

Guia basica para API Testing con POSTMAN.

API Testing con POSTMAN.

Las pruebas de API tienden a evaluar la usabilidad, la fiabilidad, la eficiencia y la seguridad de las mismas, involucran el envío de peticiones, su observación, seguimiento y monitoreo de las respuestas para asegurarse de que funcionan de la forma que se espera. Las pruebas de API son una parte fundamental del ciclo de vida del desarrollo de software.

Pruebas de funcionalidad de API's.

Pruebas de Carga de API's. (J Meter / Soup UI)

Pruebas de seguridad de API's.

API = Application programming interfaces.

Cliente <-> POST / GET <-> API <-> DataBase

Elementos:

- **URL:** representa el recurso que se va a consumer = http://www....
- **Método:** representa la operación que se va a realizar = GET / POST...
- **Código:** representa el estado o resultado del consumo del servicio = 200,400,404...

Las peticiones de API's se implementan como **peticiones HTTP**. (Hypertext transfer protocol secure).

HTTP Status Codes:

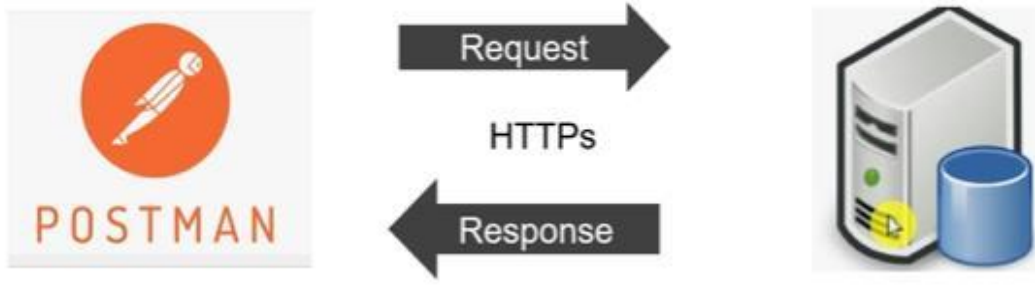
- 200 OK
- 201 CREATED
- 400 BAD REQUEST
- 401 UNAUTHORIZED
- 403 FORBIDDEN
- 404 NOT FOUND
- 500 SERVER ERROR

¿Cómo visualizar la URL del llamado a una Web API?

- Inspect -> Network -> Fetch/XHR -> headers/Preview/Response

¿Qué es POSTMAN?

Es una herramienta que permite la creación, almacenamiento, administración, pruebas y todo lo relacionado con las API's. Cuenta tanto con una versión [WEB](#) o [Desktop App](#).



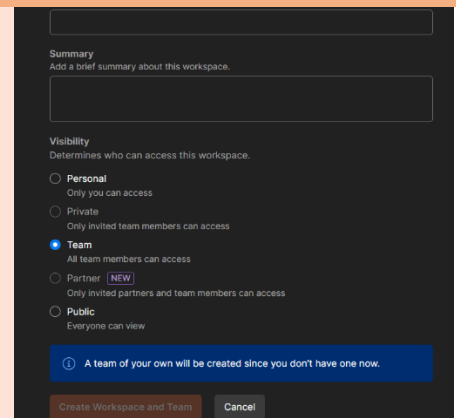
💡 Centro de aprendizaje de POSTMAN (documentación):

<https://learning.postman.com/docs/getting-started/introduction/>

- Conociendo POSTMAN:

Workspaces

Es un escritorio o un ambiente donde puedo trabajar con diferentes API's relacionada con un producto, proyecto ó equipo de trabajo en específico.



- **Collection :** A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it
- **API-s :** APIs define related collections and environments under a consistent schema.
- **Environment:** An environment is a set of variables that allows you to switch the context of your requests.
- **Mock servers:** Mock servers let you simulate endpoints and their corresponding responses in a collection without actually setting up a back end.
- **Flows:** Flows help you create API applications by connecting series of requests on an infinite canvas.
- **Monitors:** A monitor lets you run a collection periodically to check for its performance and response.
- **History:** Any request you send in this workspace will appear here.

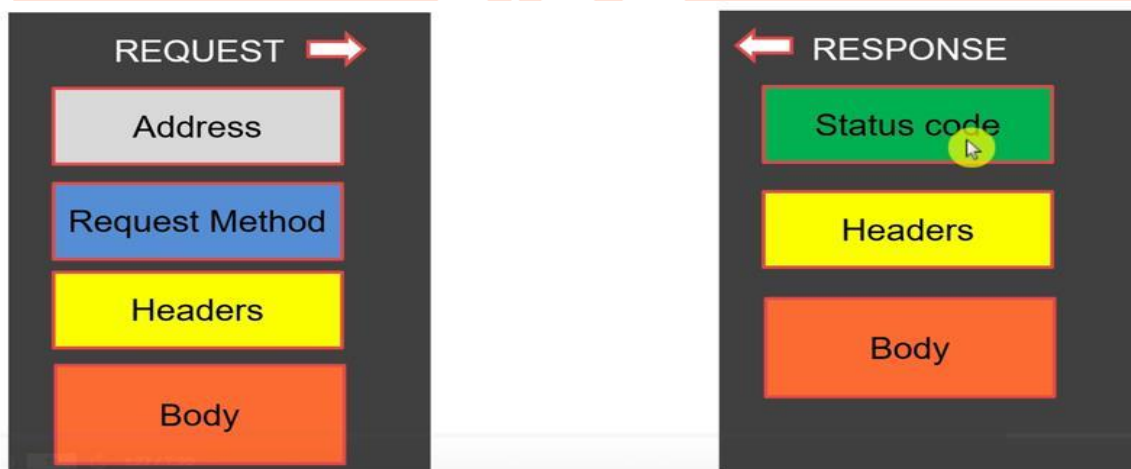
💡 Pagina para pruebas básicas de API's:

<https://reqres.in/>

- Nuestro primer API Request:

- 1- Ingresamos en el sitio web: <https://j17lt.csb.app/>
- 2- Inspect -> network -> Fetch/XHR -> F5.
- 3- En **Headers** extraemos la información necesaria para ingresar en POSTMAN y realizar la configuración de la consulta. (Request URL, Request Method...)
- 4- En POSTMAN Click en “+”
- 5- Copiamos el Request URL de nuestro interés, junto con el Request Method correspondiente para la API.
- 6- Click en “Send”
- 7- Se realizará el Request y podremos visualizar el Response del API.

- Anatomia de una API Request.



Request: es la petición que se realiza a la API.

- Address: dirección donde se encuentra alojada la API.
- Método: GET / PUT / POST...
- Headers: información requerida para el correcto consumo de la API.
- Body: información extra contenida dentro de la petición.

💡 Las respuestas de las API tipo REST vienen en formato **JSON**.

