

## Informe Taller # 1

### **Postman**

#### Autores:

Edward Javier Parada Silva

Estudiante de ingeniería de sistemas con código de estudiante 2182070

Jose Jaime Silva Martinez

Estudiante de ingeniería de sistemas con código de estudiante 2183075

Yuri Melissa García Niño

Estudiante de ingeniería de sistemas con código de estudiante 2182697

#### Docente:

Senior Dev. Jathinson Meneses

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas



Pensamiento Sistémico y Organizacional

Bucaramanga

2020

## Tabla de contenido

<b>Pregunta 1</b>	<b>3</b>
Enunciado	3
Desarrollo	3
<b>Pregunta 2</b>	<b>5</b>
Enunciado	5
Desarrollo	5
<b>Pregunta 3</b>	<b>6</b>
Enunciado	6
Desarrollo	6
<b>Pregunta 4</b>	<b>10</b>
Enunciado	10
Desarrollo	10
<b>Pregunta 5</b>	<b>16</b>
Enunciado	16
Desarrollo	16
<b>Pregunta 6</b>	<b>19</b>
Enunciado	19
Desarrollo	19

## Pregunta 1

### Enunciado

Acceder a un sitio estático usando un navegador. Los sitios a observar son:

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

<https://www.lashorasperdidas.com/>

### Desarrollo

Analizando la publicación [Página Web](#) del sitio [Wikipedia](#):

Al enviar la petición con los métodos en postman se obtuvieron dos respuestas en la parte body de la página, donde en algunos (GET, POST, entre otros) se obtenía el html de la página en sí y otro que obtenía el html de la página de error del sitio (cuando los servidores están en mantenimiento). La página maneja 3 cookies y 23 headers, entre los que se encuentran Date, un header general que contiene la fecha y hora en que se envió el mensaje; Server, que contiene información de cómo el servidor maneja las peticiones; Last-Modified, que indica la fecha y hora de la última vez que se modifica un recurso; Age, el número de segundos que un objeto ha estado en el caché de proxy; control de caché, estado de la conexión, etc.

Analizando el sitio [Las horas perdidas](#):

Siguiendo los pasos anteriores, se probaron los diferentes métodos obteniendo siempre el mismo cuerpo, el html de la página. El sitio no tiene cookies, tiene 9 headers de los cuales algunos se encuentran en los mencionados anteriormente (Date, Server, Vary, Connection) y adicionales a estos tiene Content-Type, que indica el tipo de medio del recurso; Transfer-Encoding, que especifica la forma de codificar usada para transferir la entrada al usuario, y su variación Content-Encoding, que se refiere al método usado para codificar todo el cuerpo; así como otros que especifican la tecnología que soporta la aplicación web.

Los códigos de estado de respuesta de HTTP se agrupan según su clase. Los primeros dos códigos (200 y 204) se encuentran en la clase de respuestas satisfactorias, el código 304 entra en la clasificación redirecciones y el 500 en errores de los servidores. La respuesta dada en cada caso se relaciona a continuación:

- Código 200: La solicitud ha tenido éxito (OK)
- Código 204: La petición se realizó correctamente, pero el cliente no necesita salir de su página actual, la respuesta no tiene ningún contenido (No Content)

- Código 304: Indica al cliente que la respuesta no ha sido modificada (Not Modified), así que lo redirige a la versión almacenada en caché
- Código 500: El servidor encontró una condición inesperada que no sabe cómo manejar (internal Server Error), lo cual le permite completar la petición

## Pregunta 2

### Enunciado

Entre a los sitios de Facebook y analice la autenticación.

### Desarrollo

Se entró a la dirección:

<https://www.facebook.com/photo/?fbid=10153172894140260&set=a.10150533715115260>

La cual tiene como parámetros de consulta:

- Fbid = 10153172894140260
- Set = a.10150533715115260

Al cambiar algunos números de los parámetros, la enlace se dañaba, pero al cambiar el **fbid** por uno ya existente, por ejemplo se tomó el id de la imagen en este enlace:

<https://www.facebook.com/photo?fbid=10152484276050260&set=ecnf.507320259>

Y se reemplazó en lugar del id en el primer enlace, lo que se obtuvo es que se mostraba la imagen en el segundo link. Mientras que si solo se cambiaba el **set**, no ocurría ninguna novedad en la respuesta de facebook.

Al experimentar con todos los métodos, pasó algo similar a cuando se probó en wikipedia, algunos métodos arrojaban como resultado una página de error, otros no arrojaban nada, y los que sí lo hacían daban la información de la imagen que se estaba pasando con el link.

