PETICIONES CON AJAX A UNA BASE DE DATOS MEDIANTE ESTANDARES WEB

INTEGRANTES:

DUVAN ALEXANDER ANGARITA SANCHEZ

JASSON PINCAY PÁRRAGA

JULIÁN DAVID MORA RAMOS

GABRIEL RICARDO MORENO BOLÍVAR

PRESENTADO A:

DR. ISMAEL SAGREDO OLIVENZA

UNIVERSIDAD INTERNACIONAL DE LA RIOJA

MÁSTER EN INGENIERIA DE SOFTWARE Y SISTEMAS INFORMATICOS

COMPUTACIÓN EN EL CLIENTE WEB

AÑO 2020

CONTENIDO

TABLA DE ILUSTRACIONES

OBJETIVOS

Utilizar las herramientas que posee VS Code para realizar peticiones a bases de datos cumpliendo con los estándares Web que se han creado para tal fin desde el año 2005.

Trabajar con estándares web relacionados con conexiones AJAX, funciones asíncronas y componentes para realizar peticiones a una base de datos y representar la información obtenida en una página Web.

Estudiar cada una de las maneras de hacer peticiones AJAX desde la antigüedad, hasta nuestros días para la comunicación entre clientes web y servidores.

INSTALACIÓN

Bootstrap:

Es un framework CSS front-end para crear páginas web, dispone de una serie de recursos que simplifican el desarrollo de sitios web con HTML5, CSS3 y jQuery, de manera que facilita la labor de diseño y desarrollo de toda clase de proyectos Web. En VS Code está disponible una extensión para utilizar este framework, para ello se abre la pestaña extensiones como se muestra en la Ilustración 1. (CANINOS, 2015)

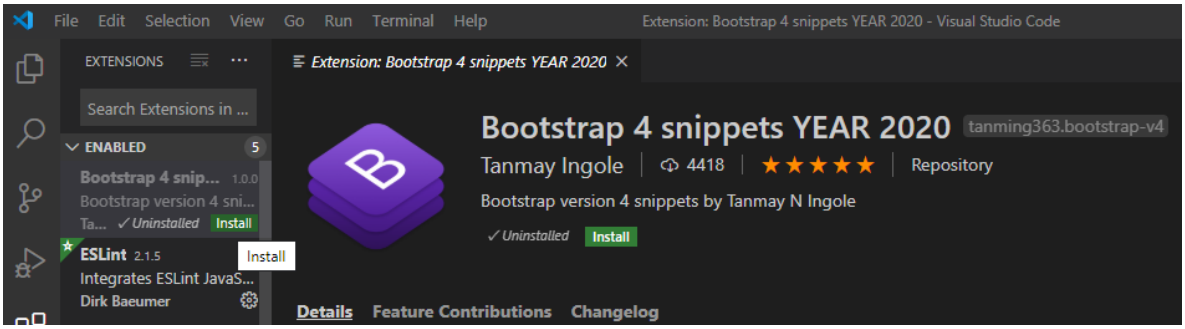

Ilustración 1. Instalación de Bootstrap en VS Code

HTML Boilerplate:

La extensión HTML5 Boilerplate (H5BP) agiliza el proceso de construcción de sitios web, permite añadir fácilmente toda una estructura de un sitio HTML5 a partir de una plantilla que sirve como base para que los desarrolladores web la adopten en sus diseños. Con H5BP podemos obtener código para la normalización de los navegadores, performance, optimizaciones para navegadores en dispositivos móviles, clases específicas para IE, clases Javascript, para estilos basados en capacidades del navegador carga de página más rápida entre otras funcionalidades que simplifican el desarrollo de software. (HTML5, 2013)

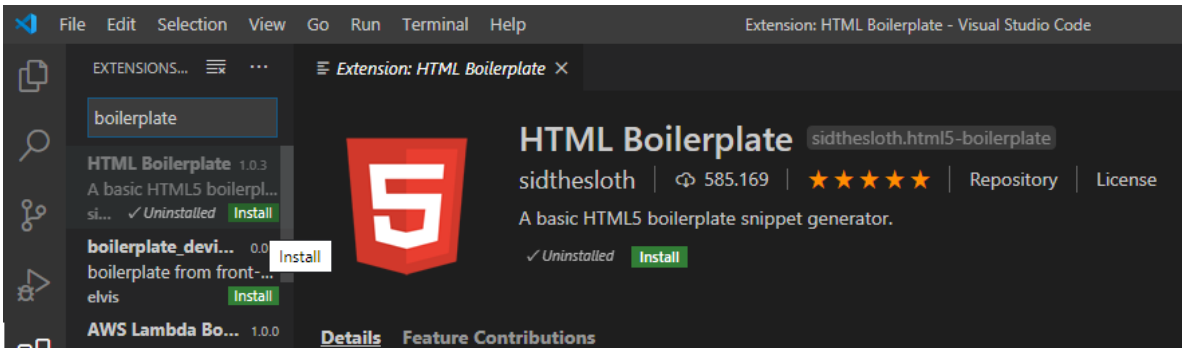

Ilustración 2. Instalación del HTML Boilerplate con VS Code.