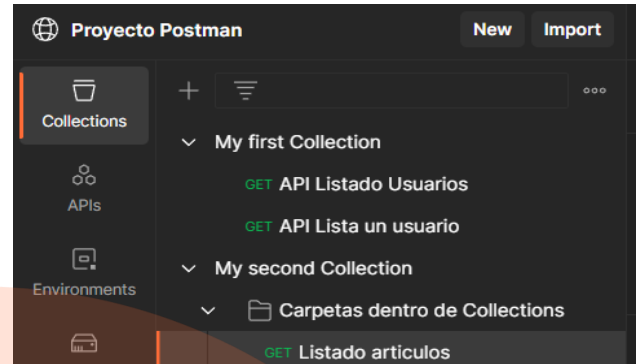
- Manejo de colecciones:

Las collections funcionan como carpetas que nos permiten organizar las Request asociadas a la misma API.

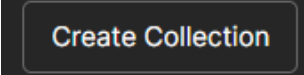
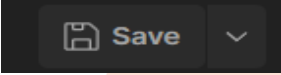
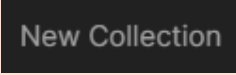
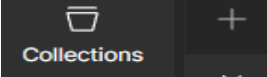
Workspace: Proyecto Postman.

Collections: My first Collection.

Request: API Listado Usuarios.



Podemos crear collections de las siguientes maneras:

- 1- 
- 2-  -> 
- 3- 

- Creación de API's REST Request:

Una API REST es una interfaz de comunicación entre sistemas de información que usa el protocolo de transferencia de hipertexto (*hypertext transfer protocol* o HTTP, por su siglas en inglés) para obtener datos o ejecutar operaciones sobre dichos datos en diversos formatos, como pueden ser XML o JSON.

Se basa en el modelo cliente-servidor donde el cliente es el que solicita obtener los recursos o realizar alguna operación sobre dichos datos, mientras que el servidor es aquel ente que entrega o procesa dichos datos a solicitud del cliente.

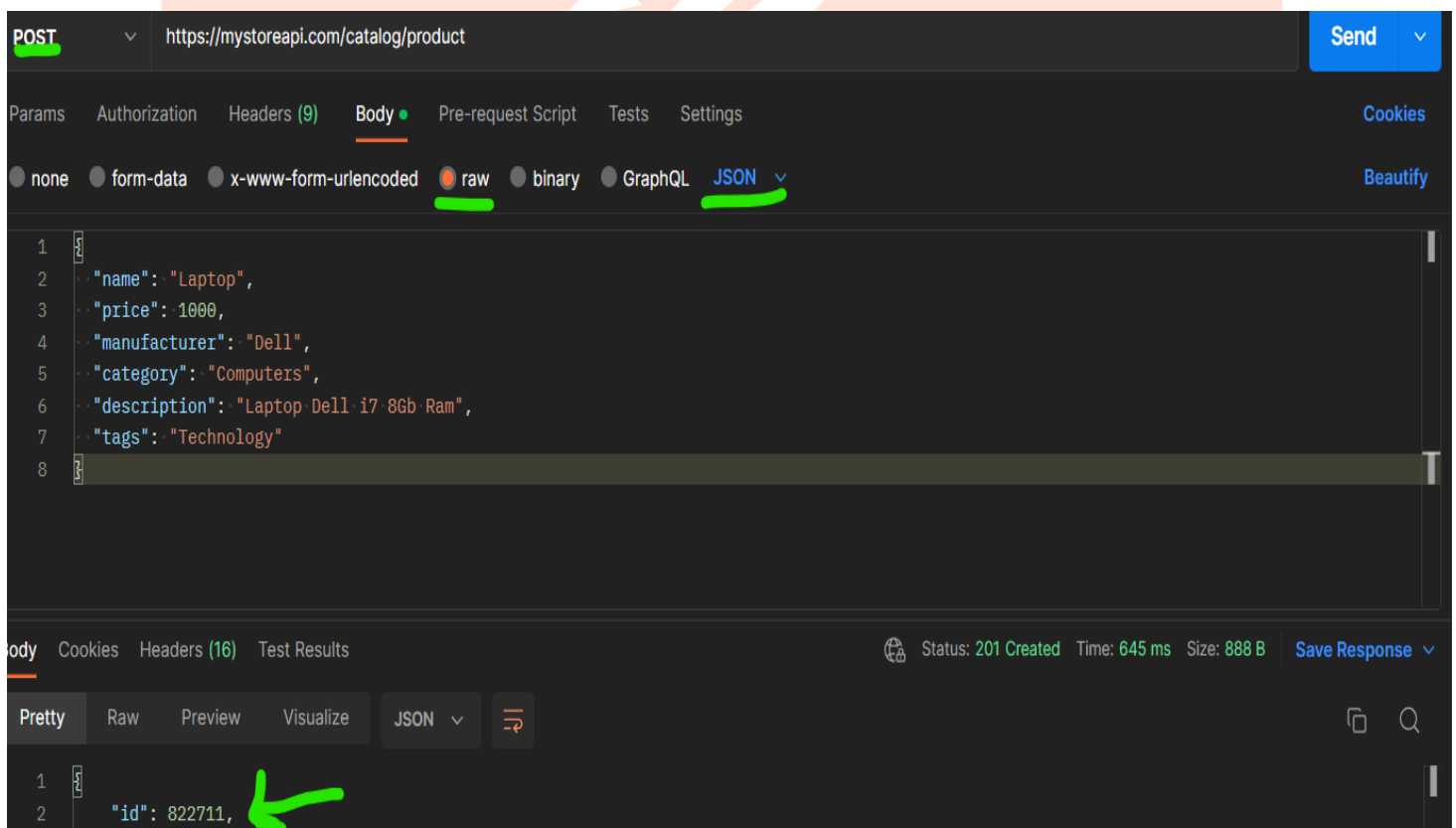


Métodos de API REST:

- **POST:** Permite crear un recurso nuevo.
- **PUT:** Permite modificar un recurso existente.
- **GET:** Permite consultar información de un recurso.
- **DELETE:** Permite eliminar un recurso determinado.
- **PATCH:** Permite modificar solo un atributo de un recurso.

