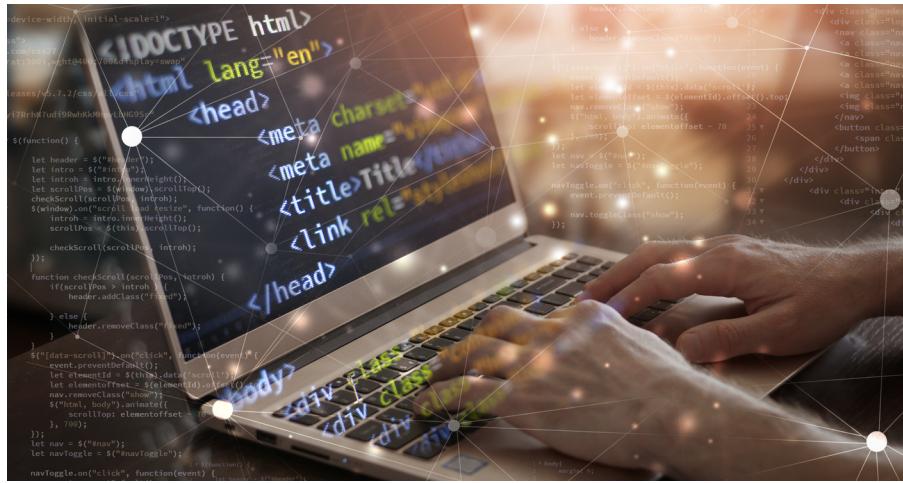




Back end nivel intermedio

**Conectando el front
end con el back end**

Para que un desarrollo web esté completo es necesario establecer la conexión entre el front end y el back end, ya que sin uno el otro no podrá funcionar. Debido a esto, existen diferentes formas de realizar dichas conexiones, la más popular y conocida es mediante JavaScript, el cual permitirá conectar el front end con el back end incluyendo API existentes o API de tu creación.



A su vez, es de suma importancia realizar las pruebas pertinentes para asegurarte que el desarrollo funciona de acuerdo con la solicitud que los clientes han realizado.

Uso de JavaScript

Dentro de un desarrollo web existen tres elementos principales:

- HTML.
- CSS.
- JavaScript.

JavaScript está enfocado en brindar la funcionalidad al sitio web y como has visto, se puede utilizar tanto en front end como en back end.

A continuación, verás cómo realizar la conexión del front end con el back end mediante JavaScript a través del siguiente caso:

Te encuentras desarrollando una página web, donde necesitas conectarte a otro servicio (back end). Este servicio es el de meteorología, ya que tu desarrollo consiste en una página web que deja escribir el país o ciudad donde estás e informa el clima.

Para lo anterior, deberás crear un archivo JavaScript que sea capaz de obtener dicha información.

1. Primero identifica la API de la cual podrás obtener los datos meteorológicos. Para este ejercicio se utilizará OpenWeatherMap.

a. Ingresa a la siguiente liga:



OpenWeather. (2022). Create New Account. Recuperado de https://home.openweathermap.org/users/sign_up

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

b. Realiza tu registro.

c. Una vez que te registres obtendrás una clave API, la cual utilizarás en el archivo JavaScript.

2. Comienza a crear el archivo JavaScript.

a. Crearás tres constantes que permitirán obtener lo escrito por el usuario:

```
const form = document.querySelector(".top-banner form");
const input = document.querySelector(".top-banner input");
const msg = document.querySelector(".top-banner .msg");
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

- b. Despues, crearás una constante que llamará al método Ajax para obtener la ciudad del usuario. La linea de código será la siguiente:

```
const nombre_de_la_costante = document.querySelector(".ajax-section .cities");
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

- c. Luego crea una constante que contenga la llave de la API que obtuviste:

```
const apiKey = "Llave de API Obtenida";
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

- d. Usa la función `fetch` para enviar un valor y obtener una respuesta.

