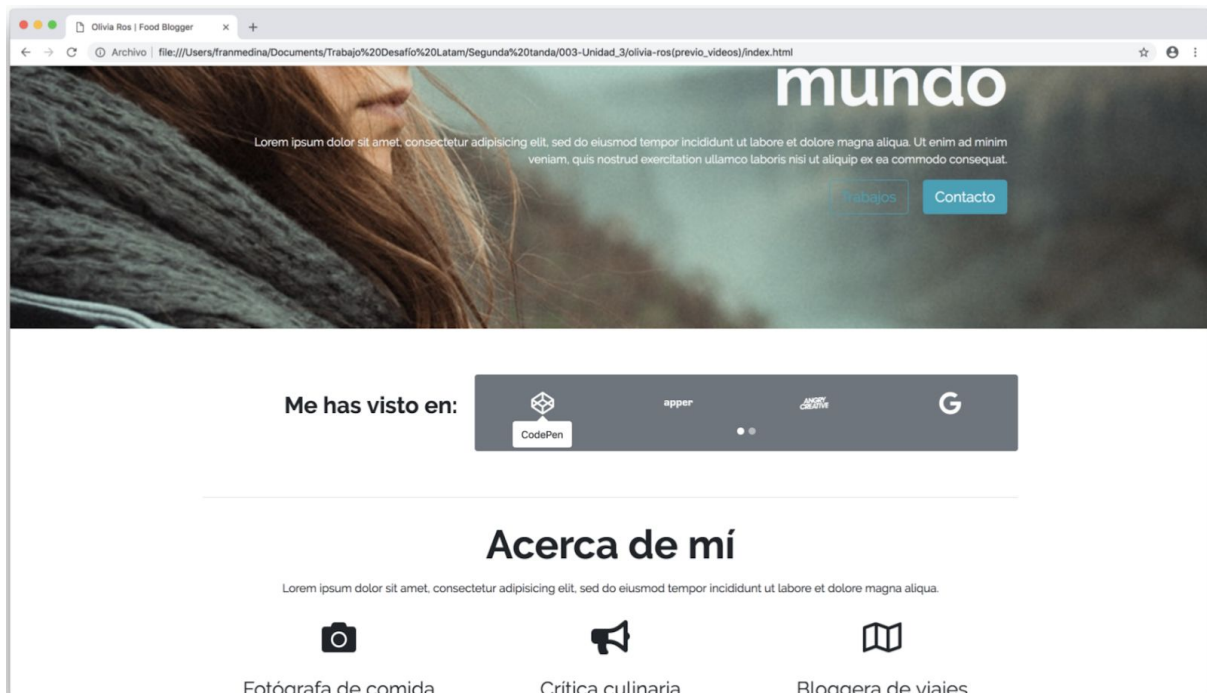


Imagen 33. Vista final de Proyecto - Íconos con popovers.
Fuente: Desafío Latam.

5. Y, finalmente, construiremos un formulario de contacto dinámico con [Typeform](#).

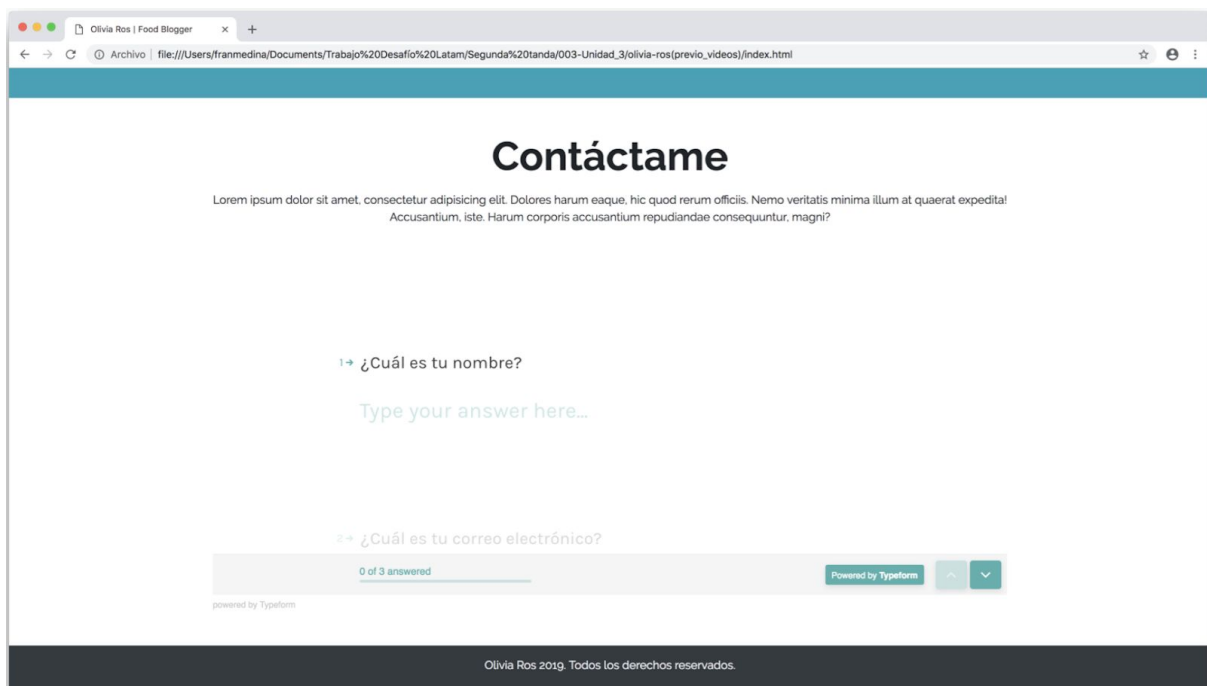


Imagen 34. Vista final del Proyecto - Formulario de contacto dinámico.
Fuente: Desafío Latam.

Entonces comencemos.

- **Paso 1:** En nuestro proyecto Olivia Ros ya integramos jQuery junto con Popper JS y Bootstrap JS.