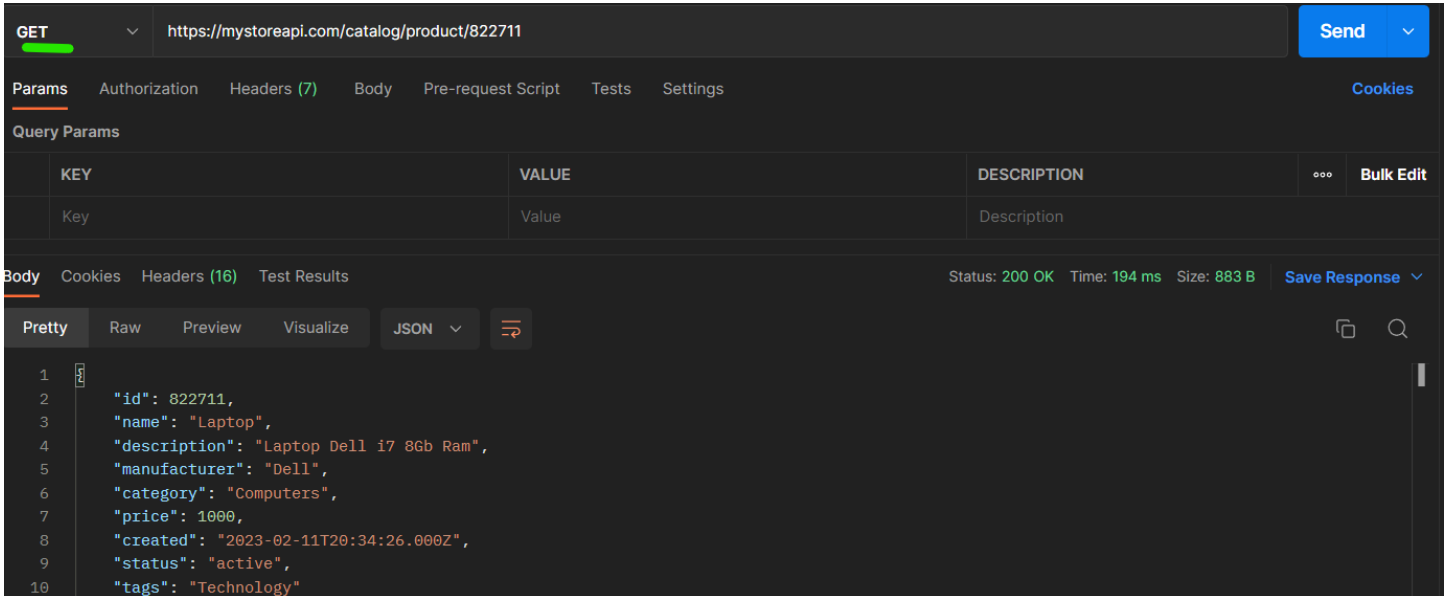
✓ METODO POST:

- 1- Ingresamos la Request URL en POSTMAN.
- 2- Seleccionamos el método POST.
- 3- Al chequear las especificaciones o documentación de la API nos dirá como es el **body que debe ser necesario incluir** para la correcta petición.
- 4- Copiamos los datos en body de POSTMAN y seleccionamos **"raw"**
- 5- Seleccionamos formato JSON / XML /Text.. o cualquier otro en base a lo requerido.
- 6- Llenamos los datos en base a la información que queramos crear.
- 7- Click "Send"
- 8- Corroboramos que el recurso se haya creado. (Ya sea con algún numero de ID o similar).



✓ METODO GET:

- 1- Ingresamos la Request URL en POSTMAN.
- 2- Seleccionamos el método GET.
- 3- Click en "Send".
- 4- Se realizará el Request y podremos visualizar el Response del API.



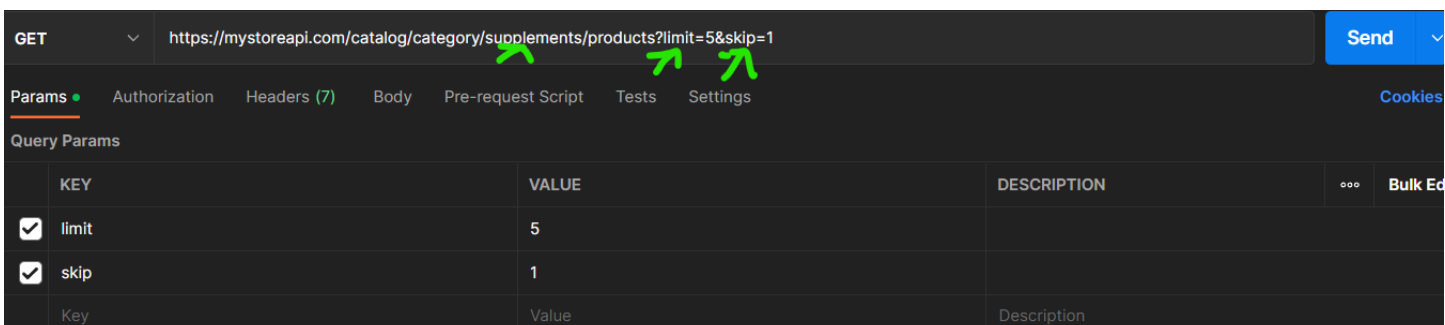
¿Cómo hacer llamado de API's Parametrizadas?

Los parámetros son cierta información que será necesario agregar para hacer una correcta petición de la API.

💡 Los parámetros son incluidos en { } dentro de la URL y deben ser llenados con datos específicos para realizar la petición.

[../catalog/category/{category}/products](https://mystoreapi.com/catalog/category/supplements/products?limit=5&skip=1)

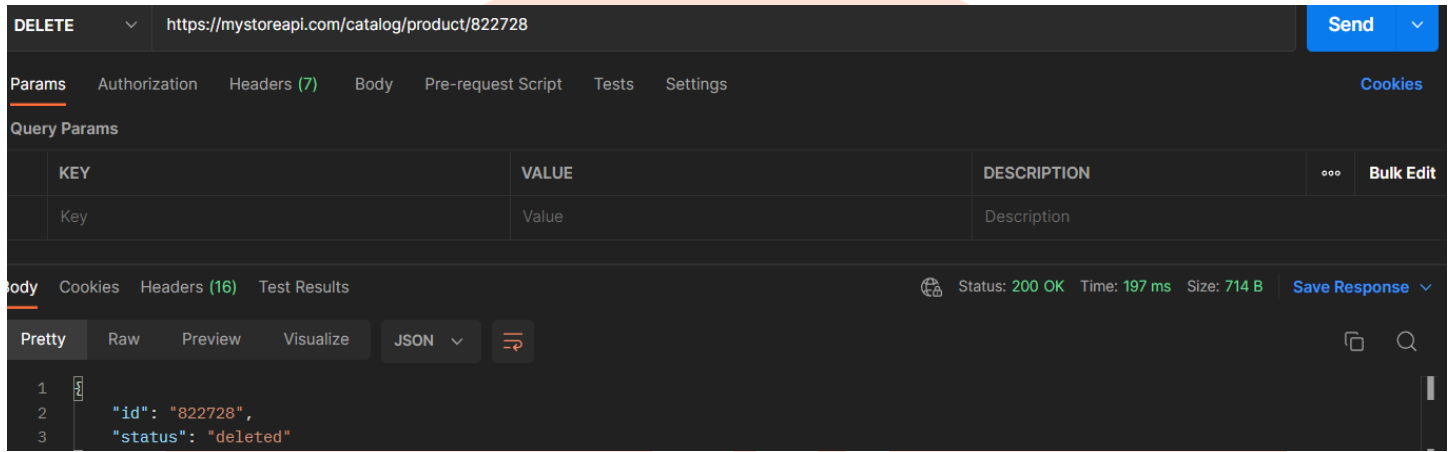
Los parámetros dentro de la URL serán reconocidos en POSTMAN y colocados y organizados dentro de la pestaña "Params"



