DOCUMENTACIÓN API RESTFUL

ARTASHES ASATRYAN

Forma de acceso

Todos los métodos indicados en el documento se forman desde la URL base "http://localhost:8080/api/v1"

Inicio de sesión

Recurso	Descripción
POST auth/signup	Crea una cuenta con el email y contraseña que se pase en el body.
POST auth/signin	Se inicia sesión con el email y contraseña que se pase en el body.

Coches

Recurso	Descripción
GET coche	Devuelve todos los coches que hay en la base de datos.
GET coche/{id}	Devuelve el coche que tenga el id pasado.
POST coche	Crea el coche pasado por body.
PUT coche/{id}	Actualiza el coche con el id buscado con los datos pasados por el body.
DELETE coche/{id}	Borra el coche con el id buscado.

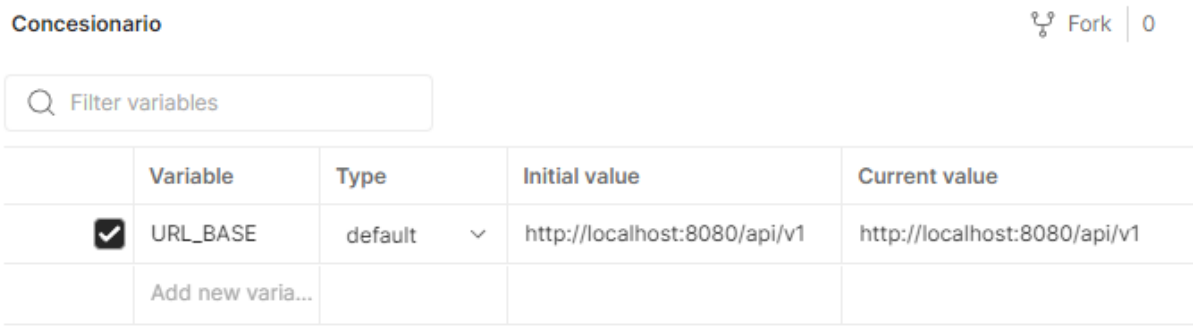
Motores

Recurso	Descripción
GET motor	Devuelve todos los motores que hay en la base de datos.
GET motor/{id}	Devuelve el motor que tenga el id

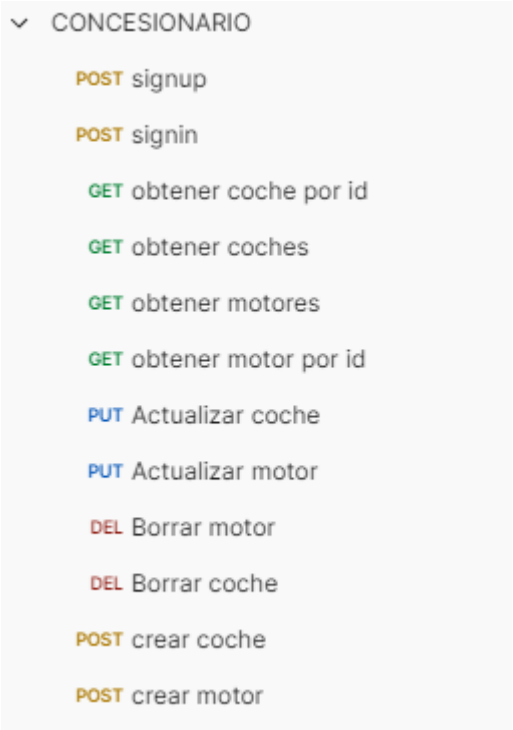
	pasado.
POST motor	Crea el motor pasado por body.
PUT motor/{id}	Actualiza el motor con el id buscado con los datos pasados por el body.
DELETE motor/{id}	Borra el motor con el id buscado.

