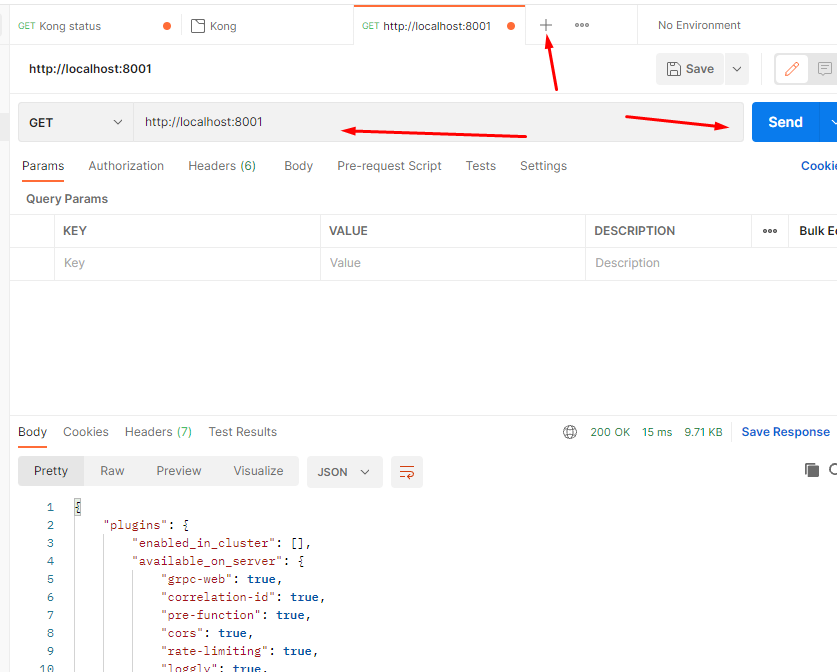
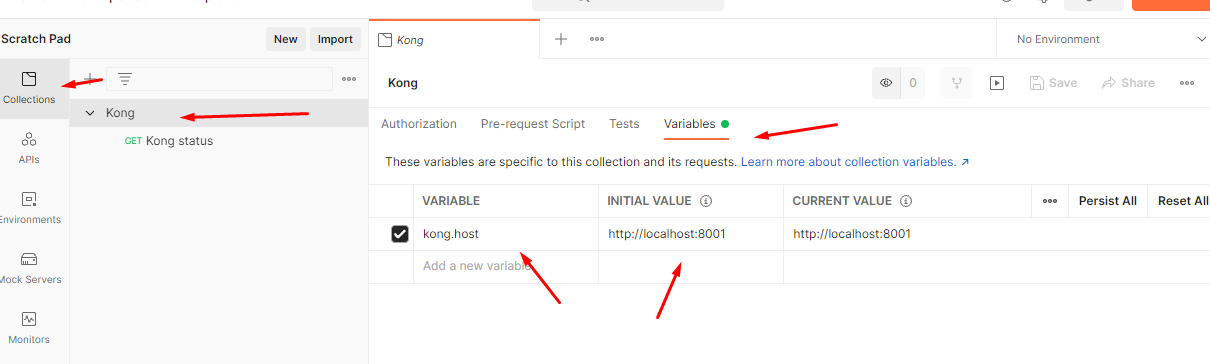
**KONG**

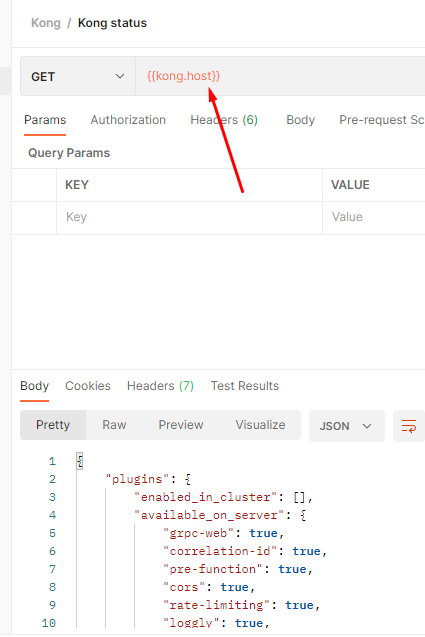
Podemos enviar solicitudes con postman



Podemos crear colecciones, guardar las solicitudes y crear variables



Las variables de escriben entre dos llaves **{{variable}}**



**Documentación**

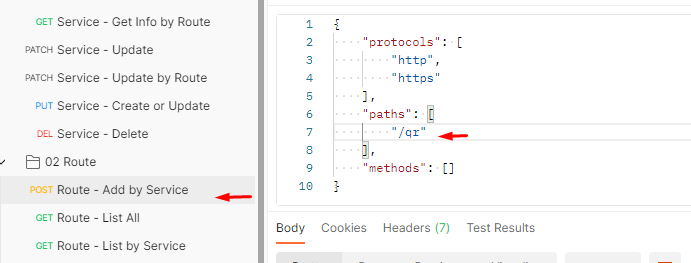
<https://docs.konghq.com/gateway-oss/1.0.x/admin-api/>

<https://goqr.me/api/doc/>

<https://pantsel.github.io/konga/>

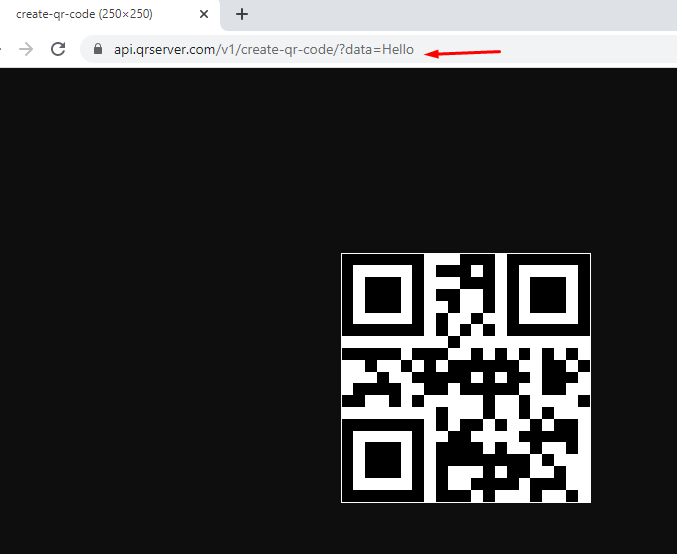
**Servicios y rutas de Kong**



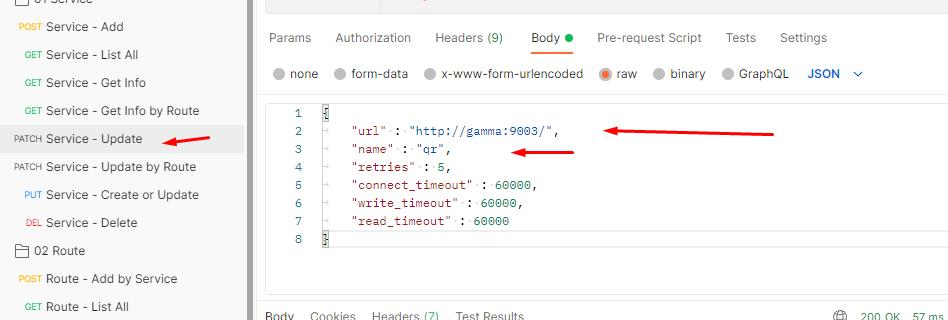


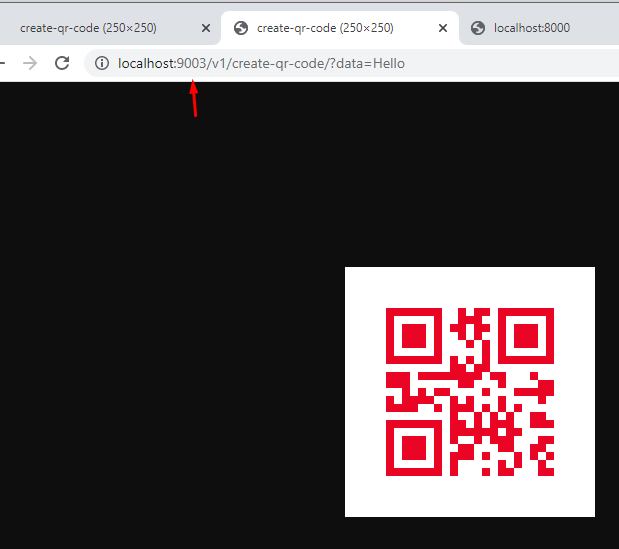
**Servicios y rutas prueba**

Creamos un qr con hola



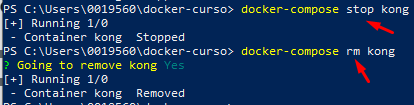
**Actualizar servicios y rutas**

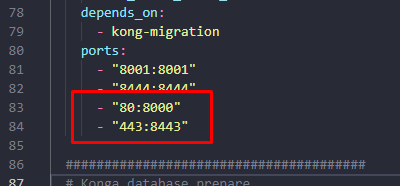




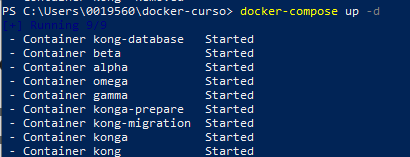
**Cambiar puertos de Kong**

Debemos parar y eliminar el Docker compose que creamos anteriormente y modificar el archivo

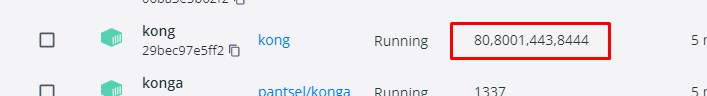




Por último, lo levantamos

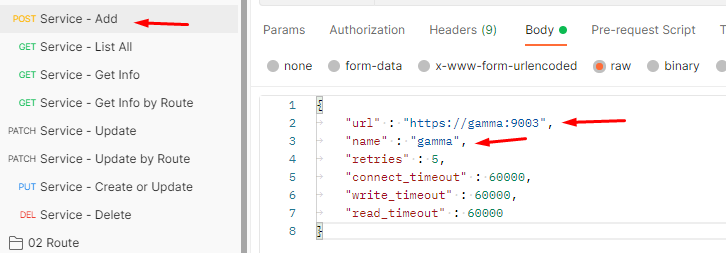


Dresde el Docker desktop vemos que ha cambiado

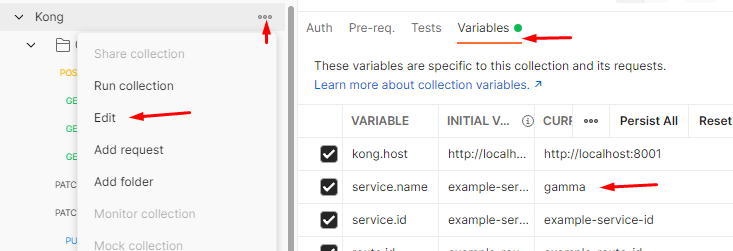


**Consumidores de kong**

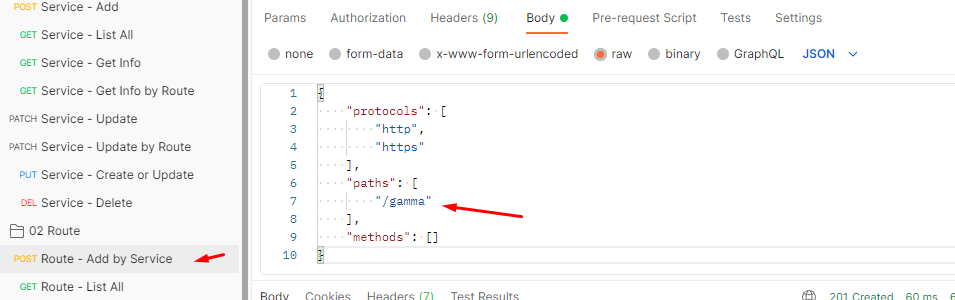
Agregamos un nuevo servicio a gamma



Modificamos la variable

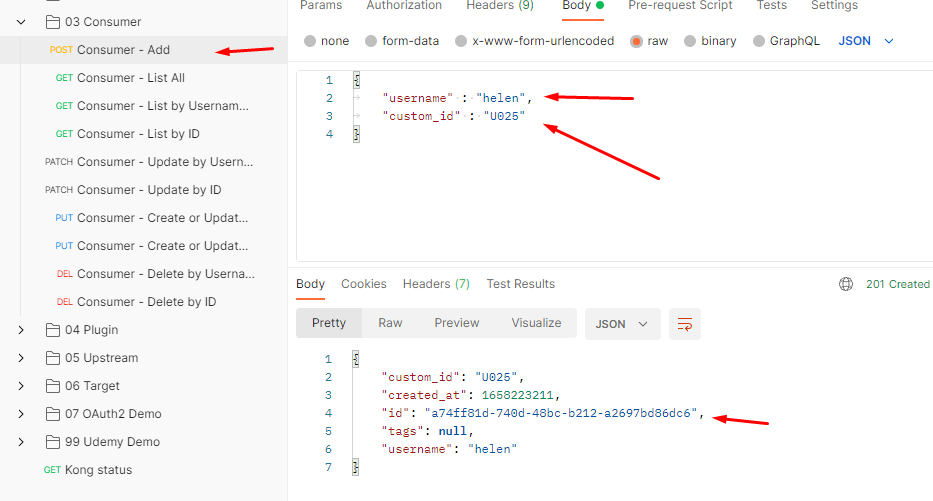


Le agregamos al servicio una ruta



Agregamos un consumidor

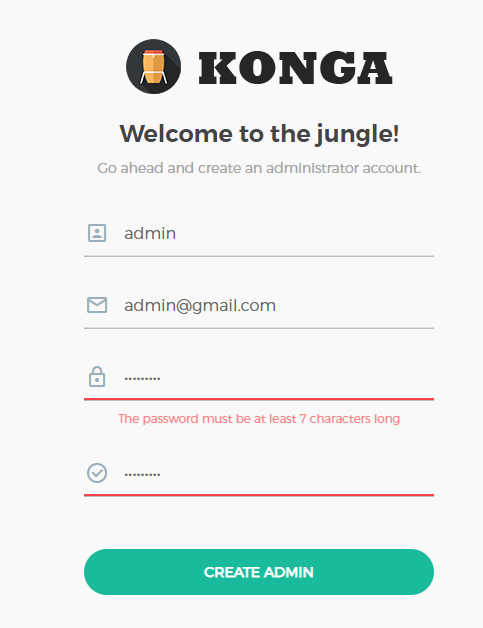
En la respuesta podemos ver que se crea otro id