### Pregunta 3

#### Enunciado

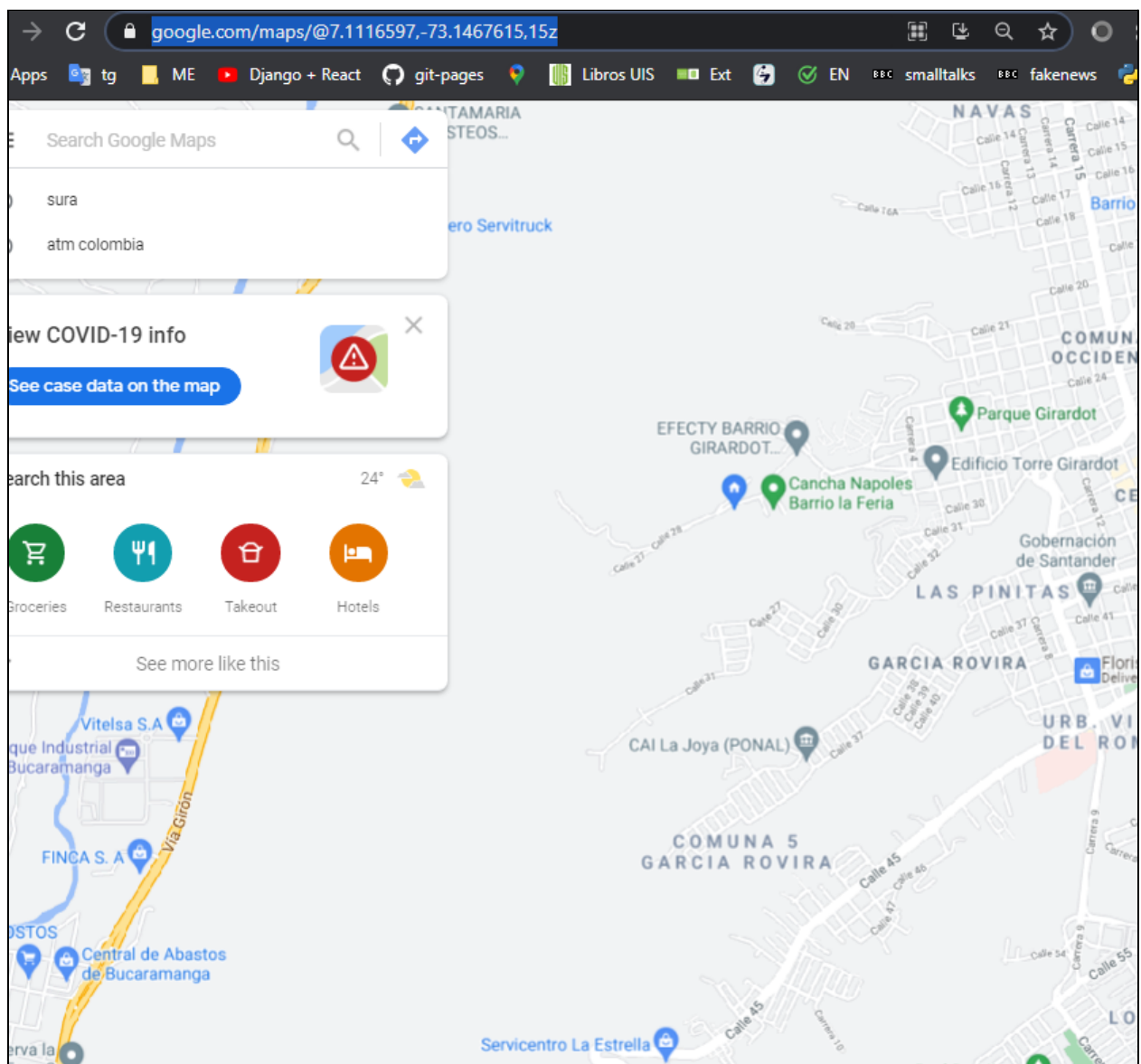
Entre a google maps mire los parámetros de la url cambielos y analice resultados cuáles son los parámetros que pudo cambiar y cuales aparecen con las funcionalidades.