Configuración

Para nuestro proyecto que se va a ejecutar en el puerto 8080 local, vamos a utilizar el entorno "Concesionario":



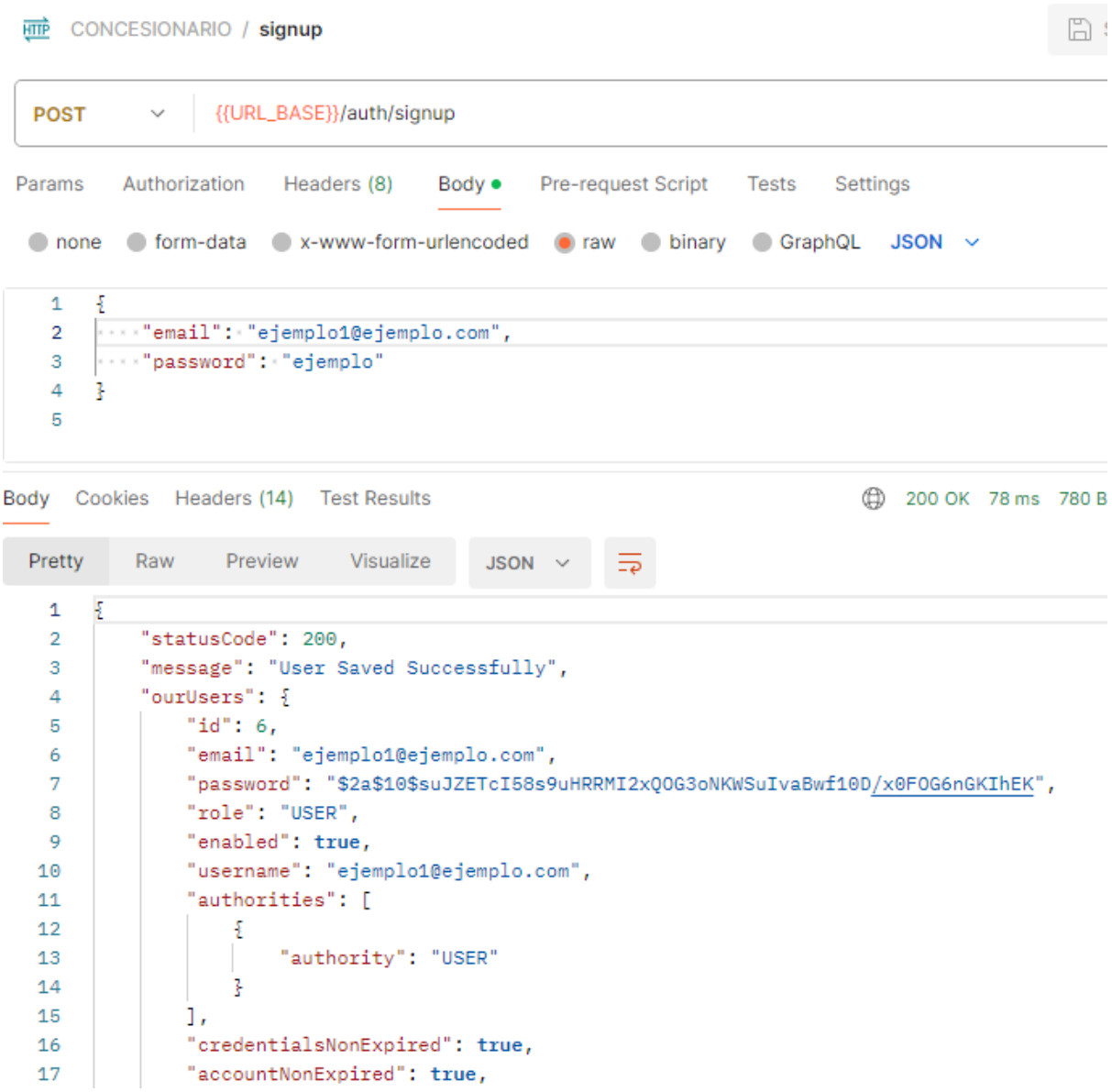
Vamos a tener los siguientes tests:



Crear cuenta

Artashes Asatryan

Primero, necesitaremos crear una cuenta. Para ello nos dirigimos al enlace de signup, pasamos el json del email y contraseña por el body y hacemos un POST:

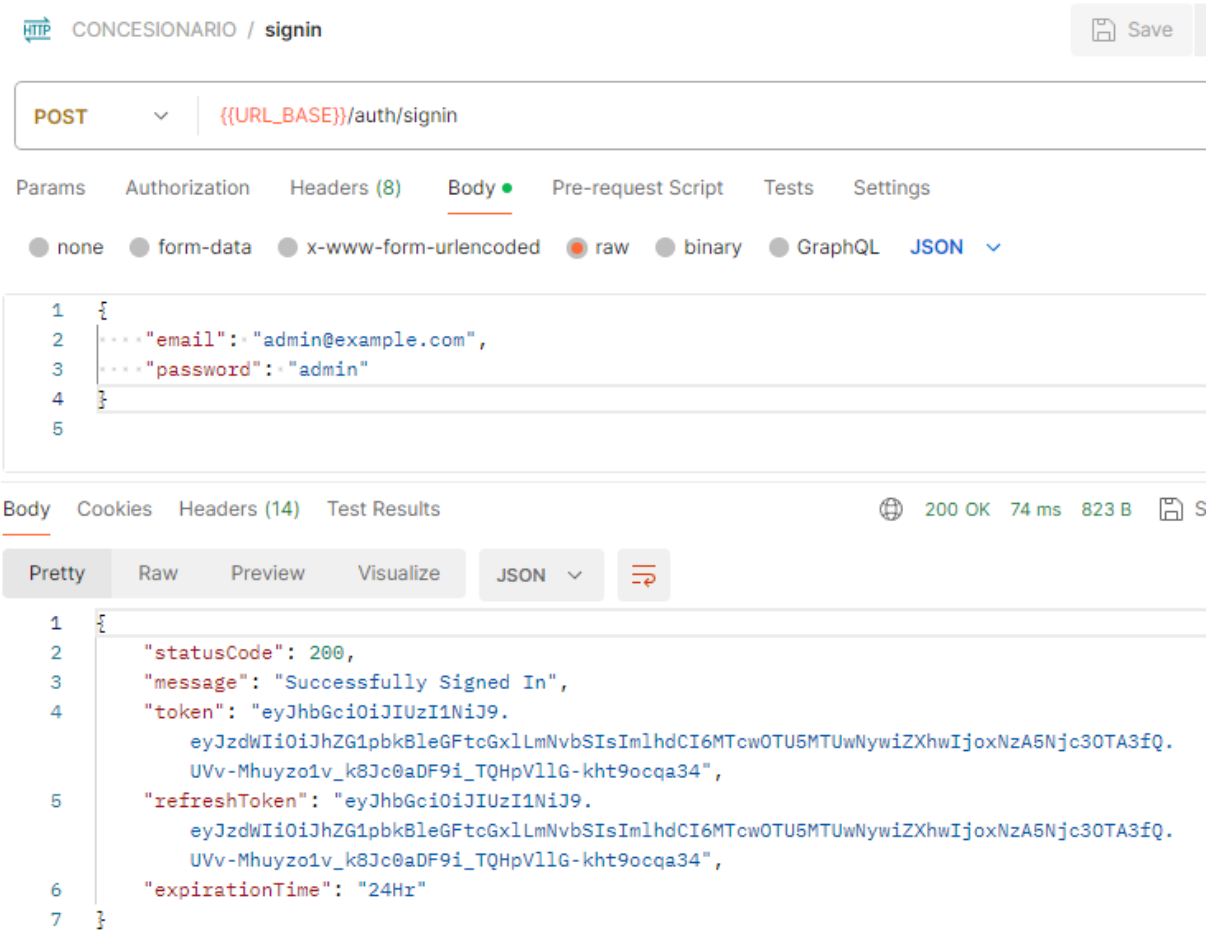


Por defecto, el usuario que se crea tiene rol de usuario normal, para que no tenga acceso al CRUD completo.