**KONGA**

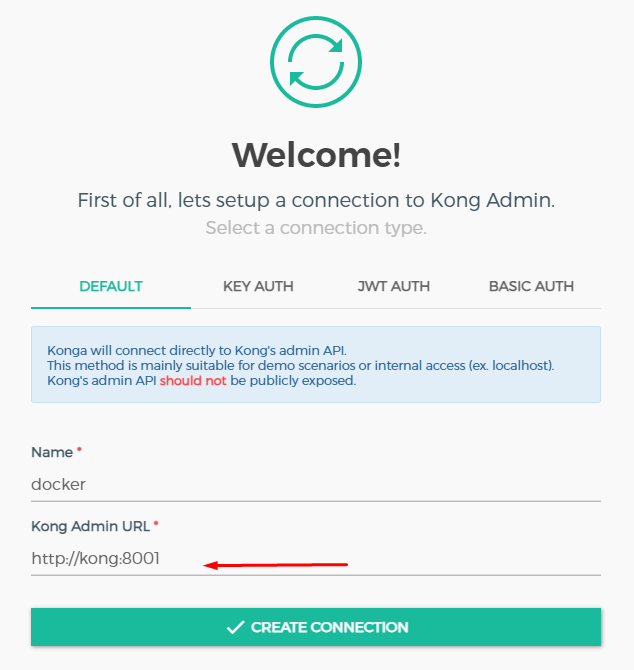
Para acceder localhost:1337

Lo primero es crear el usuario admin

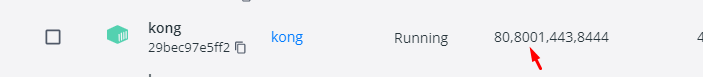


Password: 12345Ab##

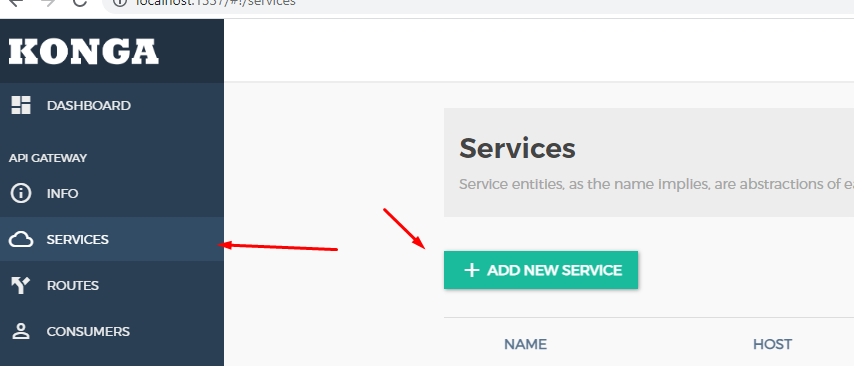
Crearemos una conexión básica



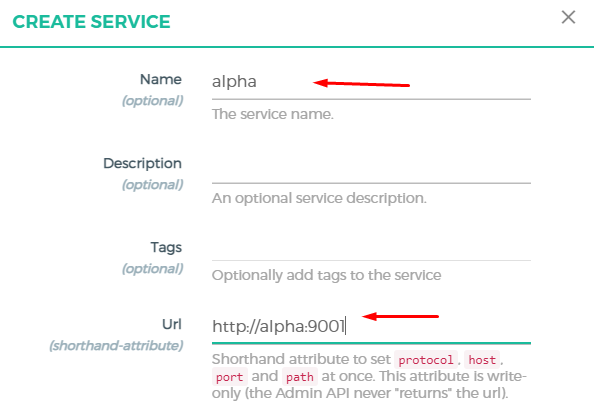
El puerto debe ser el del Kong



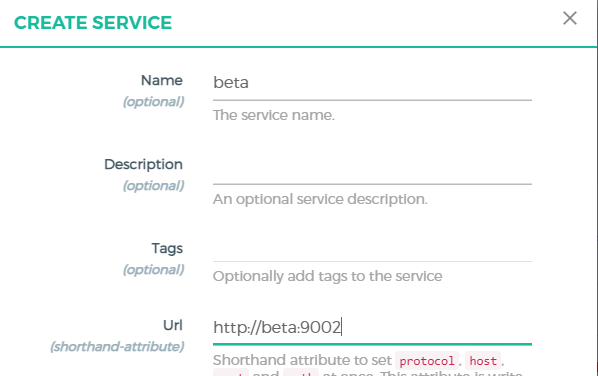
Para agregar un servicio en konga

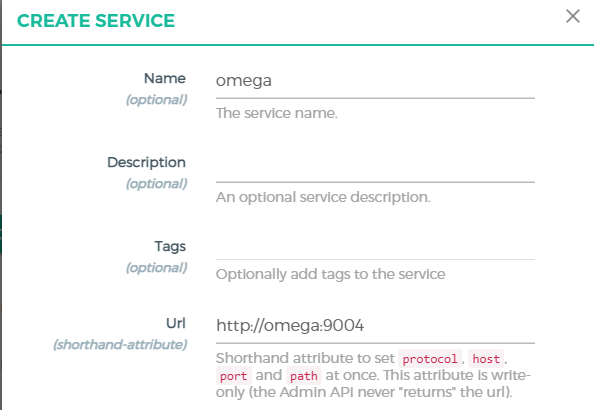


Le ponemos un nombre y la url lo demás por defecto



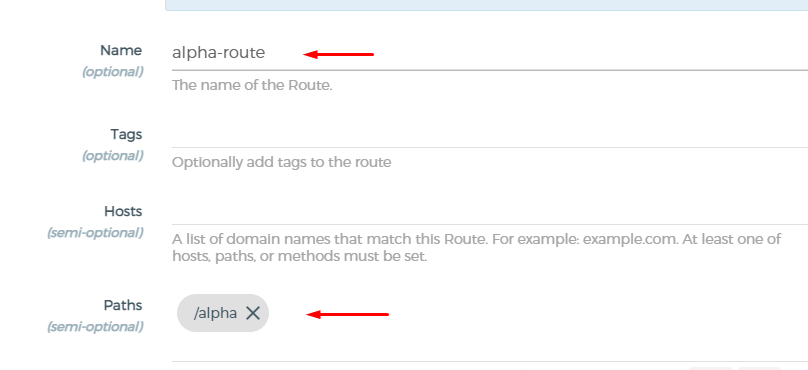
A continuación, creamos el resto de servicios





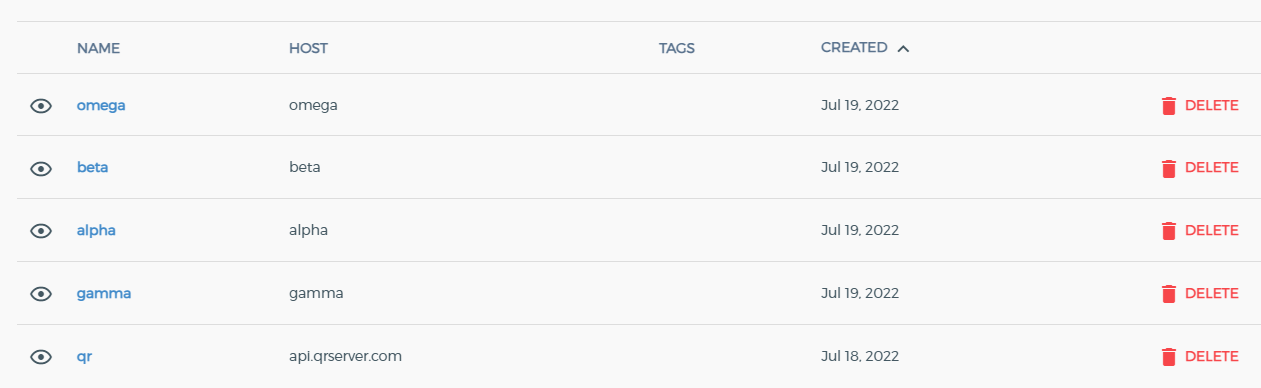
Y ahora vamos a crear una ruta para cada servicio, para ello debemos entrar en cada uno de los servicios y crear la que le corresponde, el path que tienen en este caso es el nombre del contenedor docker



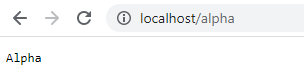


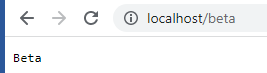
Importante para gamma borrar las rutas que hemos creado con postman que no sirven y cambiar el protocolo a http