DESARROLLO

En este laboratorio se utiliza una base de datos llamada ICNDB, inspirada en el famoso actor y experto en artes marciales conocido como Chuck Norris, que se encuentra en la página Web: http://www.icndb.com/, aquí se almacenan cientos de chistes inspirados en su carrera. Esta base de datos cuenta con una API REST o protocolo de intercambio y manipulación de datos, funciona como una interfaz entre sistemas que usan HTTP para obtener datos en múltiples formatos, como XML y JSON. (API_Market, 2016)

Ilustración 3. Logo de la base de datos ICNDb.com

Se usarán las siguientes tecnologías de comunicación entre clientes web y servidores para cumplir con los objetivos planteados:

- El objeto XMLHttpRequest.
- Las funciones AJAX del archiconocido framework jQuery.
- Un complemento específico para jQuery creado por el mantenedor del servicio al que nos vamos a conectar.
- Por último, implementación de un componente web moderno (Web component) desarrollado por un tercero que nos permitirá realizar peticiones parecidas, pero de manera increíblemente elegante.

RESOLUCIÓN DEL EJERCICIO A LA MANERA DE 2005

Mediante el objeto `XMLHttpRequest` es posible hacer peticiones AJAX a través de clientes Web hacia servidores de la siguiente manera:

```
<script type="text/javascript">
    window.onload = function(){
        xmlhttp = new XMLHttpRequest();
        xmlhttp.open('GET', 'http://api.icndb.com/jokes/random/', true);
        xmlhttp.onreadystatechange = function(){
            var textoChiste = JSON.parse(this.response).value.joke;
            console.log('chiste recibido: ' + textoChiste);
            var h1s = document.getElementsByTagName('h1');
            h1s[0].innerHTML = textoChiste;
        }
        xmlhttp.send();
    }
</script>
```

Ilustración 4. Petición con el objeto `XMLHttpRequest.`

Jumbotron es un componente ligero y flexible usado para mostrar contenido estilizado, se puede expandir en toda la ventana gráfica para mostrar mensajes personalizados. Utiliza clases de utilidad para la tipografía y el espaciado del contenido dentro de contenedores más grandes. (JUMBOTRON, s.f.)

Para aplicar este estilo y sustituir el texto del chiste en el título central de un elemento jumbotron del framework Bootstrap se hace de la siguiente manera:

```
<main role="main">
  <div class="jumbotron">
    <div class="container">
      <h1 class="display-3">Aqui va el chiste!</h1>
    </div>
  </div>
</main>
```

Ilustración 5. Elemento jumbotron para sustituir el texto del chiste.

Como se observa en la Ilustración 5, la etiqueta "`<main>`" sirve para definir el contenido principal y destacado del DOM. La clase CSS jumbotron se implementa dentro de la etiqueta

“ ”, estas crean un contenedor en bloque que se define con una clase CSS a través del atributo `class`. Al utilizar la clase “jumbotron” se muestra un título que representa el contenido destacado de una página.

