

## Tallar Postman

Brayan Andrés Qintero Pinto - 2190083

Juan Pablo Ramírez Vela - 2190076

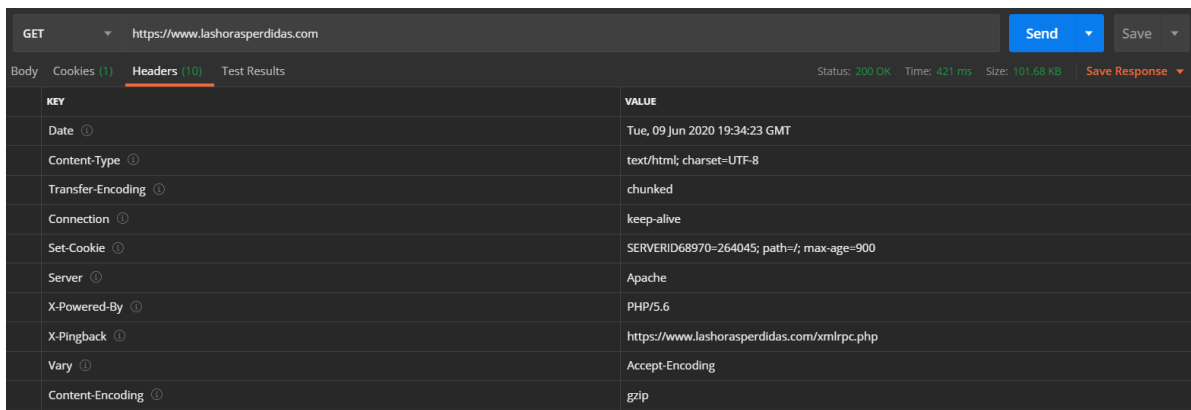
Santiago Breyner Peña Bello - 2191636

### 1. Acceder a un sitio estático usando un navegador

[https://es.wikipedia.org/wiki/P%C3%A1gina\\_web](https://es.wikipedia.org/wiki/P%C3%A1gina_web)