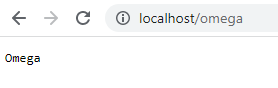
Repetimos el proceso con los demás



Después de esto ya podríamos acceder a ellos con loaclhost/path

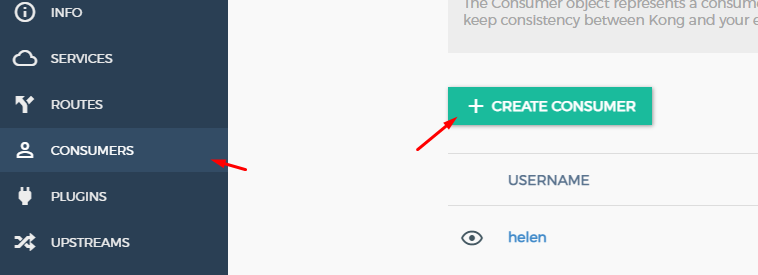


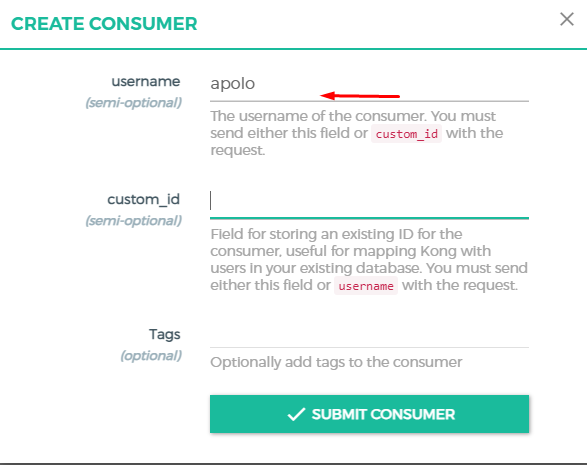




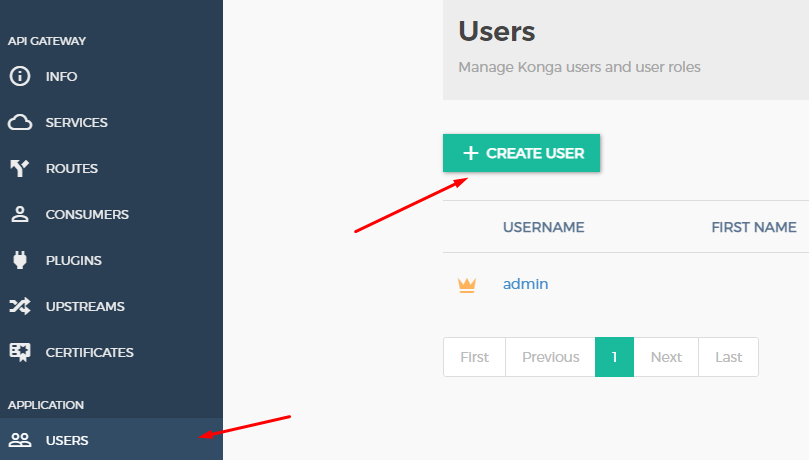


Ahora agregaremos un consumidor

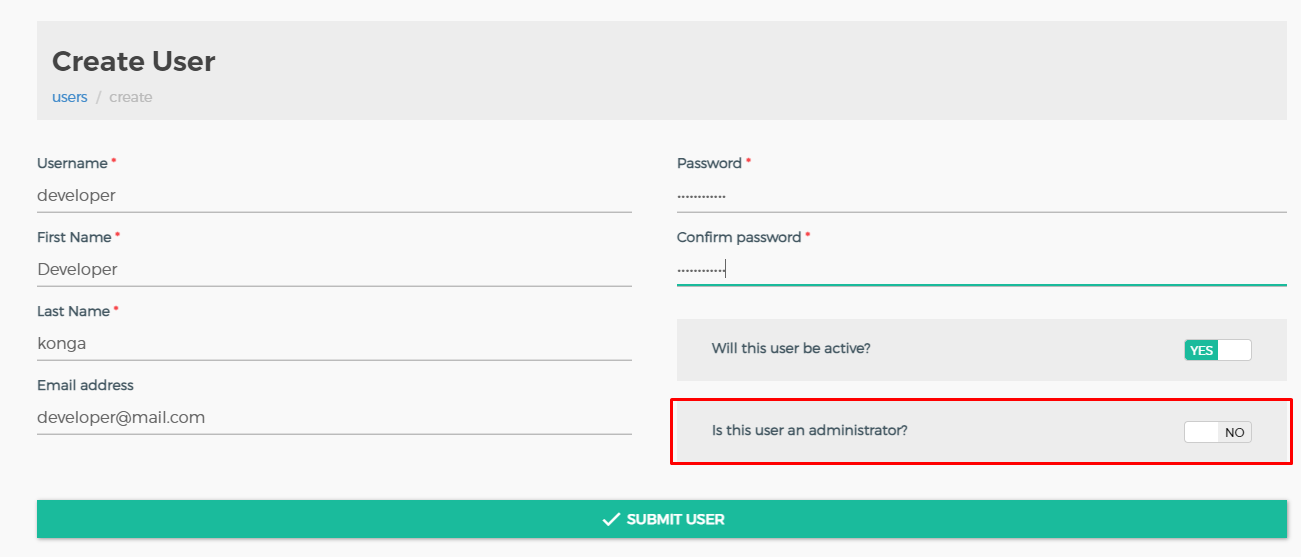




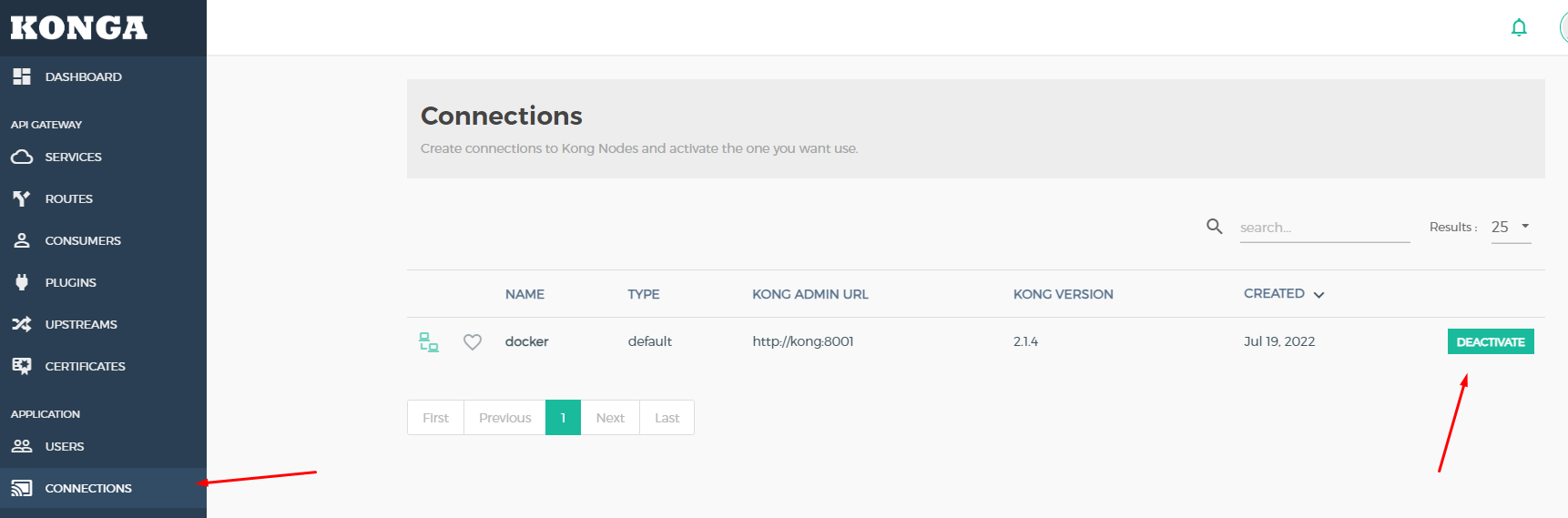
**Gestión de usuarios**



Si el usuario no es administrador solo tendrá acceso de lectura

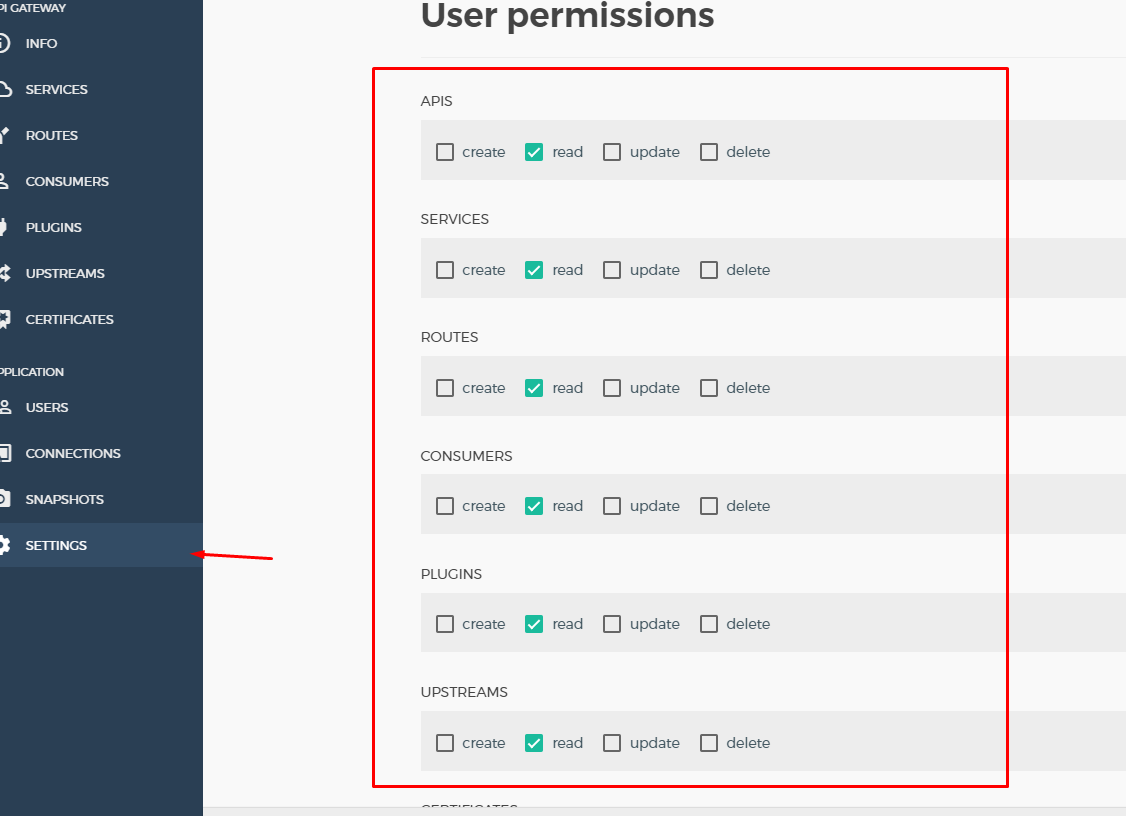


Iniciamos sesión en el usuario y lo primero es conectarnos a la conexión que creamos anteriormente.



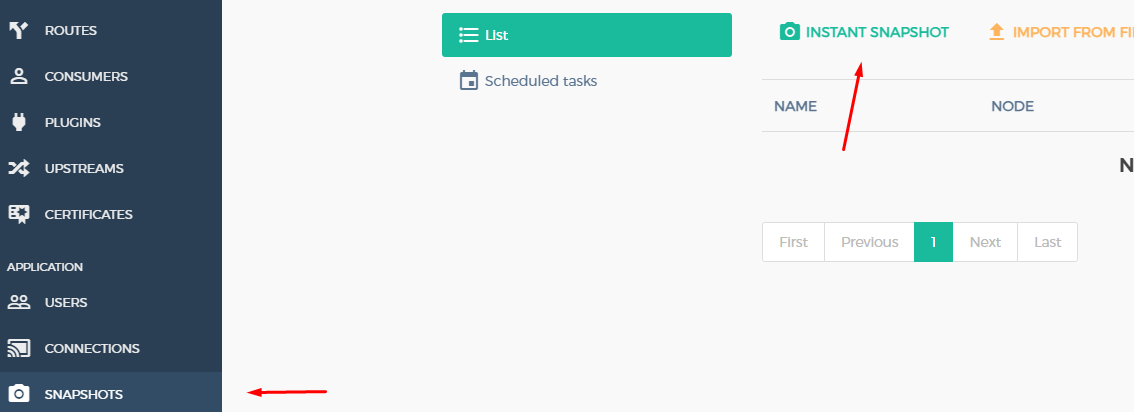
Volvemos al usuario admin

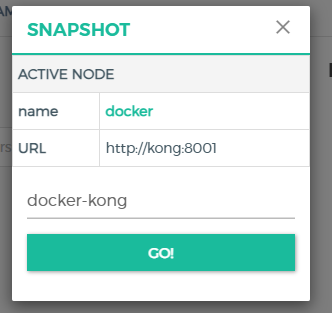
Si nos vamos a ajustes y permisos de usuarios podremos modificar los permisos que tienen los usuarios no admins



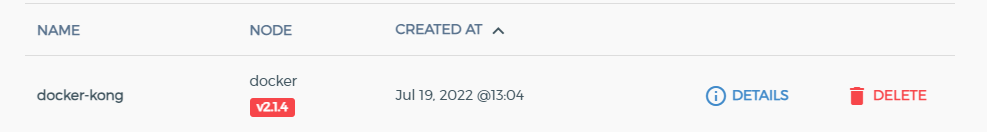
**Realizar copias de seguridad con konga**

Para ello





Si hacemos clic en el podemos bver los objetos guardados

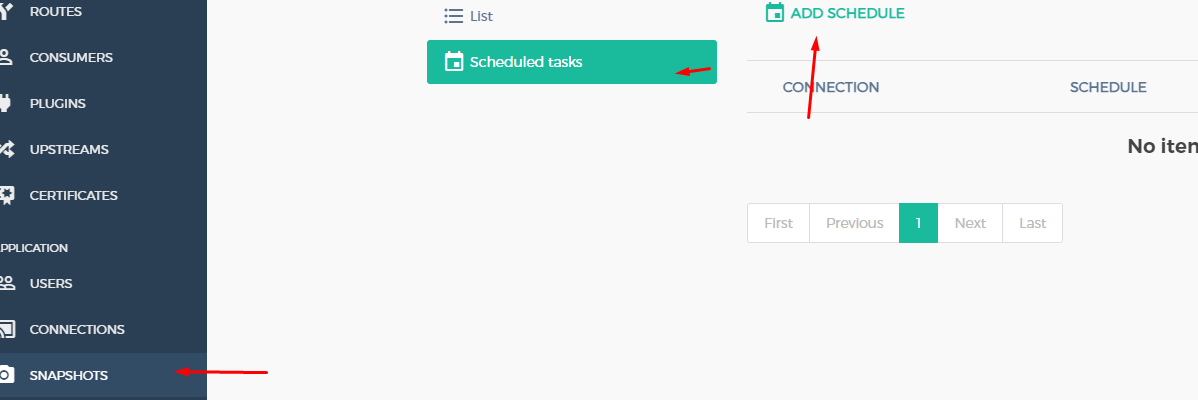


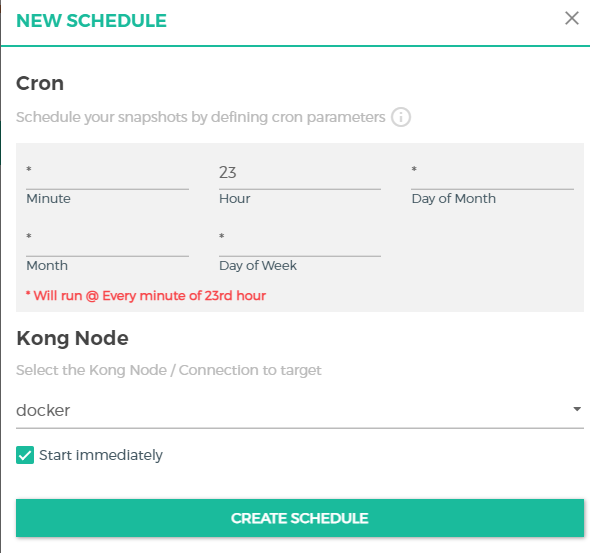
Podemos exportarlo a un archivo .json



En caso de querer restaurar una copia solo nos hará falta el archivo .json

Konga también proporciona un programador para automatizar la creación de copias de seguridad (Cron)

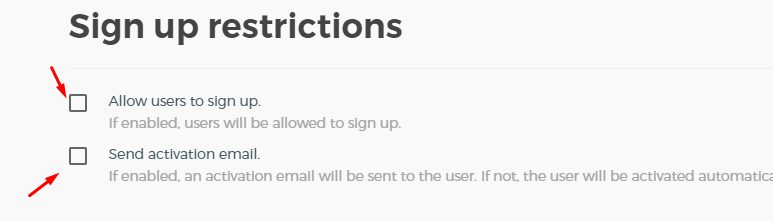


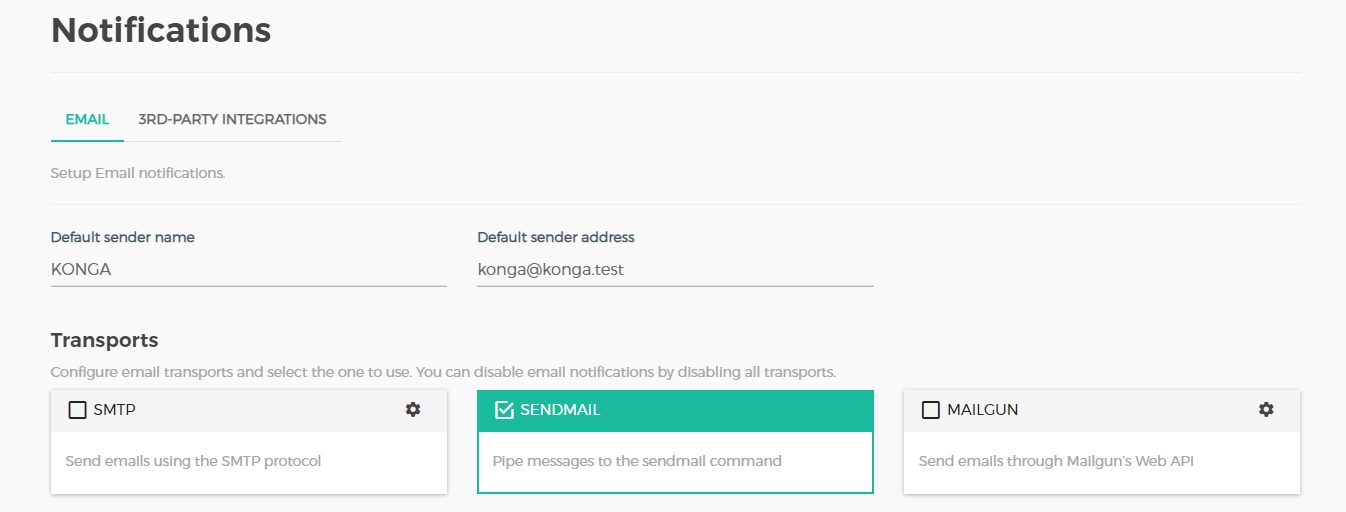


**Mas restricciones de usuarios**

La primera para habilitar que se puedan registrar usuarios

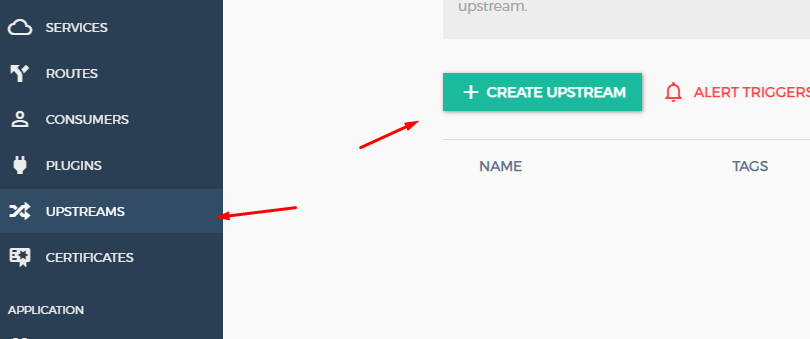
Y la segunda para que se les envíen email automáticamente