#### Desarrollo

Al entrar a google maps por la página web la url inicial es:

<https://www.google.com/maps/@7.1116597,-73.1467615,15z>

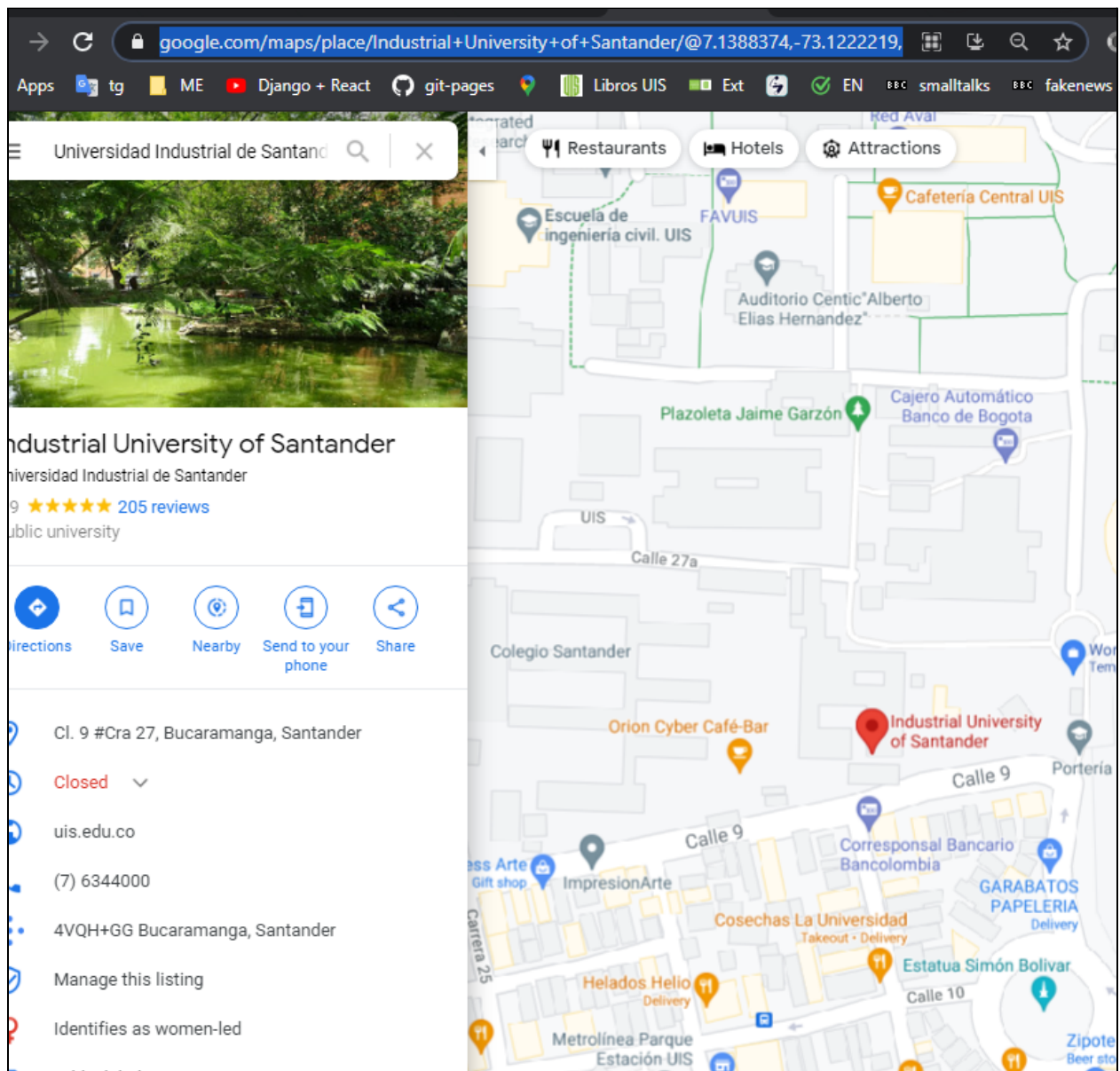
Figura 1 URL inicial



El primer parámetro después del @ (7.1116597) corresponde a la latitud, mientras que el siguiente parámetro después de la , representa la longitud (-73.1467615 en este caso) y el último parámetro corresponde a la altura en el eje z, se podría entender como el zoom.

Cuando se clickea en alguna locación, por ejemplo en la **Fig. 2** se clickeo sobre la ubicación de la UIS en Bucaramanga, la url cambia de tal forma que el nombre del lugar aparece en la URL.

**Figura 2** Estado de google maps al clickear un lugar



La URL completa es:

```
https://www.google.com/maps/place/Industrial+University+of+Santander/@7.1388374,-73.1222219,18z/data=!4m5!3m4!1s0x8e68156d1c0c8689:0x873d16bf1221d419!8m2!3d7.1388376!4d-73.1211278
```

Se pueden destacar algunos parámetros nuevos como **place** y **data**. El parámetro **place** corresponde con el nombre del lugar clickeado, mientras que el parámetro **data** no se encuentra documentado en la API de google maps. Sin embargo, con base en este [artículo](#), se podría decir que el parámetro **data** representa el ID del centroide de la región buscada.