```
fetch(url)
  .then(response => response.json())
  .then(data => {
    const { main, name, sys, weather } = data;
    const icon = `https://s3-us-west-2.amazonaws.com/s.cdpn.io/162656/${weather[0]["icon"]}.svg`;

    const li = document.createElement("li");
    li.classList.add("city");
    const markup = `
      <h2 class="city-name" data-name="${name},${sys.country}">
        <span>${name}</span>
        <sup>${sys.country}</sup>
      </h2>
      <div class="city-temp">${Math.round(main.temp)}<sup>°C</sup></div>
      <figure>
        
        <figcaption>${weather[0]["description"]}</figcaption>
      </figure>
    `;
    li.innerHTML = markup;
    list.appendChild(li);
  })
  .catch(() => {
    msg.textContent = "Escribe una ciudad válida 😞";
  });
}

msg.textContent = "";
form.reset();
input.focus();
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

- e. Previo a esto, declara una constante para la URL que se enviará con los datos del APIKEY y la ciudad escrita por la persona.

```
const url = `https://api.openweathermap.org/data/2.5/weather?q=${inputVal}&appid=${apiKey}&units=metric`;
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

f. Obtendrás el siguiente resultado.

```
const url = `https://api.openweathermap.org/data/2.5/weather?q=${inputVal}&appid=${apiKey}&units=metric`;

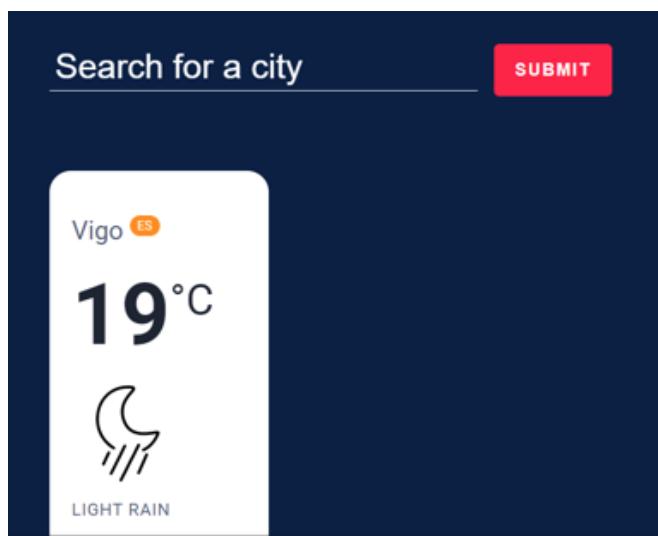
fetch(url)
  .then(response => response.json())
  .then(data => {
    const { main, name, sys, weather } = data;
    const icon = `https://s3-us-west-2.amazonaws.com/s.cdpn.io/162656/${weather[0]["icon"]}.svg`;

    const li = document.createElement("li");
    li.classList.add("city");
    const markup = `
      <h2 class="city-name" data-name="${name},${sys.country}">
        <span>${name}</span>
        <sup>${sys.country}</sup>
      </h2>
      <div class="city-temp">${Math.round(main.temp)}<sup>°C</sup></div>
      <figure>
        
        <figcaption>${weather[0]["description"]}</figcaption>
      </figure>
    `;
    li.innerHTML = markup;
    list.appendChild(li);
  })
  .catch(() => {
    msg.textContent = "Escribe una ciudad válida 😞";
  });

msg.textContent = "";
form.reset();
input.focus();
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

g. Ejecuta tanto el HTML, como el CSS y el JS, para obtener un resultado similar al siguiente:



Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

Lo anterior, es uno de muchos ejemplos que existen para realizar una conexión con otro servicio.

Recuerda que si tú creas el servicio back end, este debe encontrarse en la red; de tal forma que JavaScript pueda establecer la conexión mediante la URL enviada en el fetch.

Uso de API

El desarrollo web involucra mucho trabajo y mucha determinación, por lo cual existen ciertas funcionalidades que son complicadas de elaborar desde cero. Debido a esto, surgen las API (Application Programming Interface). Estas API permitirán establecer una conexión entre tu front end y otro servicio back end.

Un ejemplo de una API podría ser el ejemplo que acabas de ver de meteorología. Alguien se dio a la tarea de poder realizar una conexión con un satélite que se encuentra en el espacio, de tal forma que ese satélite regresa a la API la información del clima al front end; en este caso el back end es el servidor al cual se encuentra conectado el satélite y es el que regresará la información a la API para que ahí vaya al front end.

De acuerdo con Amazon Web Services (s.f.), existen diferentes clasificaciones para las API, estas clasificaciones están basadas en su orientación. Para este tema te enfocarás en las API de servicios web. Estas API se desarrollan para brindar acceso a un servicio a través de una URL.