Para utilizar Bootstrap se debe asegurar que las páginas están configuradas con los últimos estándares de diseño y desarrollo usando un doctype HTML5 e incluir una etiqueta “meta” para configuración del “viewport” que a su vez se encuentra dentro de la etiqueta “<head>”, así se obtiene el comportamiento adecuado para las diferentes resoluciones de pantalla que disponen los dispositivos modernos. (CARBALLO, 2012)

```
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```

Ilustración 6. Estándares de diseño, desarrollo y meta etiqueta de ventana grafica para el uso de Bootstrap.

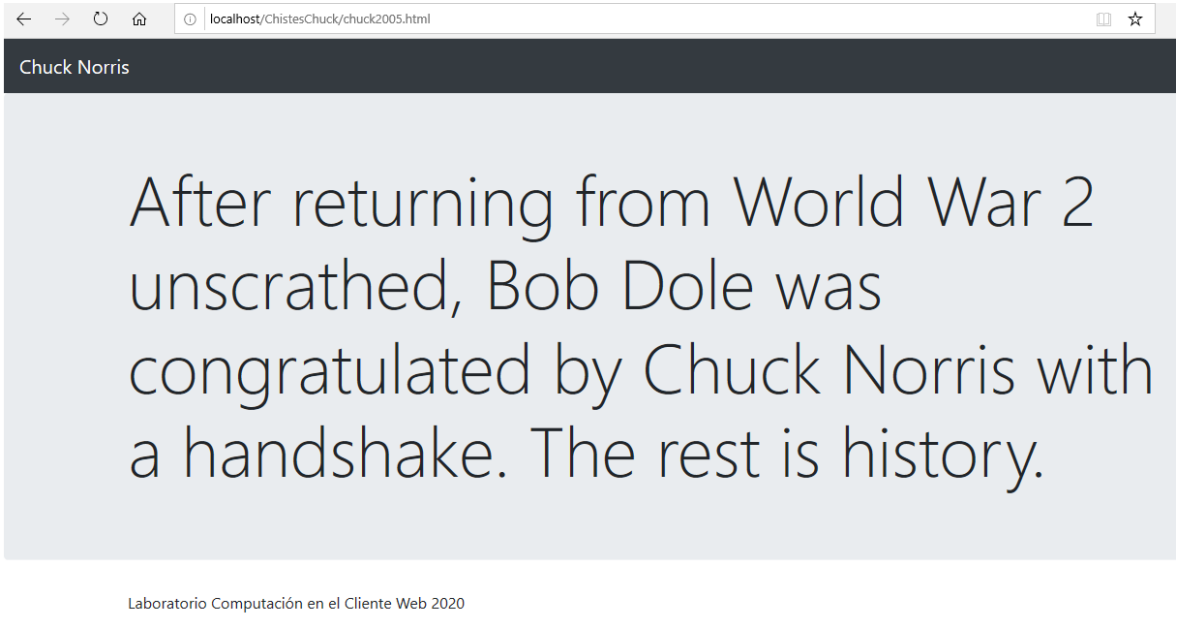

Ilustración 7. Página Web con el resultado de la petición a la manera 2005.

RESOLUCIÓN DEL EJERCICIO A LA MANERA DE 2006

Al implementar la librería jQuery, se incorpora una nueva instrucción muy útil para realizar peticiones tal como se hacían en el 2006, para ello se debe enlazar el sitio Web con el recurso que provee dicha librería, por lo que se procede a crear otro fichero llamado "chuck2006.html" que utiliza el código JavaScript mostrado en la Ilustración 8 para conseguir un funcionamiento parecido al del ejercicio anterior (Ilustración 4):

```
<script type="text/javascript">
    window.onload = function(){
        $.get("http://api.icndb.com/jokes/random", (response) => {
            var textoChiste = response.value.joke;
            $('h1').text(textoChiste);
        });
    }
</script>
```

Ilustración 8. Instrucción `$.get()` del framework jQuery.

Para enlazar con la biblioteca se utiliza la etiqueta "`<script>`" y mediante el atributo `src` se identifica la URL en donde se encuentra alojado jQuery. Cabe resaltar que el atributo `integrity` permite que el navegador verifique la fuente del archivo para asegurarse de que el código nunca se cargue si la fuente ha sido manipulada. Define un valor hash de un recurso (como una suma de comprobación) que debe coincidir para que el navegador lo ejecute. El hash asegura que el archivo no se modificó y contiene los datos esperados. De esta manera, el navegador no cargará recursos diferentes (por ejemplo, maliciosos). El atributo `crossorigin` permite configurar las peticiones CORS de los datos que se cargan. El Intercambio de Recursos de Origen Cruzado (CORS) es un mecanismo que utiliza cabeceras HTTP adicionales para permitir que un user agent obtenga permiso para acceder a recursos seleccionados desde un servidor, en un origen distinto (dominio) al que pertenece. Un agente crea una petición HTTP de origen cruzado cuando solicita un recurso desde un dominio distinto, un protocolo o un puerto diferente al del documento que lo generó. (MDN, 2019)