✓ METODO DELETE:

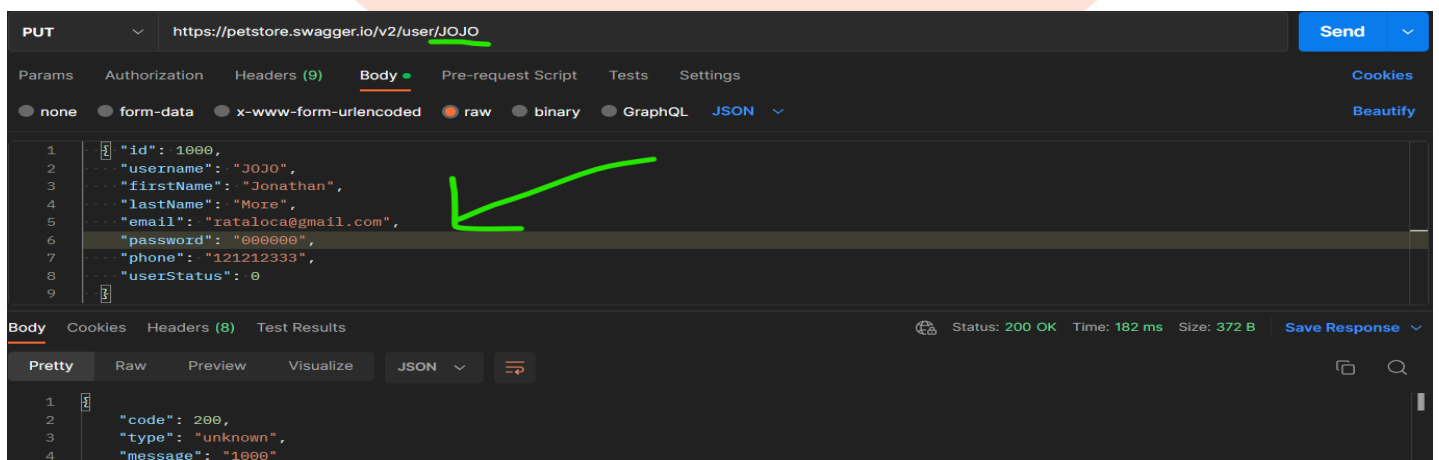
- 1- Ingresamos la Request URL en POSTMAN.
- 2- Seleccionamos el método DELETE.
- 3- Modificamos los parámetros en "Params" *en base a en que manera está configurada la URL para eliminar un recurso.*
- 4- Click en "Send".
- 5- Se realizará el Request y podremos visualizar el Response del API.

[.../catalog/product/{id}](#)



✓ METODO PUT:

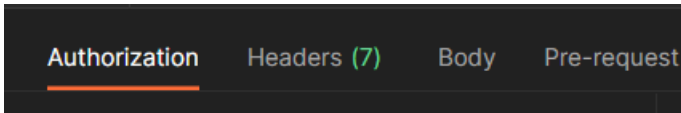
- 1- Ingresamos la Request URL en POSTMAN.
- 2- Seleccionamos el método PUT.
- 3- Modificamos los parámetros de la URL *en base a en que manera está configurada para la modificación de un recurso.*
- 4- Click en "Send".
- 5- Se realizará el Request y podremos visualizar el Response del API



- Autenticación y Autorización en POSTMAN:

Las API's tienen mecanismos de seguridad que deben ser configuradas previamente para poder consumirlas.

Existen diferentes tipos de Autorizaciones en POSTMAN y las mismas se encuentra en:



✓ BASIC AUTH:

La Autorización Básica consiste en la configuración de la Request con:

Username:

Password:

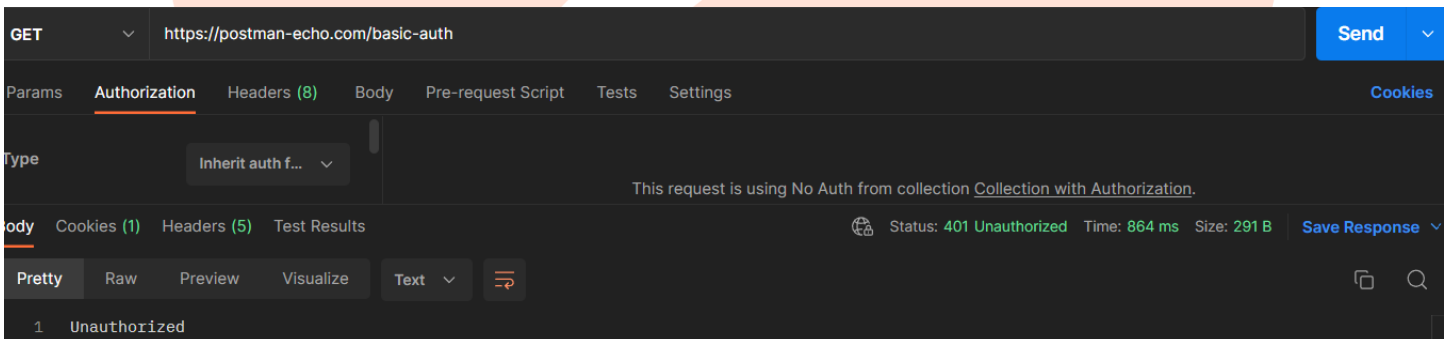
Al estar configurado correctamente el Request traerá como Response:

200 ok

Caso contrario traerá como Response:

401 unauthorized

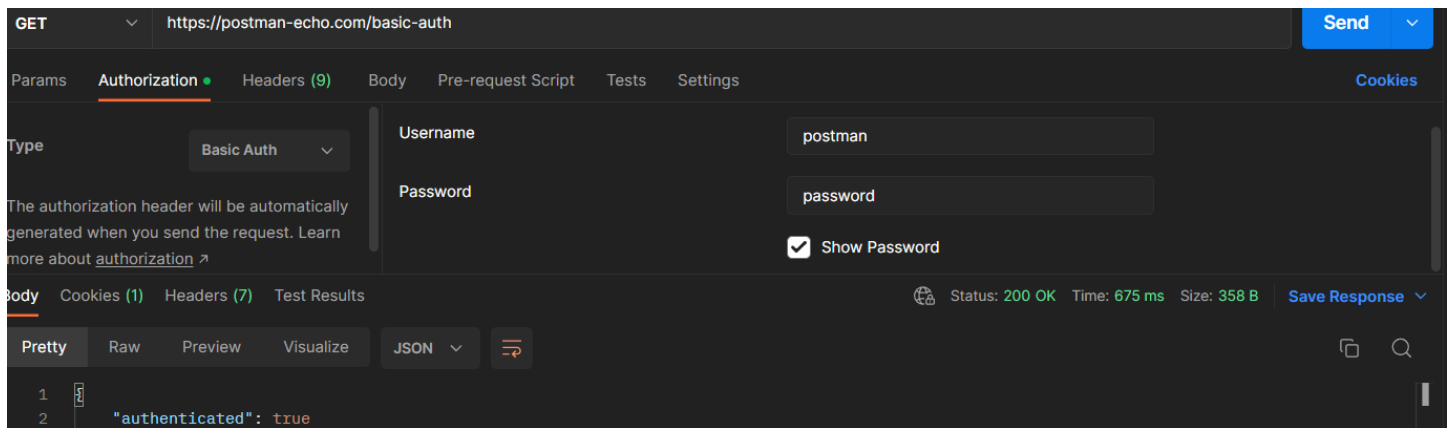
ANTES DE LA CONFIGURACION:



¿Cómo se configuran las Autorizaciones?