Lo que haremos entonces será cambiar jQuery slim por jQuery normal que sacaremos desde [Code jQuery](#). ¿Por qué? Porque jQuery slim pierde algunas propiedades de jQuery normal, como las animaciones, que necesitaremos para el desarrollo de la actividad. De todas formas copiaremos el CDN de jQuery minificado, para ahorrar tiempos de carga y peso del archivo.

```
<!--jQuery 3.5.1 (sin slim)-->
<script src="https://code.jquery.com/jquery-3.5.1.min.js"
integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
crossorigin="anonymous"></script>
<!--Popper JS 1.14.3-->
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.
min.js"
integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/18
WvCWIPm49" crossorigin="anonymous"></script>
<!-- Bootstrap 4.5.3 CSS -->
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.mi
n.css"
integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlIDPvg
6uqKI2xXr2" crossorigin="anonymous"></script>
```

- **Paso 2:** A continuación vamos a generar un smooth scrolling entre las secciones de la página principal del sitio Olivia Ros.

Lo primero que debemos hacer es generar las anclas por id. Eso lo haremos generando un id por las secciones correspondientes y vinculándose con los ítems del menú.

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Olivia Ros | Food Blogger</title>
    <!-- Bootstrap 4.5.3 CSS -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.mi
n.css"
integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg
6uqKI2xXr2" crossorigin="anonymous">
    <!--Google Font: Raleway (Regular, Bold)-->
    <link href="https://fonts.googleapis.com/css?family=Raleway:400,700"
rel="stylesheet">
    <!--Font Awesome 5.6.3-->
    <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.6.3/css/all.css"
integrity="sha384-UHRtZLI+pbxtHCWp1t77Bi1L4ZtiqrqD80Kn4Z8NTRSryMA2Fd33n5d
Q8lWUE00s/" crossorigin="anonymous">
    <!--Mi estilo-->
    <link rel="stylesheet" href="assets/css/style.css">
  </head>
  <body>
    <!--Navbar agregado-->
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top">
      <div class="container">
        <a class="navbar-brand" href="#"></a>
        <button class="navbar-toggler" type="button"
data-toggle="collapse" data-target="#navbarNav"
aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle
navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
          <ul class="navbar-nav ml-auto">
            <li class="nav-item">
              <a class="nav-link" href="#about">Acerca de mí</a>
            </li>
            <li class="nav-item">
              <a class="nav-link" href="#work">Trabajos</a>
            </li>
          </ul>
        </div>
      </div>
    </nav>
  </body>
</html>
```

```
        </li>
        <li class="nav-item">
            <a class="nav-link" href="#contact">Contacto</a>
        </li>
    </ul>
</div>
</div>
</nav>

<header class="hero text-right text-light pt-5">
    <div class="container pt-5">
        <h1 class="display-1 font-weight-bold my-3">Comidas del fin del
        mundo</h1>
        <p class="my-3">Lorem ipsum dolor sit amet, consectetur
        adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore
        magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco
        laboris nisi ut aliquip ex ea commodo consequat.</p>
        <a class="btn btn-lg btn-outline-info mr-3"
        href="#work">Trabajos</a>
        <a class="btn btn-lg btn-info" href="#contact">Contacto</a>
    </div>
</header>

<section id="about" class="container text-center my-5">
    <h2 class="display-4 font-weight-bold my-3">Acerca de mí</h2>
    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
    eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
    <div class="row">
        <div class="col-md-4">
            <i class="fas fa-3x fa-camera my-3"></i>
            <h3 class="my-3">Fotógrafa de comida</h3>
            <p class="my-3">Lorem ipsum dolor sit amet, consectetur
            adipisicing elit. Esse, eos perspiciatis qui consequuntur sint ducimus
            itaque inventore dolorem error dignissimos, illo magnam temporibus
            laborum voluptatem deserunt perferendis consectetur aut at.</p>
        </div>
        <div class="col-md-4">
            <i class="fas fa-3x fa-bullhorn my-3"></i>
            <h3 class="my-3">Crítica culinaria</h3>
            <p class="my-3">Lorem ipsum dolor sit amet, consectetur
            adipisicing elit. Provident libero reiciendis quod, quidem sequi ducimus
            dolore odio porro, quibusdam commodi, laborum ipsum! Voluptas velit
            cumque facilis blanditiis enim, maxime repellendus.</p>
        </div>
    </div>
</section>
```

```
<div class="col-md-4">
  <i class="far fa-3x fa-map my-3"></i>
  <h3 class="my-3">Bloggera de viajes</h3>
  <p class="my-3">Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Eum aliquid qui dolores veritatis quia facere aliquam
cumque ex, maxime sunt incidunt, fugiat praesentium, ut obcaecati
dignissimos voluptate. Cupiditate, et, vero.</p>
</div>
</div>
</section>

<section id="work" class="container-fluid my-5 p-5 bg-info">
  <h2 class="display-4 text-center font-weight-bold text-light
mb-3">Mis trabajos</h2>
  <div class="card-deck">
    <div class="card">
      
      <div class="card-body">
        <h5 class="card-title">Asado en la Patagonia</h5>
        <p class="card-text">Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Explicabo iure sint, perspiciatis distinctio molestias
nihil repudiandae neque rem adipisci repellendus libero illo quo
blanditiis, velit quasi ad, recusandae voluptates officia.</p>
        <button type="button" class="btn btn-outline-info">Ver
más</button>
      </div>
    </div>
    <div class="card">
      
      <div class="card-body">
        <h5 class="card-title">Papas del fin del mundo</h5>
        <p class="card-text">Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Deserunt a culpa doloremque voluptatem dicta nam.
Maxime ipsum, consequatur quos error fuga excepturi quo. Laudantium
optio pariat laboriosam modi, ea molestias?</p>
        <button type="button" class="btn btn-outline-info">Ver
más</button>
      </div>
    </div>
    <div class="card">
      
```



```
<div class="card-body">
  <h5 class="card-title">Salmones de Alaska</h5>
  <p class="card-text">Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Obcaecati facere quibusdam sint id laborum magni velit
ducimus corporis facilis qui architecto dolor, accusamus eveniet
exercitationem fugit. Quod autem neque ducimus.</p>
  <button type="button" class="btn btn-outline-info">Ver
más</button>
</div>
</div>
</div>
</section>