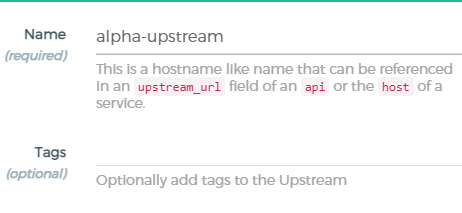




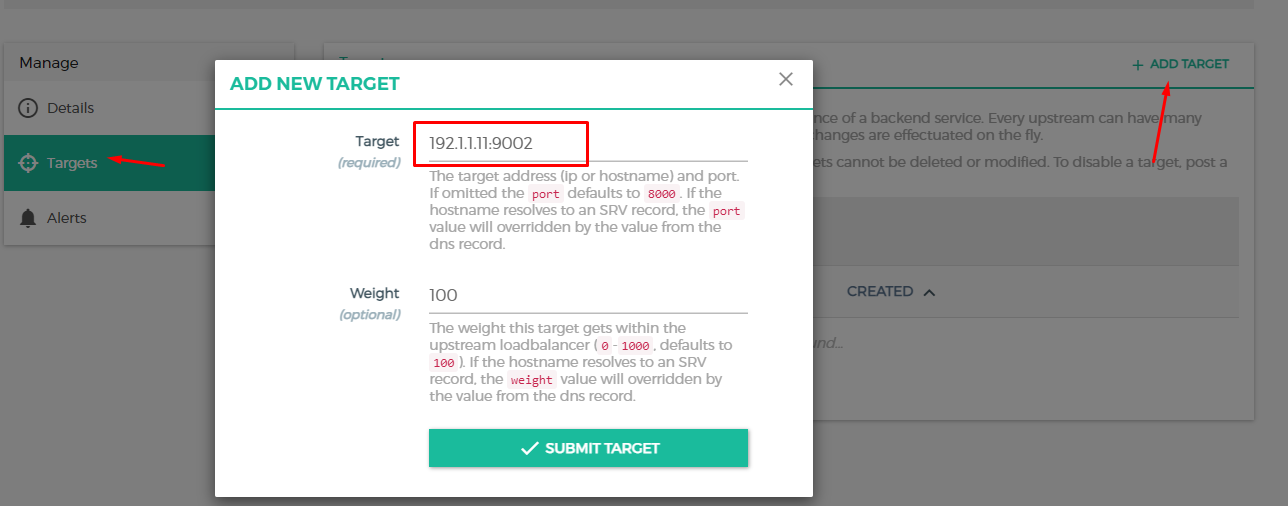
**Upstream y Target**

Con konga debemos de hacer lo siguiente

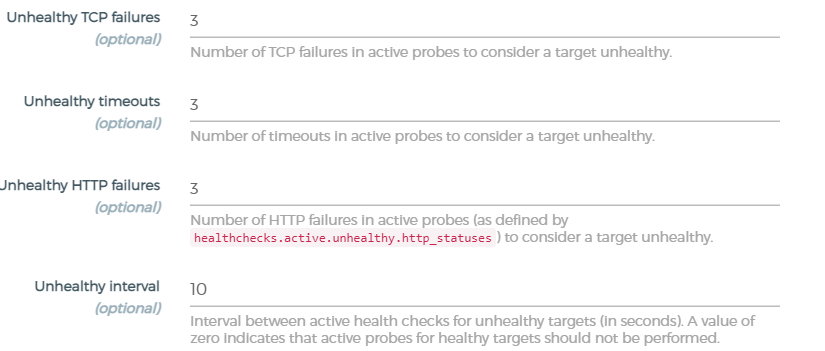


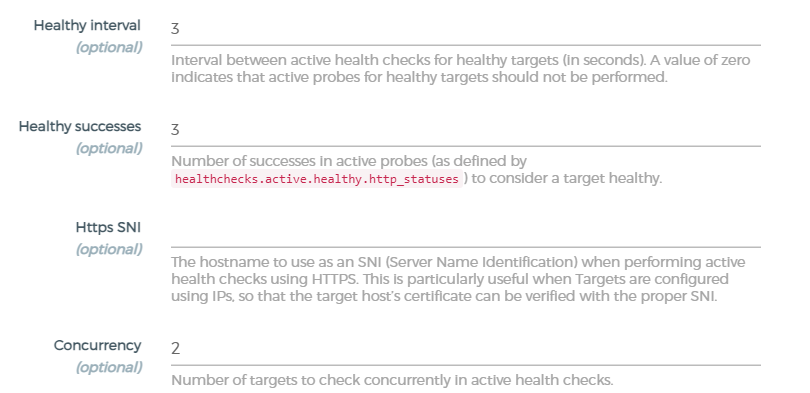


Si entramos en los detalles podemos añadir un target, la ip la obtenemos del Docker-compose



En detalles podemos configurar un health check, por ejemplo, que haga un ping cada 60 segundos

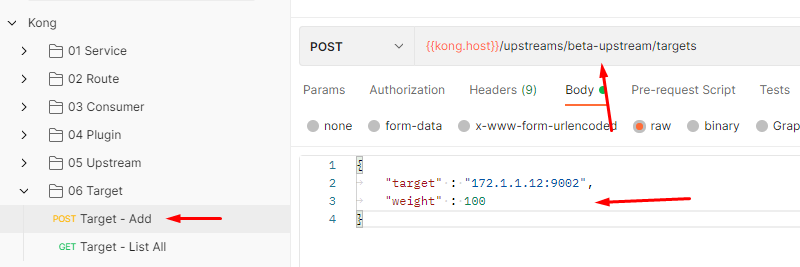




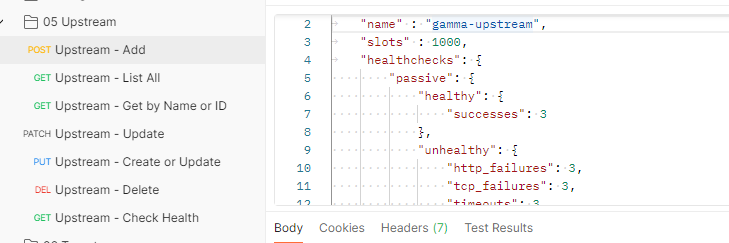
Para hacerlo con postman debemos, usaremos un health check pasivo con lo cual borramos el activo

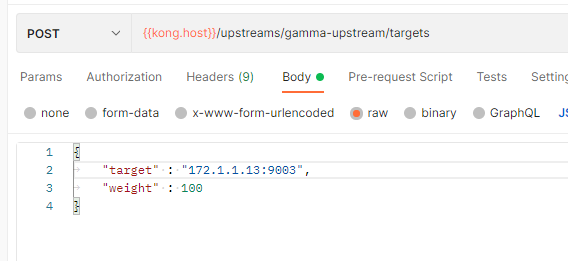


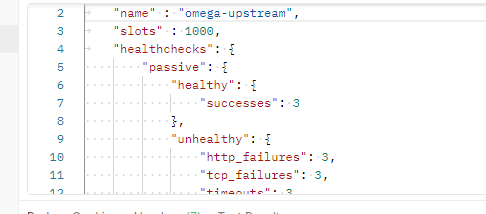
Ahora agregamos el destino o target, la ip la sacamos del Docker-compose

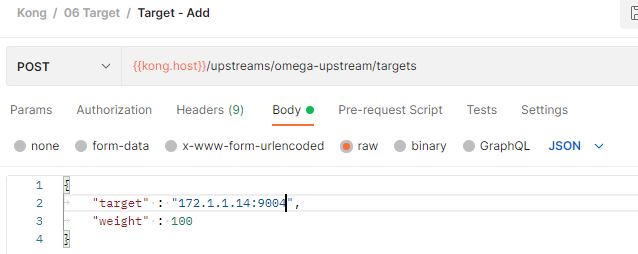


Repetimos el mismo proceso con los otros









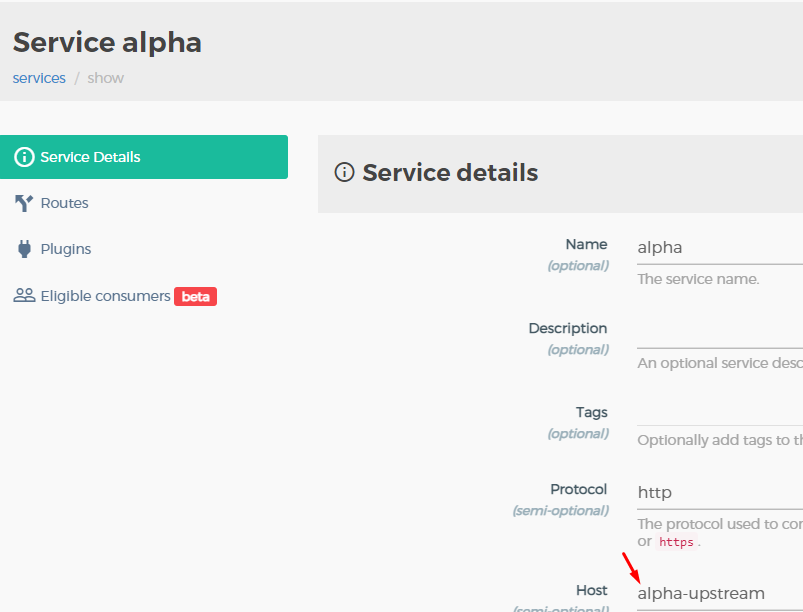
Desde konga podemos comprobar que es correcto

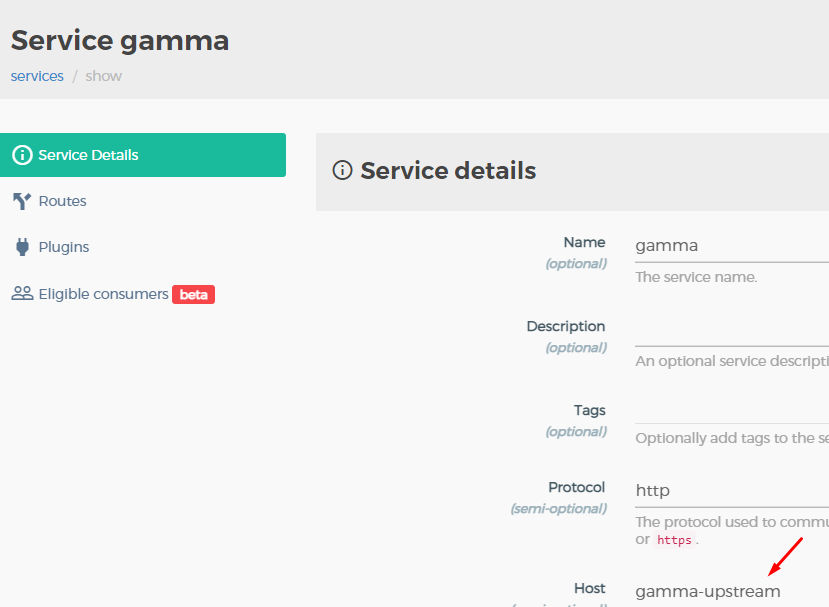


Lo siguiente que haremos será cambiar el host de todos los servicios creados



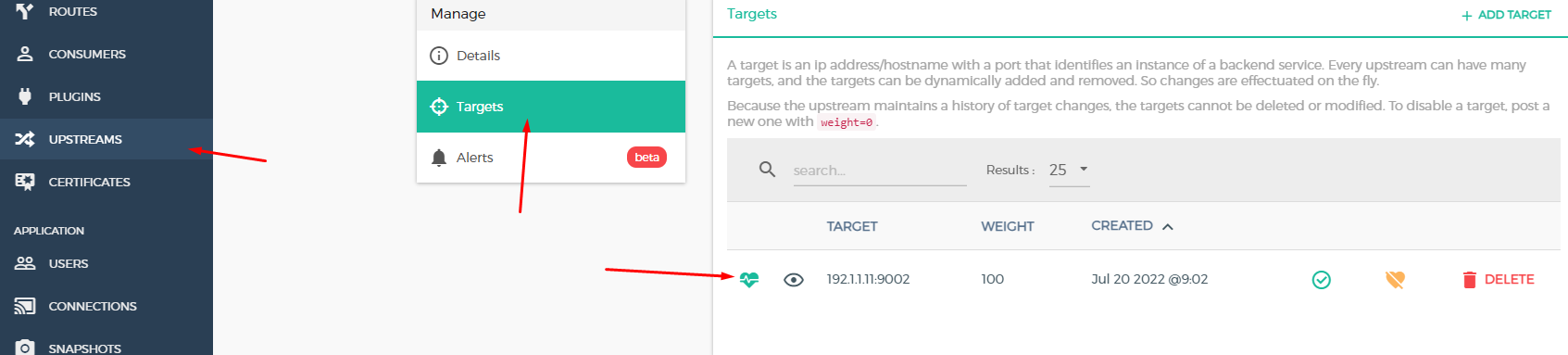






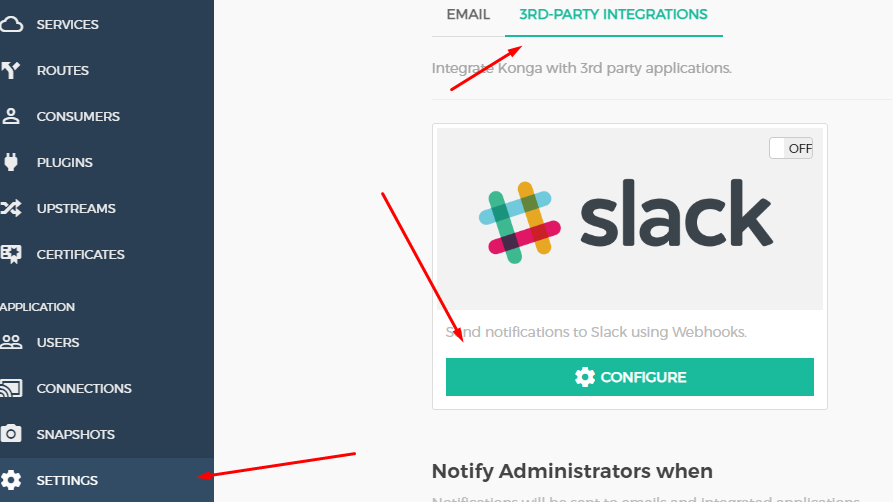
**Ver el estado de upstreams**

Para ello nos vamos a la parte de upstream y entramos en detalles de alguno

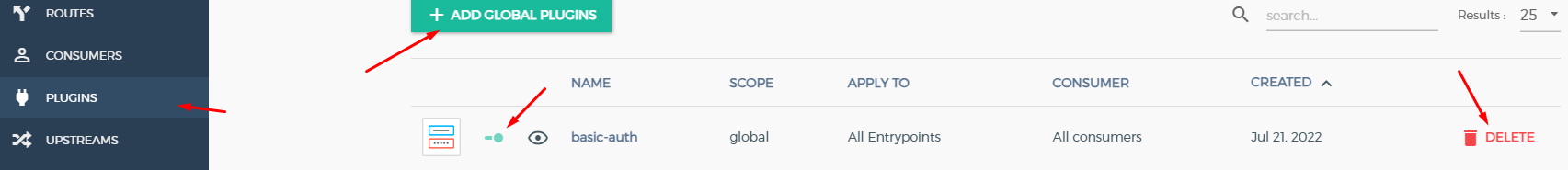


Para activar las notificaciones cuando un objetivo está en mal estado

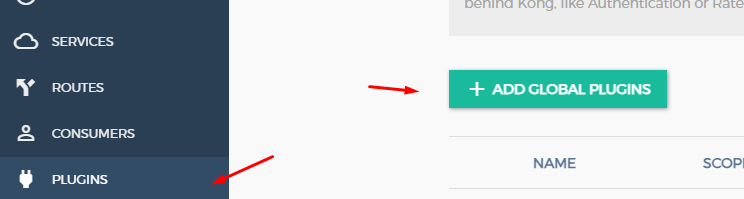
Ajustes 🡪 notificaciones 🡪tener una cuenta de slack

