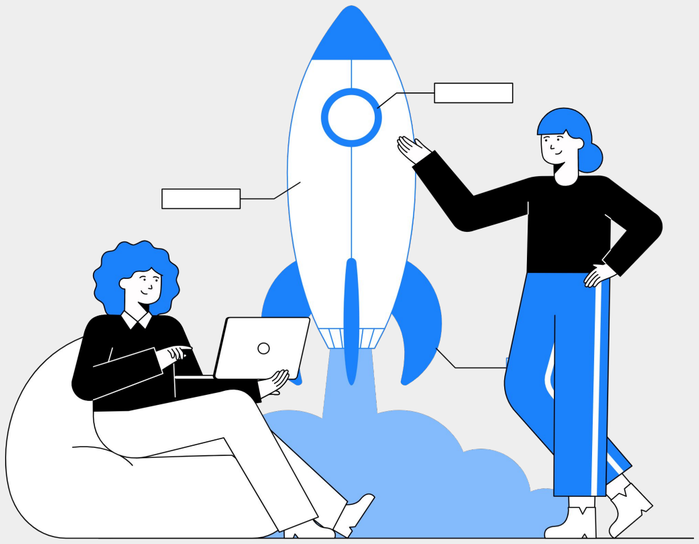
¡Javascript!

guayerd beta hub

guayerd

Agenda del día



01

APIs

Definición.

Instalar Thunder Client.

API REST

Definición.

Recursos.

Verbos.

Body.

API FETCH

Definición.

GET

POST

MOCKAPI.

02

Ejercitación

Hacer un fetch de una orden de compra.

.

guayerd

daily

¿Cómo venimos?

¿Algo nos bloquea?

¿Cómo seguimos?

guayerd

API

Interfaz de Programación de Aplicaciones

guayerd

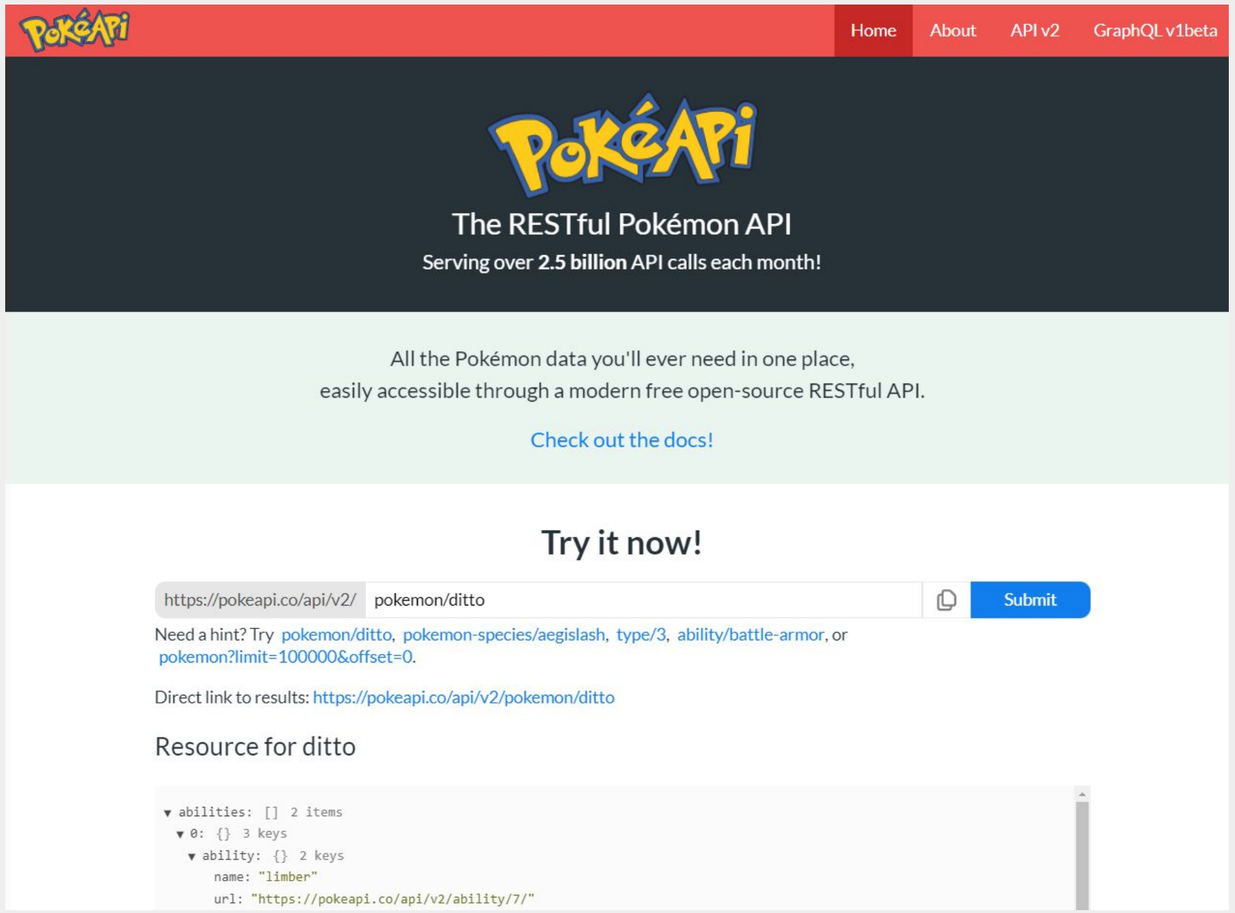
SPRINT 3

API

Una API (Interfaz de Programación de Aplicaciones) es una implementación de reglas y protocolos que permite la comunicación entre programas o consumir servicios.