Por otro lado, se observa que al buscar algún lugar por nombre en el buscador de la app, esta cambia la URL y agrega el parámetro **search** y el texto ingresado como se observa en la **Fig. n**. La URL completa es:

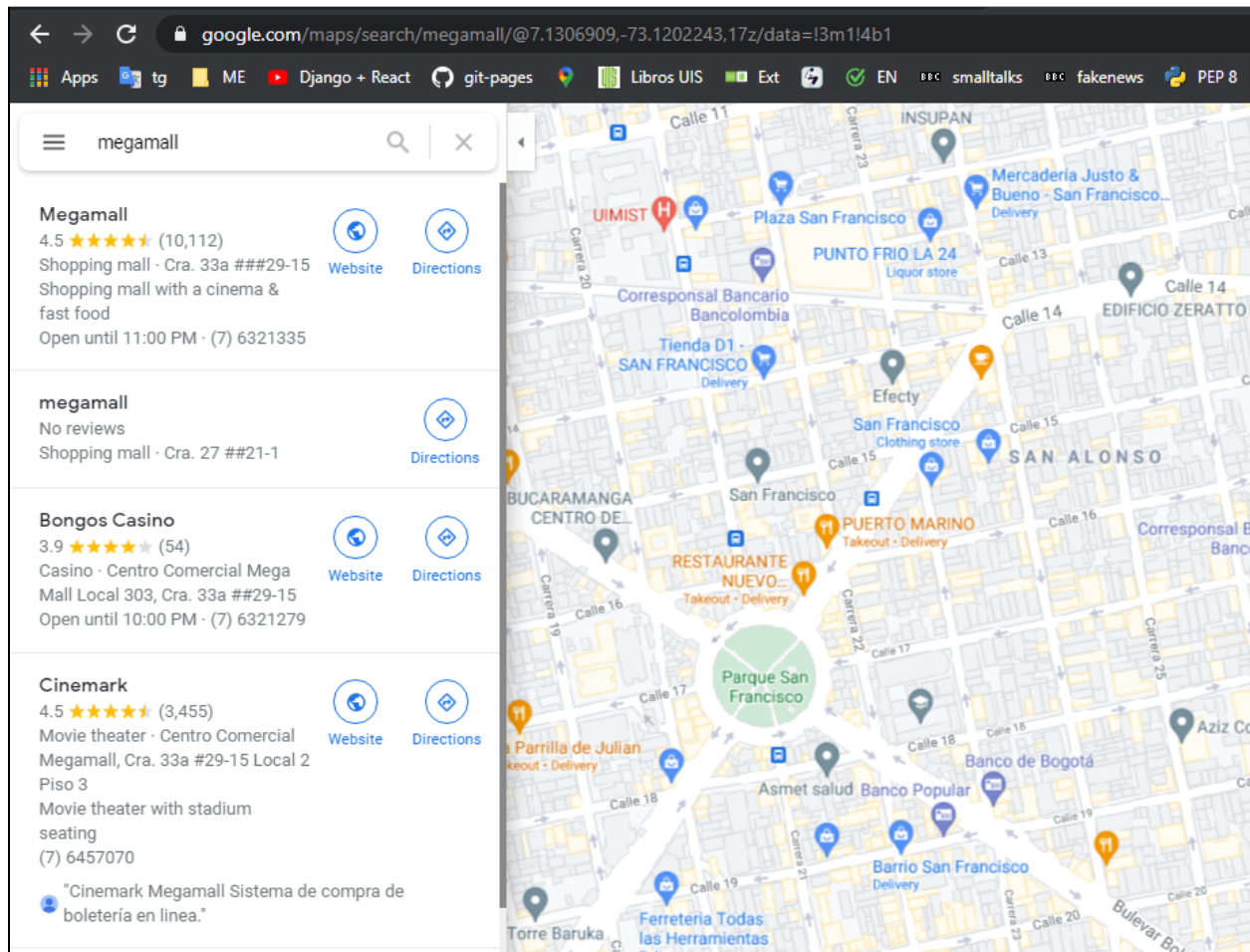
```
https://www.google.com/maps/search/megamall/@7.1306909,-73.1202243,17z/data=!3m1!4b1
```