<section id="contact" class="container my-5">
  <h2 class="display-4 text-center font-weight-bold
my-3">Contáctame</h2>
  <p class="text-center">Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Dolores harum eaque, hic quod rerum officiis. Nemo
veritatis minima illum at quaerat expedita! Accusantium, iste. Harum
corporis accusantium repudiandae consequuntur, magni?</p>
  <form>
    <div class="form-group">
      <label for="formGroupExampleInput"
class="font-weight-bold">Nombre</label>
      <input type="text" class="form-control"
id="formGroupExampleInput" placeholder="Ingresa tu nombre">
    </div>
    <div class="form-group">
      <label for="exampleInputEmail1" class="font-weight-bold">Correo
electrónico</label>
      <input type="email" class="form-control"
id="exampleInputEmail1" aria-describedby="emailHelp"
placeholder="Ingresa tu correo electrónico">
    </div>
    <div class="form-group">
      <label for="exampleFormControlTextarea1"
class="font-weight-bold">Mensaje</label>
      <textarea class="form-control" id="exampleFormControlTextarea1"
rows="3" placeholder="Escribe tu mensaje"></textarea>
    </div>
    <div class="contact-button-inner text-right">
      <button type="button" class="btn btn-lg
btn-info">Enviar</button>
    </div>
```

```
</form>
</section>
<!-- Footer -->
<footer class="bg-dark text-center text-light p-3">
  <p>Olivia Ros 2019. Todos los derechos reservados.</p>
</footer>

<!--jQuery 3.5.1 (sin slim)-->
<script src="https://code.jquery.com/jquery-3.5.1.min.js"
integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
crossorigin="anonymous"></script>
<!--Popper JS 1.14.3-->
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.
min.js"
integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/18
WvCWIPm49" crossorigin="anonymous"></script>
<!-- Bootstrap 4.5.3 CSS -->
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.mi
n.css"
integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfiDPvg
6uqKI2xXr2" crossorigin="anonymous"></script>
</body>
</html>
```


- **Paso 3:** Luego, generamos la carpeta `js` dentro de la carpeta `assets` y ahí crearemos el archivo `script.js`. A este archivo lo llamaremos desde el final del `<body>`, justo después del llamado al `Bootstrap JS`.

```
<!--jQuery 3.5.1 (sin slim)-->
<script src="https://code.jquery.com/jquery-3.5.1.min.js"
integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
crossorigin="anonymous"></script>
<!--Popper JS 1.14.3-->
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.
min.js"
integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/18
WvCWIPm49" crossorigin="anonymous"></script>
<!-- Bootstrap 4.5.3 CSS -->
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.mi
n.css"
integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAY5uIKz4rEkgIXeMed4M0jlfIDPvg
6uqKI2xXr2" crossorigin="anonymous"></script>
<!-- Mi script -->
<script src="assets/js/script.js"></script>
```

- **Paso 4:** Dentro del archivo `script.js` vamos a declarar la función de `document` `Document Ready Event`. Recordemos que con las nuevas versiones la función se escribe así:

```
$(function(){

  // métodos de jQuery...

});
```

- **Paso 5:** Dentro de la función vamos a seleccionar a los links `<a>` con el selector de etiqueta `$("#a")`.

El evento que desencadenará la función será el evento `click`, o sea, cuando el usuario clickee la etiqueta `<a>`.

```
$(function(){  
  
  $("#a").click(function(event) {  
  
  });  
  
});
```

- **Paso 6:** Ahora debemos evaluar una condicional. Para ello utilizaremos el método `hash`, que evalúa el atributo `href` de esto, y obtiene la parte de la URL que comienza con `#`.

Hagamos una prueba sólo para revisar. Cuando clickeamos un link nos aparecerá una alerta con lo que sigue después del `#` en el atributo `href`.

```
$(function(){  
  
  $("#a").click(function(event) {  
    alert(this.hash)  
  });  
  
});
```

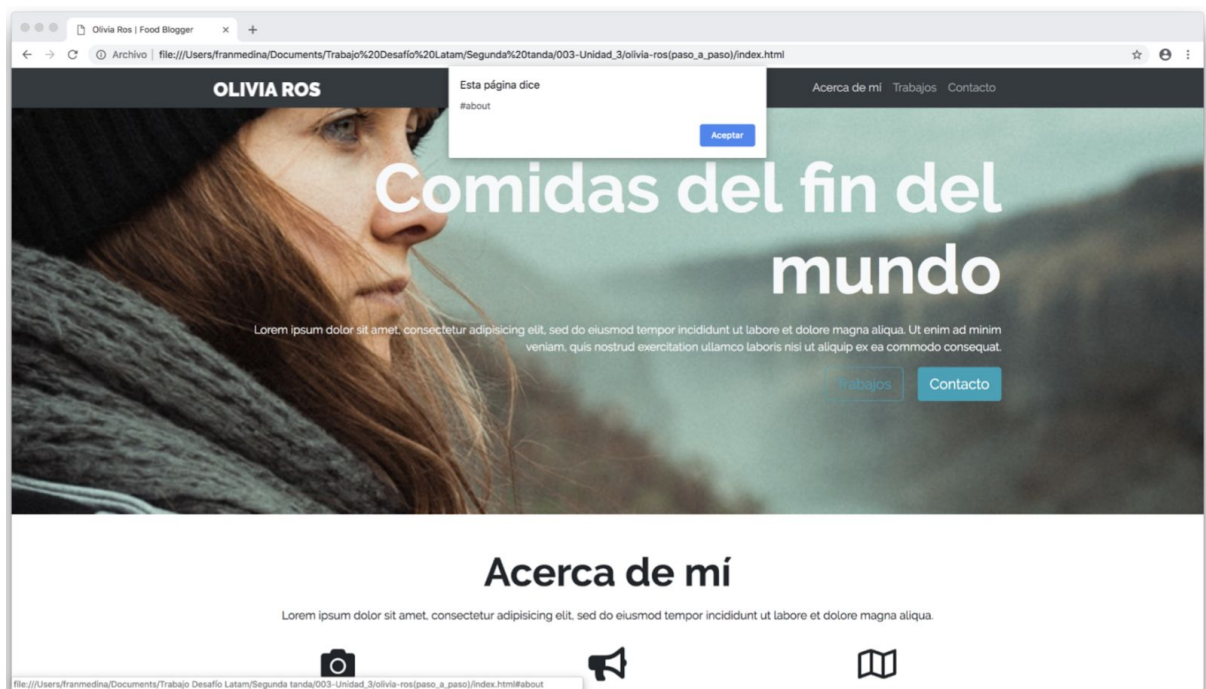


Imagen 35. Primera prueba.
Fuente: Desafío Latam.

Ya los tenemos identificados.

- **Paso 7:** Ahora escribiremos lo siguiente:

```
$(function(){  
  
  $("a").click(function(event) {  
    // Con este if se asegura que this.hash tenga un valor antes de anular  
    el comportamiento predeterminado  
    if (this.hash !== "") {  
      // Previene el comportamiento de click predeterminado  
      event.preventDefault();  
      // Guarda el valor del hash en una variable llamada gato  
      var gato = this.hash;  
      // Usa el método animate para animar el scroll y hacerlo de una  
      forma suave  
      // El número opcional (800) especifica el número de milisegundos que  
      se demorará en llegar hasta el área  
      $('html, body').animate({  
        scrollTop: $(gato).offset().top  
      }, 800, function(){
```

```
// Agrega hash (#) a la URL cuando haya terminado de desplazarse  
(comportamiento de click predeterminado)  
    window.location.hash = gato;  
    });  
} // Fin del if  
});  
  
});
```

Ahora guardamos y recargamos el navegador y podemos ver que se genera el scroll suave entre secciones.

El siguiente paso es implementar ventanas modales al proyecto de Olivia Ros: Tiene que sentido que en la sección de "Mis trabajos", cada botón de cada card genere una ventana modal con mayor información de ese trabajo en específico. Por lo tanto el botón será el que abrirá la ventana modal.

- **Paso 8:** Primero lo primero. Leer la [documentación](#) de las ventanas modales.

Según lo que leemos aquí a un botón debemos añadirle los atributos `data-toggle="modal"` `data-target="#exampleModal"`. Tenemos que reemplazar el valor `exampleModal` por el `id` que le demos al `div` contenedor de la ventana modal.

Cómo haremos una por cada trabajo nos convendría ponerle un `id` relacionado al plato que se quiere ver en mayor profundidad. El primero es "Asado en la Patagonia" así que le pondremos `asado` como `id`.

Por lo tanto al botón le daremos esos atributos: `<button type="button" class="btn btn-outline-info" data-toggle="modal" data-target="#asado">Ver más</button>`.

```
<div class="card">
  
  <div class="card-body">
    <h5 class="card-title">Asado en la Patagonia</h5>
    <p class="card-text">Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Explicabo iure sint, perspiciatis distinctio molestias
nihil repudiandae neque rem adipisci repellendus libero illo quo
blanditiis, velit quasi ad, recusandae voluptates officia.</p>
    <button type="button" class="btn btn-outline-info"
data-toggle="modal" data-target="#asado">Ver más</button>
  </div>
</div>
```

- **Paso 9:** Y luego escribiremos la modal fuera de la sección:

```
<div class="modal fade" id="asado" tabindex="-1" role="dialog"
aria-labelledby="exampleModallLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModallLabel">Modal title</h5>
        <button type="button" class="close" data-dismiss="modal"
aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        ...
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary"
data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save
changes</button>
      </div>
    </div>
  </div>
</div>
```

Ojo que debemos reemplazar su id de exampleModal por asado.

Probemos qué sucede:

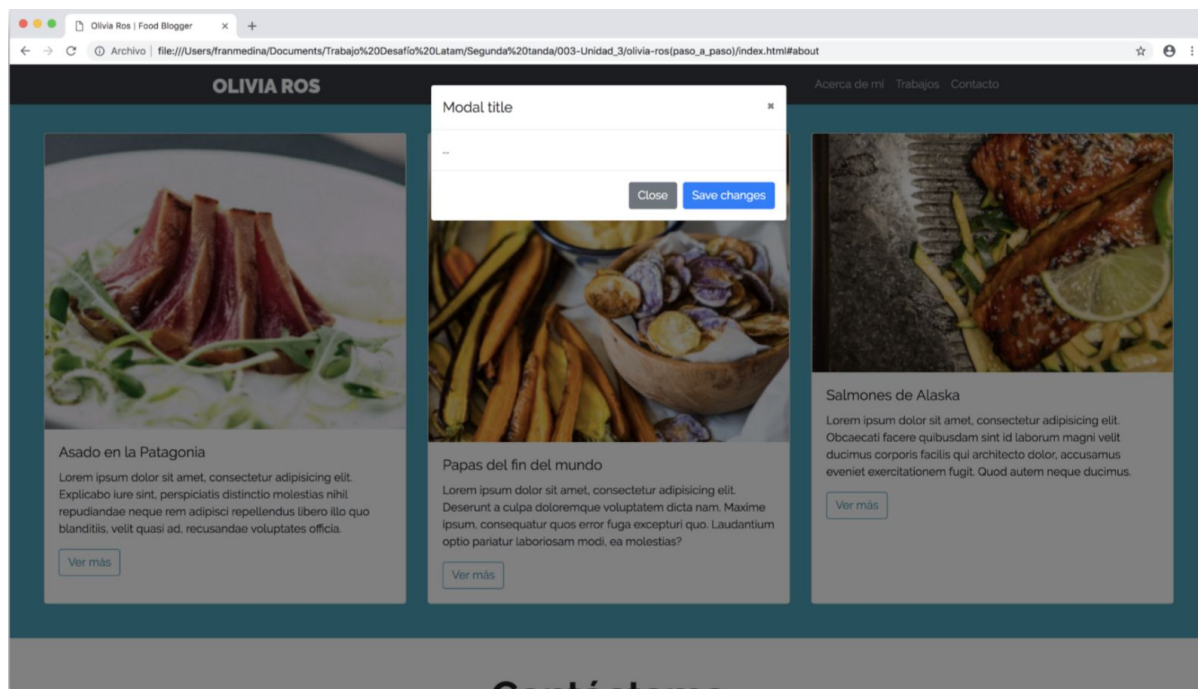


Imagen 36. Modificando Modal Title.

Fuente: Desafío Latam.

Ahora sólo falta rellenarla con contenido.

- **Paso 10:** Modal title lo reemplazaremos por el título de la card: Asado en la Patagonia.

Dentro de `<div class="modal-body">...</div>` colocaremos la imagen correspondiente con la clase `img-fluid` para que ocupe el ancho del contenedor y la clase `my-3` para que genere espaciado vertical.

- **Paso 11:** Luego, también dentro de `<div class="modal-body">...</div>`, colocaremos 3 párrafos con `Lorem Ipsum`. Además le sacaremos el botón de Save changes.