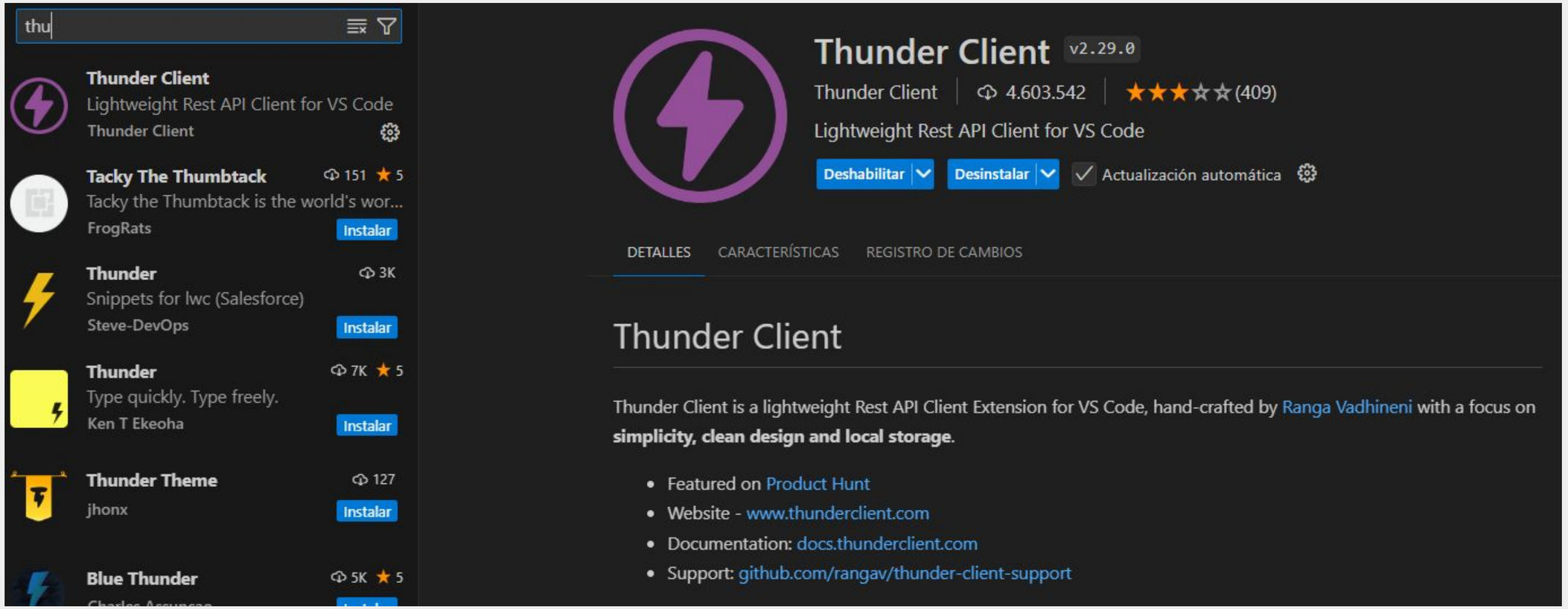
guayerd

Ejemplo



guayerd

Instalar Thunder Client en VSCode



guayerd

Servidores

Clientes

guayerd

API REST

Representational State Transfer

API REST

Es una arquitectura para diseñar servicios web que permiten comunicar datos o información entre clientes y servidores.

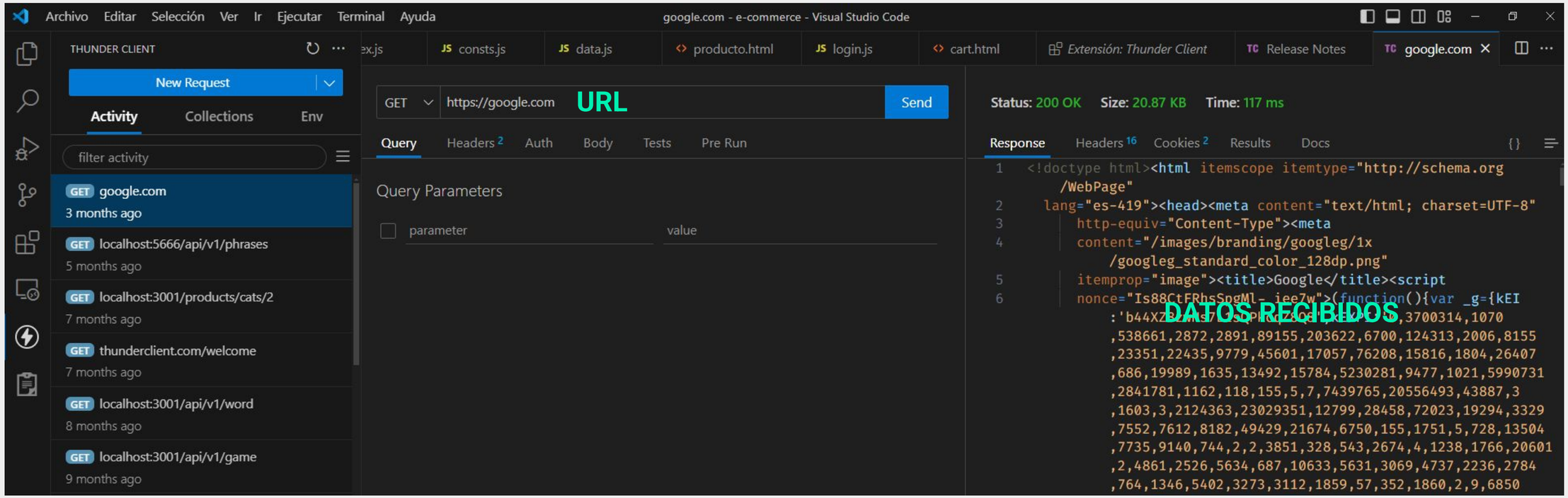
Está compuesto por:

- Recursos
- Métodos o verbos
- Parámetros
- Headers
- Body

guayerd

Recursos

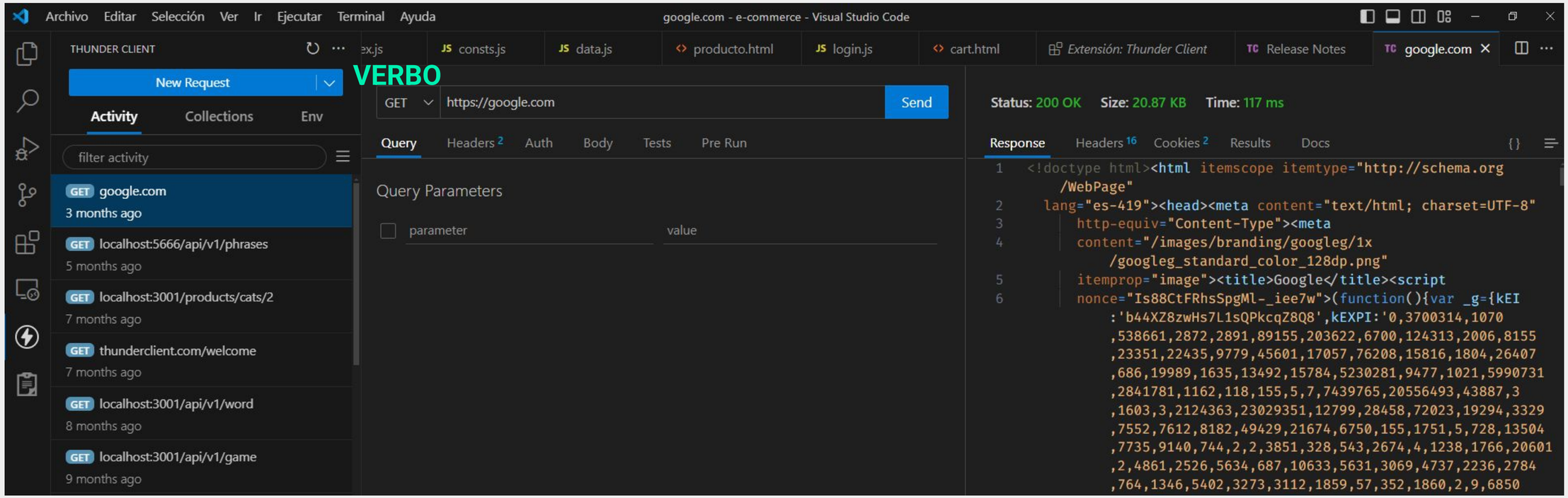
Nos comunicamos con una API a través de una URL y obtenemos los recursos representados como datos utilizables.