Por otro lado, cuando se piden las indicaciones a un lugar, google maps agrega el parámetro **dir** el cual es un string que contiene las coordenadas de latitud y longitud del lugar de origen. Y al. Por ejemplo, al pedir indicaciones desde mi casa, cuyas coordenadas son (7.119579, -73.142387) a la UIS, la URL resultante es:

```
https://www.google.com/maps/dir/'7.119579,-73.142387'/Industrial+University+of+Santander,+Cl.+9+%23Cra+27,+Bucaramanga,+Santander/@7.1321619,-73.1380388,16z/data=!4m12!4m11!1m3!2m2!1d-73.142387!2d7.119579!1m5!1m1!1s0x8e68156d1c0c8689:0x873d16bf1221d419!2m2!1d-73.1211278!2d7.1388376!3e0
```



**Figura 3** Estado de google maps al clickear un lugar



## Pregunta 4

### Enunciado

Entre a la siguiente dirección:

<https://docs.postman-echo.com/?version=latest>

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

- GET GET Request
- POST POST Raw Text
- POST POST Form Data
- PUT PUT Request
- PATCH PATCH Request
- DEL DELETE Request

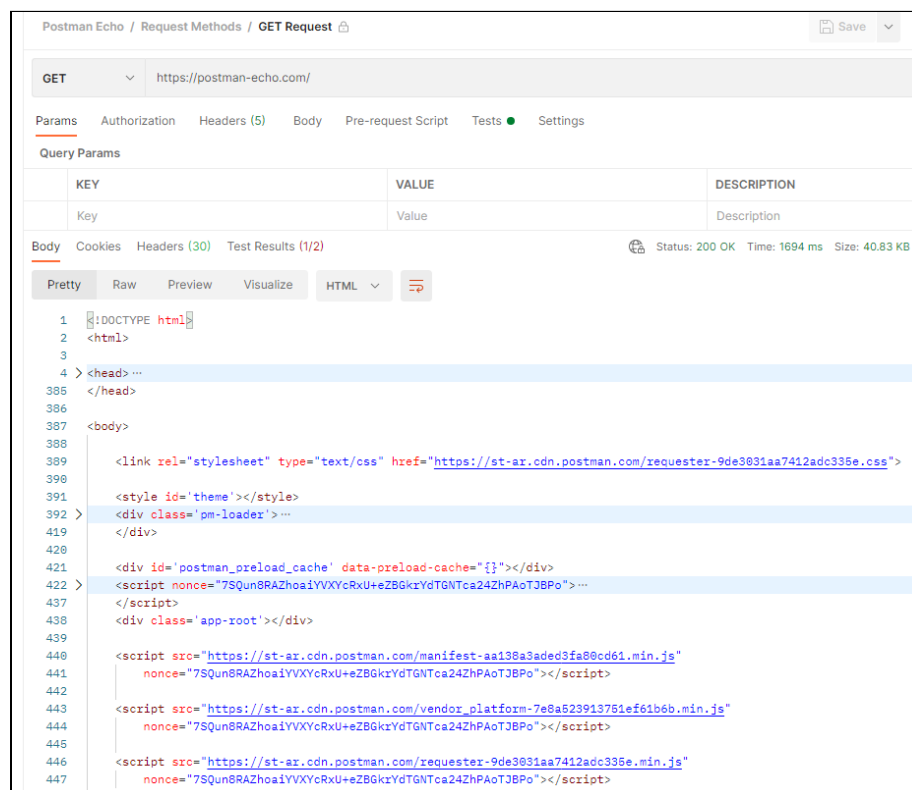
Analice cada comando y que función tiene dentro de una página web.

### Desarrollo

#### GET GET Request

Al realizar una GET request a <https://postman-echo.com/>, se observa que el servidor responde con el archivo HTML de la dirección web solicitada.

**Figura 4** Respuesta del servidor



The screenshot shows the Postman Echo interface for a GET request to `https://postman-echo.com/`. The response is displayed in the 'Body' tab, showing the HTML structure of the page. The status is 200 OK, with a response time of 1694 ms and a size of 40.83 KB.

```
1 <!DOCTYPE html>
2 <html>
3
4 <head> ...
385 </head>
386
387 <body>
388
389 <link rel="stylesheet" type="text/css" href="https://st-ar.cdn.postman.com/requester-9de3831aa7412adc335e.css">
390
391 <style id="theme"></style>
392 <div class="pm-loader"> ...
419 </div>
420
421 <div id="postman_preload_cache" data-preload-cache="{}"></div>
422 <script nonce="7SQun8RAZhaoiYVXYcRxU+eZBGkrYdTGNTca24ZhPaotJBpO"> ...
437 </script>
438 <div class="app-root"></div>
439
440 <script src="https://st-ar.cdn.postman.com/manifest-aa138a3aded3fa80cd61.min.js"
441 nonce="7SQun8RAZhaoiYVXYcRxU+eZBGkrYdTGNTca24ZhPaotJBpO"></script>
442
443 <script src="https://st-ar.cdn.postman.com/vendor_platform-7e8a523913761ef61b6b.min.js"
444 nonce="7SQun8RAZhaoiYVXYcRxU+eZBGkrYdTGNTca24ZhPaotJBpO"></script>
445
446 <script src="https://st-ar.cdn.postman.com/requester-9de3831aa7412adc335e.min.js"
447 nonce="7SQun8RAZhaoiYVXYcRxU+eZBGkrYdTGNTca24ZhPaotJBpO"></script>
```