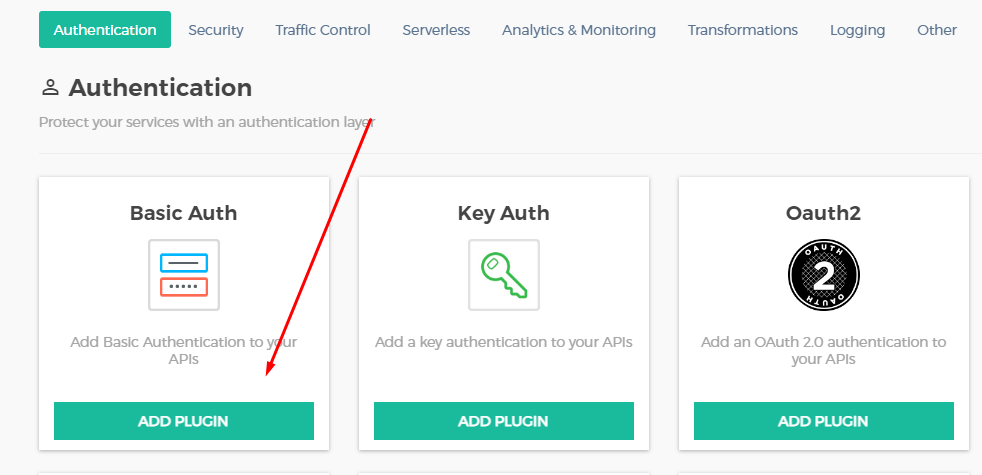


**Habilitar deshabilitar y eliminar plugins**



Plugin autenticación básica

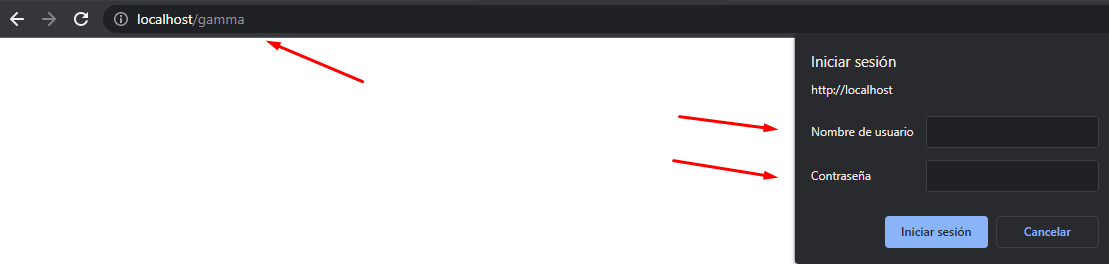




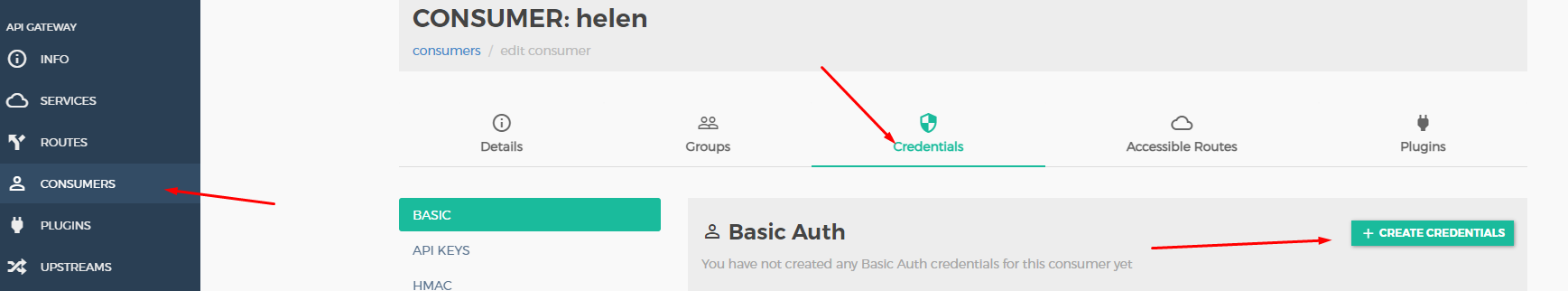
Lo dejamos en blanco para que afecte a todo

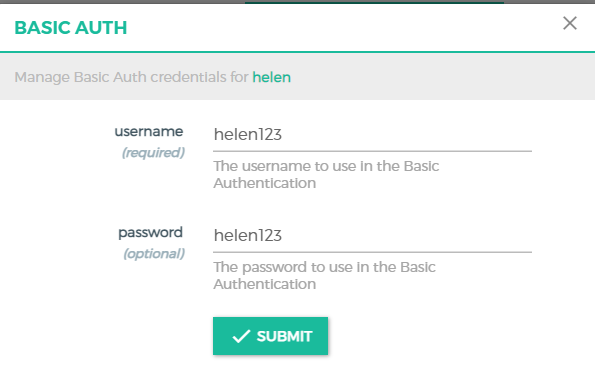


Ahora si intentamos acceder a gamma por ejemplo veremos que nos pide un usuario y una contraseña



Para poder acceder vamos a consumers y en Helen por ejemplo entramos en sus detalles y creamos sus credenciales



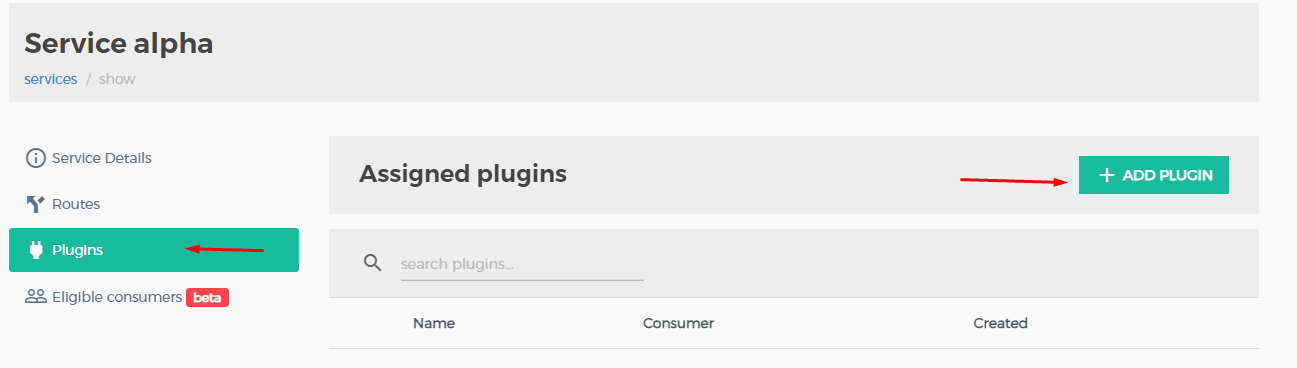


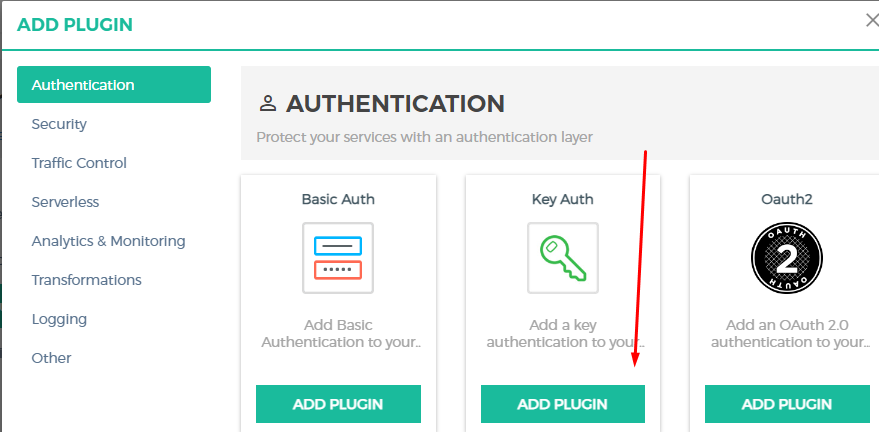
Con esto ya podríamos acceder, ahora si queremos acceder desde postman debemos



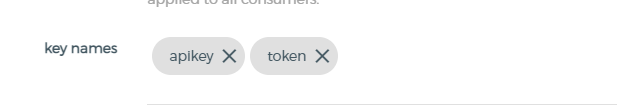
API key

Esta vez vamos a agregar un plugin a un servicio específico, para ello entramos en los detalles del servicio Alpha

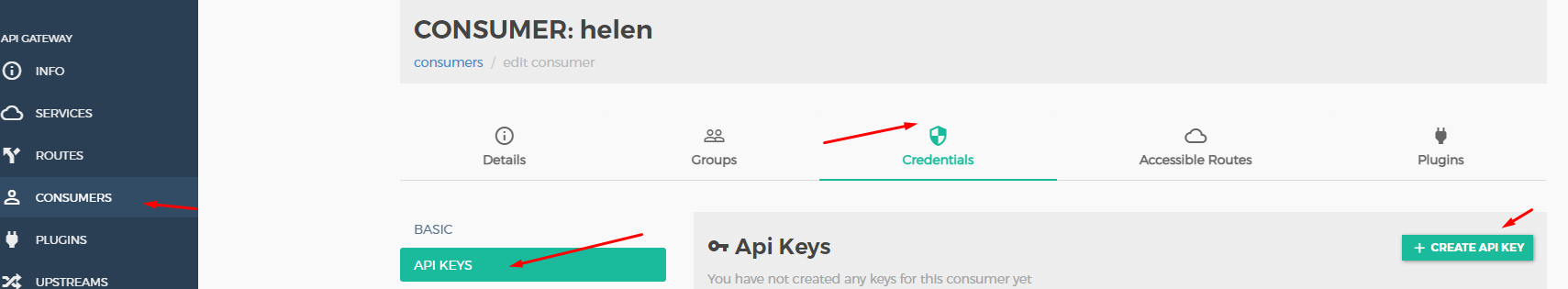




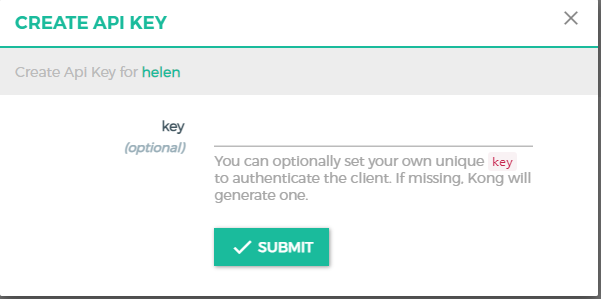
En consumidor no introducimos nada ya que lo estamos agregando dentro de alfa



Para poder acceder nos vamos a los detalles del consumidor helen

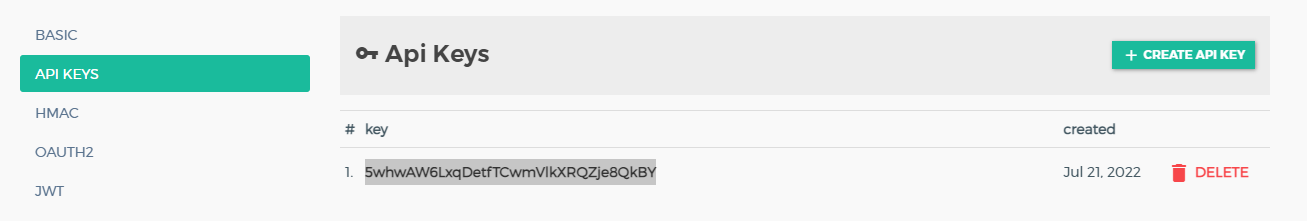


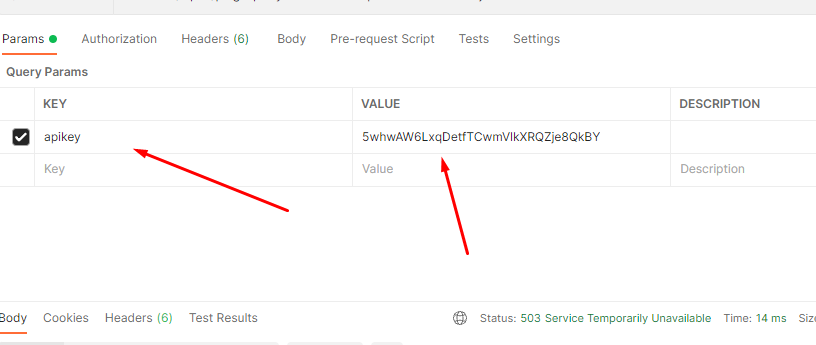
Dejamos el campo en blanco para que Kong genere una clave API



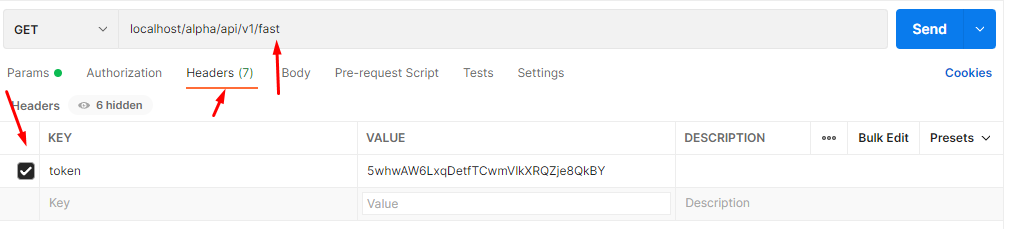
Para acceder desde postman debemos

Copiar la API key generada



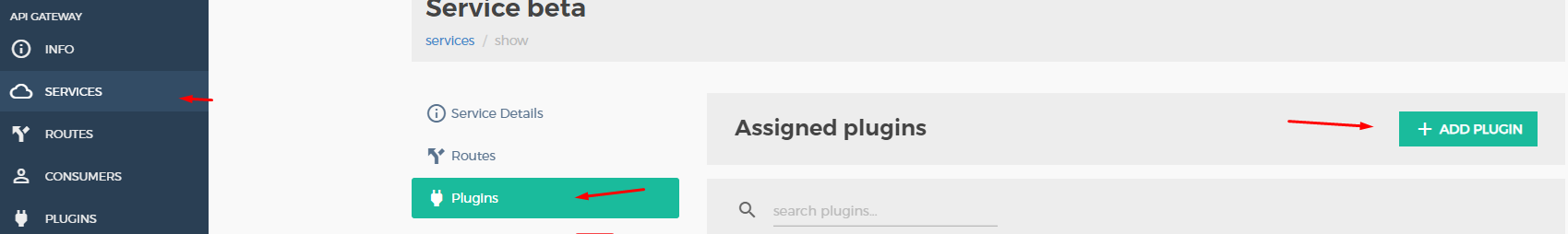


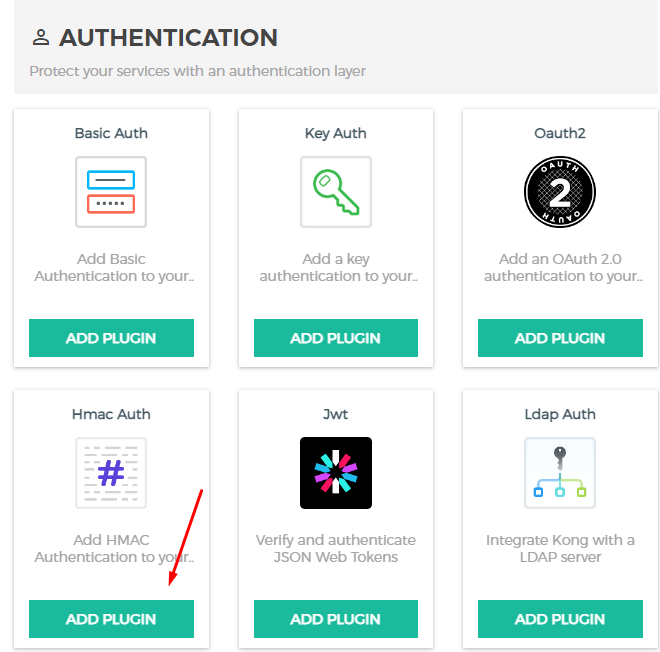
O también



Autenticación HMAC (hash)