Existen diferentes servicios:

- a. REST: son las más populares y consisten en que el usuario envía ciertas solicitudes al servidor y son entregadas como datos. Este servidor usa dichos datos para iniciar las funciones internas y devuelve la información solicitada al cliente.
- b. SOAP: consiste en el intercambio de información mediante XML, actualmente no es tan utilizada.
- c. XML-RPC: conocida como procedimientos remotos y consiste en finalizar una función o procedimiento en el servidor (por parte del cliente), de tal forma que el servidor devuelva el resultado a dicho cliente. Esto se realiza mediante XML.
- d. JSON-RPC: conocida como procedimientos remotos y consiste en la finalización de una función o procedimiento en el servidor (por parte del cliente), de tal forma que el servidor devuelva el resultado a dicho cliente. Esto se realiza mediante objetos JSON.

El ejercicio realizado anteriormente contiene una REST API, ya que directamente desde JavaScript se está enviando la información a la API, esta lo procesa y regresa lo solicitado.

Uso de JSON

De acuerdo con JSON (s.f.), es un formato que permite el intercambio de datos. Sus siglas en inglés significan *notación de objetos de JavaScript*. Dado que su estructura es muy fácil de utilizar, los usuarios pueden leer y escribir fácilmente, además los sistemas lo pueden interpretar y generar de forma simple.

La estructura de un archivo JSON está compuesta de la siguiente forma:



Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

Retoma el ejemplo de la página web que brinda el clima de acuerdo con la ciudad que hayas escrito. ¿qué pasaría si el cliente solicita que automáticamente indique el clima de acuerdo con la ciudad donde se encuentra el usuario?

En ese caso será de mucha ayuda el archivo JSON. Dentro del archivo definirás la longitud y la latitud donde se encuentra el usuario. Para esto, conectarás tu desarrollo web a otra API mediante la siguiente URL:



JSONPlaceholder. (2022). *freegeoip*.
Recuperado de <https://freegeoip.app/json/>

Los siguientes enlaces son externos a la Universidad Tecmilenio, al acceder a ellos considera que debes apegarte a sus términos y condiciones.

Esta devolverá la ubicación exacta del usuario mediante un archivo JSON. Este archivo será enviado mediante JavaScript para hacer la conexión con la API y así obtener el resultado esperado. Importante considerar lo siguiente: la liga de JSON mostrará el resultado si se siguen los pasos del tema, de lo contrario saldrá como error.

El archivo de JavaScript sería el siguiente:

```
const API_ENDPOINT = "https://freegeoip.app/json/";
fetch(API_ENDPOINT)
.then(response => response.json())
.then(datosUbicacion => {
    // Imprimir los datos de la ubicación
    console.log(datosUbicacion);
    // Recuerda que podemos acceder a latitud y longitude, entre otros
    const latitud = datosUbicacion.latitude,
          longitud = datosUbicacion.longitude;

    console.log(`Tus coordenadas son ${latitud},${longitud}`);
});
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

El archivo JSON obtenido sería el siguiente:

```
{
  "ip": "12.34.56.78",
  "country_code": "MX",
  "country_name": "México",
  "region_code": "ASD",
  "region_name": "NombreRegión",
  "city": "Ciudad",
  "zip_code": "Código postal",
  "time_zone": "America/Mexico_City",
  "latitude": 0.0,
  "longitude": 0.0,
  "metro_code": 0
}
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

Una vez enviado el contenido de JSON, puedes crear un archivo JavaScript que establezca la conexión con la API del clima.

El archivo sería el siguiente:

```
const requestURL = 'https://mdn.github.io/learning-area/javascript/oojs/json/nombredelarchivo.json';
const request = new XMLHttpRequest();
request.open('GET', requestURL);
request.responseType = 'json';
request.send();
request.onload = function() {
  const ubicacion = request.response;
}

const url = `https://api.openweathermap.org/data/2.5/weather?q=${ubicacion}&appid=${apiKey}&units=metric`;
```

Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

De esta forma estás combinando JSON y JavaScript para establecer una conexión con una API, para obtener la información de forma dinámica, ya que estás enviando la ubicación real del usuario en tiempo real.

Pruebas de testing

Para cualquier desarrollo de software existe un ciclo de vida, el cual está conformado de diferentes etapas. Una de las últimas etapas es conocida como software testing, consiste en realizar diferentes pruebas al desarrollo para asegurar que el producto cumple con la funcionalidad requerida y que además lo hace de manera óptima.