El comando GET tiene como función realizar una petición al servidor, en este caso cuando vamos a la URL realizamos un GET a <https://postman-echo.com/> lo que hacemos es pedir el archivo html de la página.

### POST POST Raw Text

Cuando se realiza un POST Raw Text a <https://postman-echo.com/post>, el servidor nos devuelve el siguiente JSON como respuesta:

```
{
  "args": {},
  "data": "This is expected to be sent back as part of response body.",
  "files": {},
  "form": {},
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",
    "x-amzn-trace-id": "Root=1-6084c87b-56af81f8518da8a52824fea9",
    "content-length": "58",
    "content-type": "text/plain",
    "user-agent": "PostmanRuntime/7.28.0",
    "accept": "*/*",
    "cache-control": "no-cache",
    "postman-token": "fed09190-4dcc-447c-bd2c-2546ac10ebc6",
    "accept-encoding": "gzip, deflate, br",
    "cookie":
"sails.sid=s%3Ais2AqmrUuz07uEFH2Ajst5CkHiqSaPG.%2B6nGdwJ9e5FLNYa5%2Fg%2Bl
VaRnrUy0tWRQjwE8RFd6FhY"
  },
  "json": null,
  "url": "https://postman-echo.com/post"
}
```

El comando POST Raw Text se utiliza para enviar texto o strings directamente al servidor sin la necesidad del uso de formularios de HTML.

### POST POST Form Data

Cuando se realiza un POST mediante un formulario a <https://postman-echo.com/post>, el servidor nos devuelve el siguiente JSON como respuesta:

```
{
  "args": {},
  "data": "",
  "files": {},
  "form": {
    "foo1": "bar1",
    "foo2": "bar2"
  },
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",
    "x-amzn-trace-id": "Root=1-6084d246-586303f409f0e4af3ec69431",
    "content-length": "19",
    "user-agent": "PostmanRuntime/7.28.0",
    "accept": "*/*",
    "cache-control": "no-cache",
    "postman-token": "b6851703-9ab9-4b84-b342-40a41eaa3bd7",
    "accept-encoding": "gzip, deflate, br",
    "cookie":
"sails.sid=s%3AseKJWa_z70h5pvS1UVmLyCPkc0FTs3g.9ylo1kNdoIGa26iHEks4zCqK
ML2e7SVCqFhvh4PROVI",
    "content-type": "application/x-www-form-urlencoded"
  },
  "json": {
    "foo1": "bar1",
    "foo2": "bar2"
  },
  "url": "https://postman-echo.com/post"
}
```

Se destaca que en este caso la respuesta del servidor contiene un formulario con dos parámetros, y el valor de **data** es vacío. El comando POST Form Data permite enviar datos al servidor a través de formularios, lo cual es útil en el caso de HTML para recibir datos por parte del usuario.

### PUT PUT Request

Cuando se realiza un PUT a <https://postman-echo.com/put>, el servidor nos devuelve el siguiente JSON como respuesta:

```
{
  "args": {},
```

```

    "data": "This is expected to be sent back as part of response
body.",
    "files": {},
    "form": {},
    "headers": {
      "x-forwarded-proto": "https",
      "x-forwarded-port": "443",
      "host": "postman-echo.com",
      "x-amzn-trace-id": "Root=1-6084d389-11edf6b25c8484ee4c965973",
      "content-length": "58",
      "content-type": "text/plain",
      "user-agent": "PostmanRuntime/7.28.0",
      "accept": "*/*",
      "cache-control": "no-cache",
      "postman-token": "6cdea80e-5ebd-4646-ac54-1366652832cc",
      "accept-encoding": "gzip, deflate, br",
      "cookie":
"sails.sid=s%3AunUJnv90a9rrTNJh8k4LBnNTIzNC1dAd.2MXLqTdXu6XuuISIKC8ZkVRh
JrN3WTu58dB%2Fy7hNvEs"
    },
    "json": null,
    "url": "https://postman-echo.com/put"
  }

```

Se observa que entre los headers del JSON que nos devuelve el servidor se encuentra un content-type distinto al que teníamos en POST, esto es porque el comando PUT se utiliza para transferir datos al servidor, como subir un archivo, o foto.