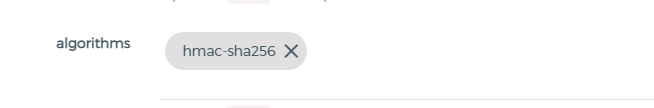
Para ellos vamos a servicios entramos en los detalles de beta por ejemplo y le agregamos un plugin





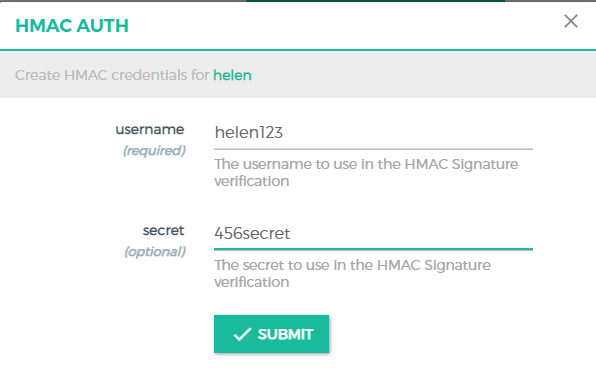
Introduciremos lo siguiente, el consumidor no hace falta indicarlo





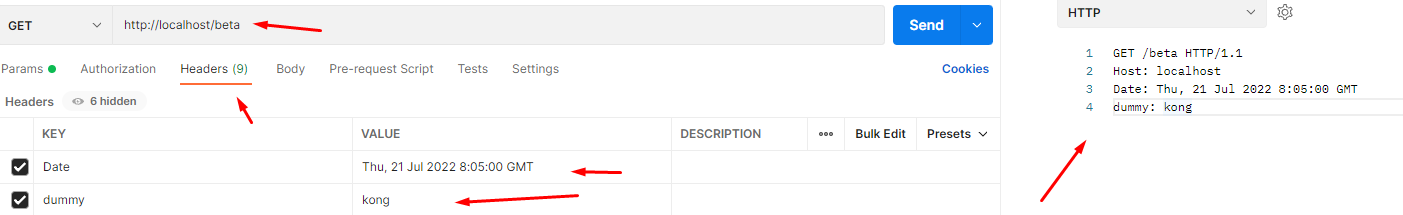
Ahora vamos a los detalles del consumidor Helen y le agregamos las credenciales





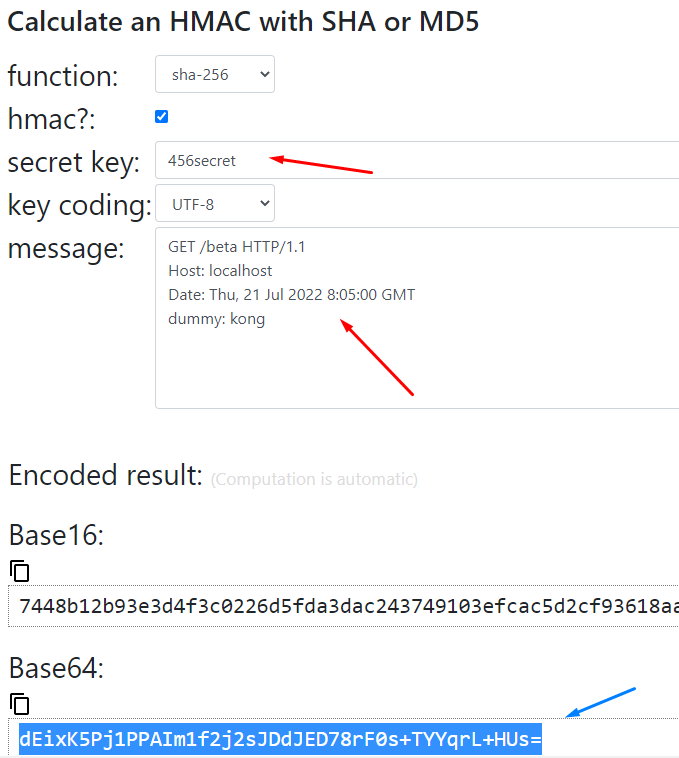
Para enviar una solicitud desde postman debemos hacer lo siguiente

Lo primero es generar el código en postman, el formato de la hora es GMT

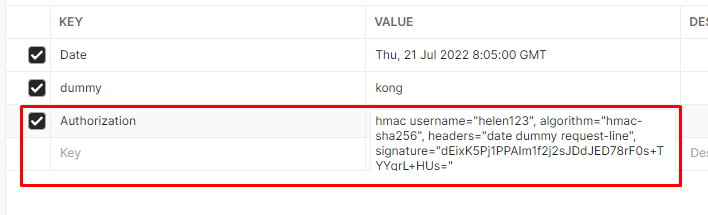


Una vez lo tenemos generar el HMAC

<https://dinochiesa.github.io/hmachash/index.html>

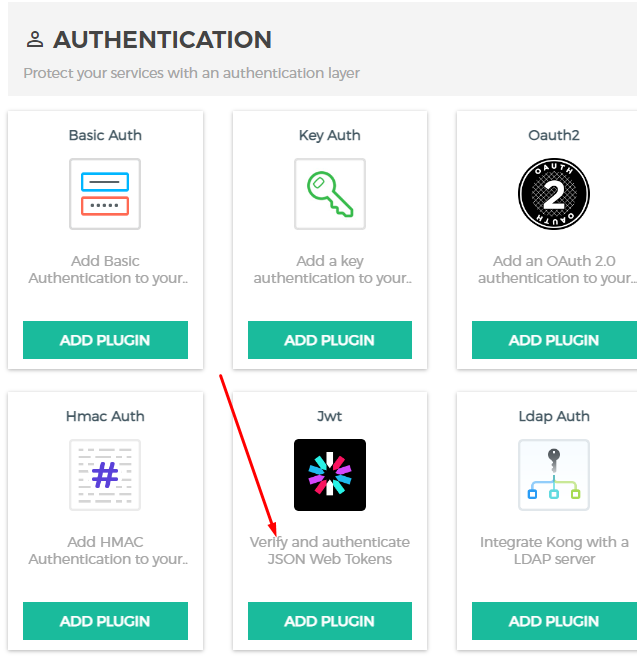


Por último, agregamos la última entrada y listo

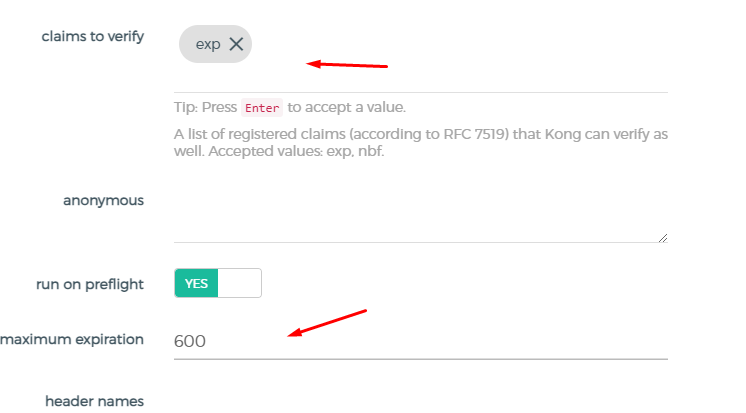


Plugin autenticación JWT

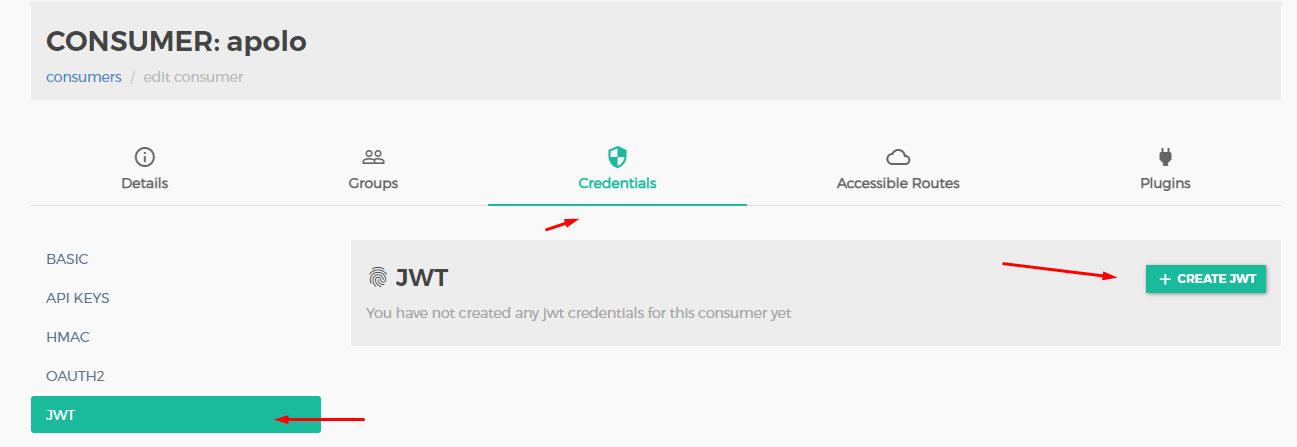
Para ello debemos ir a un servicio (En mi caso gamma) y agregar el siguiente plugin



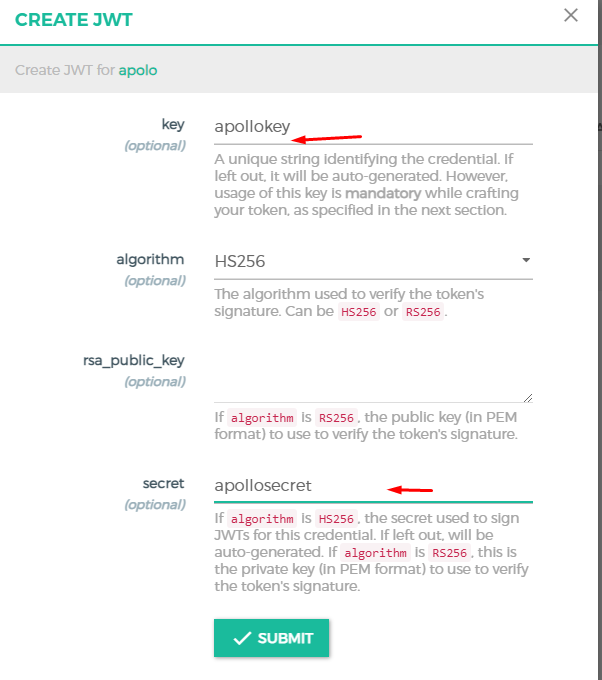
Rellenamos estos dos campos, no hace falta indicar nada más porque estamos a nivel de servicio



Tras esto nos vamos a un consumidor (Apolo en mi caso) y creamos la credencial JWT



Rellenamos los campos, si los dejamos en blanco lo hará de forma automática



Tras esto accedemos a un generador de firma JWT

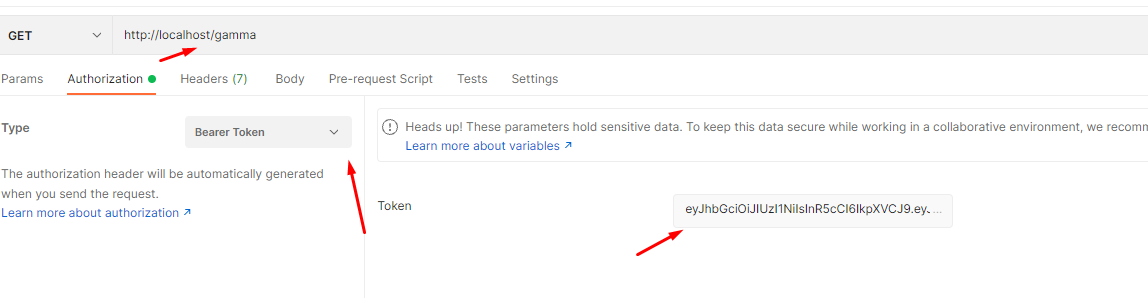
<https://jwt.io/>

exp es la hora, para obtenerla en este formato podemos usar este conversor

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

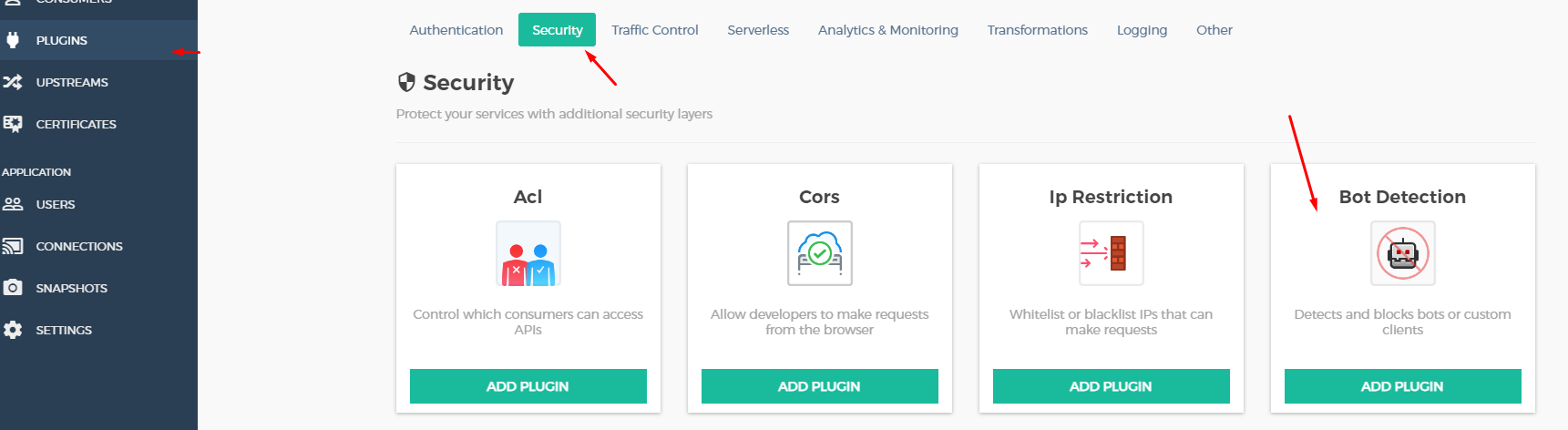


Por último, nos vamos a postman e introducimos el token generado.