```
<script src="https://code.jquery.com/jquery-3.5.0.min.js"
integrity="sha256-xNzN2a4ltkB44Mc/Jz3pT4iU1cmeR0FkXs4pru/JxaQ="
crossorigin="anonymous"></script>
```

Ilustración 9. Atributos `src`, `integrity` y `crossorigin` para enlazar con la biblioteca jQuery.

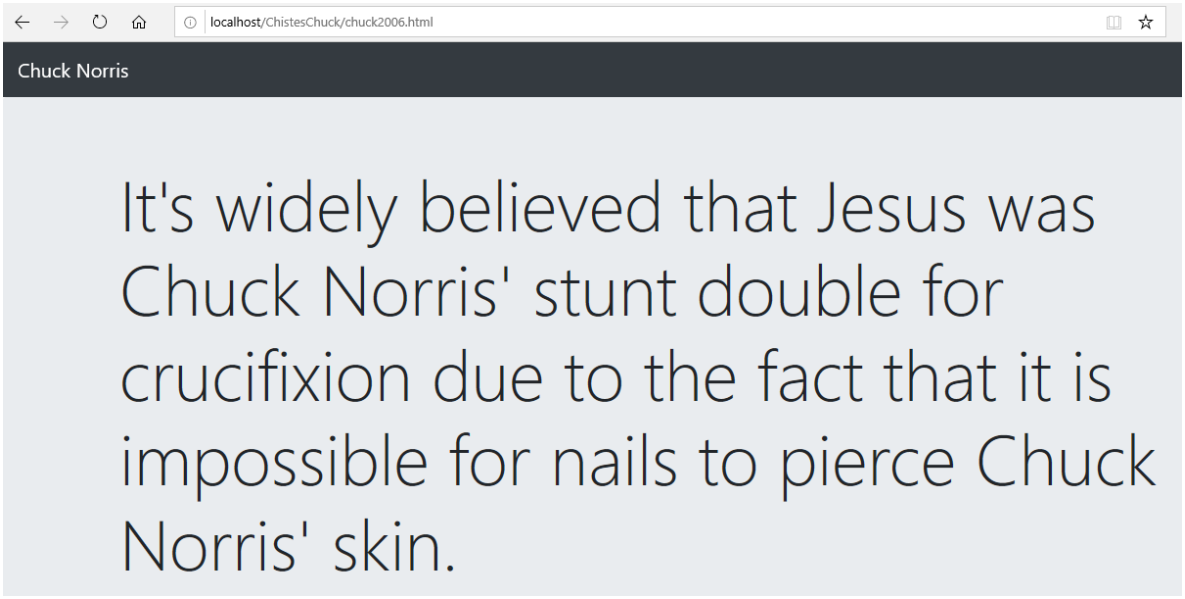

Ilustración 10. Página Web con el resultado de la petición a la manera 2006.

¿Qué diferencias ves con respecto al ejercicio anterior?

Al usar el framework de JQuery se reduce la cantidad de código, de igual manera este mismo se presenta más limpio.

¿Cómo simplifica la vida jQuery?

JQuery nos permite de una manera mucho más fácil el desarrollo y la interacción con el DOM, de igual manera es intuitivo y fácil de usar.

¿Qué ocurre si tenemos varios tags h1?

Al tener varios tags h1, el código permite insertar el resultado obtenido (el chiste) e insertarlo en cada uno de los h1 que encuentre en el DOM.

RESOLUCIÓN CON *PLUGIN* DE jQuery (HASTA 2014)

Este plugin que de manera general se puede definir como una función que se le agrega al objeto jQuery, se ejecuta la instrucción para extraer una lista de chistes, cuando el documento DOM termina de cargarse y activa `function().` Mediante el método `$.icdn.getRandomJokes()` se obtiene una lista de chistes (10 en este caso) utilizando la propiedad `number` de la base de datos gracias a la instrucción `<script`

src="Bootstrap/js/jquery.icndb.min.js" ></script>. Cada elemento que se obtiene se va agregando a la lista mediante el atributo append sin que desaparezca el elemento anterior.