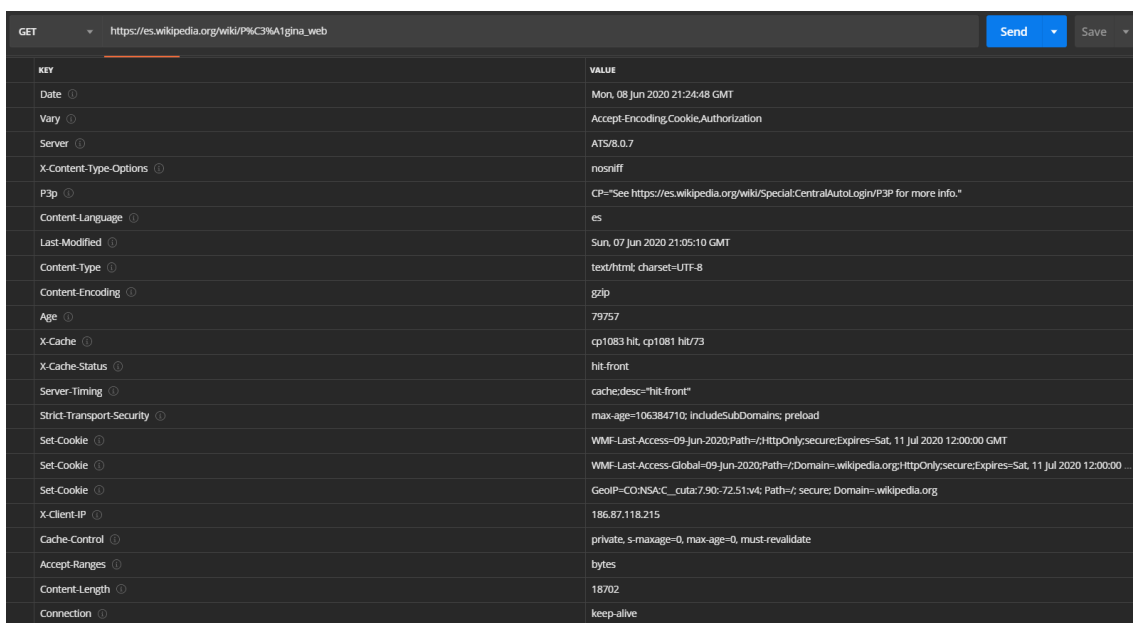
<https://www.lashorasperdidas.com>

- (mirar headers e info interesante):
  - Entre las dos páginas consultadas existen Headers como: Date, Server, X-Content-Type-Options, P3p, Content-Language, Vary, Content-Type, Content-Encoding, Age, X-Cache, X-Cache-Status, Server-Timing, Strict-Transport-Security, X-Client-IP, Cache-Control, Content-Length y Connection. Dichos headers almacenan información relacionada con el contenido de la página, la solicitud, la conexión e información del servidor. Estos pueden ser generales como Date o más específicos de cada página como X-Cache.



KEY	VALUE
Date	Tue, 09 Jun 2020 19:34:23 GMT
Content-Type	text/html; charset=UTF-8
Transfer-Encoding	chunked
Connection	keep-alive
Set-Cookie	SERVERID68970-264045; path=/; max-age=900
Server	Apache
X-Powered-By	PHP/5.6
X-Pingback	https://www.lashorasperdidas.com/xmlrpc.php
Vary	Accept-Encoding
Content-Encoding	gzip

Ilustración 2: Headers de un blog en Postman



KEY	VALUE
Date	Mon, 08 Jun 2020 21:24:48 GMT
Vary	Accept-Encoding, Cookie, Authorization
Server	ATS/8.0.7
X-Content-Type-Options	nosniff
P3p	CP="See https://es.wikipedia.org/wiki/Special:CentralAutologin/P3P for more info."
Content-Language	es
Last-Modified	Sun, 07 Jun 2020 21:05:10 GMT
Content-Type	text/html; charset=UTF-8
Content-Encoding	gzip
Age	79757
X-Cache	cp1083 hit, cp1081 hit/73
X-Cache-Status	hit:front
Server-Timing	cache:desc="hit:front"
Strict-Transport-Security	max-age=106384710; includeSubDomains; preload
Set-Cookie	WMF-Last-Access=09-jun-2020; Path=/; HttpOnly; secure; Expires=Sat, 11 Jul 2020 12:00:00 GMT
Set-Cookie	WMF-Last-Access-Global=09-jun-2020; Path=/; Domain=.wikipedia.org; HttpOnly; secure; Expires=Sat, 11 Jul 2020 12:00:00 GMT
Set-Cookie	GeoIP=CO:NSA:C_cuta:7.90-72.51-v4; Path=/; secure; Domain=.wikipedia.org
X-Client-IP	186.87.118.215
Cache-Control	private, s-maxage=0, max-age=0, must-revalidate
Accept-Ranges	bytes
Content-Length	18702
Connection	keep-alive

Ilustración 1: Headers de un artículo de Wikipedia en Postman

*Ilustración 3: GET de facebook.com en Postman (línea 76)*

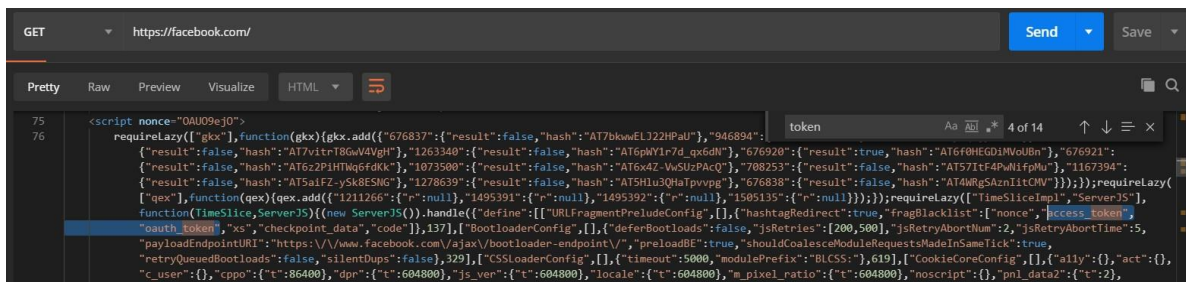


Ilustración 4: GET de facebook.com en Postman (línea 365)