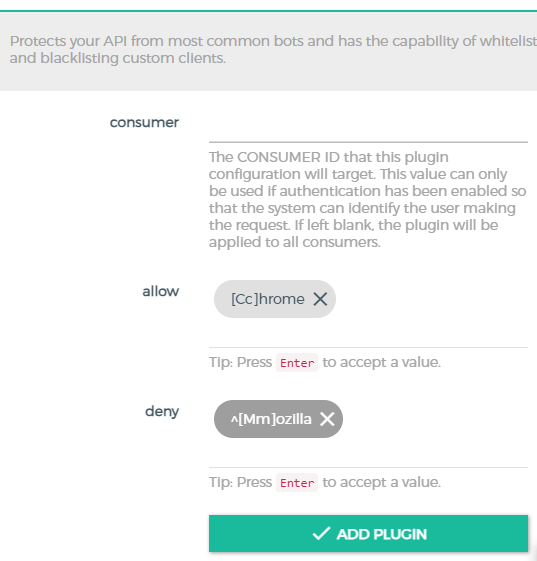


**Detección de Bots**

Para añadir la detección de bots debemos agregar un plugin nuevo

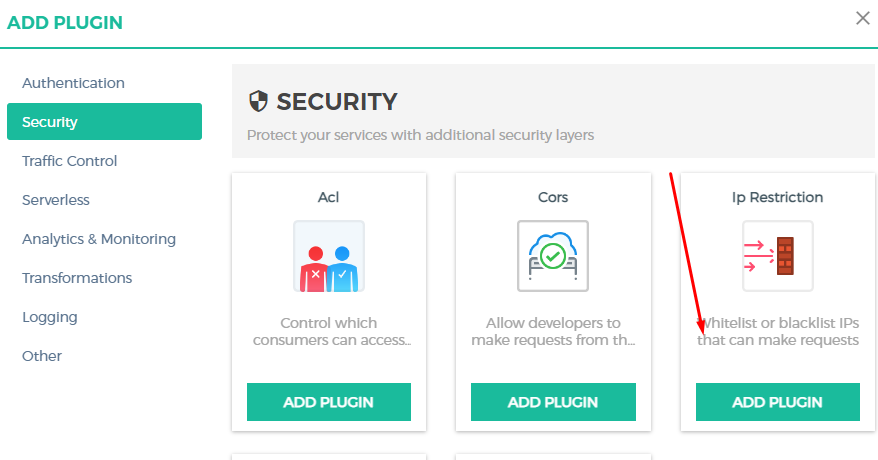


Agregamos las normas para permitir o denegar que queramos

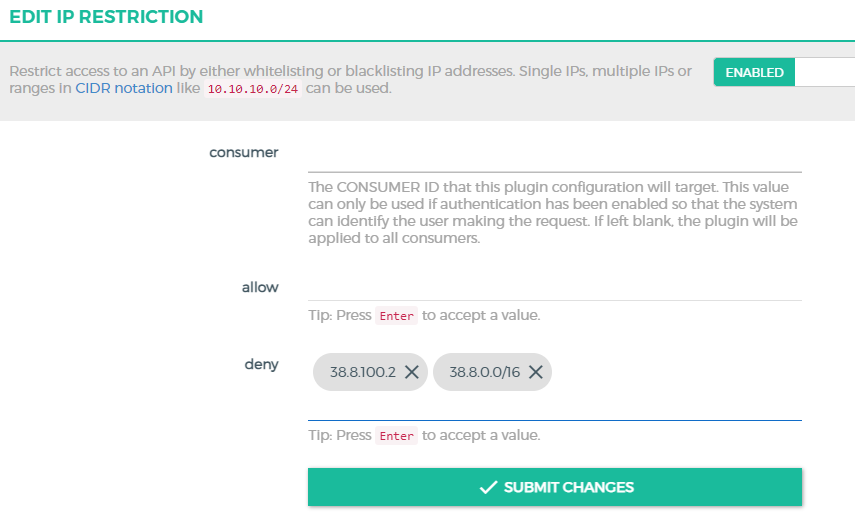


**Restricción de IP**

Con esto podremos dar acceso o denegarlo a una dirección IP, para ello debemos agregar un nuevo plugin al servicio que queramos en mi caso gamma

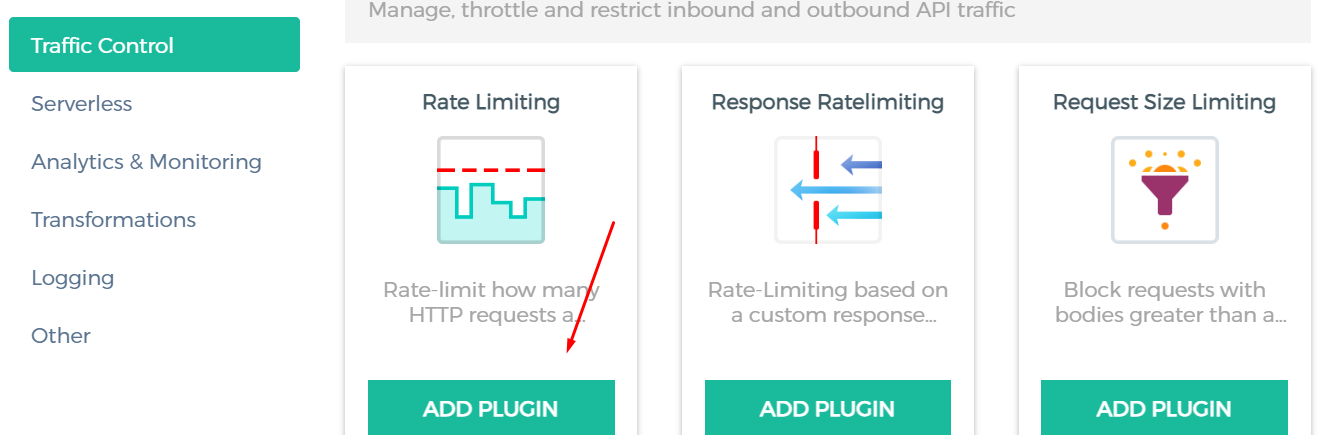


Y ya nos quedaría indicar las Ip o el rango que queremos denegar



**Limitación de velocidad (Rate limiting)**

Para ellos entramos dentro de un servicio (omega en mi caso) y agregamos el siguiente plugin

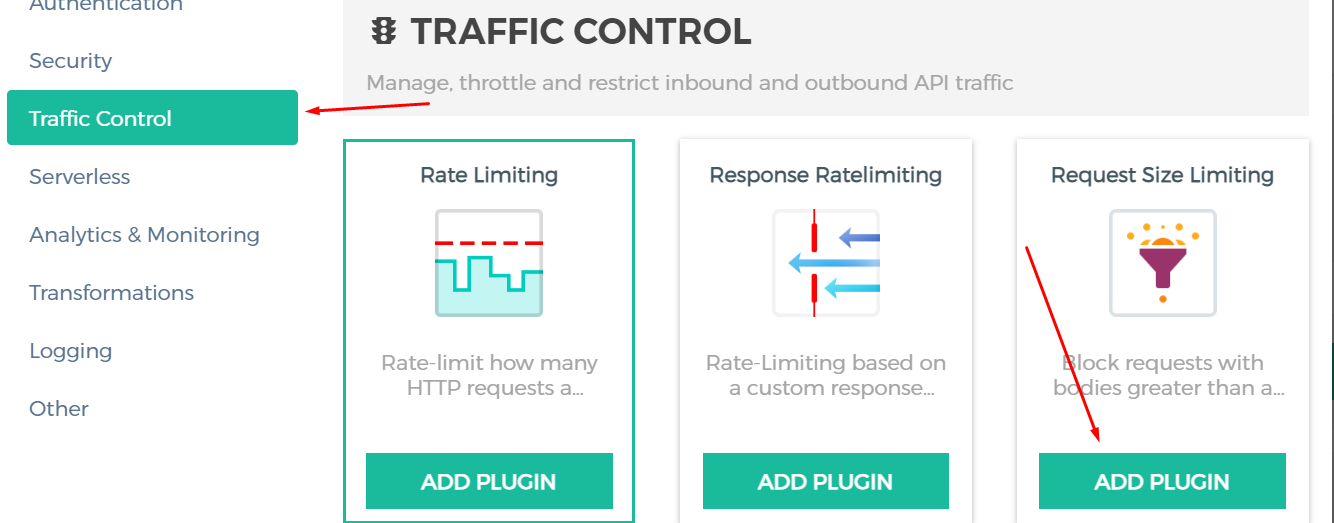


Estableceremos que solo se puedan hacer 5 solicitudes por minuto

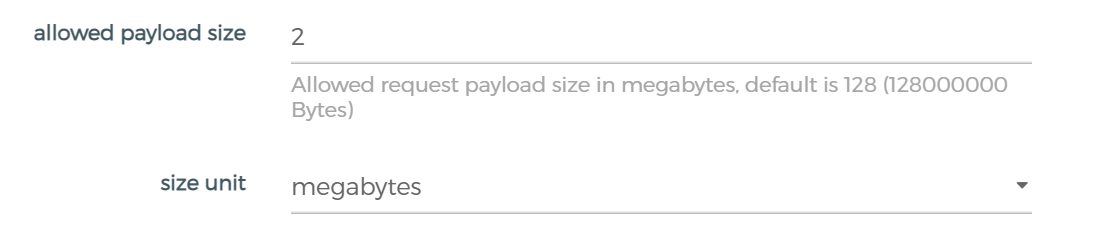


**Limitación de tamaño de la solicitud (Request size limiting)**

Para llevar a cabo esto añadiremos un plugin de forma global

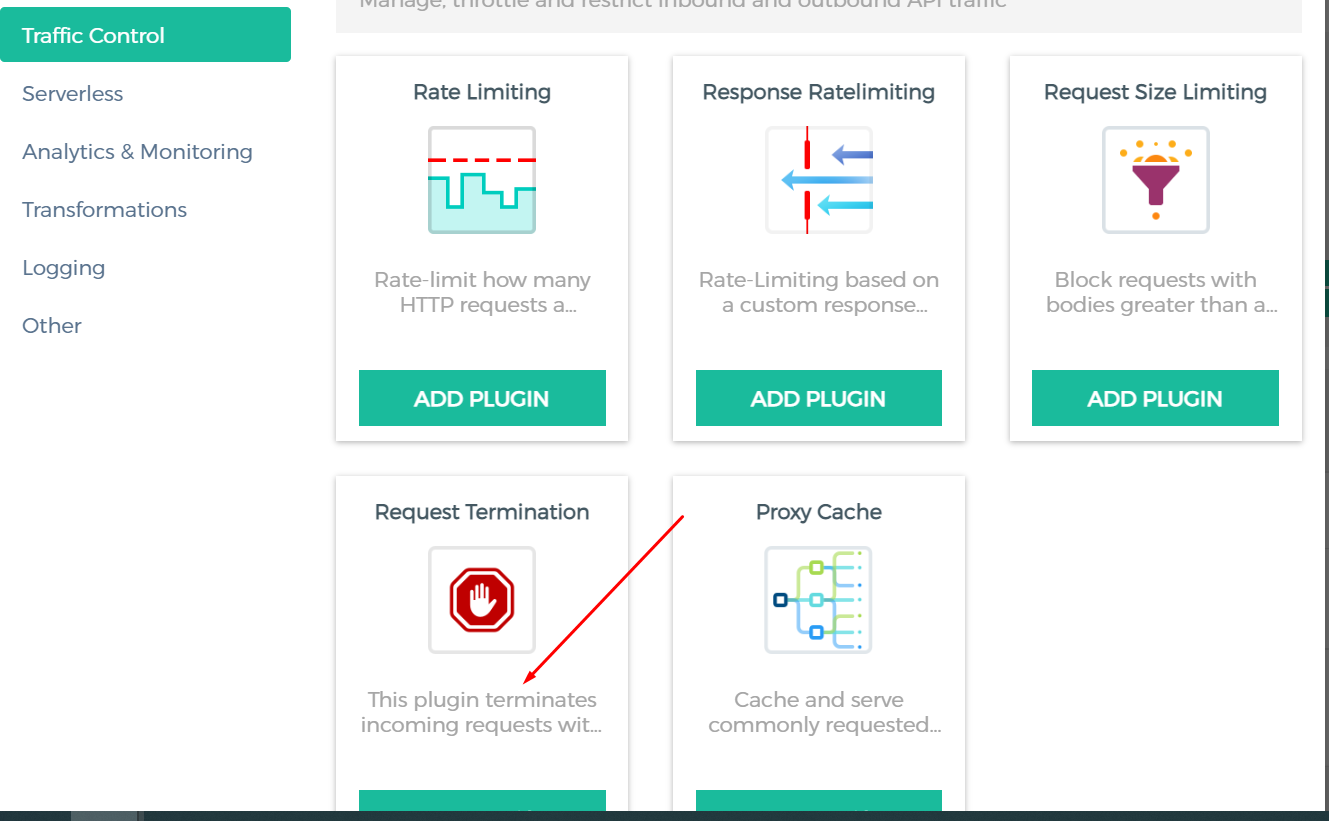


Establecemos el limite del tamaño de la solicitud, en este caso 2 Mb

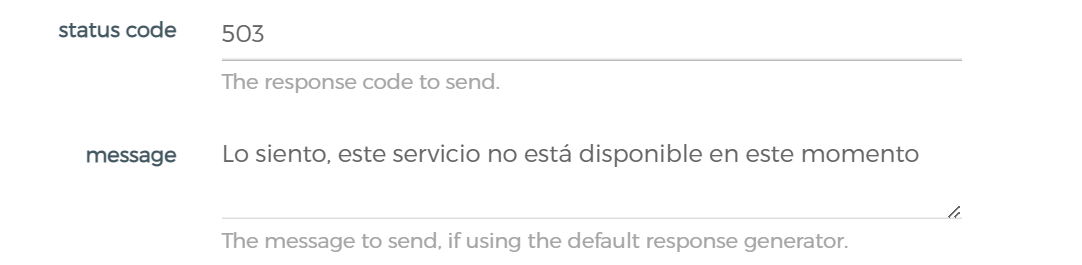


**Solicitud de recisión (Request termination)**

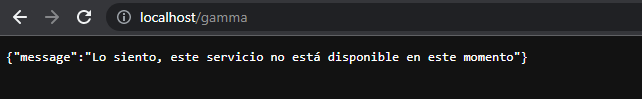
Para ello debemos de dirigirnos a un servicio (En mi caso gamma) y agregar el siguiente plugin