guayerd

Métodos o verbos: GET

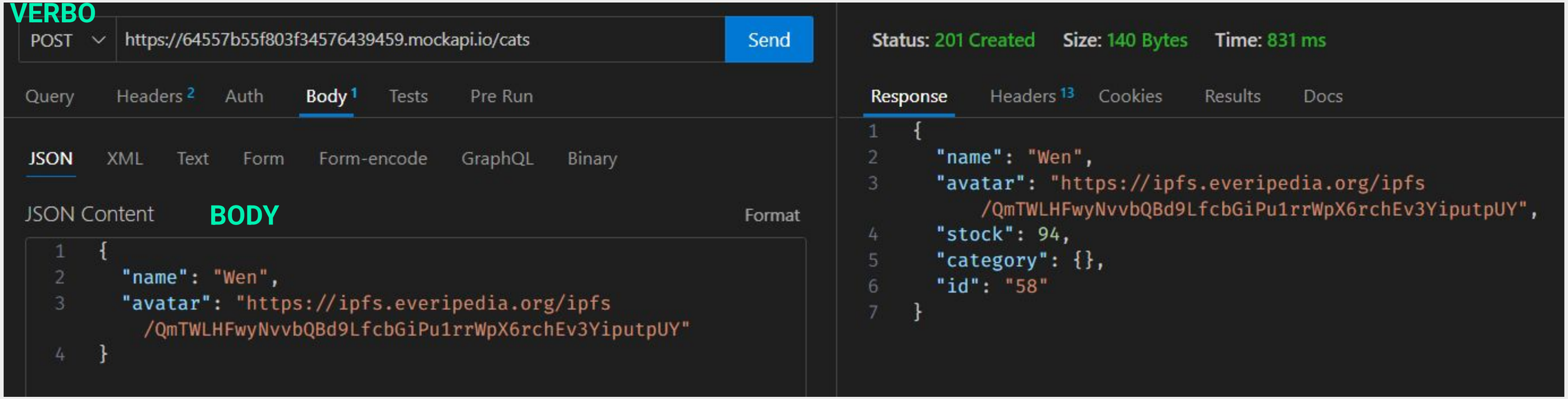
Permiten comunicar distintas intenciones al servidor. El más usado es GET, que trae directamente un recurso. Lo usamos comúnmente en la barra de direcciones del navegador.



guayerd

Métodos o verbos: POST

Permite enviar información al servidor para crear un recurso en el servidor. Enviamos la información en formato JSON y usamos el campo body.



guayerd

API FETCH

Navegador

API FETCH

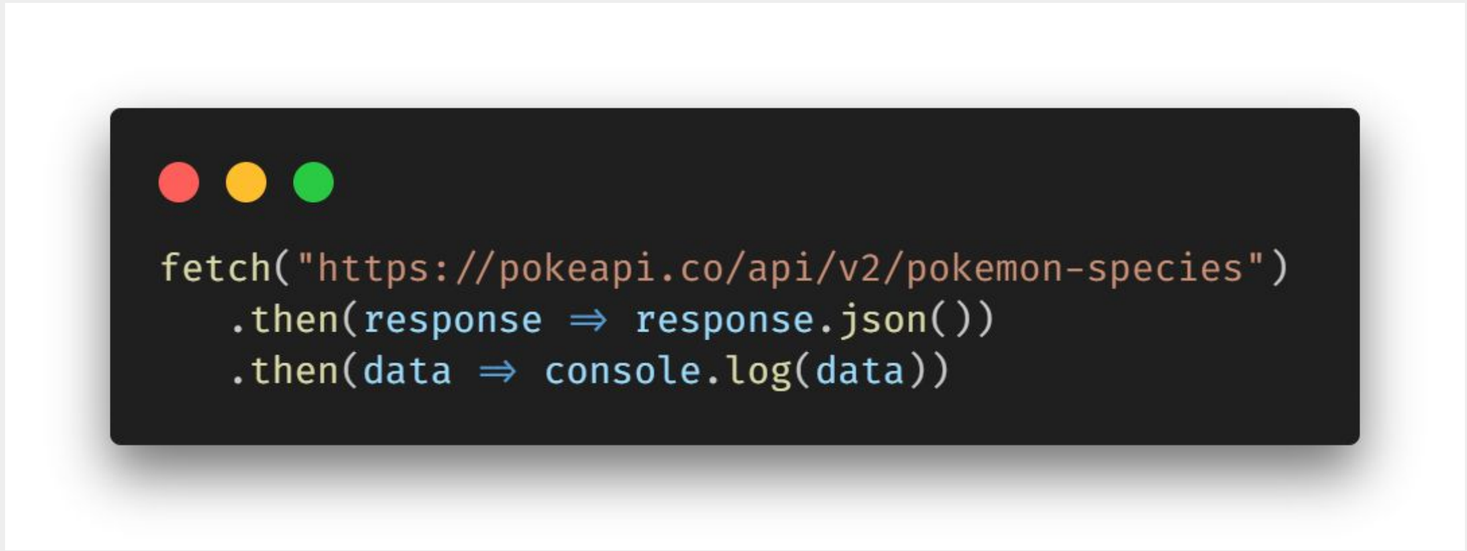
Es una API de navegador que permite comunicarnos con APIS (servidores) desde el navegador.

De este modo integramos nuestras aplicaciones con datos de la **nube**.

Fetch funciona asincrónicamente, es una **promesa**, por lo cual, encadena los métodos: then, catch, finally.

GET

El verbo get del método fetch, es el verbo predeterminado, por eso no se declara. Fetch recibe como primer argumento la URL. En el primer then, transformamos lo que devuelve el servidor en un formato que Javascript puede entender. El segundo then obtiene la data.