- Parámetros de la URL: Mediante la modificación de las URLs se pueden acceder a distintos sitios en una misma página web o modificar lo que se visualiza actualmente:
  - Facebook for Developers ([https://developers.facebook.com/docs/facebook-login/access-tokens/?locale=es\\_LA](https://developers.facebook.com/docs/facebook-login/access-tokens/?locale=es_LA)): En este caso se le añade el prefijo “developers.” a facebook.com, para indicarle que a ese apartado de su web nos queremos dirigir. También se le añade la función “/docs” para indicarle que miraremos su apartado de documentación y el documento en específico que deseamos ver “/facebook-login/access-tokens”, para finalizar se especifica el idioma que queremos con “?locale=es\_LA” para Español Latinoamericano, o pudiendo cambiar estos últimos cinco caracteres por es\_ES para Castellano o eliminándolos para acceder al idioma original.
  - Facebook Marketplace (<https://www.facebook.com/marketplace>): Con la extensión “/marketplace” podemos acceder a una pestaña de Facebook.com especializada en la compra y venta de productos.
  - Facebook Watch (<https://www.facebook.com/watch/>)
  - Facebook Groups (<https://www.facebook.com/groups/feed/>)
  - Página (<https://www.facebook.com/EDteamLat>): Se agrega un identificador textual único luego de la URL base de Facebook para acceder a cada página.
  - Imagen (<https://www.facebook.com/photo?fbid=2957626554292902&set=a.634364769952437>): Para acceder a una imagen en Facebook, es necesario especificar su tipo “/photo” y agregarle el identificador como imagen de Facebook tal que: “?fbid=XXXXXXXXXXXXXXXXXX” el cual contiene 16 dígitos y ya solo con eso bastaría, ya que al modificar el siguiente parámetro (&set=a...) no se perciben cambios visuales.
  - Video (<https://www.facebook.com/634354456620135/videos/1180257568986069> ó <https://www.facebook.com/watch/?v=1180257568986069>): Hay dos formas de acceder a un video, ya sea desde Facebook Watch o desde el sitio específico donde fue publicado, pero el identificador único del video se compone de 16 dígitos y se ingresa después de “/watch/?v=”. Si se varía dicho identificador, tanto en imágenes como en video, se accede a otro sitio con otra imagen o video, o incluso a un sitio como el siguiente:

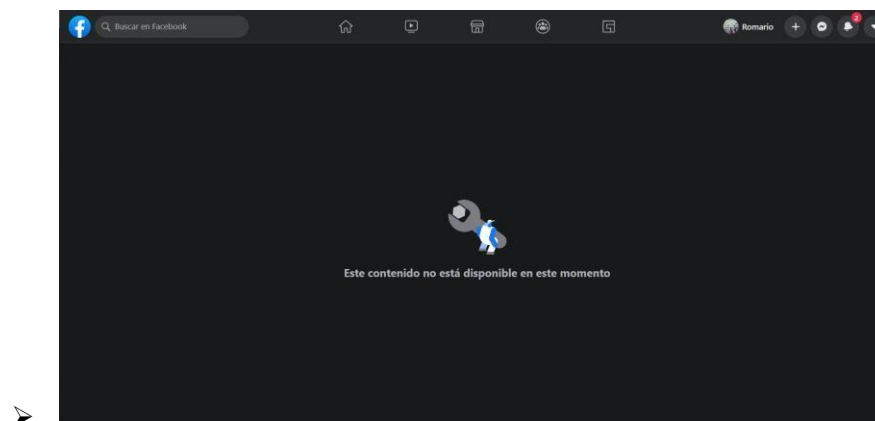


Ilustración 5: Identificador de imagen no encontrado en Facebook

3. Entre a Google Maps mire los parámetros de la URL, cámbielos y analice resultados cuales son los parámetros que pudo cambiar y cuales aparecen con las funcionalidades.
- **Funciones:** Basándose en la URL inicial de Google Maps (<https://www.google.com/maps>) se pueden agregar algunas de las funciones soportadas por esta plataforma anteponiendo un “/” antes de las mismas y por supuesto, después de la URL base. También es necesario, luego de seleccionar la función especificar luego de un slash que se hará uso de la api=1, tal que: /?api=1. Dichas funciones son:
  - **search:** Se utiliza para especificarle a Google Maps que lo que se quiere hacer es una búsqueda de un sitio, con diferentes parámetros que se agregan después de un “&” como palabras que describan o nombren al lugar (separadas por %20 ó +, que cumplen las funciones del espacio vacío en una URL).
  - **dir:** Esta función tiene como objetivo pedir direcciones y mostrarlas mediante la interfaz gráfica de Google Maps.
  - **map\_action=map:** Se agrega de modo que la URL quedé: /@?api=1&map\_action=map, luego del maps y teniendo en cuenta que aún faltan los parámetros para que Google Maps inicie sin marcadores ni indicadores.
  - **map\_action=pano:** Muestra un panorama de Street View, el cuál permite tener una visión como si se estuviera presente en dicho sitio especificado en los parámetros.
- **Parámetros:** Existen algunos parámetros que varían lo que muestra la página de Google Maps, entre ellos tenemos:
  - **query:** Define el sitio a resaltar en el mapa, mediante la búsqueda por nombre.
  - **query\_place\_id:** Identificador único de los lugares en GMaps. (mayor precisión)
  - **origin:** Establece el sitio inicial de salida para dar las direcciones e indicaciones.
  - **origin\_place\_id:** Identificador único del sitio de salida de una trayectoria.
  - **destination:** Establece el sitio de destino del trayecto.
  - **destination\_place\_id:** Identificador único para el sitio de destino.
  - **travelmode:** Define el método de viaje (driving, walking, bicycling, transit).
  - **dir\_action=navigate:** Da indicaciones precisas para la ruta a recorrer.
  - **waypoints:** Establece lugares por los que el usuario necesita pasar a pesar de que no estén en la ruta inicial (Floridablanca%2CSantander%7CGiron%2CSantander).
  - **waypoint\_place\_ids:** Identificadores únicos de los sitios por los cuales el usuario desea pasar.
  - **center:** Define el centro del mapa en la ventana mostrada (-0.2656,32.6598)
  - **zoom:** Define el nivel de acercamiento del mapa (0 a 21).
  - **basemap:** Define el tipo de mapa a mostrar (roadmap, satellite, terrain)
  - **layer:** Define que información adicional mostrar (none, transit, traffic, bicycling)
  - **viewpoint:** Define a la ubicación del panorama mas cercano en Street View.
  - **pano:** Especifica el identificador del panorama fotografiado a mostrar.
  - **heading:** Indica la orientación de brújula en sentido horario (-180° a 360°)
  - **pitch:** Indica el ángulo de la cámara (-90° a 90°)
  - **fov:** Determina el campo de visión horizontal de la imagen (10° a 100°)

4. Entre a la siguiente dirección: <https://docs.postman-echo.com/?version=latest>

Use <https://postman-echo.com> Realice

- **GET Request:** El comando contiene dos parámetros de consulta {“foo1”: “bar1”, “foo2”: “bar2”}, los cuales se usan para verificar uno de los dos tests que se realizan, uno de ellos es verificar que el código de respuesta sea el 200 y el otro es para revisar si se encuentran los parámetros en el jsonBody o más específicamente en el objeto “args”. El método GET se usa para exclusivamente para recuperar datos del servidor.
- **POST Raw Text:** El comando no contiene parámetros como tal pero si un Raw Text en el Body en el cuál dice: “This is expected to be sent back as part of response body.” Además, contiene dos tests, el primero verifica que

todo haya ido bien (200 OK) y el otro que en el objeto “data” se encuentre el texto enviado como entidad en el servidor mediante el método de petición POST.