```
<div class="modal fade" id="asado" tabindex="-1" role="dialog"
aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
```

```
<h5 class="modal-title" id="exampleModallabel">Asado en la
Patagonia</h5>
<button type="button" class="close" data-dismiss="modal"
aria-label="Close">
  <span aria-hidden="true">&times;</span>
</button>
</div>
<div class="modal-body">
  
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit
esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
est laborum.</p>
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit
esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
est laborum.</p>
  <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea
commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit
esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat
cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id
est laborum.</p>
</div>
<div class="modal-footer">
  <button type="button" class="btn btn-secondary"
data-dismiss="modal">Close</button>
</div>
</div>
</div>
```


Guardemos y recarguemos el navegador.

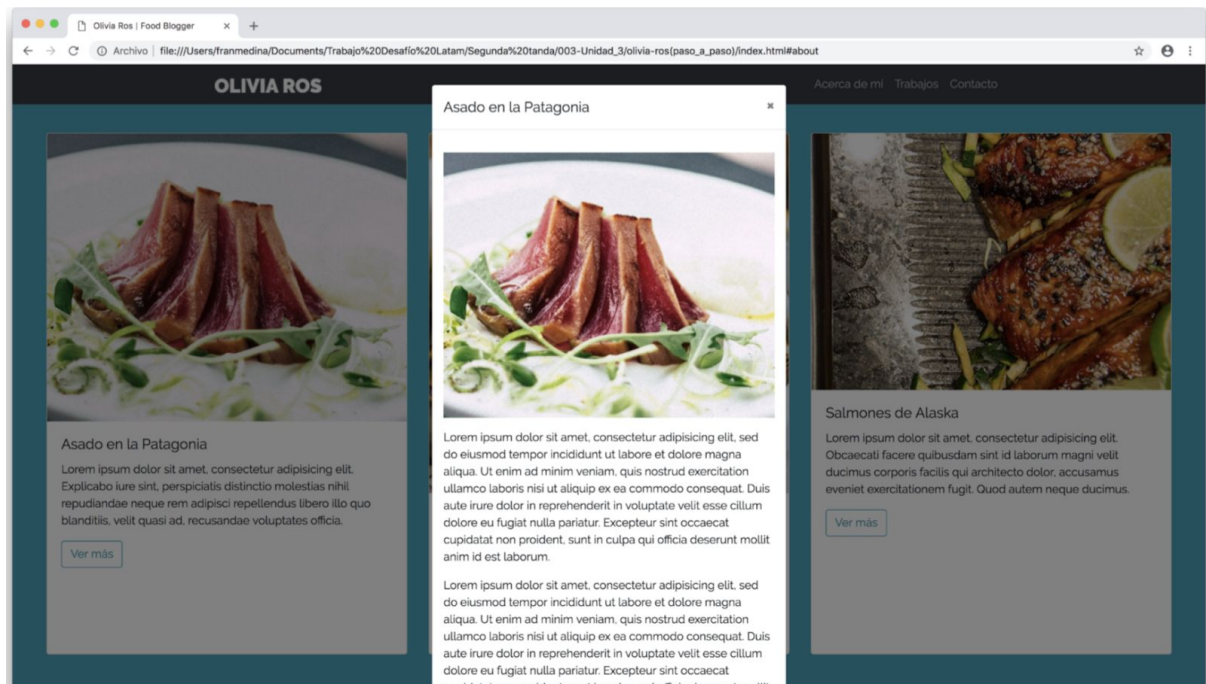


Imagen 37. Modal completo.
Fuente: Desafío Latam.

¿Te animas a generar otros modales?

- **Paso 12:** Ahora es el momento de agregar el carousel. Queremos lograr lo siguiente:

Me has visto en:



Imagen 38. Ejemplo de carousel.
Fuente: Desafío Latam.

Utilizaremos los siguientes íconos de Font Awesome y a esos íconos le daremos la clase `fa-2x` para agrandarlos:

- `fab fa-codepen.`
- `fab fa-apper.`
- `fab fa-angrycreative.`
- `fab fa-google.`
- `fab fa-500px.`
- `fab fa-apple.`
- `fab fa-aviato.`
- `fab fa-d-and-d.`

Este código lo colocaremos entre el `<header>` y la sección de Acerca de mí.

- **Paso 13:** Podemos identificar un divisor con clase `container` que contendrá la frase `"Me has visto en:"` y el carousel. A ese divisor también le daremos la clase `my-5 py-3` para generar márgenes y paddings verticalmente.
- **Paso 14:** Generaremos un `row` (columna) donde pondremos dos `divs` con sus respectivas columnas.

Una será ocupando 4 columnas (la del texto `"Me has visto en:"`) y la segunda ocupará 8 columnas (carousel).

- **Paso 15:** Al primer `div` le pondremos la clase `text-right` para alinear el texto a la derecha y `p-4` para generar paddings a todos lados. Además de un `<h2 class="font-weight-bold">Me has visto en:</h2>` como contenido.

Al segundo `div` le pondremos, aparte de la clase para asignarle el tamaño de columnas, las clases: `bg-secondary text-white rounded pt-4 pb-5`, para generar el color de fondo, las letras blancas, el borde redondeado y los padding top y bottom.