### **PATCH PATCH Request**

Cuando se realiza un PATCH a <https://postman-echo.com/patch>, el servidor nos devuelve el siguiente JSON como respuesta:

```

{
  "args": {},
  "data": "This is expected to be sent back as part of response
body.",
  "files": {},
  "form": {},
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",

```

```

    "x-amzn-trace-id": "Root=1-6084d488-6a2e18dc6e0a764f462ce8bd",
    "content-length": "58",
    "content-type": "text/plain",
    "user-agent": "PostmanRuntime/7.28.0",
    "accept": "*/*",
    "cache-control": "no-cache",
    "postman-token": "d1ab8069-a679-466b-a366-eb444273ae24",
    "accept-encoding": "gzip, deflate, br",
    "cookie":
"sails.sid=s%3A4bhEtK0sXK-Cx5FEUTzFE0lJ9XsmRcyc.Nmt6lFeM0Of8VyNGjx6zXfar
KRpQxgv6HBQfdXNI%2BXI"
  },
  "json": null,
  "url": "https://postman-echo.com/patch"
}

```

El comando PATCH se utiliza para actualizar recursos en el servidor, estos recursos pueden ser archivos como imágenes, videos, etc.

### **DEL DELETE Request**

Cuando se realiza un DELETE a <https://postman-echo.com/delete>, el servidor nos devuelve el siguiente JSON como respuesta:

```

{
  "args": {},
  "data": "This is expected to be sent back as part of response
body.",
  "files": {},
  "form": {},
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",
    "x-amzn-trace-id": "Root=1-6084d50d-704b23611c984e6318421cc5",
    "content-length": "58",
    "content-type": "text/plain",
    "user-agent": "PostmanRuntime/7.28.0",
    "accept": "*/*",
    "cache-control": "no-cache",
    "postman-token": "d506249f-15db-4bfc-ad05-a647f8705dde",
    "accept-encoding": "gzip, deflate, br",
    "cookie":

```

```
"sails.sid=s%3Ahptx-pv8kONDW-TvRdUqz6uyTcU_Jn5B.fDVfRJ%2FE7ANk9Rc7UmcoeK
Ti%2FYJerCXFJtPgOkQQM9I"
  },
  "json": null,
  "url": "https://postman-echo.com/delete"
}
```

El comando DELETE se emplea para borrar o eliminar recursos de un servidor.

## Pregunta 5

### Enunciado

De postman-echo realice un get de respuesta json, analice la respuesta. Ahora compárelo contra el siguiente: [linkjsongis](https://linkjsongis.com)

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

### Desarrollo

Considerando la documentación existente, con postman-echo se realiza un GET al siguiente enlace:

```
https://postman-echo.com/get?foo1=bar1&foo2=bar2
```

La respuesta en JSON es la siguiente:

```
{
  "args": {
    "foo1": "bar1",
    "foo2": "bar2"
  },
  "headers": {
    "x-forwarded-proto": "https",
    "x-forwarded-port": "443",
    "host": "postman-echo.com",
    "x-amzn-trace-id": "Root=1-6084bcbc-40e70590466cca6d1706c9f1",
    "user-agent": "PostmanRuntime/7.26.10",
    "accept": "*/*",
    "postman-token": "e3c38b49-fe9a-4c1b-beeb-c0f7fdc14b91",
    "accept-encoding": "gzip, deflate, br",
    "cookie":
"sails.sid=s%3AfwB9E55LaSBlG_iZ25q_8p5y5jTW043h.nZAHAdB0de9nGJR3wF%2Bu
zeKM%2FJ%2FASXyeMaajAFWhPFA"
  },
  "url": "https://postman-echo.com/get?foo1=bar1&foo2=bar2"
}
```



En esta respuesta se encuentran los parámetros foo y bar, que se encontraban en el enlace usado, aparecen como parte de la respuesta en "args", del mismo modo en el encabezado (headers) el tipo de protocolo usado (HTTPS), el puerto accedido, la página host y el enlace sobre el que el get se hizo entre otros elementos.