- POST Form Data: Este comando tiene en su Body un x-www-form-urlencoded con: {"foo1": "bar1", "foo2": "bar2"} los cuales se envían al servidor mediante un POST y se revisa su respuesta con dos funciones, la que verifica que todo haya salido bien y la que revisa que en Formulario enviado se encuentre en el objeto “form” del JSON recibido.
- PUT Request: Este comando se encarga de reemplazar los valores del objeto “data” por "This is expected to be sent back as part of response body." En caso de que ya existiera con otro valor o de crear a “data” para darle dicho valor. Al final, verifica que todo haya salido bien (200 OK) y que se haya actualizado “data” con el valor enviado por el body.
- PATCH Request: Se hace uso del método PATCH para hacer una modificación parcial del objeto “data” en el JSON recibido como respuesta. En el body de éste comando se encuentra el Raw Text destinado a ser el nuevo valor de “data”: “This is expected to be sent back as part of response body.” Se mira si se cumple con que todo haya salido bien y que la modificación parcial se haya realizado en los tests.
- DEL DELETE Request: Con este comando se busca eliminar un recurso que se especifica previamente.

5. De Postman-Echo realice un GET de respuesta JSON, analice la respuesta.

Ahora compárelo contra el siguiente. [linkjsongis](#)

¿Qué diferencias en los parámetros encuentra y que diferencia en la respuesta?

- Se analizaron ambos GETs y se pudo observar:
  - En cuanto a los parámetros, el GET Request de Postman-Echo contiene como parámetros de consulta a {"foo1": "bar1", "foo2": "bar2"} y el GET de la URL suministrada contiene los siguientes parámetros de consulta: {"service": "WFS", "version": "1.1.0", "request": "GetFeature", "typename": "osm:water\_areas", "outputFormat": "application/json", "srsname": "EPSG:3857", "bbox": "-8938009.785172544,5370452.51819444,-8879764.769619238,5393383.626679991, EPSG:3857"}.
  - Las respuestas JSON de ambas peticiones son completamente diferentes, empezando por comparar sus tamaños, la primera nombrada anteriormente (GET Request) contiene solo 18 líneas y unos cuantos objetos predominando los headers, pero en el otro caso, se observan más de 10000 líneas compuestas de objetos y arrays como por ejemplo, coordenadas.