```
<script src="bootstrap/js/jquery.icndb.min.js" ></script>
<script type="text/javascript">
    window.onload = function(){
            $.icndb.getRandomJokes({
                number: 10,
                success: (response) => {
            response.forEach(element => { $("ul.list-group").append('<li class="list-group-item">' + element.joke + '</li>'); });
            }});
    }
</script>
```

Ilustración 11. Uso del pluggin jQuery ICNDB para crear una lista de chistes.

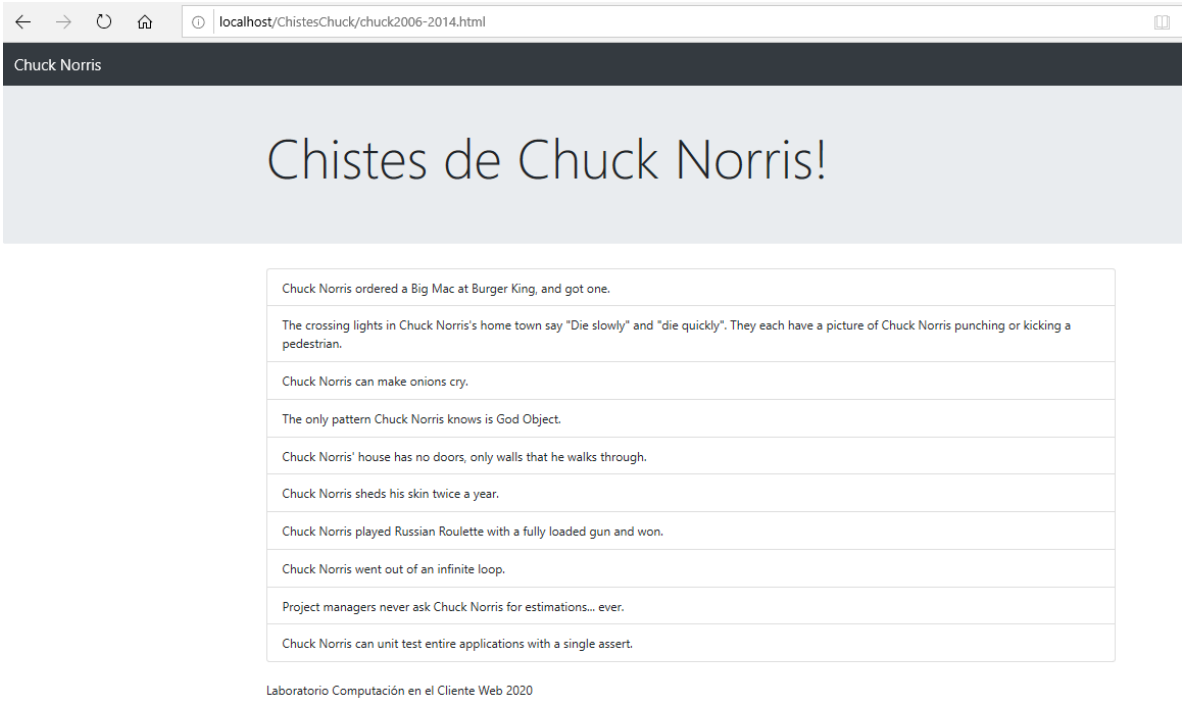

Ilustración 12. Página Web con el resultado de la petición usando el plugin de jQuery.

```
<div class="container">
  <div class="row">
    <div class="col-12">
      <ul class="list-group">
      </ul>
    </div>
  </div>
</div>
```

Ilustración 13. Clase container para crear añadirle características a la lista.

Mediante la clase "`container`" se crea un contenedor de anchura fija y con la clase "row" se crean unas rejillas dentro del contenedor que en este caso se usan para separar cada chiste de la lista teniendo una división de un ancho definido con la clase "`col-12`".

¿cómo se escribían las funciones en las versiones de ECMAScript previas a la versión 6?

En las versiones de ECMAScript previas a la 6, solo se podían escribir las funciones de una forma tradicional con la palabra reservada `function`, dándole un nombre y permitiendo parámetros de entrada. En la versión 6 se agregaron una variedad de formas para escribir funciones, como por ejemplo las funciones flecha (del inglés, Arrow functions) que son fácilmente identificados al implementar la sintaxis "=>".

RESOLUCIÓN EN 2014

Para desarrollar este ejercicio se utilizará el método `fetch`, el cual fue definido por la WHATWG, y es un nuevo estándar de interacción por HTTP basado en promesas el cual funciona desde `window` como desde worker.

Para su implementación hacemos su llamado como una propiedad de `window`, se pasa como parámetro la URL del servicio y variables de configuración como el método (en este caso el método GET). Posteriormente, el resultado de la promesa es convertido a formato JSON y se devuelve como promesa para que a través del iterador `forEach` sea insertado cada ítem (elemento `li`) a la lista del DOM.

```
<script type="text/javascript">
    window.fetch('http://api.icndb.com/jokes/random/10',
    {
        method: 'get'
    })
    .then(function(response) {
        return response.json();
    })
    .then(function(data) {
      var ul = document.getElementById("list");
      jokes = data.value;
      jokes.forEach(element => { ul.innerHTML += '<li class="list-group-item">' + element.joke + '</li>'; });
    });
</script>
```

Ilustración 14. Uso de fetch.