- 1- ingresamos el Request URL en POSTMAN.
- 2- Click pestaña "Authorization"
- 3- Seleccionamos el Type (en este caso Basic Auth)
- 4- Configuramos USERNAME & PASSWORD

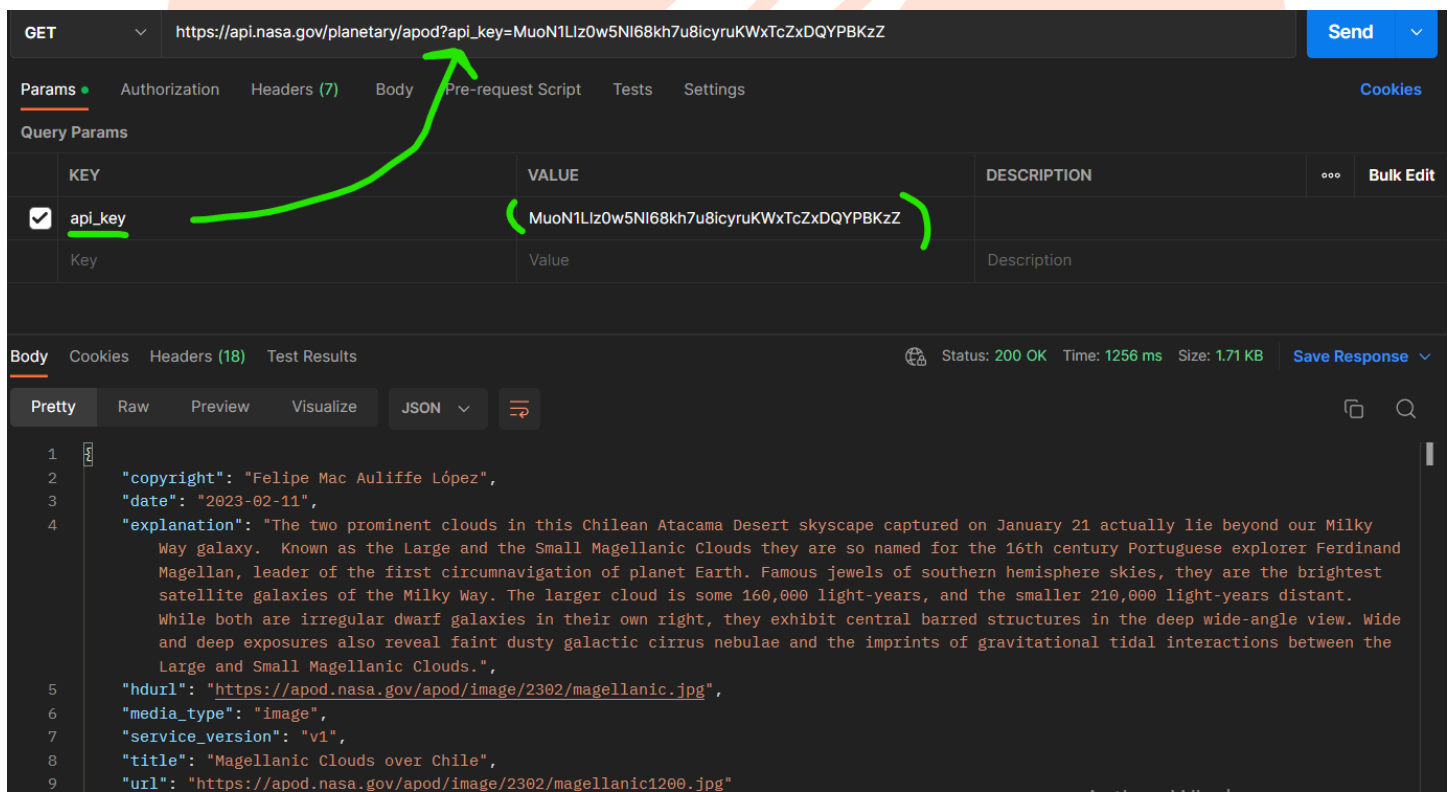
LUEGO DE LA CONFIGURACION:



✓ API KEY:

Al usar las API KEY la autorización se realiza mediante la URL, cuando la misma tiene “?” es reconocida por POSTMAN y permite configurar a la misma.

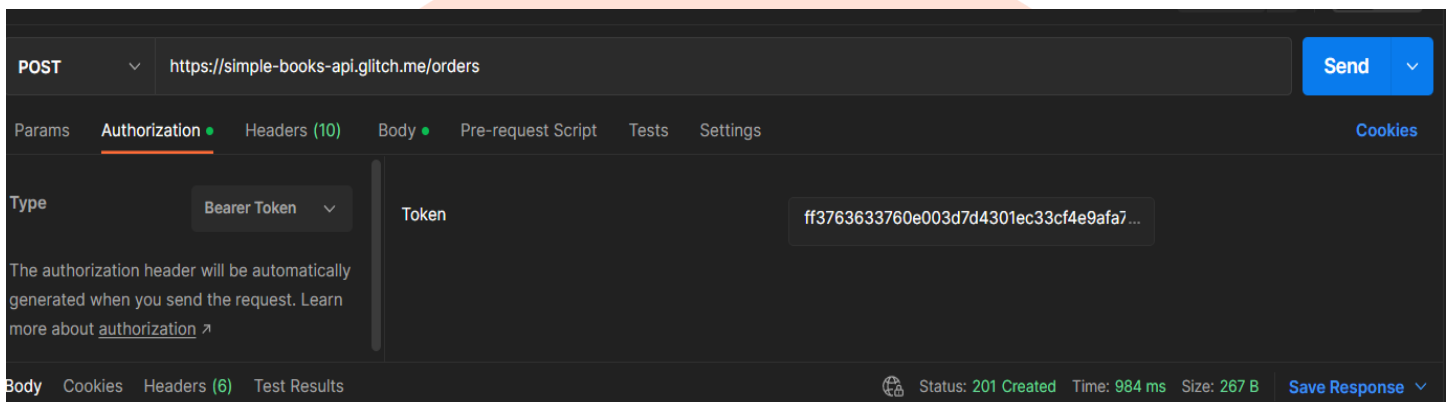
La contra de este tipo de autorización es que la misma queda expuesta en la URL.