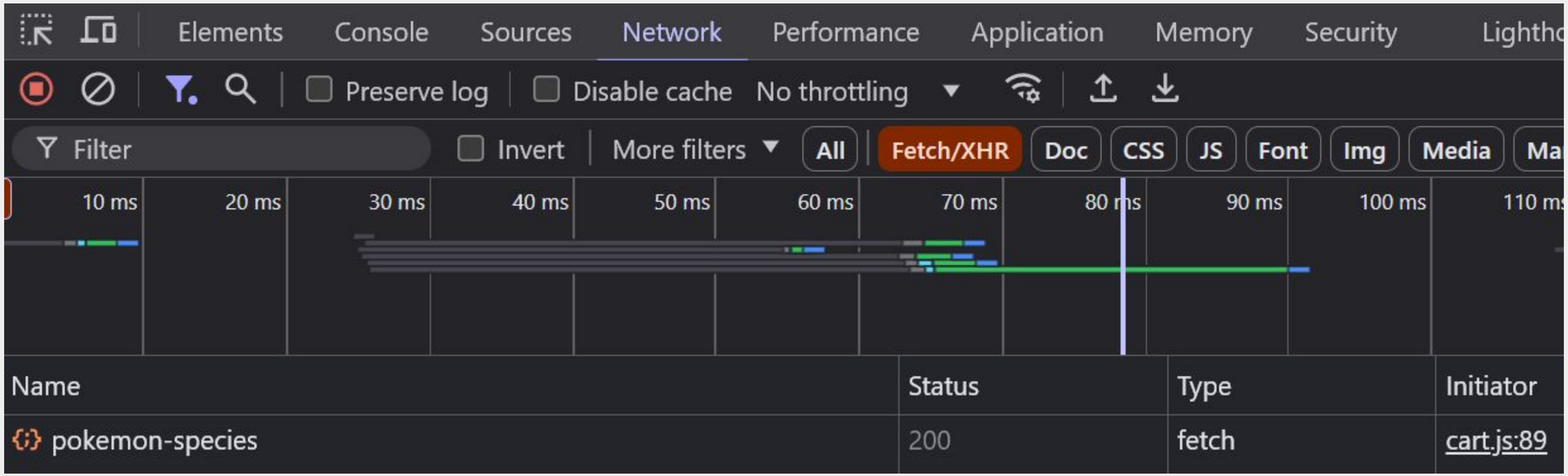


guayerd

GET

En Dev Tools, en la pestaña Red (Network) podemos ver la petición.

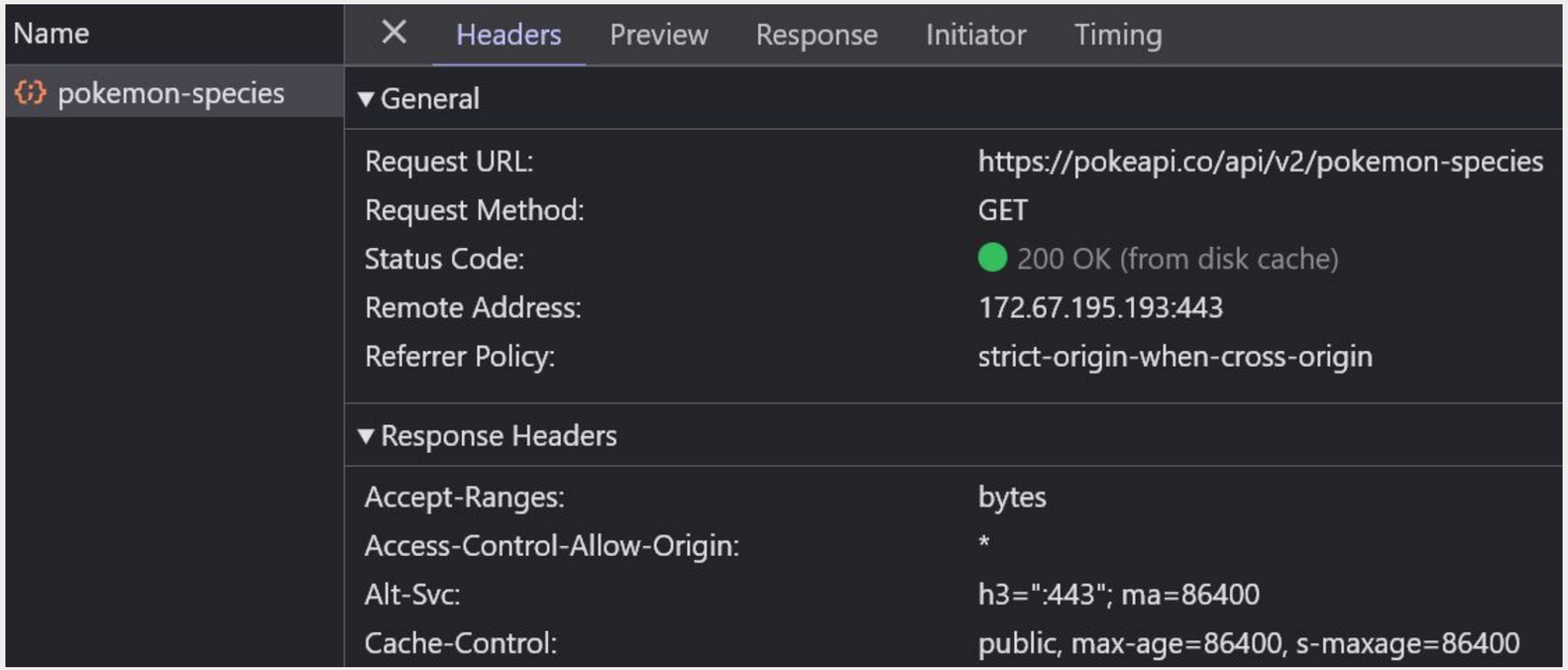
El status 200 indica que la comunicación funcionó.