```
<div class="container">
  <div class="row">
    <div class="col-12">
      <ul class="list-group" id="list">
      </ul>
    </div>
  </div>
</div>
```

Ilustración 15. Lista a aplicar resultado.

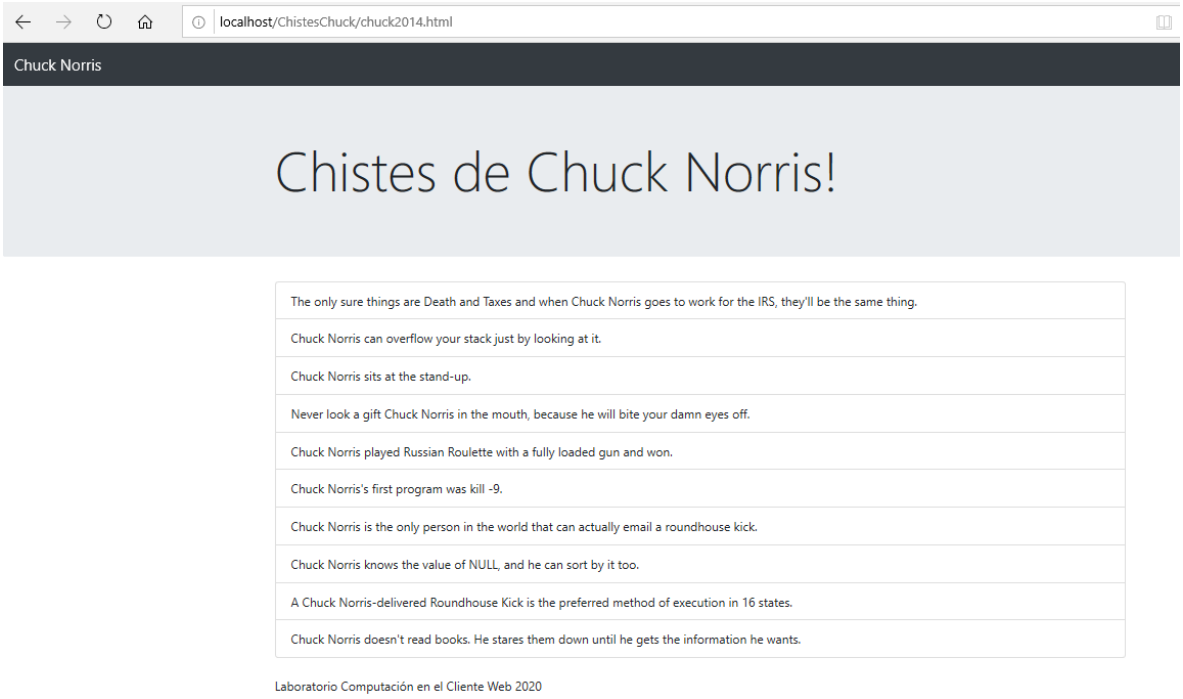

Ilustración 16. Página Web con el resultado de la petición a la manera 2014.

Para realizar la prueba a través del fetch con NodeJS, debemos haber instalado node en nuestro equipo, el cual se puede descargar desde el siguiente enlace: https://nodejs.org/es/download/ .

Después creamos un archivo .js para este caso chuck-node.js el cual contendrá el siguiente código:

```
chuck-node.js > ...
1   const fetch = require('node-fetch');
2
3   fetch('http://api.icndb.com/jokes/random/10',
4           {
5               method: 'get'
6           })
7           .then(function(response) {
8               return response.json();
9           })
10          .then(function(data) {
11            jokes = data.value;
12            jokes.forEach(element => {  console.log("- " + element.joke + "\n"); });
13          });
```

Ilustración 17 Codigo fetch-node

El siguiente paso es abrir una consola e ir a la ruta de la carpeta donde se encuentra el archivo creado y ejecutamos el siguiente comando para agregar el módulo de node:

`npm i node-fetch --save`

Por último, ejecutamos el archivo creado con el comando `node chuck-node.js`

```
DEBUG CONSOLE  PROBLEMS  OUTPUT  TERMINAL                    1: powershell
Debug Console (Ctrl+Shift+Y)
PS C:\Users\duvan\Downloads\ChistesChuck-master> node chuck-node.js
- When Arnold says &quot;I'll be back&quot; in Terminator movie it is implied that he's going to ask Chuck No

- Chuck Norris does not need to know about class factory pattern. He can instantiate interfaces.
PS C:\Users\duvan\Downloads\ChistesChuck-master> node chuck-node.js
                                                                                - Chuck Norris sheds
 his skin twice a year.

- Chuck Norris is the reason why Waldo is hiding.

- Scientifically speaking, it is impossible to charge Chuck Norris with &quot;obstruction of justice.&quot; This is be
cause even Chuck Norris cannot be in two places at the same time.

- All arrays Chuck Norris declares are of infinite size, because Chuck Norris knows no bounds.

- While urinating, Chuck Norris is easily capable of welding titanium.

- Chuck Norris has the greatest Poker-Face of all time. He won the 1983 World Series of Poker, despite holding only a
Joker, a Get out of Jail Free Monopoly card, a 2 of clubs, 7 of spades and a green #4 card from the game UNO.

- Noah was the only man notified before Chuck Norris relieved himself in the Atlantic Ocean.

- The word 'Kill' was invented by Chuck Norris. Other words were 'Die', 'Beer', and 'What'.

- Chuck Norris already went to Moon and Mars, that's why there are no signs of life.

- Chuck Norris programs occupy 150% of CPU, even when they are not executing.