Retoma el ejemplo anterior del desarrollo web para obtener el clima:

Después de meses invertidos por fin terminas el desarrollo y estás listo para mostrarlo al cliente. Llegas a la reunión y al momento de empezar a ejecutarlo te das cuenta de que no funciona del todo bien. Los clientes empiezan a molestarse e impacientarse, dado que se les había comunicado que todo estaba listo y que saldrían de esa reunión con un software 100% funcional. Tu jefe se ve disgustado por lo que está pasando y decide terminar la reunión antes de lo previsto, pide una disculpa a los clientes y les comenta que no volverá a ocurrir, y en 24 horas tendrán su producto funcionando.

Si tan solo hubieras invertido y organizado un tiempo durante esos meses de desarrollo para ir probando y asegurar que todo funcionara de forma correcta, no hubiera ocurrido este problema. La empresa no vería afectada su reputación, al igual que la tuya, ni la de tu equipo. Los clientes no se hubieran frustrado ni amenazado con irse a otra empresa para que le hagan el producto esperado. La empresa no hubiera tenido que gastar más de lo contemplado para poder solventar tu error y eso no te hubiese costado tu puesto de trabajo.

Lo anterior es un vivo ejemplo de la importancia de realizar pruebas de testing. Ahora bien, ¿qué es el software testing? De acuerdo con IBM (s.f.), el software testing es un proceso en el cual se verifica si el producto tiene un correcto funcionamiento y si lleva a cabo el propósito que le fue definido en un inicio.

Existen siete principios de las pruebas de software, los cuales fueron creados por la ISTQB, la cual es una institución encargada de establecer todos los parámetros, métricas y todo lo relacionado con el tema de software testing. De acuerdo con Black, Graham y Van (2019), se pueden aplicar a cualquier prueba que vayas a realizar:

1. *Las pruebas muestran la presencia de defectos, no su ausencia:* el que tu realices pruebas a un software te ayudará a encontrar los errores que pueden existir, sin embargo, esto no te asegura que tu software se encuentre libre de errores. Las pruebas te ayudarán a minimizar el riesgo de un mal funcionamiento.
2. *Las pruebas exhaustivas son imposibles:* es imposible que puedas probar todas las posibles combinaciones de inputs y outputs del software, por lo cual, lo que debes hacer es un análisis de riesgo y de esta forma enlistar y darle prioridad a las pruebas que tienen mayor importancia.
3. *Las pruebas tempranas ahorran tiempo y dinero:* si decides esperar a que todo el proyecto esté terminado para poder realizar las pruebas te darás cuenta de la infinidad de errores que encontrarás, es por esto por lo que es importante que en cuanto realices un componente o un módulo de tu software empieces a probarlo. Esto permitirá que al momento en que integres el software reduzcas considerablemente los errores, el tiempo en resolverlos y el dinero que invertirás en arreglarlos.

4. *Agrupación de defectos*: los errores suelen agruparse o encontrarse en algún módulo o zona específica del software, es posible que cuando arregles un módulo, surja otro error en otro módulo, esto debido a la conectividad que existen entre estos. Es por esto por lo que es sumamente importante priorizar las pruebas de estos módulos.

5. *La paradoja del pesticida*: este principio es muy parecido a los insectos o inclusive a los virus de las enfermedades. Si utilizas el mismo pesticida o medicamento, el insecto o el virus se van haciendo cada vez más resistentes, lo mismo pasa con los errores. Si ejecutas las mismas pruebas repetidas, a veces dejarás de encontrar errores nuevos, pero esto no te asegura que no existan. Es por esto por lo que debes de cambiar las pruebas para encontrarlos.

6. *Las pruebas dependen del contexto*: de acuerdo con este principio debes adecuar tus pruebas conforme el objetivo, por el cual fue creado tu software. No puedes utilizar la misma prueba para un software que está enfocado en una empresa afianzadora que para una empresa automotriz.