guayerd

GET

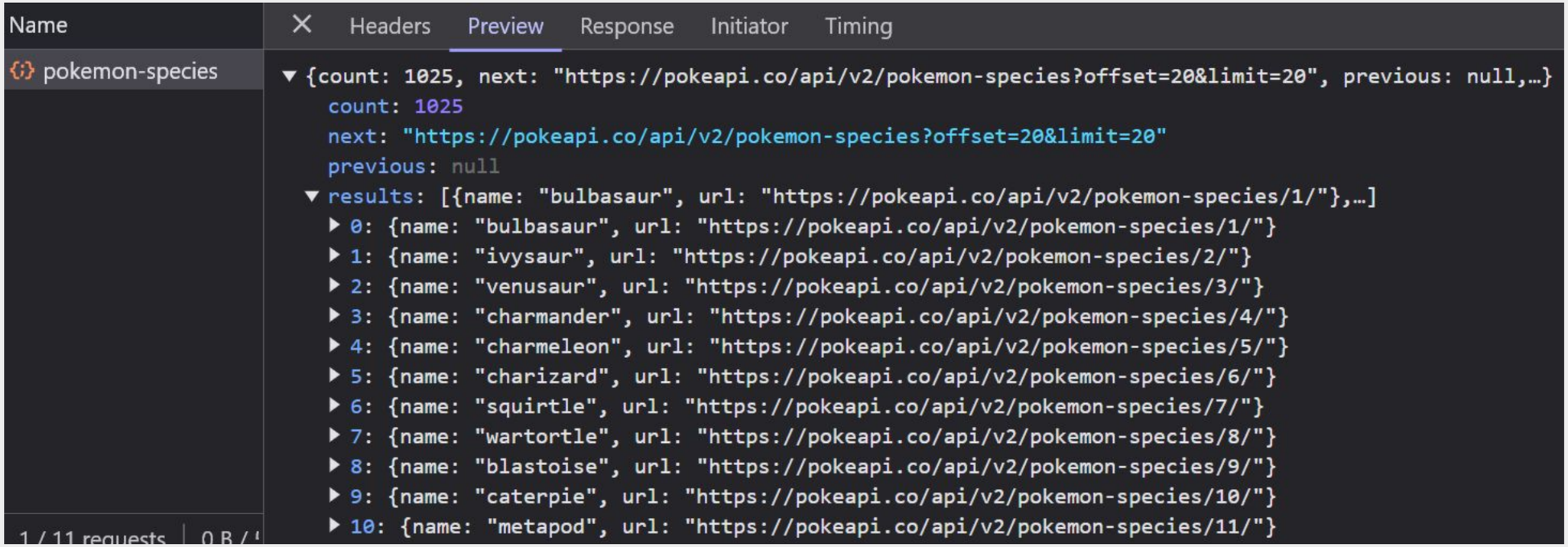
Si damos clic en la petición podremos ver información en varias pestañas.



guayerd

GET

Si damos clic en la petición podremos ver información en varias pestañas.



guayerd

POST

En el segundo argumento del método fetch se incluye un objeto de opciones. El verbo POST lo especificamos con la clave method y el body con la data que queremos enviar stringificada.

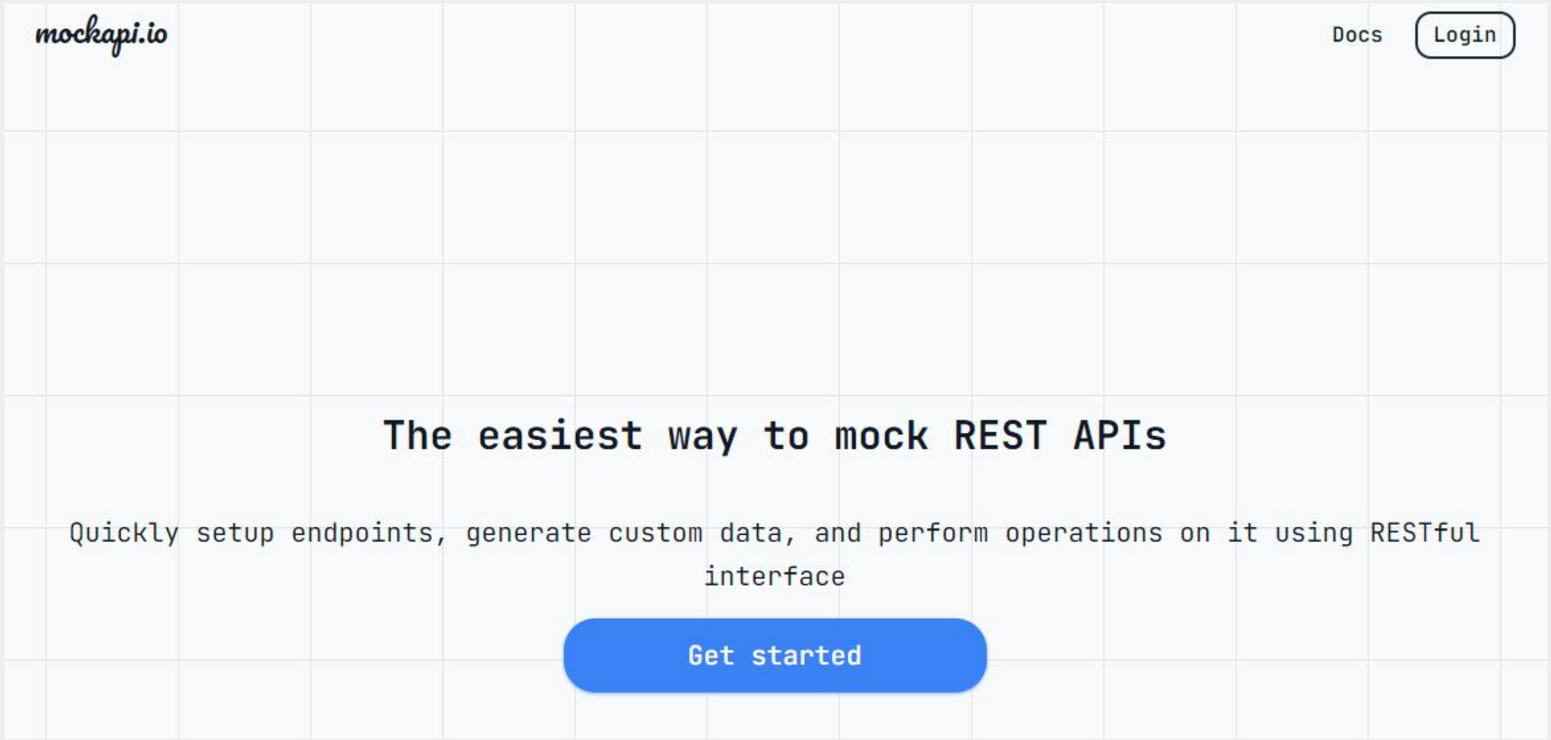
```
const recurso = {
   name: "Wen",
   avatar: "https://ipfs.everipedia.org/ipfs/QmTWLHFwyNvvbQBd9LfcbGiPu1rrWpX6rchEv3YiputpUY",
}

fetch("https://64557b55f803f34576439459.mockapi.io/cats", {
   method: "POST",
   body: JSON.stringify(recurso),
})
   .then(response ⇒ response.json())
   .then(data ⇒ console.log(data))
```