```
<div class="container my-5 py-3">
  <div class="row">
    <div class="col-4 p-4 text-right">
      <h2 class="font-weight-bold">Me has visto en:</h2>
    </div>
    <div class="col-8 bg-secondary text-white rounded pt-4 pb-5">
    </div>
  </div>
</div>
```

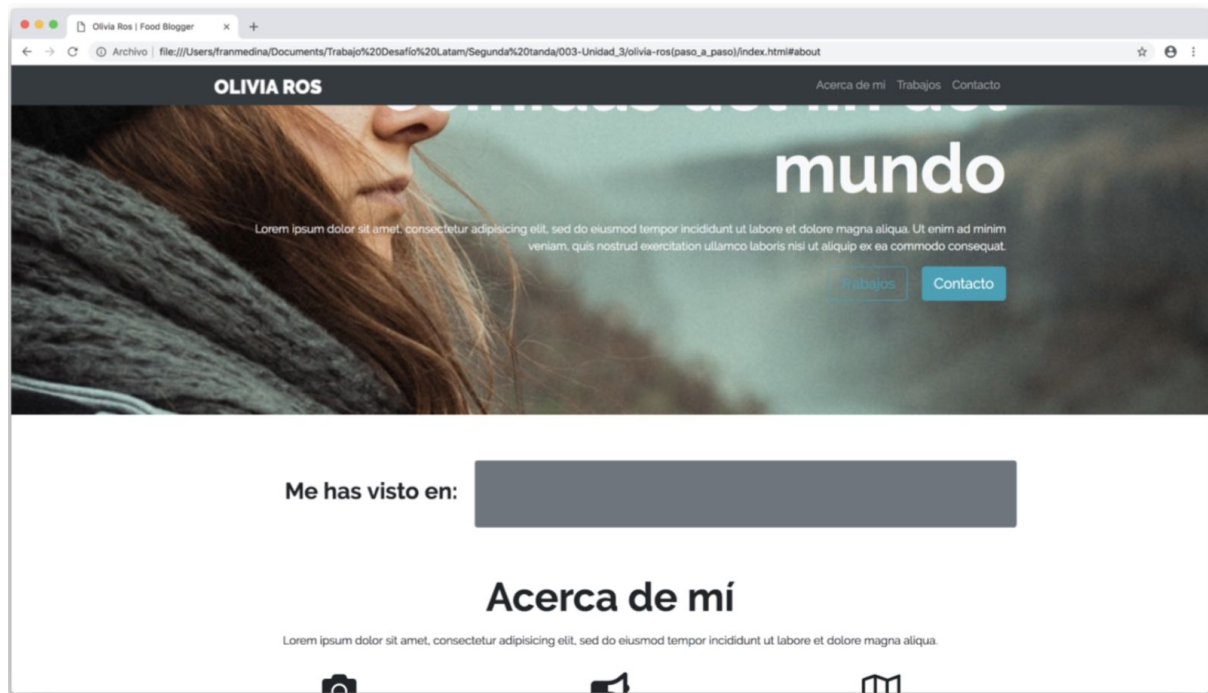


Imagen 39. Carousel incorporado, parte 1.
Fuente: Desafío Latam.

- **Paso 16:** Ahora dentro del `div` de 8 columnas pegaremos el siguiente código que analizaremos poco a poco:

```
<div id="blogCarousel" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
    <li data-target="#blogCarousel" data-slide-to="0"
class="active"></li>
    <li data-target="#blogCarousel" data-slide-to="1"></li>
  </ol>
  <div class="carousel-inner">
    <div class="carousel-item active text-center">
      <div class="row">
        <div class="col-3">
          <i class="fab fa-codepen fa-2x"></i>
        </div>
        <div class="col-3">
          <i class="fab fa-apper fa-2x"></i>
        </div>
        <div class="col-3">
          <i class="fab fa-angrycreative fa-2x"></i>
        </div>
        <div class="col-3">
```