✓ TOKEN:

- 1- ingresamos el Request URL en POSTMAN.
- 2- Click pestaña "Authorization"
- 3- Seleccionamos el Type (en este caso Beared Token)
- 4- Ingresamos el Token
- 5- Enviamos la Request

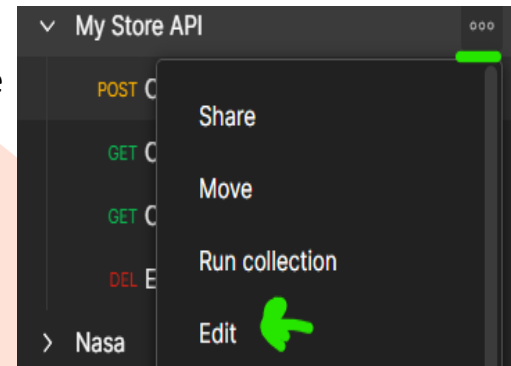
```
{  
  "accessToken": "ff3763633760e003d7d4301ec33cf4e9afa7bb622b5231705716604b4bee06f3"  
}
```



Manejo de Variables en POSTMAN

- Creación de Variables de Colección:

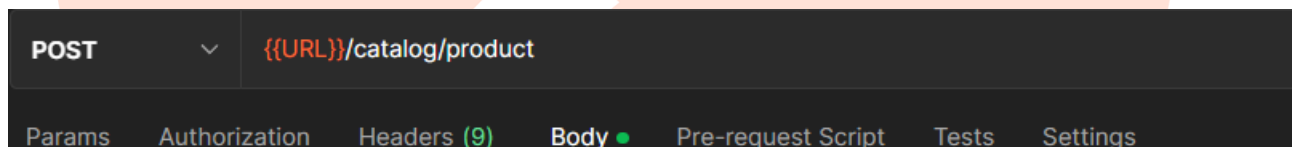
- 1- Selecciono la Colección donde quiero configurar mi Variable
- 2- Click en "edit" ->
- 3- Click en la pestaña "Variables"
- 4- Defino las variables que irán dentro de la Collection.



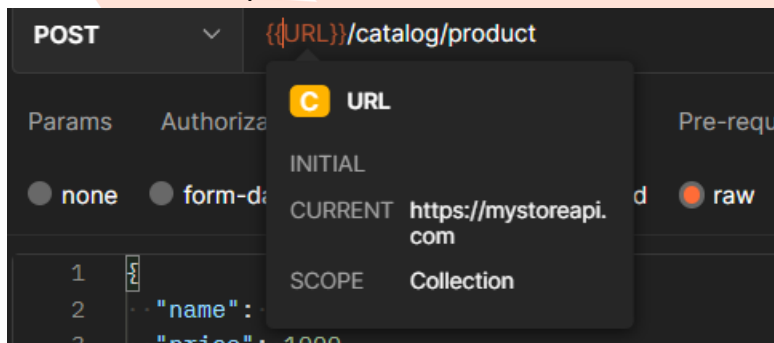
These variables are specific to this collection and its requests. Learn more about [collection variables](#)

	VARIABLE	INITIAL VALUE ⓘ	CURRENT VALUE ⓘ	...	Persist All	Reset All
<input checked="" type="checkbox"/>	URL		https://mystoreapi.com			

- 5- Para instanciar una variable se debe usar `{{ }}` y el nombre de la variable dentro.



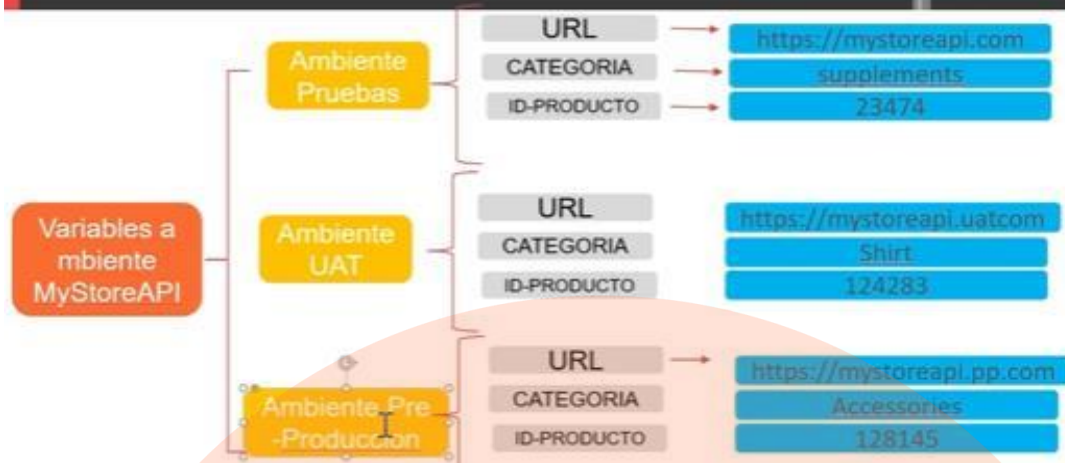
- 6- Corroboramos que la instancia a la variable sea correcto:



- Creación de Variables de Ambientes:

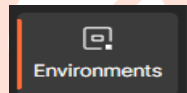
💡 Las API's se alojan en HOST's específicos, por lo que la dirección serán diferentes, es por ello que deben ser configurados mediante ambientes, de esta manera se pueden probar en diferentes instancias del desarrollo del software..

Manejo de Variables de ambiente



¿Cómo se crean los ambientes de prueba en POSTMAN?

- 1- Seleccionamos la pestaña "Environments"
- 2- Click "Create Environment"
- 3- Configuramos las variables del ambiente.



Create Environment

Ambiente pruebas 1

Fork

0

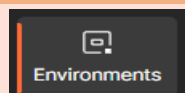
Save

	VARIABLE	TYPE ⓘ	INITIAL VALUE ⓘ	CURRENT VALUE ⓘ	...	Persist
<input checked="" type="checkbox"/>	URL	default	⌵	https://mystoreapi.com		
<input checked="" type="checkbox"/>	CATEGORIA	default	⌵	supplements		
<input checked="" type="checkbox"/>	ID-PRODUCTO	default	⌵	23474		

- 4- Seleccionamos el ambiente en el cual queremos probar las API's.
- 5- Las Variables de Ambientes se instancias de igual manera -> `{{ nombre_variable }}`.

- Creación de Variables Globales:

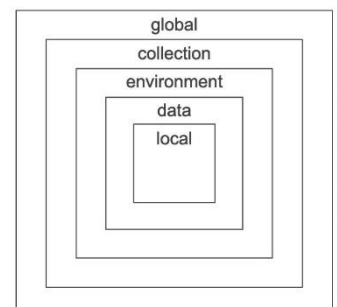
- 1- Seleccionamos la pestaña "Environments"
- 2- Click "Globals"
- 3- Configuramos las variables del ambiente.