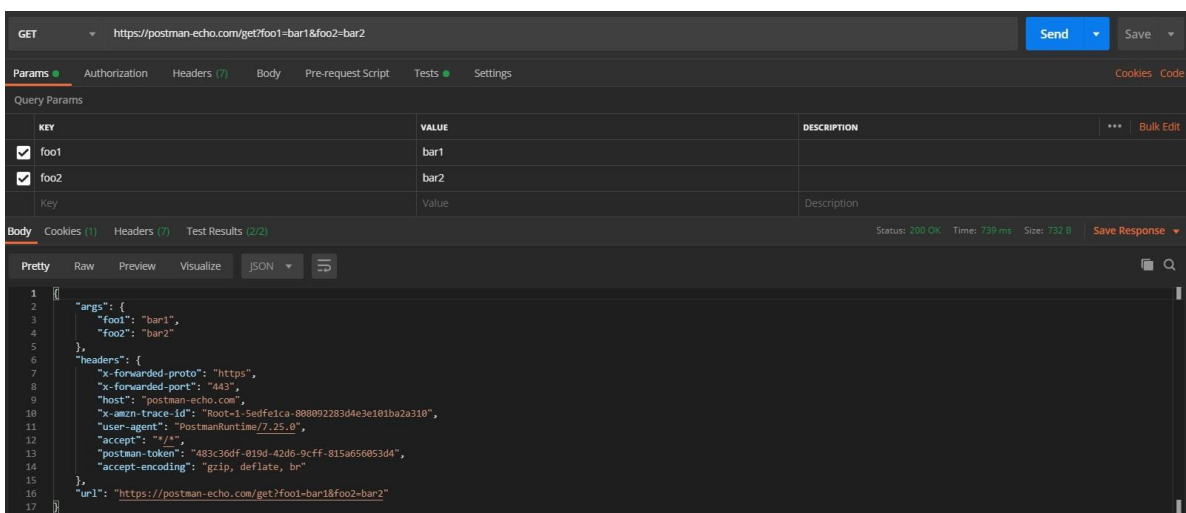


Ilustración 6: GET Request de Postman-Echo

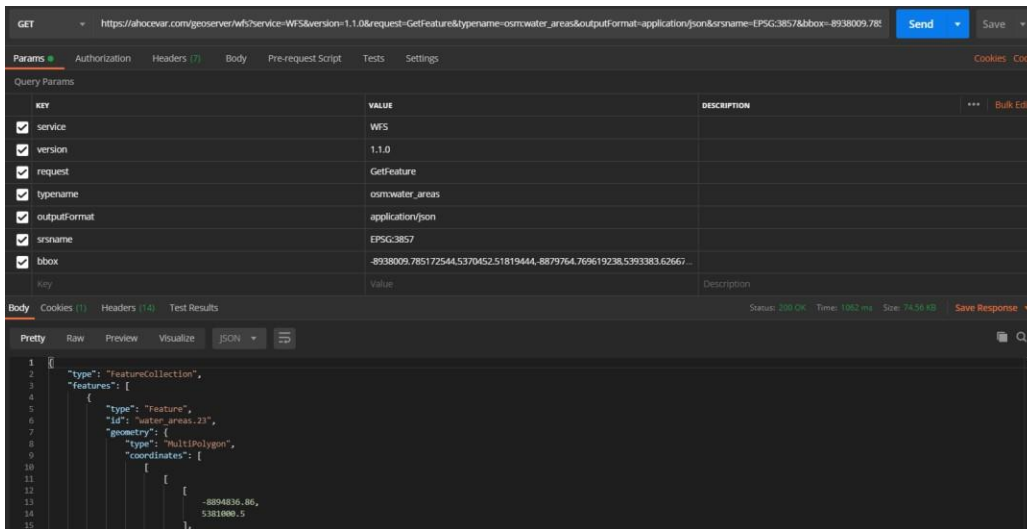


Ilustración 7: GET al link señalado

## 6. Metodos de Autenticación realice en Postman-Echo GET Basic Auth

Analice los parametros.

- Al realizar un GET Basic Auth del Postman-Echo se le pasan dos parametros importantes para poder realizar dicha autenticación como lo son el Username: “postman”, y el Password: “password”. De modo que al aplicar el comando se verifica si todo salió como debía (200 OK) y si en el JSONBody de respuesta el objeto “authenticated” tiene como valor “true”, lo que quiere decir que las credenciales son correctas y se ha logrado iniciar sesión de manera correcta.

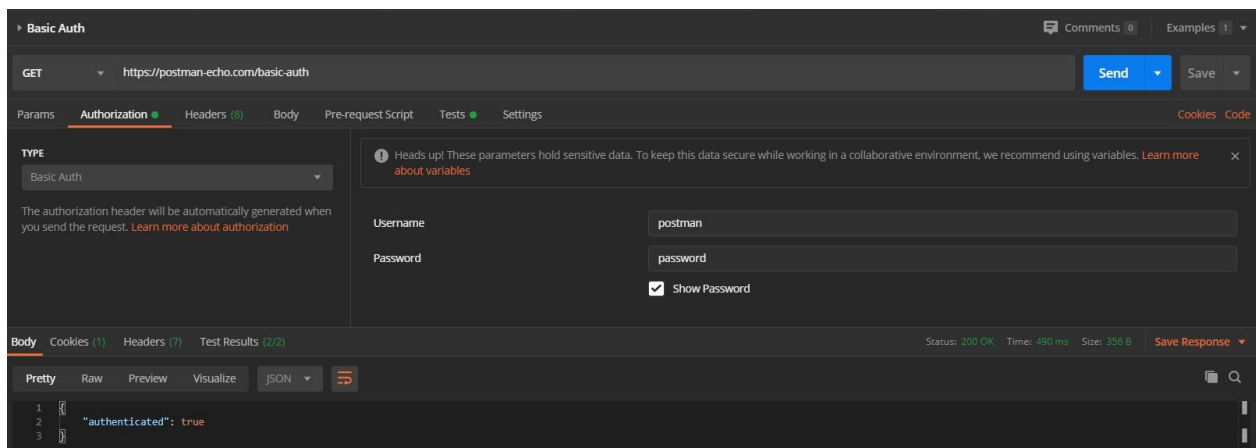


Ilustración 8: GET Basic Auth del Postman-Echo