```

        <i class="fab fa-google fa-2x"></i>
      </div>
    </div>
  </div>

  <div class="carousel-item text-center">
    <div class="row">
      <div class="col-3">
        <i class="fab fa-500px fa-2x"></i>
      </div>
      <div class="col-3">
        <i class="fab fa-apple fa-2x"></i>
      </div>
      <div class="col-3">
        <i class="fab fa-aviato fa-2x"></i>
      </div>
      <div class="col-3">
        <i class="fab fa-d-and-d fa-2x"></i>
      </div>
    </div>
  </div>
</div>
</div>
</div>

```

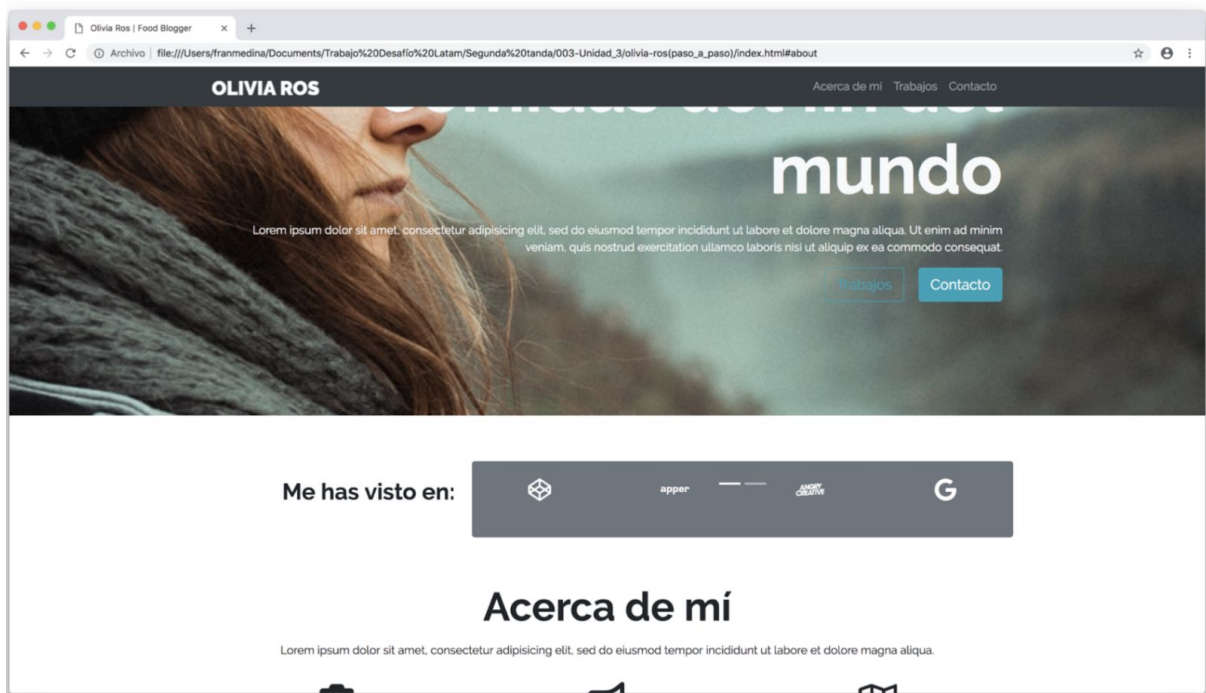


Imagen 40. Carousel incorporado, parte 2.
Fuente: Desafío Latam.

- Primero lo que hicimos fue crear el `<div id="blogCarousel" class="carousel slide" data-ride="carousel">`. Esto genera un carousel.
- Dentro de ese div generamos una lista ordenada contenedora de los indicadores `<ol class="carousel-indicators">`, que son esas líneas para cambiar el ítem del carousel.
- Éstas parten del cero. Como tenemos dos ítems del carousel escribimos: `<li data-target="#blogCarousel" data-slide-to="0" class="active">` y `<li data-target="#blogCarousel" data-slide-to="1">` en representación de los dos ítems del carousel. La primera, o sea, la 0 tiene la clase active, que corresponde a ser la primera activa.
- Después generamos un `<div class="carousel-inner">` donde estarán los ítems del carousel. Ahí generamos dos `<div class="carousel-item text-center">`, les asignamos el `text-center` para centrar los iconos y el primero tiene la clase active por ser el primero y el que debería ser mostrado en primer lugar.
- Dentro de este `carousel-item` generamos un `<div class="row">` con 4 divs en su interior con clase `col-3`. Eso es para usar 3 columnas y por ende poder hacer 4 contenedores. Dentro de cada uno se colocó la etiqueta `<i>` con su clase correspondiente para cada ícono.

- **Paso 17:** Mirando el producto final nos faltaría cambiar la forma de los indicadores y ponerlos por bajo los íconos. Para ello usaremos CSS.

```
.carousel-indicators {  
  top: 50px;  
}  
  
.carousel-indicators li {  
  width: 10px;  
  height: 10px;  
  border-radius: 50%;  
}
```

Ahora guardemos y recarguemos:

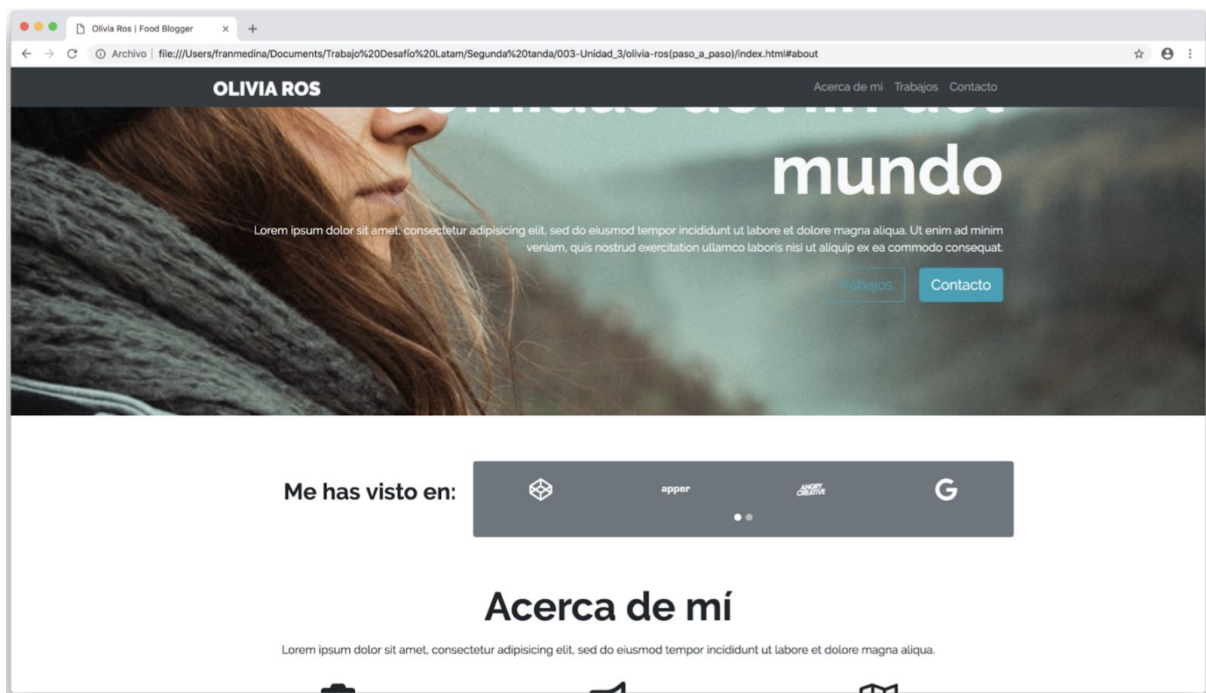


Imagen 41. Carousel incorporado, parte 3.
Fuente: Desafío Latam.

- **Paso 18:** Lo siguiente que haremos será implementar popovers a cada ícono. Para ello debemos leer la [documentación](#).

Lo primero que debemos hacer es colocar el script que piden en la documentación.

Lo pondremos dentro de nuestro script.js:

```
$( '[data-toggle="popover"]' ).popover();
```

Como ya tenemos la función de Document Ready Event no necesitamos volver a colocarla.

- **Paso 19:** Ahora copiaremos los atributos necesarios para generar un popover que se coloque bajo el elemento `data-container="body" data-toggle="popover" data-placement="bottom" data-content="Vivamus sagittis lacus vel augue laoreet rutrum faucibus."`.

Reemplazamos el valor del `data-content` por el nombre de la marca:

```
<i class="fab fa-codepen fa-2x" data-container="body"  
data-toggle="popover" data-placement="bottom"  
data-content="Codepen."></i>
```

Veamos si funciona:

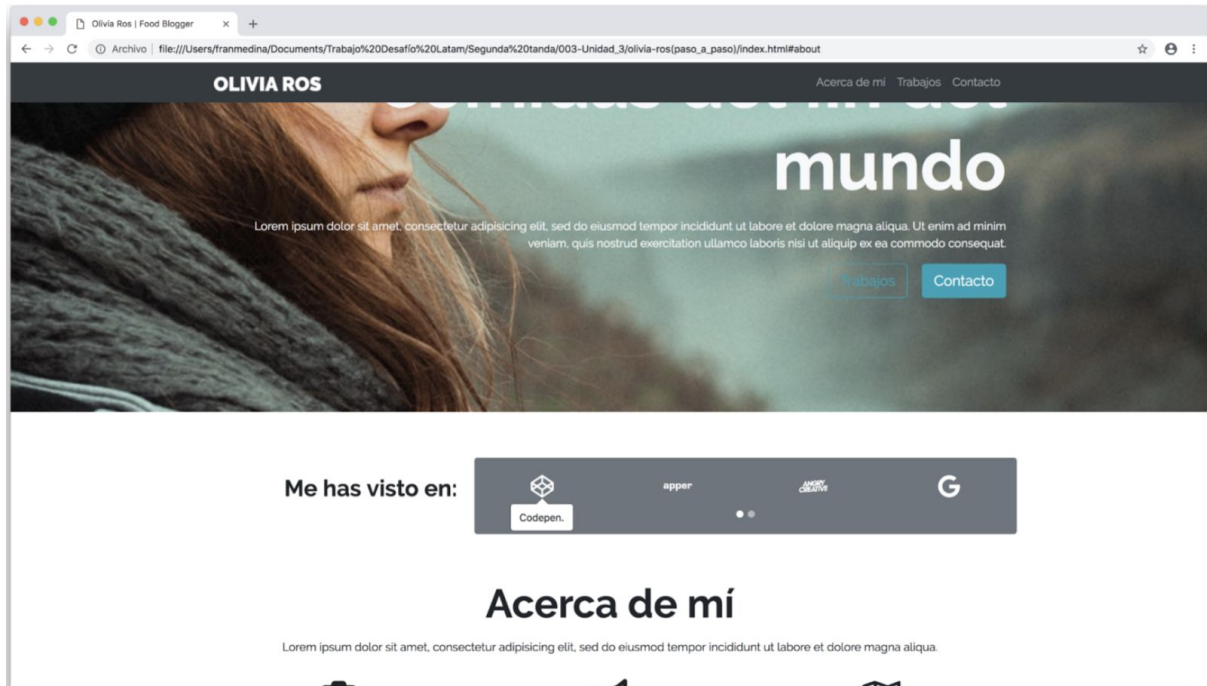


Imagen 42. Implementado popovers.
Fuente: Desafío Latam.

- **Paso 20:** Ahora le agregaremos el atributo `data-trigger="hover"` para que aparezca al hacer el hover, o sea, al pasar el mouse por encima y no al clikearlo.

```
<i class="fab fa-codepen fa-2x" data-container="body"  
data-toggle="popover" data-trigger="hover" data-placement="bottom"  
data-content="Codepen."></i>
```

¿Te animas a hacerlo con las otras marcas?.

- **Paso 21:** Finalmente vamos a agregar el typeform al proyecto, para esto copiamos el código que hemos creado en la lectura.

Vamos a nuestro `index.html` donde está nuestro formulario de contacto, que ahora reemplazamos por nuestro nuevo código.

```
<div class="typeform-widget"
data-url="https://form.typeform.com/to/vPsgLvoF?typeform-medium=embed-snippet" style="width: 100%; height: 500px;"></div> <script> (function() {
var qs,js,q,s,d=document, gi=d.getElementById, ce=d.createElement,
gt=d.getElementsByTagName, id="typef_orm",
b="https://embed.typeform.com/"; if(!gi.call(d,id)) {
js=ce.call(d,"script"); js.id=id; js.src=b+"embed.js";
q=gt.call(d,"script")[0]; q.parentNode.insertBefore(js,q) } })()
</script>
```

Guardemos el archivo y recarguemos nuestro navegador para ver nuestro nuevo formulario.

- **Paso 22:** Ahora lo vamos a probar. Llenaremos los datos, presionaremos enviar y volveremos a la página de Typeform para verificar los datos. Esto lo podemos hacer en las pestaña de resultados.



Imagen 43. Resultados.

Fuente: [Typeform](https://typeform.com).

Solución de Ejercicios Propuestos

Solución Ejercicio Propuesto (1)

1. Seleccionar las etiquetas "img".

```
$("img")
```

2. Seleccionar los elementos con id="imagen".

```
$("#imagen")
```

3. Seleccionar los elementos con clase="centrado".

```
$(".centrado")
```

4. Seleccionar los elementos con id="imagen" con clase="centrado" y que son etiqueta "img".

```
$("#img#imagen.centrado")
```

5. Seleccionar las etiquetas "a" con clase="encabezado" dentro de una etiqueta "p".

```
$("p a.encabezado")
```

6. Seleccionar los elementos con id="imagen" o clase="centrado".

```
$("#imagen,.centrado")
```

7. Seleccionar todos los elementos.

```
$("#*")
```

8. Seleccionar los elementos con el atributo href.

```
$("#[href]")
```

9. Seleccionar todos los input de tipo text.

```
$("#:text")
```

10. Un evento que al hacer click en un párrafo, imprime por consola "click".

```
$('#p').click(function() {  
    console.log('click');  
});
```

11. Modifica el precio del texto para que imprima 100.000:

```
$('.details').text('$100.000');
```

12. Al código anterior, ponle color amarillo de fondo a todos los elementos de la clase "vacaciones".

```
$('.vacations').css({color: 'yellow'});
```