Globals

💡 Las variables Globales funcionan exclusivamente dentro del Workspace.

💡 Orden de prioridad de variables en POSTMAN.



- Creación de Variables Locales:

- 1- Dentro de la Request seleccionamos: **Pre-request Script**
- 2- Escribimos el script, ejemplo: `pm.variable.set("name", "TV");`
- 3- Se instancias las variables de igual manera `{{ }}`.

```
Params  Authorization  Headers (9)  Body ●  Pre-request Script ●  Tests  Settings

1  pm.variables.set("Name", "TV");
2  pm.variables.set("Precio", 1500);
```

```
1  {
2    "name": "{{Name}}",
3    "price": "{{Precio}}",
4    "manufacturer": "Dell",
```

💡 En POSTMAN también podemos manipular variables de diferentes niveles mediante scripts preformados llamados “Snippets”. Algunos ejemplos serian:

`pm.globals.set("parametro1", "parametro2");`

`pm.environment.set("parametro1", "parametro2");`

`pm.globals.get("NOMBRE_VARIABLE");`

Para información mas extensiva sobre el uso de scripts:

<https://learning.postman.com/docs/sending-requests/generate-code-snippets/#generating-code-snippets-in-postman>

```
GET  {{URL}}/catalog/category/{{CATEGORIA}}/products?limit=20&skip=1  Send

Params  Authorization  Headers (7)  Body  Pre-request Script ●  Tests  Settings  Cookies

1  //Creacion de variables
2  pm.environment.set("variable_key", "variable_value");
3  pm.globals.set("variable_key", "variable_value");
4  pm.collectionVariables.set("variable_key", "variable_value");
5
6  //llamado de variables
7  console.log(pm.environment.get("variable_key"));
8  console.log(pm.globals.get("variable_key"));
9  console.log(pm.variables.get("variable_key"));

Pre-request scripts are written in JavaScript, and are run before the request is sent. Learn more about pre-request scripts ↗

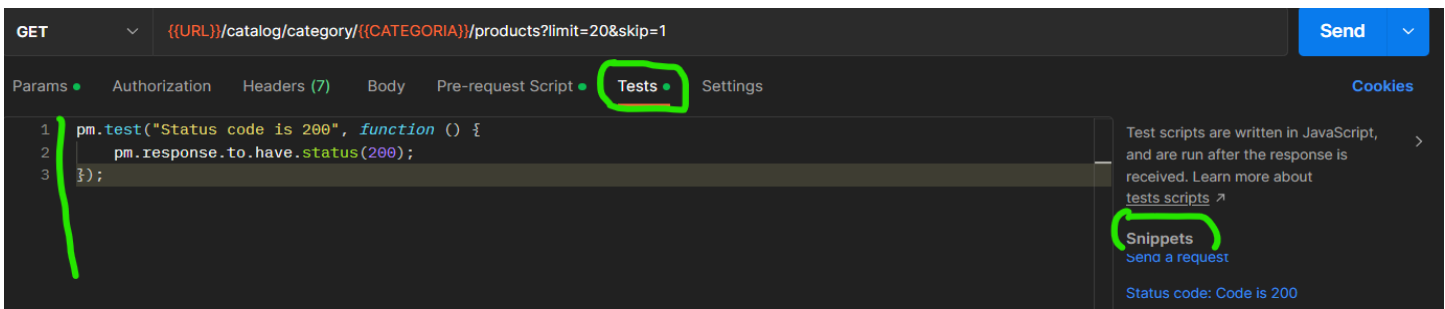
Snippets
Get an environment variable
Get a global variable
```

Tests y Automatización con POSTMAN

Los módulos anteriores nos permitieron realizar la configuración básica de API's en POSTMAN.

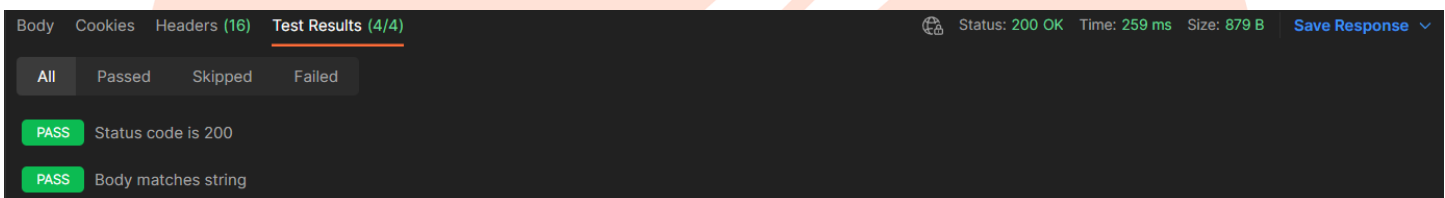
¿Cómo realizamos las pruebas de API?

- 1- Ingresamos en la Request de la API que queramos testear.
- 2- Seleccionamos **Tests**.
- 3- Dentro del cuerpo de Tests podemos generar los scripts de pruebas, o bien usar los scripts pre creados por POSTMAN, llamados **"SNIPPETS"**.

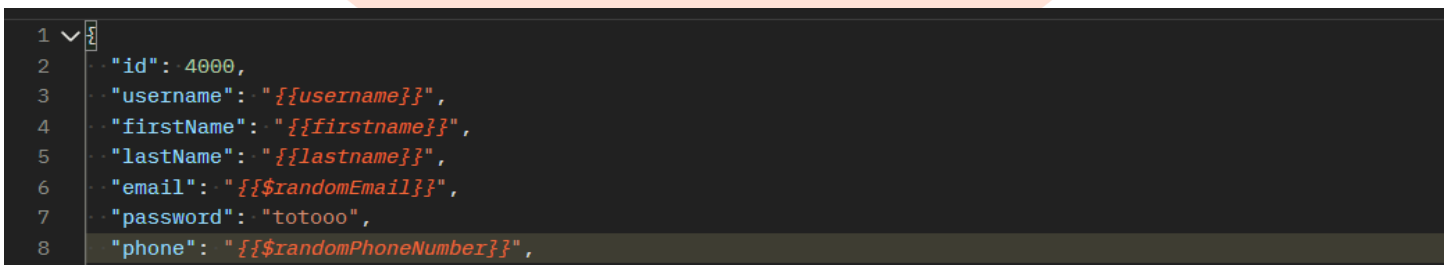
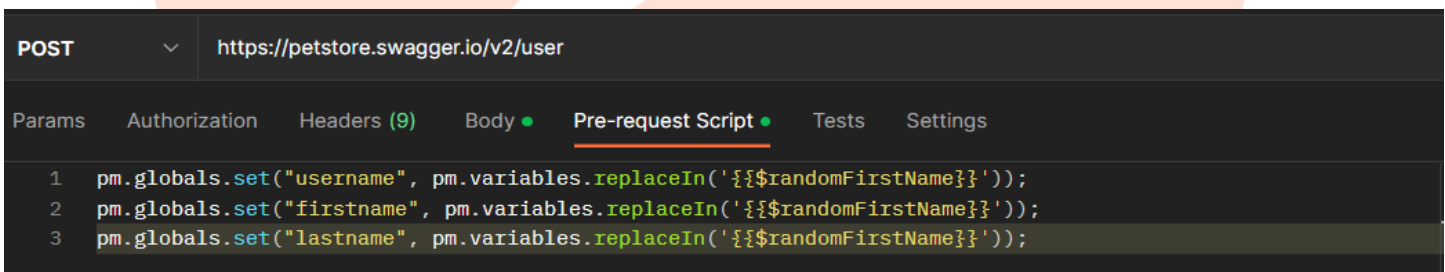


💡 Los scripts en POSTMAN son codificados en el lenguaje JavaScript.