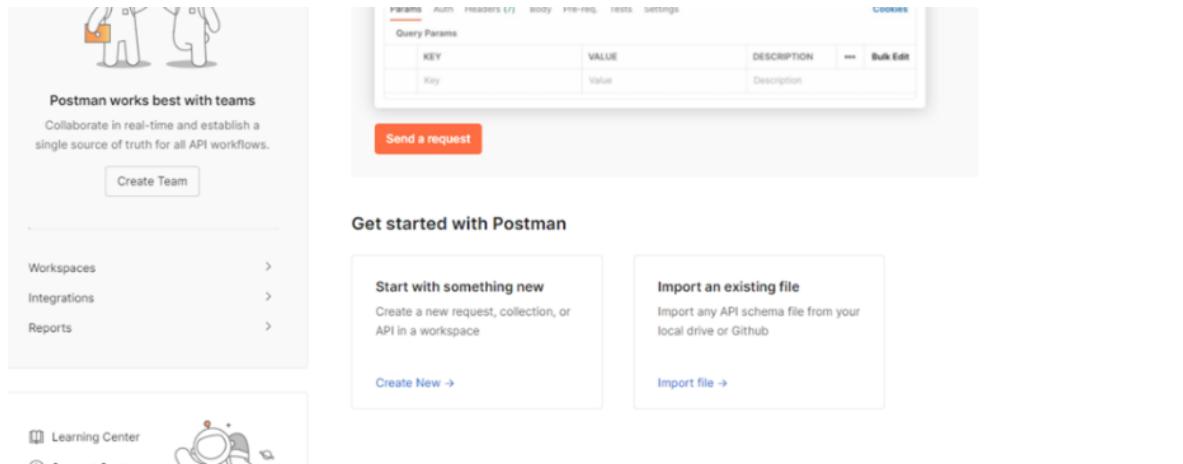
7. *La falacia de ausencia de errores*: este principio explica que, aunque ya hayas corregido todos los errores, esto no exenta a tu producto de que no sea exitoso. Puede ocurrir que tu producto no cumpla con lo que el cliente realmente quería.

Aunado a estos principios, existen diferentes softwares que te ayudarán a realizar las pruebas de testing a tu desarrollo. Algunos de estos softwares se enlistan a continuación:

- Selenium
- Gatling
- Testim
- HeadSpin
- Test Studio
- LoadRunner
- WebLoad
- Blazemeter
- JMeter
- Xray
- TestRail
- Postman

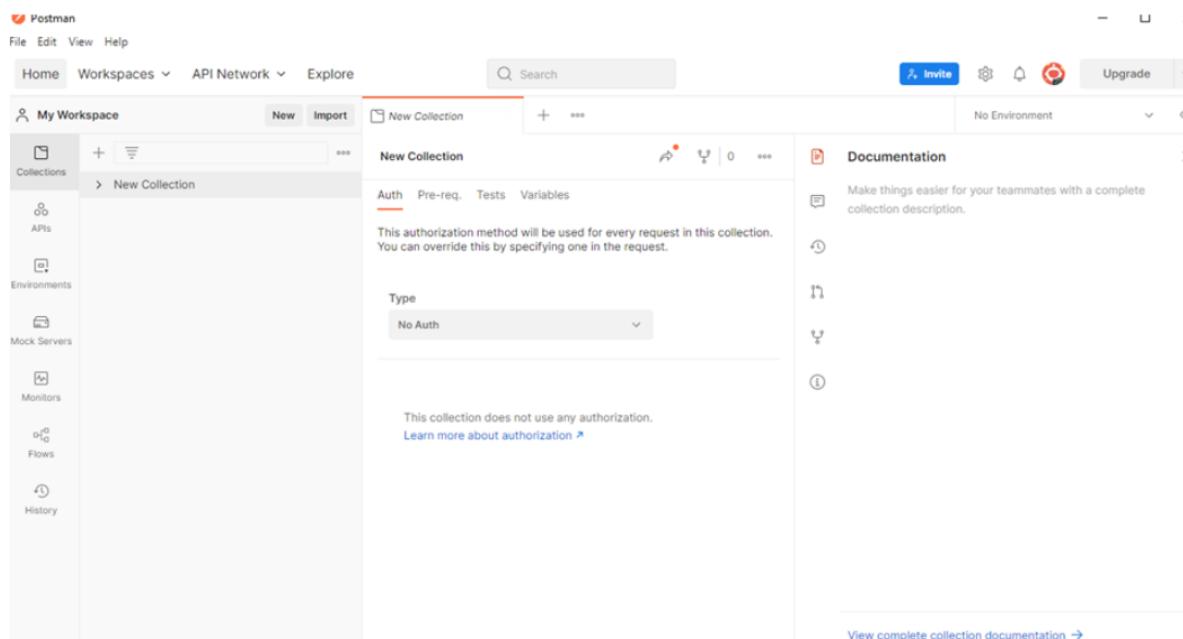
Observa un ejemplo de cómo funcionan este tipo de softwares. Se utilizará Postman para dicho ejemplo. Se va a obtener la longitud y latitud de un usuario.