guayerd

MOCKAPI

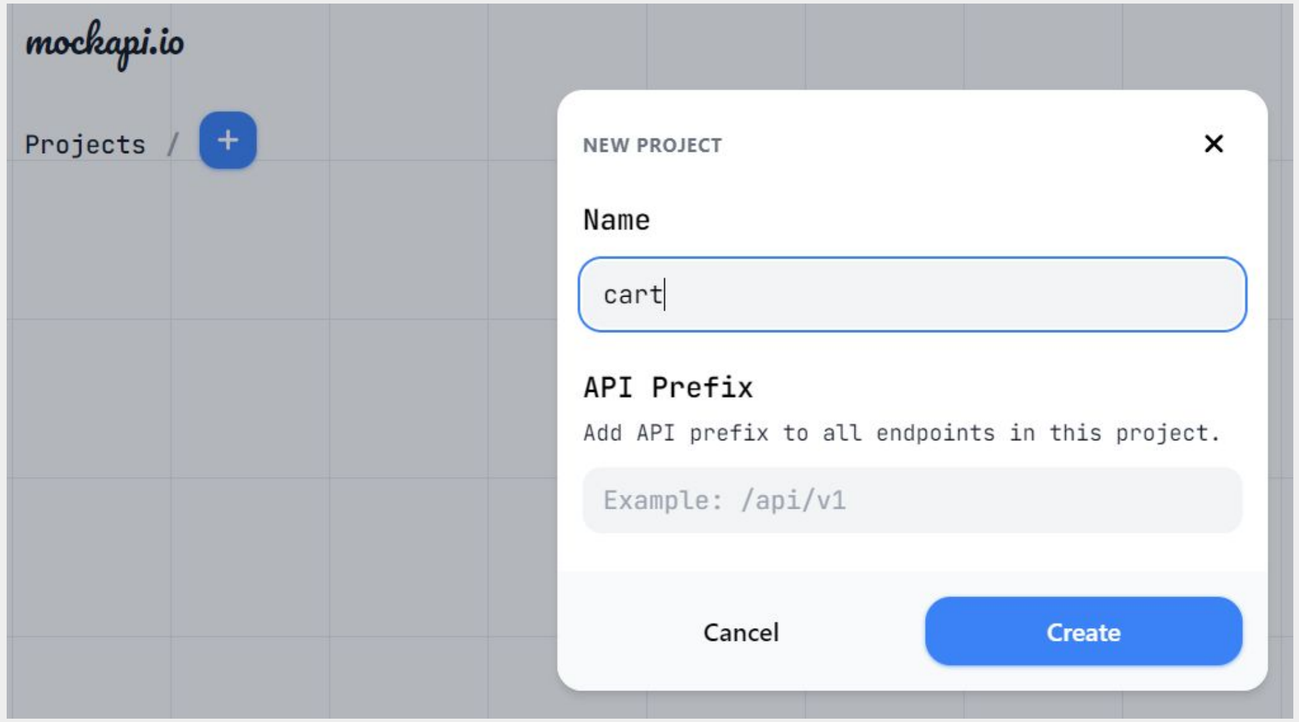
Creamos una API de prueba. Ver: https://mockapi.io/projects



guayerd

MOCKAPI

Damos un clic al botón + y ponemos en name: cart.



MOCKAPI

Damos un clic a la flecha de la derecha.

guayerd

MOCKAPI

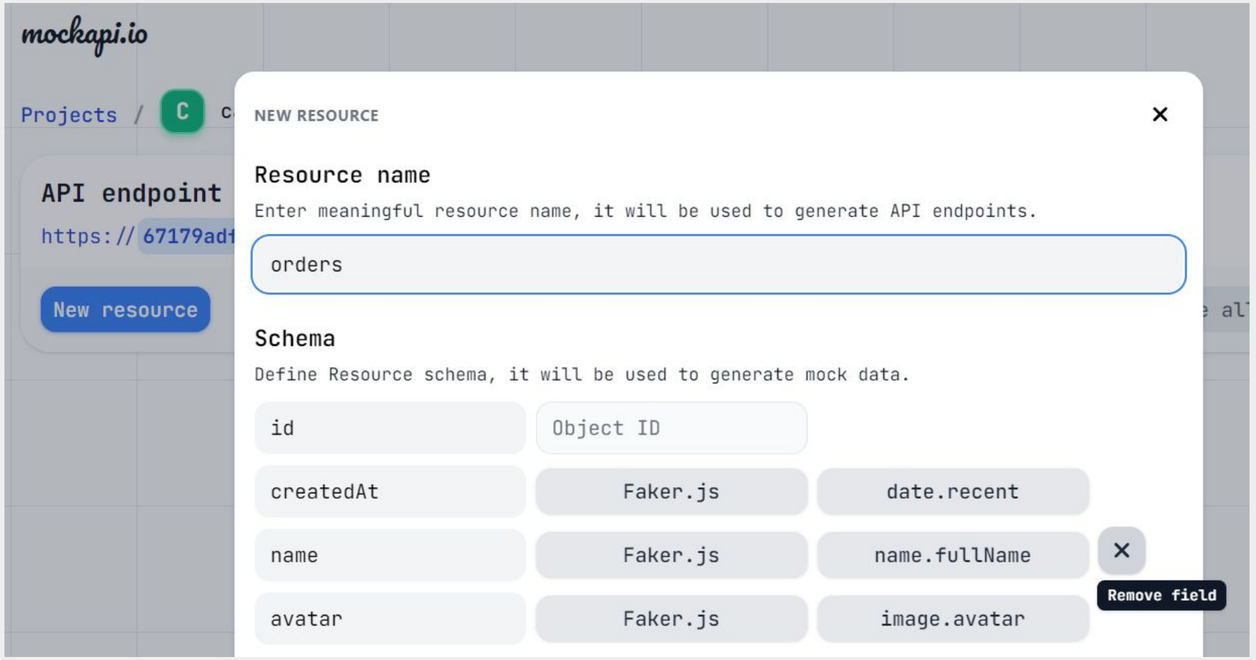
Agregamos el campo items como Object y el campo user como string.

guayerd

MOCKAPI

Damos un clic en New Resource.