- 4- Se evalúa los resultados en las pruebas en:



¿Cómo crear Variables con datos aleatorios?



<https://learning.postman.com/docs/writing-scripts/script-references/variables-list/>

Uso de Assertions:

<https://learning.postman.com/docs/writing-scripts/script-references/test-examples/>

↳ Using multiple assertions

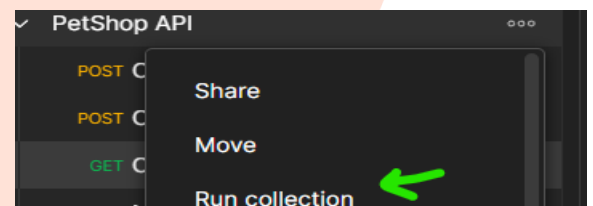
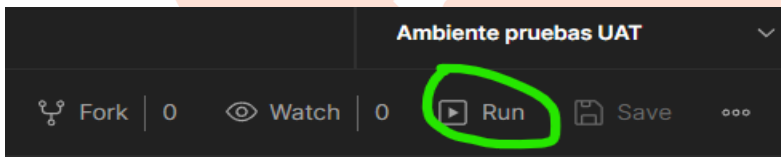
Your tests can include multiple assertions as part of a single test. Use this to group together related assertions:

```
pm.test("The response has all properties", () => {  
  //parse the response JSON and test three properties  
  const responseJson = pm.response.json();  
  pm.expect(responseJson.type).to.eql('vip');  
  pm.expect(responseJson.name).to.be.a('string');  
  pm.expect(responseJson.id).to.have.lengthOf(1);  
});
```

If any of the contained assertions fails, the test as a whole will fail. All assertions must be successful for the test to pass.

- Automatización de ejecuciones de pruebas en POSTMAN:

- 1- Seleccionamos la Collection que queremos ejecutar.
- 2- Tenemos 2 opciones:



- 3- Configuramos el RUN de la Collection.

Run configuration

- 4- Ejecutamos el RUN

Run PetShop API

💡 **Monitors** son una herramienta más potente para la ejecución automatizada de pruebas.

-Herramientas de automatización para POSTMAN:

-NEWMAN: es un modulo de NodeJs que nos permite ejecutar y probar Collections de POSTMAN, directamente desde la línea de comando.

1- Instalar **Node JS**. -> CMD -> node -v

Agregar a Node Js como una variable de entorno de Windows

Panel -> sistema y seguridad -> sistema -> configuración avanzada de sistema->

Variables de entorno -> path -> editar -> nuevo ->copiar la dirección de carpeta de Node JS -> aceptar.

2-Instalar Newman -> **-npm install -g newman**

3-Instalar Newman HTML reports -> **-npm install -g newman -reporter-htmlextra.**

¿Como ejecutar las Request de POSTMAN en CMD?

1- Abrir CMD

2- Navegar en CMD: d: -> cd postman(carpeta previamente creada) + la collection previamente guardada -> dir: -> **Newman run + nombre de la collection.json**

3- Enter.

Para descargar el reporte:

1- Abrir CMD

2- Navegar en CMD: d: -> cd postman(carpeta previamente creada) + la collection previamente guardada -> dir: -> **Newman run + nombre de la collection.json + -r htmlextra.**

3- Enter.

-JENKINS: es una herramienta de *Integración Continua (CI)*, así como AzureDevOps, Bamboo, etc. Cuando un desarrollador hace un cambio en cualquier lugar del código nuestra herramienta CI detecta un cambio y dispara una suite de pruebas automatizada.

1- Descargamos e Instalamos Java JRE.

2- Descargar e instalar Jenkins versión LTS.

3- Seleccionar un ambiente donde se correrá Jenkins (dominio o personal).

4- Ejecutamos Jenkins:

a. CMD

b. dir:

- c. Navegamos hasta la carpeta donde quedo Jenkins instalado.
- d. Dentro de la carpeta Jenkins ejecutamos: **java -jar jenkins.war --httpPort=8080**
(OPCIONAL SUMARLE: **--enable-future-java**)
- e. Abrimos browser.
- f. localhost:8080
- g. **a**Configuramos la seguridad de la cuenta: abriendo la dirección que te da y copiamos y pegamos la password.
- h. Seleccionamos los plugins a instalar.
- i. Creamos y configuramos un usuario.

En caso de problemas en la instalación: https://www.youtube.com/watch?v=-Z1L2j2-6Ac&ab_channel=SilentSolution

¿Cómo se crea un PIPELINE en JENKINS para pruebas de POSTMAN?

Parte1:

- 1- Abrimos la carpeta donde Jenkins esta instalado.
- 2- Copiamos la dirección.
- 3- CMD.
- 4- Cd..
- 5- Cd..
- 6- Raiz
- 7- Pegamos la direccion.
- 8- Dentro de la carpeta Jenkins ejecutamos: **java -jar jenkins.war --httpPort=8080**
(OPCIONAL SUMARLE: **--enable-future-java**)
- 9- Abrimos un browser -> LocalHost:8080.

Parte2:

- 1- Nueva Tarea.
- 2- Nombre del proyecto.
- 3- Crear un proyecto estilo libre.
- 4- OK.
- 5- Configuramos el proyecto.
- 6- Seleccionamos Disparadores de ejecuciones.
- 7- Ejecutar -> comando de Windows. -> node -v
- 8- Ejecutar -> comando de Windows. -> npm -v
- 9- Ejecutar -> comando de Windows-> **newman run + nombre de la collection.json + -r htmlextra.**

Parte3:

- 1- Guardamos.
- 2- Seleccionamos Construir ahora.
- 3- Jenkins creara el proyecto.

💡 En Jenkins podemos instalar plugins extras -> **Administrar Jenkins** -. **Administrar Plugins** -> **Todos los plugins** -> **Build monitor view** -> seleccionamos -> install.

💡 Para usar el plugin: **mis vistas** -> + -> **le damos un nombre** -> **Build monitor view** -> **seleccionamos los Jobs (tareas o proyectos que tenemos)** -> creamos el tablero

https://www.youtube.com/watch?v=mZGS-DF0J8c&ab_channel=SachinGupta