PS C:\Users\duvan\Downloads\ChistesChuck-master>
```

Ilustración 18 Ejecución node-fetch

Podemos evidenciar como se nos muestra en consola el resultado de la petición.

¿Qué es el WHATWG (https://whatwg.org/)?

El WHATWG (Web Hypertext Application Technology Working Group) es una comunidad de personas y empresas interesadas en la evolución de HTML y tecnologías conexas.

RESOLUCIÓN DEL EJERCICIO CON WEB COMPONENTS

Mediante el uso de Web Components se puede obtener el contenido de la API de ICNDB. Esto se puede realizar de dos maneras:

- Haciendo una referencia mediante el tag `<link rel="import">` hacia el repositorio del sitio web del componente.
- O haciendo el import de manera local instalando el Web Component a través de un gestor de paquetes como Bower por ejemplo.

Link a URL

Bajo el primer método basta con insertar la siguiente línea, que permite importar del Web component `<chuck-norris-joke>` y poderlo utilizar dentro del DOM:

```
<link rel="import"
href="https://raw.githubusercontent.com/erikringsmuth/chuck-norris-
joke/master/chuck-norris-joke.html">
```

Ilustración 19. Referencia para importar del Web Component.

Así mismo, se puede referenciar el framework de maquetado Skeleton mediante las instrucciones

```
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/skeleton/2.0.4/skeleton.css">
```

Ilustración 20. Referencia del framework de maquetado Skeleton.

Hecho esto, se puede generar el contenido de la tabla en la que se insertarán las bromas obtenidas haciendo uso de la etiqueta `<chuck-norris-joke>`. No obstante, para hacerlo de una manera más dinámica se desarrolló una función que genere el contenido de la tabla. Antes de codificar las funciones se declaran dos variables, una para validación y otra que contendrá la etiqueta que hace referencia al Web Component importado previamente.

```
//variable de validación
var bool = false;
//declaración de una variable que contendrá el tag y a la que se llamará
var joke = '<chuck-norris-joke>'
```

Ilustración 21. Variables de validación y para contener el tag.

```
//declaración de la función
function generateJokeTable(){
//inicio del ciclo para la generación de 10 registros de bromas
    for(var i = 0; i <= 10; i++){
//inserción por primera vez del contenido
        $('table').append('<tr><td>'+joke+'</td></tr>')
        console.log(joke)
    }
    bool = true;
}
```

Ilustración 22. Función para generar el contenido inicial.

```
//declaración de la función para agregar mas bromas al inicio de la tabla
function addjokes(){
//inicio del ciclo para agregar 10 registros nuevos de bromas
    for(var i = 0; i <= 10; i++){
//el método prepend permite agregar registros por encima del
//primer 'hijo' del elemento al que se busca
        $('table').prepend('<tr><td>'+joke+'</td></tr>')
        console.log('adding'+joke)
    }
}
```

Ilustración 23. Función para agregar contenido después de una primera ejecución.

```
//declaración de función global que invocará a las otras funciones
function getJokes(){
// validación de existencia de registros en la tabla
    if(bool === true){
        addjokes();
    } else{generateJokeTable();}
}
```

Ilustración 24. Función que invocara a las funciones anteriores.

```
//llamada a la funcion para generar el contenido al cargar la ventana
window.onload = function(){
    generateJokeTable();
}
```

Ilustración 25. Función para generar el contenido.

```
<!-- custom skeleton -->
<style>
    body {
        font-size: 1.5em;
        line-height: 1.6;
        font-weight: 400;
        font-family: "Raleway", "HelveticaNeue", "Helvetica Neue", Helvetica, Arial, sans-serif;
        color: #222;
    }