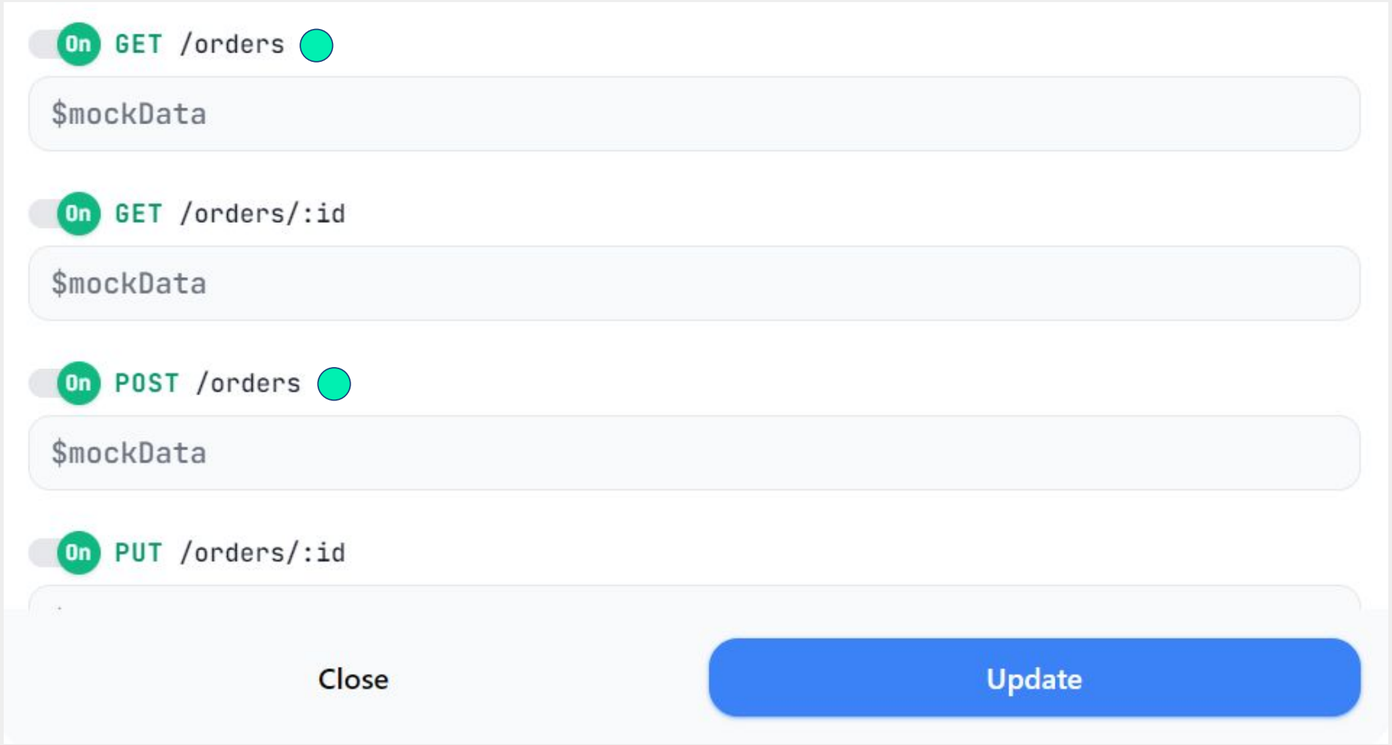
En Resource name ponemos: orders y removemos el campo name y avatar.



guayerd

MOCKAPI

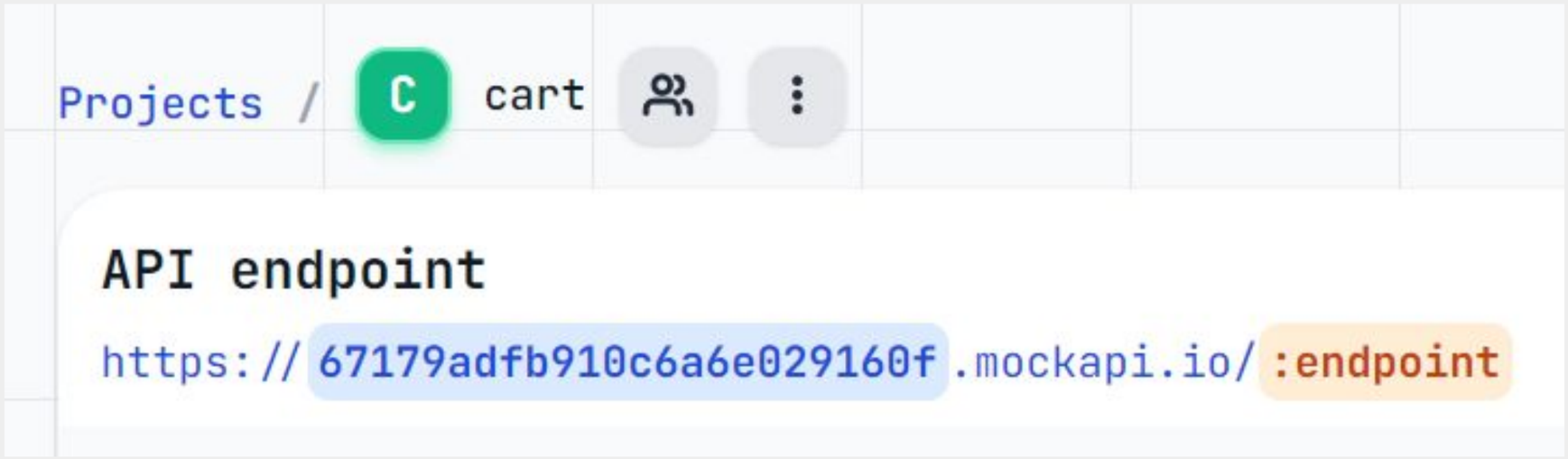
Hacemos scroll y vemos los métodos. Finalmente damos clic en Update.