1. De acuerdo con Alvarado (2020), lo primero que se debe de realizar es la creación de una colección. Para realizar dicha acción deberás hacer clic en “Create Collection”.



Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

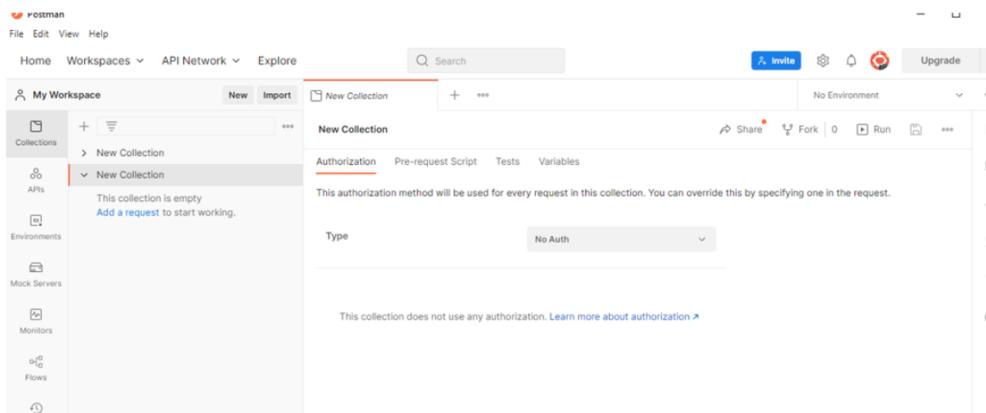
2. Configura los datos referentes a la colección.



Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

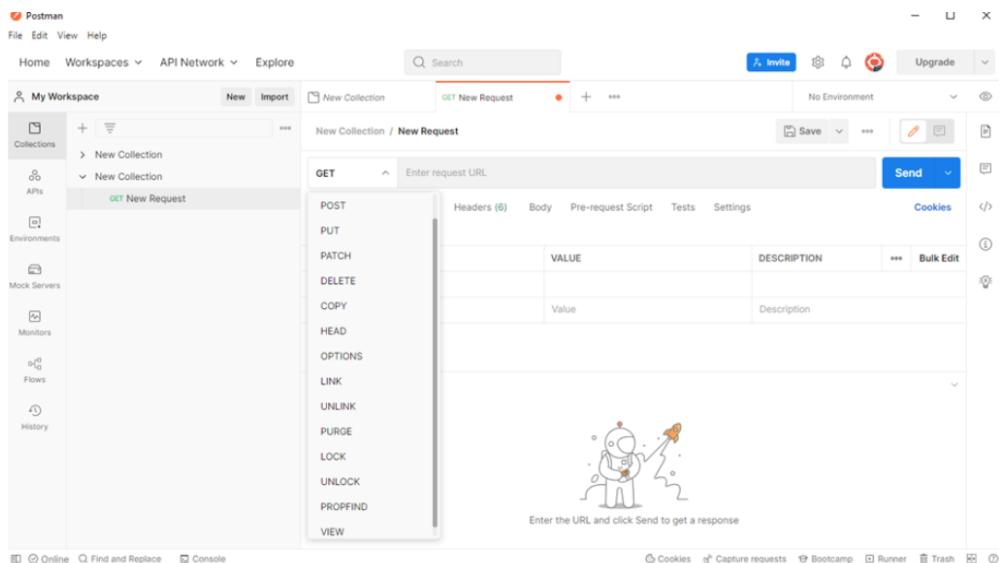
Posterior a esto, deberás crear tu primera solicitud.

3. Para realizar dicha acción debes ir a la colección creada y dar clic en “Add request”.



Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

4. Selecciona el verbo GET y coloca la URL a evaluar.



Esta pantalla se obtuvo directamente del software que se está explicando en la computadora, para fines educativos.

Con esto podrás ver los resultados que te brinda la conexión con dicha URL.

La conexión entre el front end y el back end hará que tu proyecto de desarrollo web se encuentre completo. Para dicha conexión existen diferentes formas de hacerlo, las más comunes son el uso de JavaScript junto con JSON.