Escribimos el mensaje que queremos que aparezca

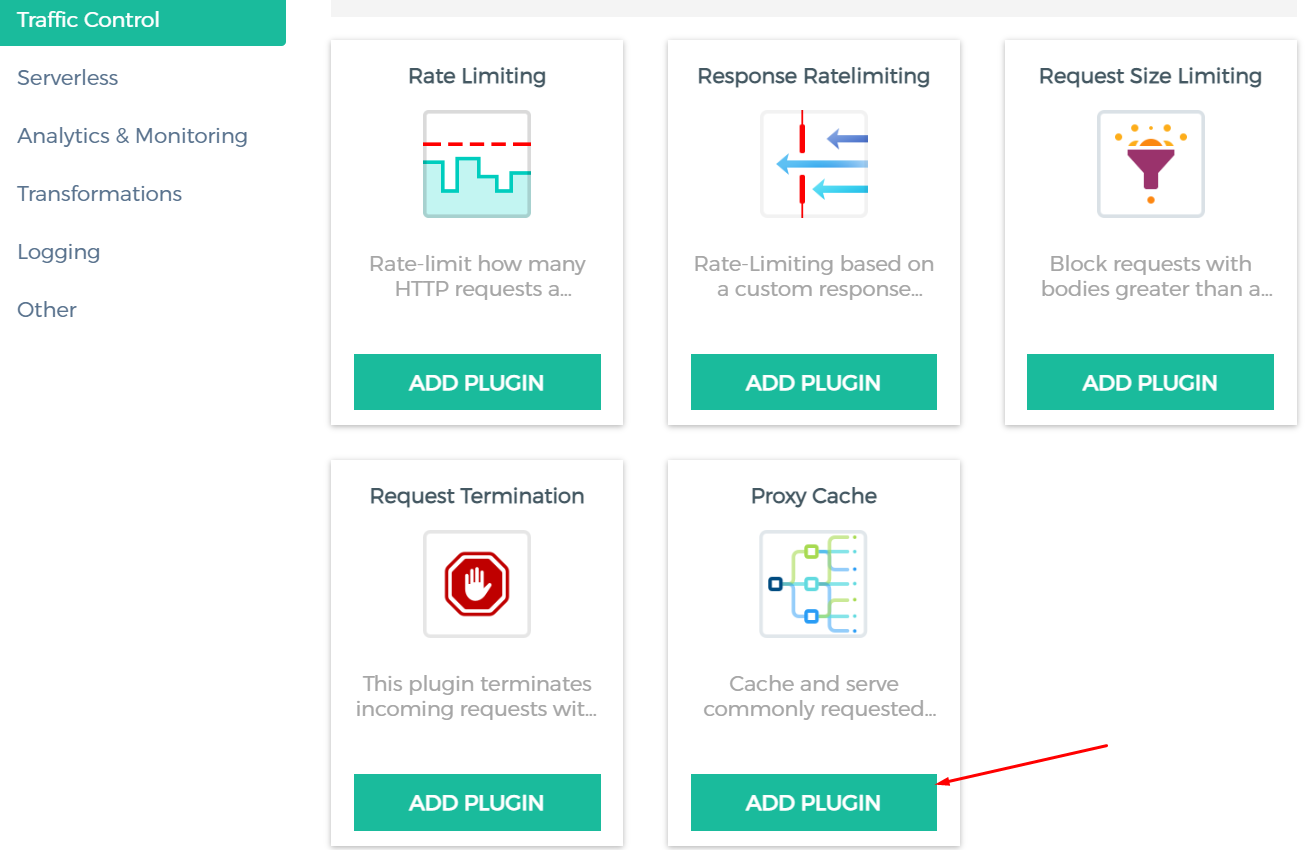


Si accedemos a gamma veremos el resultado



**Proxy Caché**

Para ello agregaremos un plugin a uno de nuestros servicios en mi caso beta



Completamos los siguientes parámetros, que quieren decir que guardará la información en memoria caché 300 segundos

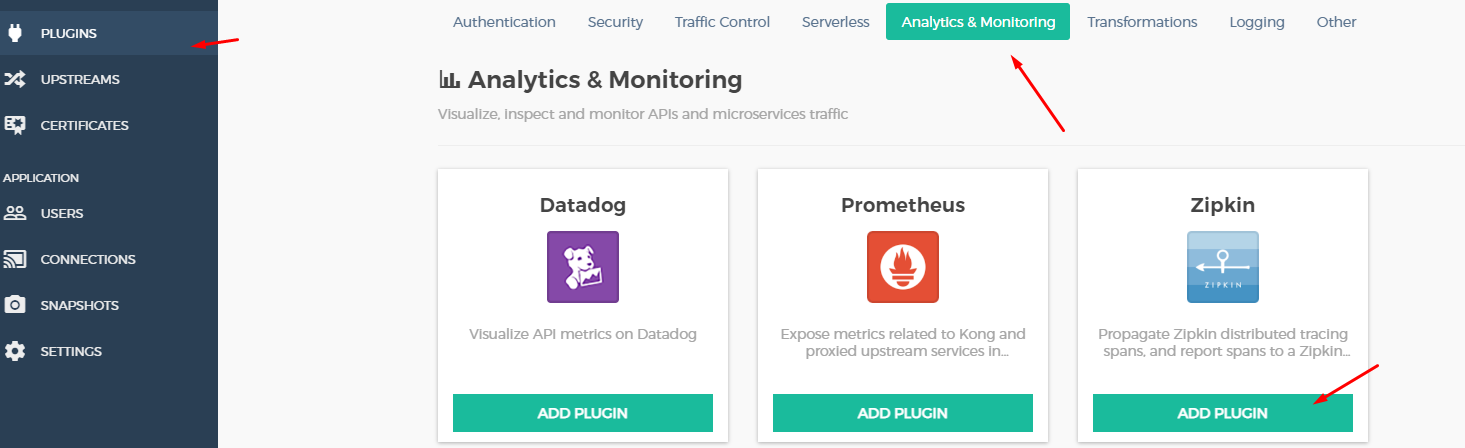


**Zipkin**

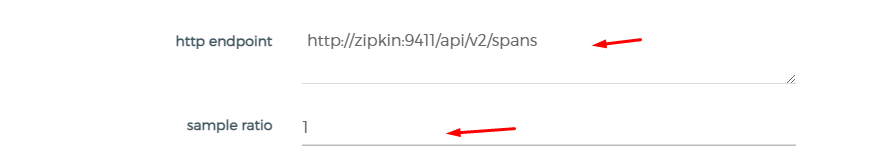
Lo primero será tener Zipkin



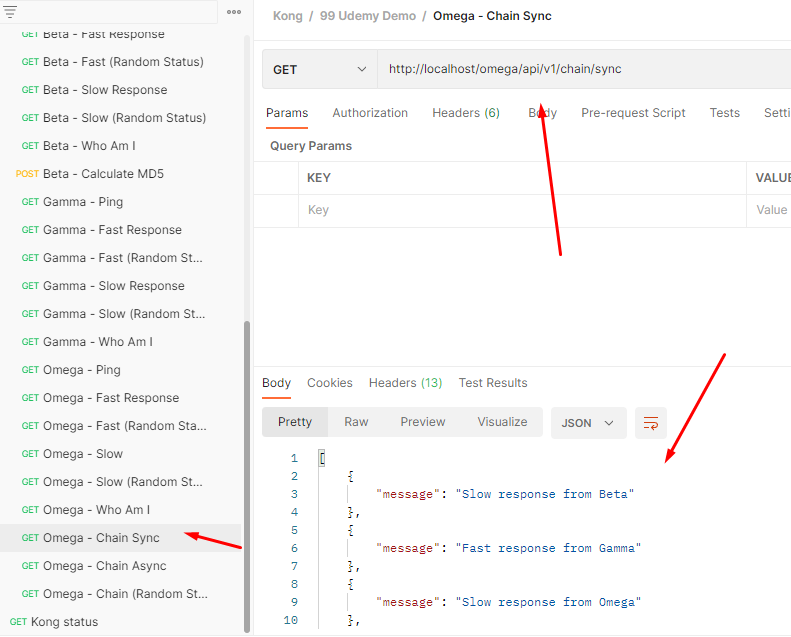
A continuación, agregamos el plugin



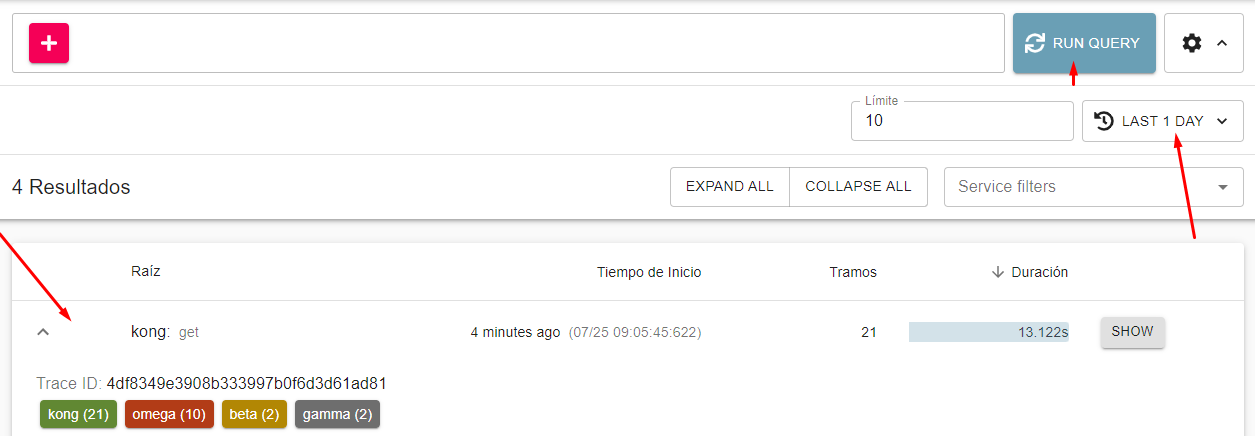
Rellenamos los datos



Desde postman vamos a crear un punto ficticio



Ahora accedemos a zipkin a través de <http://localhost:9411> y ver la traza



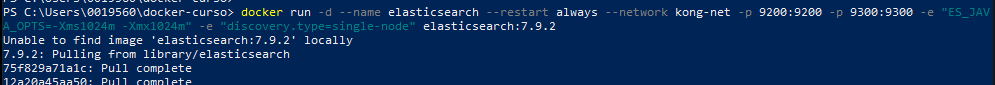
**Elastic Stack (Elasticsearch, Logstash y** **Kibana)**

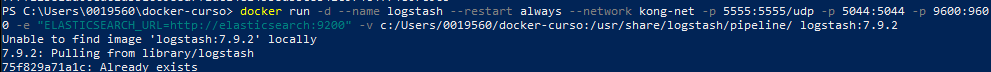
**Elasticsearch:** Almacenar datos

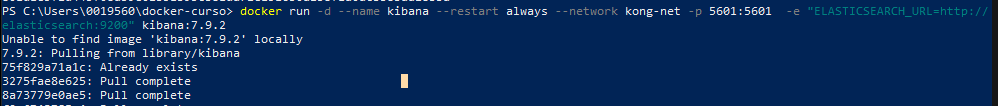
**Logstash:** Recopilar y enviar datos

**Kibana:** Visualizar datos en formato legible

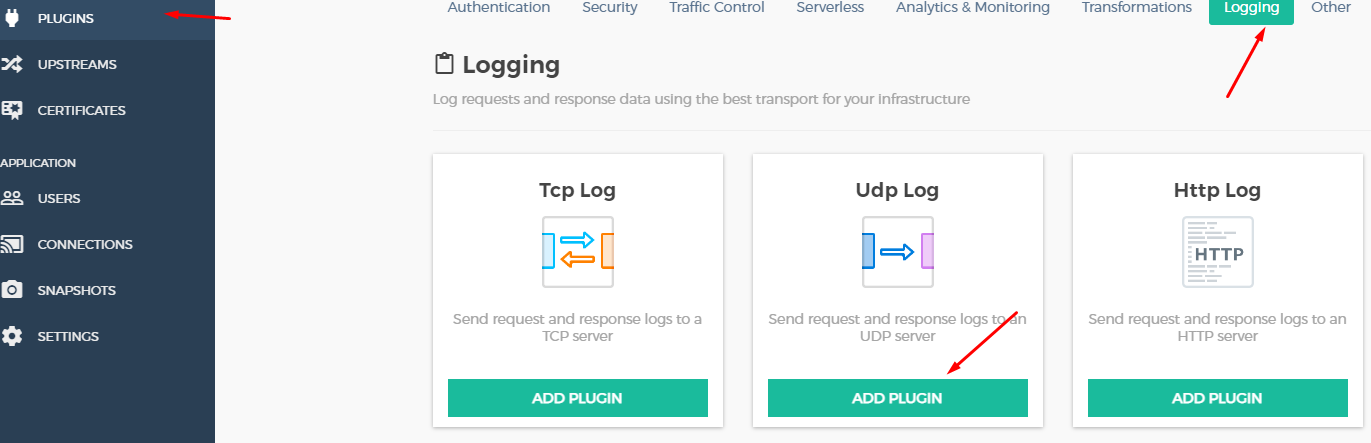
Lo primero será instalar las 3 herramientas



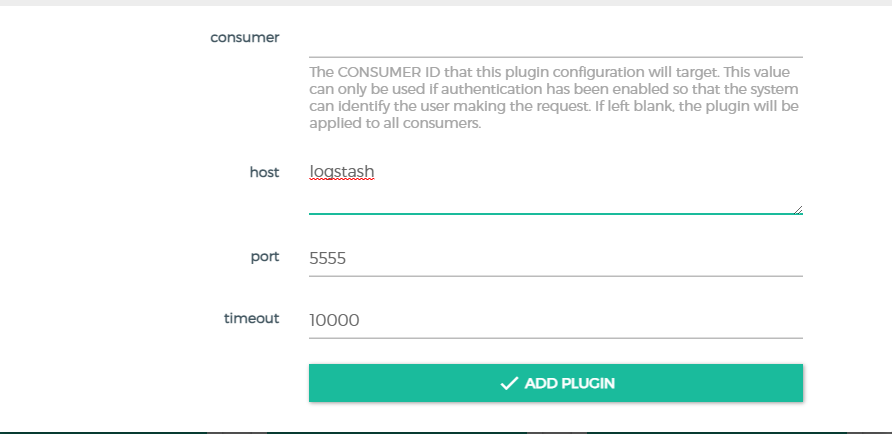




Necesitaremos agregar e siguiente plugin global en konga



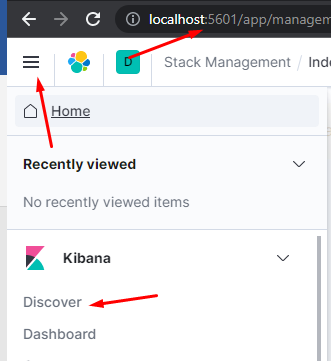
Con los siguientes puertos, el puerto está definido en el archivo pipelines.conf



Tras esto acedemos a postman para enviar un dato



Tras esto accedemos a ElasticSearch



Y creamos como una tabla de BBDD relacional con lo cual nos pedirá primero que introduzcamos en que índice estamos trabajando (logstash\*), tras eso nos pedirá crear el campo que contiene el filtro del tiempo seleccionamos *@timestamp* y creará el índice



**Clean up ElascticSearch**

*docker container stop elasticsearch logstash kibana*

*docker container rm elasticsearch logstash kibana*

*docker image ls*

*docker image rm [image-id-elasticsearch] [image-id-logstash] [image-id-kibana]*