guayerd

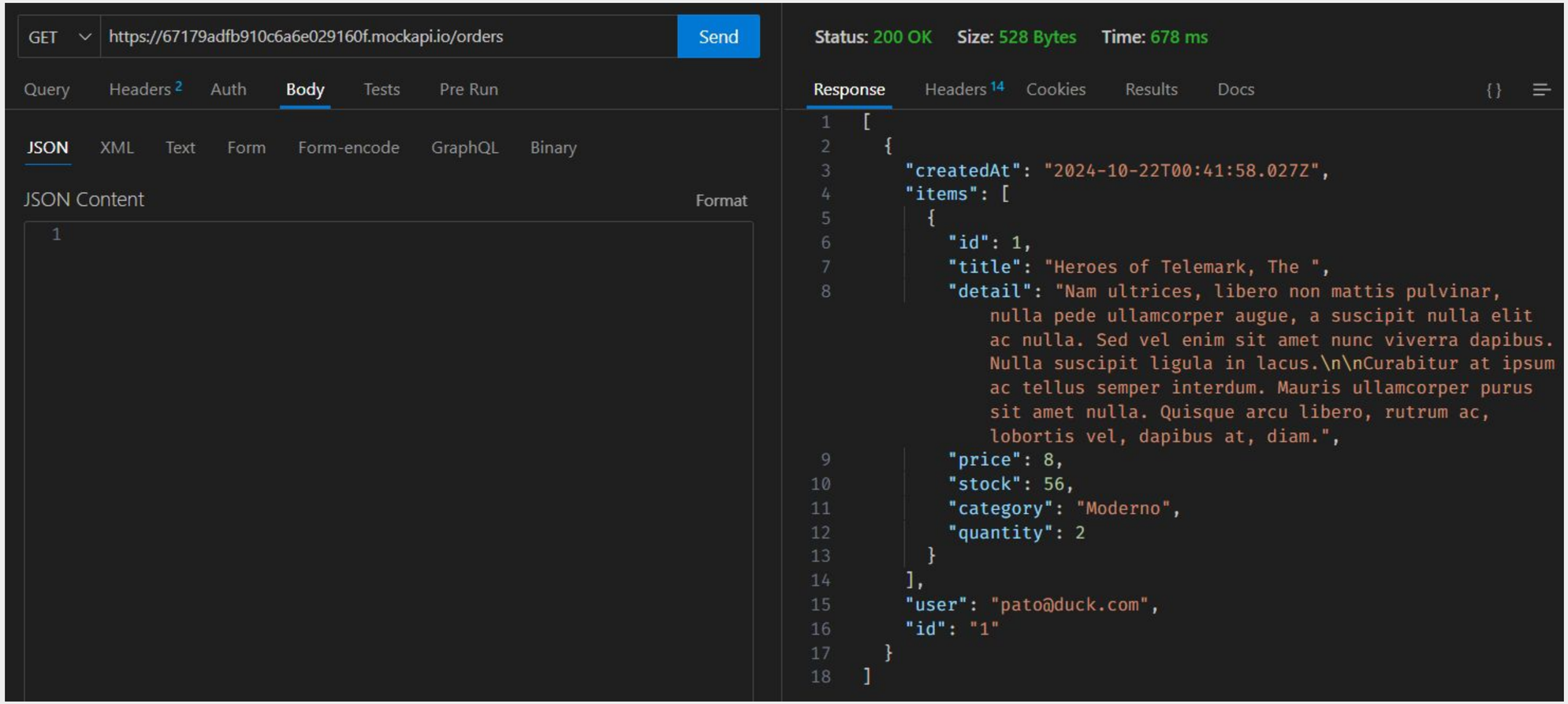
MOCKAPI

Endpoints.



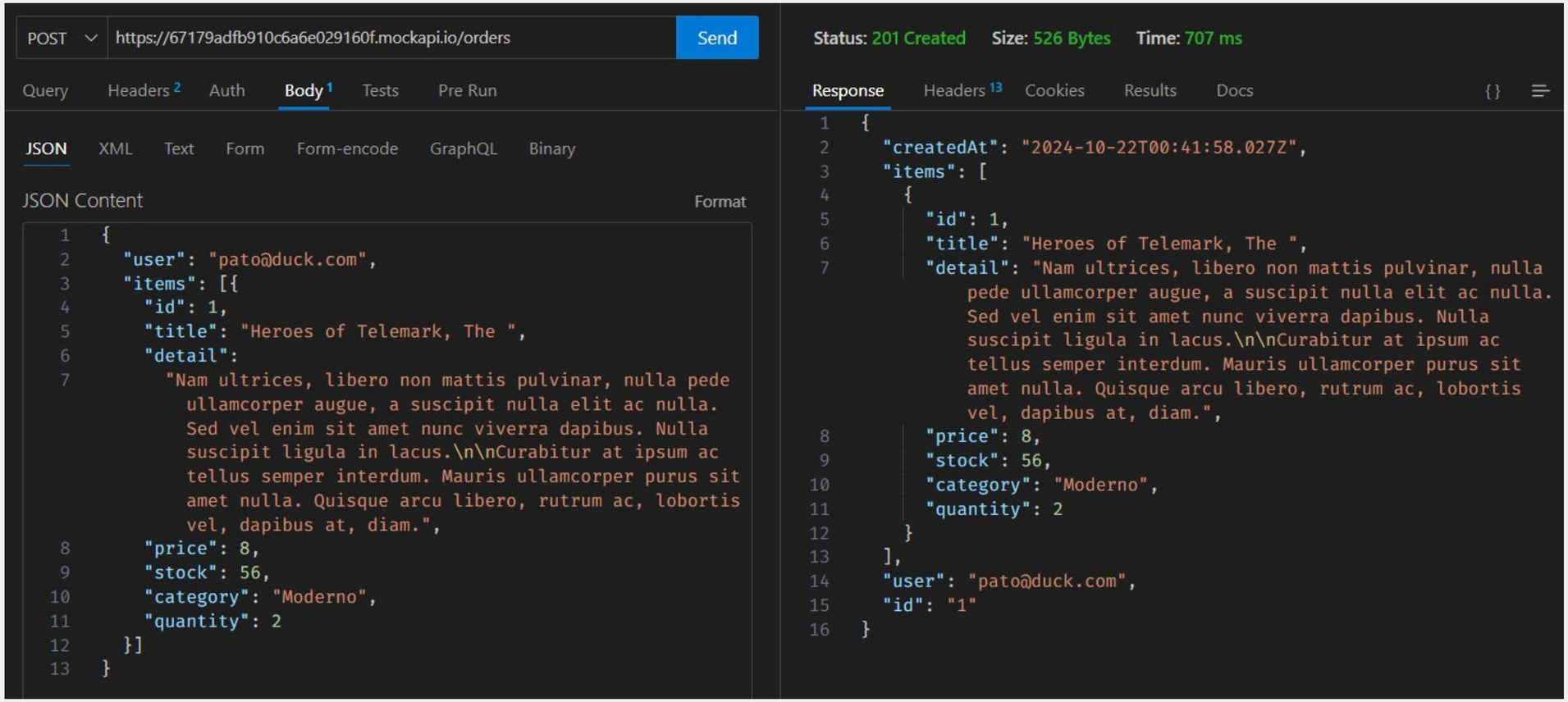
guayerd

MOCKAPI GET



guayerd

MOCKAPI POST




That's all Folks!

guayerd

¡Manos a la obra!

guayerd

SPRINT 3

Fetch

Entregable

- Crear una mockapi por grupo.
- Al dar clic en el botón Checkout de la página cart enviar un fetch para crear un recurso.
- Si **funciona**, mostrar un SweetAlert que incluya el email del usuario y el número de orden creada. Vaciar el carrito.
- Si **falla** qué catch muestre un mensaje al usuario con SweetAlert.

guayerd

retro

¿Cómo nos fué?

¿Qué cosas no quedaron claras y necesitamos repasar la próxima?