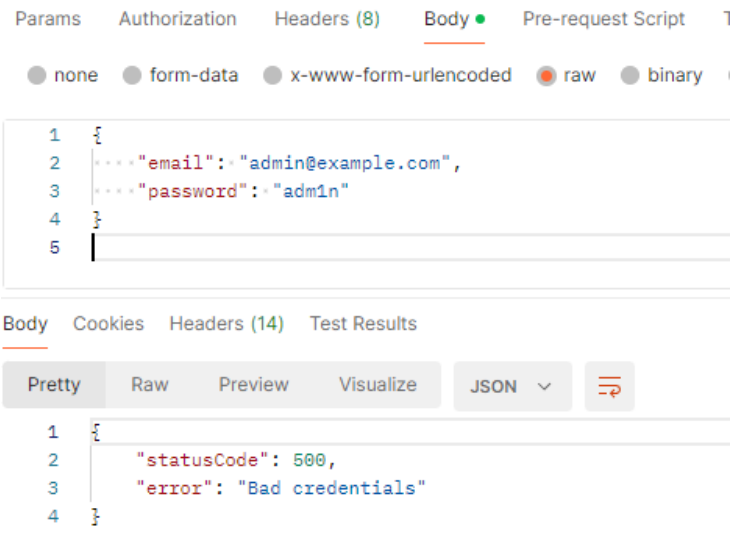
Iniciar sesión

Ya podemos iniciar sesión con esta cuenta:

Nos dirigimos al signin y hacemos un POST con el json del usuario creado en el body:



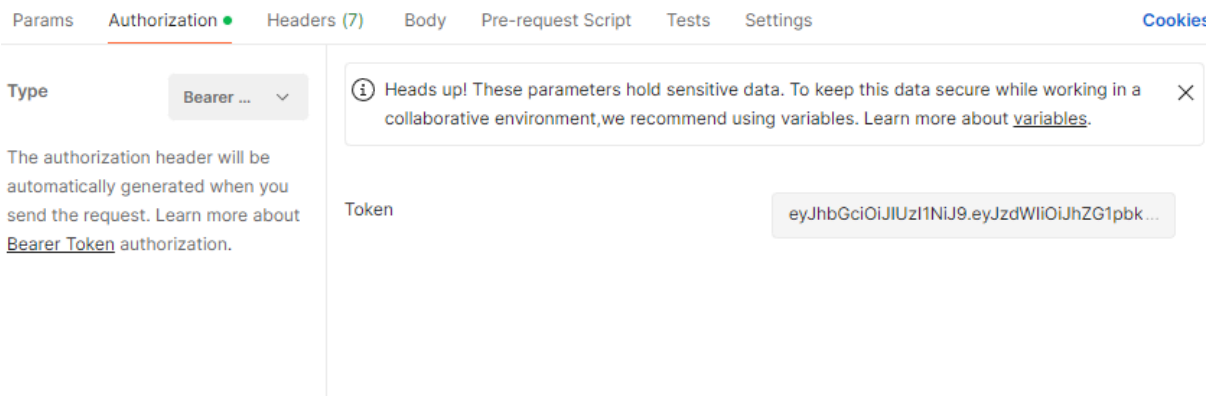
Nos devuelve el mensaje de que se ha iniciado sesión y nos da un token.



Si introducimos credenciales incorrectas, nos avisará. Para realizar los demás tests, iniciamos sesión con la cuenta del admin y guardamos el token.