A su vez, es de suma importancia que conforme avances en el desarrollo, vayas realizando las pruebas de testing, con el objetivo de asegurar que se cumpla con el propósito que fue definido por el cliente, a la vez su funcionamiento sea el adecuado.

¿Consideras que es necesario realizar las pruebas de testing?

¿Consideras que las pruebas de testing son parte de la conexión entre el front end y el back end?

Referencias bibliográficas

- Alvarado, A. (2020). *Primeros pasos - ¿Como usar Postman? (Serie Postman - Parte 1)*. Recuperado de [https://modest-payne-93a7eb.netlify.app/blog/start-postman-series-1/#:~:text=Crear%20nuestra%20primera%20solicitud%20\(request,documentar%20lo%20que%20estamos%20realizando\)](https://modest-payne-93a7eb.netlify.app/blog/start-postman-series-1/#:~:text=Crear%20nuestra%20primera%20solicitud%20(request,documentar%20lo%20que%20estamos%20realizando)).
- Amazon Web Services. (s.f.). *¿Qué es una API?* Recuperado de <https://aws.amazon.com/es/what-is/api/>
- Black, R., Graham, D., y Van, V. (2019). *Foundations of Software Testing ISTQB Certification* (4^a ed.). Estados Unidos: Cengage.
- JSON. (s.f.). *Introducción a JSON*. Recuperado de <https://www.json.org/json-es.html>
- IBM. (s.f.). *What is software testing?* Recuperado de <https://www.ibm.com/topics/software-testing>

Para saber más

Lecturas

Para conocer más acerca de **conexión entre Front End y Back End**, te sugerimos leer lo siguiente:

- mdn web docs. (s.f.). *Trabajando con JSON*. Recuperado de <https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON>
- geeksforgeeks. (2022). *JavaScript Backend basics*. Recuperado de <https://www.geeksforgeeks.org/javascript-backend-basics/>

Videos

Para conocer más acerca de **conexión entre Front End y Back End**, te sugerimos revisar lo siguiente:

- Raydelto Hernandez (2018, 4 de febrero). *Conectando el Frontend al backend mediante Javascript* [Archivo de video]. Recuperado de https://www.youtube.com/watch?v=4y_0HzAuws&t=332s
- The Coder Cave esp. (2020, 12 de abril). *Curso de JSON - ¿Qué es JSON y para qué sirve?* [Archivo de video]. Recuperado de <https://www.youtube.com/watch?v=ZPsav9k0p2I>

Checkpoints

- Comprender la utilidad de JavaScript para conectar el front end con el back end.
- Comprender cómo utilizar JSON para obtener información de una API REST.
- Comprender la importancia de las pruebas de testing para asegurar el correcto funcionamiento del desarrollo.

Requerimientos técnicos

- JavaScript.
- ipapi.
- Postman.
- Visual Studio.

Prework

- Tener instalado Visual Studio.
- Tener instalado JavaScript.
- El registro en Postman e ipapi se realizará el día de la actividad, sin embargo, se recomienda investigar sus páginas con anticipación.
- Revisar archivo con códigos para la actividad.

La obra presentada es propiedad de ENSEÑANZA E INVESTIGACIÓN SUPERIOR A.C. (UNIVERSIDAD TECMILENIO), protegida por la Ley Federal de Derecho de Autor; la alteración o deformación de una obra, así como su reproducción, exhibición o ejecución pública sin el consentimiento de su autor y titular de los derechos correspondientes es constitutivo de un delito tipificado en la Ley Federal de Derechos de Autor, así como en las Leyes Internacionales de Derecho de Autor.

El uso de imágenes, fragmentos de videos, fragmentos de eventos culturales, programas y demás material que sea objeto de protección de los derechos de autor, es exclusivamente para fines educativos e informativos, y cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por UNIVERSIDAD TECMILENIO.

Queda prohibido copiar, reproducir, distribuir, publicar, transmitir, difundir, o en cualquier modo explotar cualquier parte de esta obra sin la autorización previa por escrito de UNIVERSIDAD TECMILENIO. Sin embargo, usted podrá bajar material a su computadora personal para uso exclusivamente personal o educacional y no comercial limitado a una copia por página. No se podrá remover o alterar de la copia ninguna leyenda de Derechos de Autor o la que manifieste la autoría del material.