Continuando con la actividad se hizo lo mismo con el siguiente enlace:

```
https://ahocevar.com/geoserver/wfs?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
```

La respuesta en JSON obtenida constó de más de 10000 líneas, por brevedad se muestran las primeras y últimas.

```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "id": "water_areas.23",
      "geometry": {
        "type": "MultiPolygon",
        "coordinates": [
          [
            [
              [
                -8894836.86,
                5381000.5
              ],
              [
                -8894825.07,
                5381032.26
              ]
            ]
          ]
        ]
      },
      "geometry_name": "the_geom",
      ...
    }
  ]
}
```

```

        "properties" : {
            "osm_id" : 25465158,
            "natural" : "natural",
            "waterway": null,
            "landuse" : "reservoir",
            "name"    : "Forest Grove Storm Water Management Pond"
        }
    },
    ],
    "totalFeatures" : 100,
    "numberMatched" : 100,
    "numberReturned": 100,
    "timeStamp"     : "2021-04-25T00:35:15.796Z",
    "crs"           : {
        "type"      : "name",
        "properties": {
            "name": "urn:ogc:def:crs:EPSG::3857"
        }
    }
}

```

En el mismo enlace se notó la existencia de varios parámetros (sobre los cuales Postman avisó), estos fueron **service**, **version**, **request**, **typename**, **outputFormat**, **srsname** y **bbox**, considerando que la respuesta esta diseñada para mostrar algún tipo de gráfico (dada la existencia de elementos como **geometry** y **MultiPolygon**), es probable que estos influyesen en la representación de dicho gráfico.

Notablemente el encabezado mostró algunas similitudes de acuerdo a Postman, como la existencia de una fecha y una codificación.

## Pregunta 6

### Enunciado

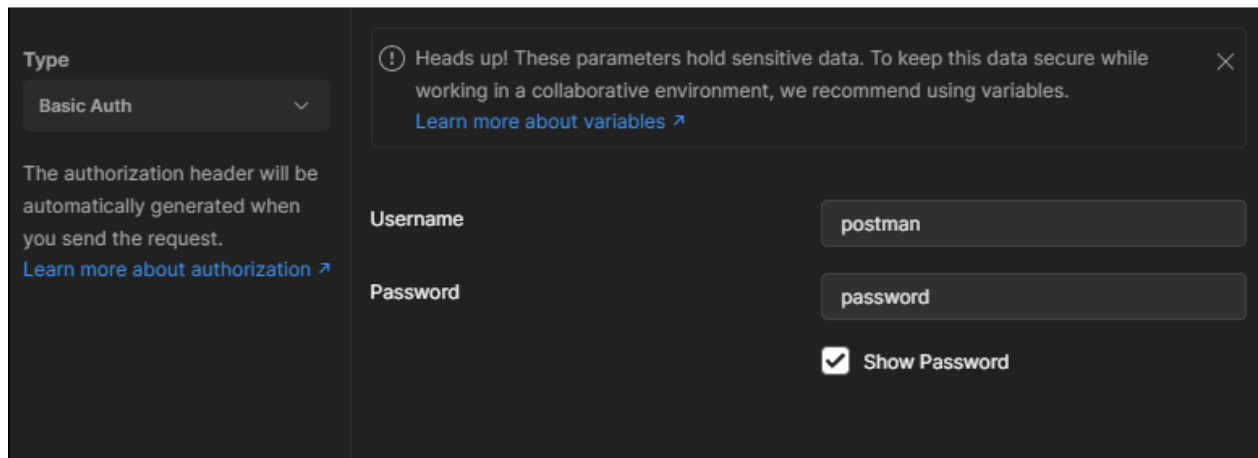
Métodos de autenticación: Realice en postman-echo GET Basic Auth. Analice los parámetros.

### Desarrollo

De acuerdo a la documentación de postman-echo, el GET Basic Auth va a devolver un status code de 200 siempre que se envíen los parámetros de usuario y contraseña del siguiente modo:

Username: postman
Password: password

De lo contrario el status code será 401 *Unauthorized*, esto para el GET significa que antes de enviarse primero es necesario dirigirse a la sección de Auth e insertar las credenciales mostradas.



Type: Basic Auth

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [Learn more about variables](#)

Username: postman

Password: password

☒ Show Password

En la sección de scripts se muestra que con esto se genera un encabezado de autorización, que de acuerdo a la documentación corresponden a las credenciales insertadas en base64, el GET en cURL se muestra a continuación:

```
curl --location \
--request GET 'https://postman-echo.com/basic-auth' \
--header 'Authorization: Basic cG9zdG1hbJpwYXNzd29yZA==' \
```

```
--header 'Cookie:  
sails.sid=s%3A4hDFLjZSgLoHgiQ-iney-XbS5dM1eSvW.14VCkLrdLw1PUK6reTAyuyN  
Mfl7xl2amQjrDnPymQTA'
```

Si todo sale bien en la respuesta se encontrará el código de status 200 *Ok* y el siguiente JSON:

```
{  
  "authenticated": true  
}
```

Notablemente en este caso no se usan parámetros del mismo modo que se hizo anteriormente.