    .button {
        border-radius: 100px;
    }

    .section-heading,
    .section-description {
        margin-top: 1.5em;
        margin-bottom: 1.2em;
    }
</style>
```

Ilustración 26. Sobreescritura y personalización de los estilos de Skeleton.

Para visualizar la petición con Web Components se pueden usar los navegadores Opera, Blisk o Edge en su ultima versión.

Hecho esto, se obtiene el siguiente resultado:

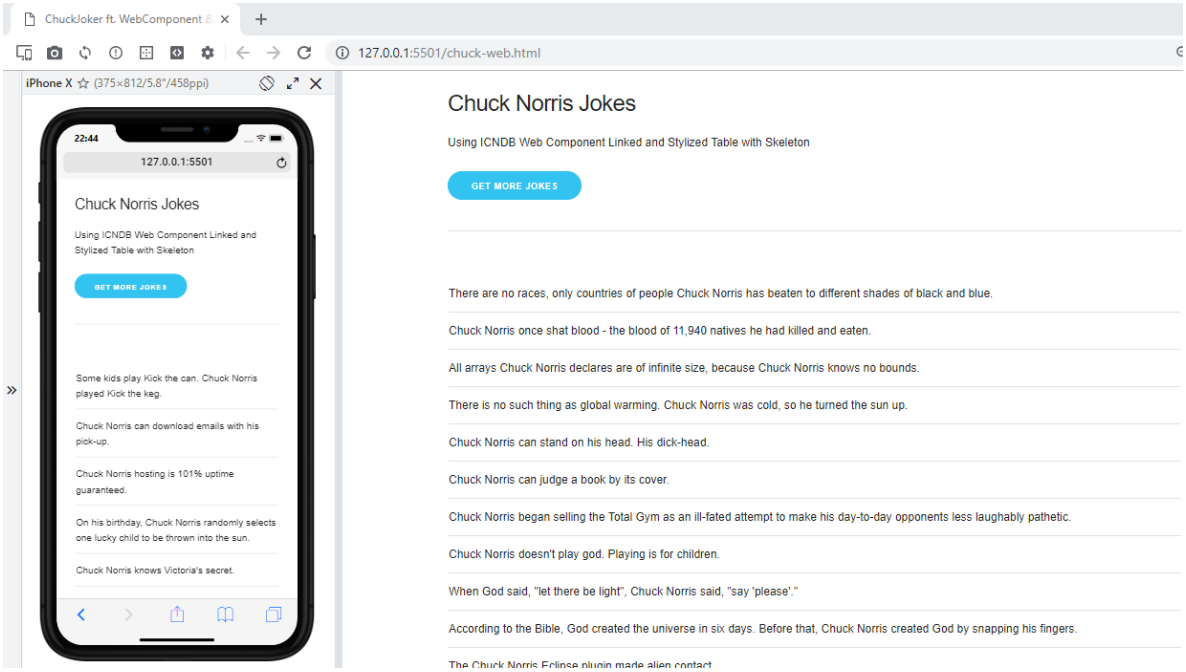

Ilustración 27. Página Web con el resultado de la petición con Web Components.

Link al web Component Local

Bajo este método es necesario instalar el componente de manera local con la siguiente línea

```
bower install chuck-norris-joke --save
```

Esto genera un archivo: bower.json, en el que se almacenan todas las dependencias del proyecto para poder actualizarlas cuando sea necesario. Así mismo, genera una estructura de directorio:

```
+bower_components
 L chuck-norris-jokes
  L .bower.json (dependencias para el gestor de paquetes p.e. npm)
  L bower.json (dependencias para bower)
  L chuck-norris-joke.html (código del Tag requerido)
  L demo.html
  L LICENSE.md
  L README.md
```

Seguido de esto, basta con hacer la referencia tal cual el método anterior, pero hacia el directorio local en el atributo `[href]`.

De la misma manera se puede instalar el framework Skeleton con la siguiente línea

```
bower install skeleton-framework --save
```

Finalmente, las referencias hacia el Web Component y el framework Skeleton se escriben así

```
<!-- Skeleton -->
<link rel="stylesheet" href="./Skeleton/css/normalize.css">

<!-- web component -->
<link
rel="import"
href="./bower_components/chuck-norris-joke/chuck-norris-joke.html">
```

Ilustración 28. Referencias hacia el Web Component y el framework Skeleton.

BIBLIOGRAFIA

API_Market, B. (23 de MARZO de 2016). *API REST*. Obtenido de API REST: https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos

CANINOS, N. (06 de SEPTIEMBRE de 2015). *bootstrap*. Obtenido de bootstrap: https://www.negocioscaninos.com/que-es-bootstrap-bootstrap-framework-front-end/

CARBALLO, R. G. (07 de NOVIEMBRE de 2012). *ETIQUETAS*. Obtenido de TODO HTML: https://sites.google.com/site/dwebhtml/reference

HTML5. (2013). *Que es HTML5 Boilerplate*. Obtenido de Que es HTML5 Boilerplate: https://www.htmlcinco.com/html5-boilerplate/#comments

JUMBOTRON, C. (s.f.). *jumbotron*. Obtenido de jumbotron: https://www.tutorialesprogramacionya.com/cssya/bootstrapya/detalleconcepto.php?codigo=149

MDN. (06 de MAYO de 2019). *MDN web docs* . Obtenido de Control de acceso HTTP (CORS): https://developer.mozilla.org/es/docs/Web/HTTP/Access_control_CORS

CONCLUSIONES

VS Code es un potente editor de código usado en desarrollo, tiene incorporado multitud de funcionalidades que facilitan el trabajo del desarrollo Web, es así como se han utilizado en este laboratorio las extensiones HTML 5 Boilerplate y Bootstrap, que trabajan con los estándares usados en la mayoría de las aplicaciones Web.

Se ha desarrollado el código para realizar peticiones a la base de datos ICNDB y se han comprendido las características propias de cada uno de los estándares Web que han surgido desde el año 2005.