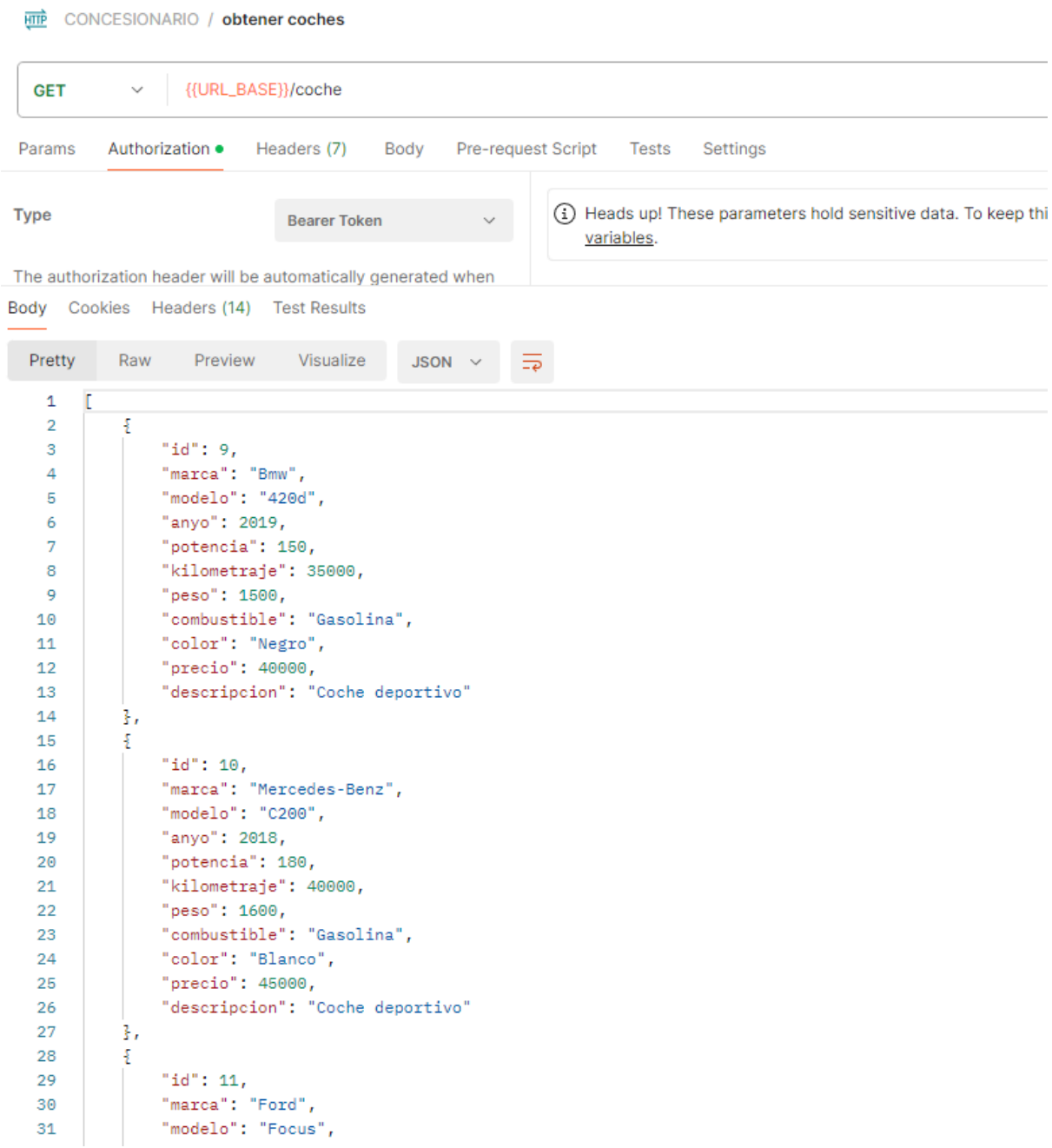
Configuración tests

Para los siguientes tests, necesitamos configurar el token en el apartado de Autorización de cada test que vayamos a hacer:



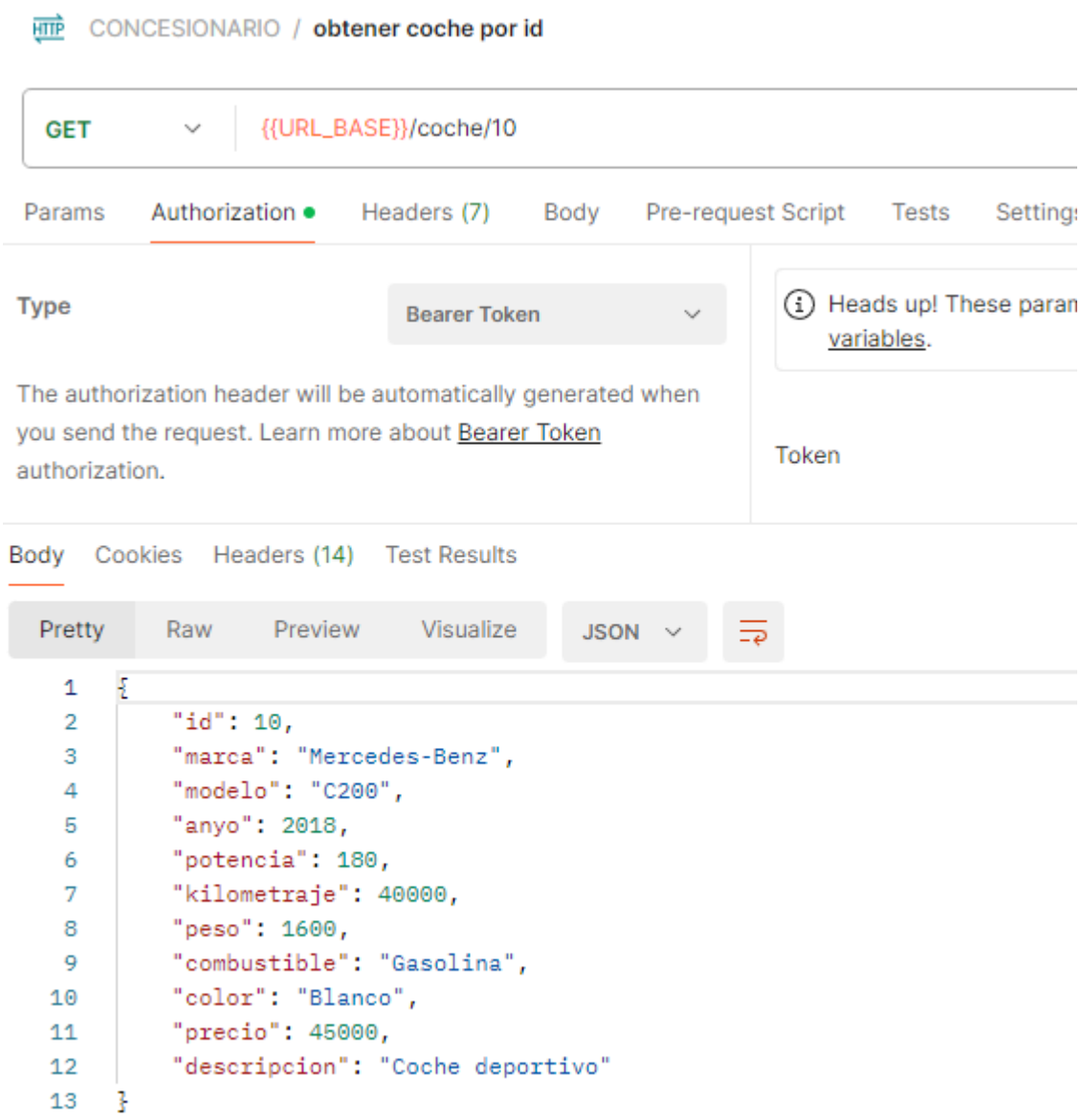
1. **Obtener coches**

Nos dirigimos a “URL/coche” y realizamos un GET para recuperar todos los coches.



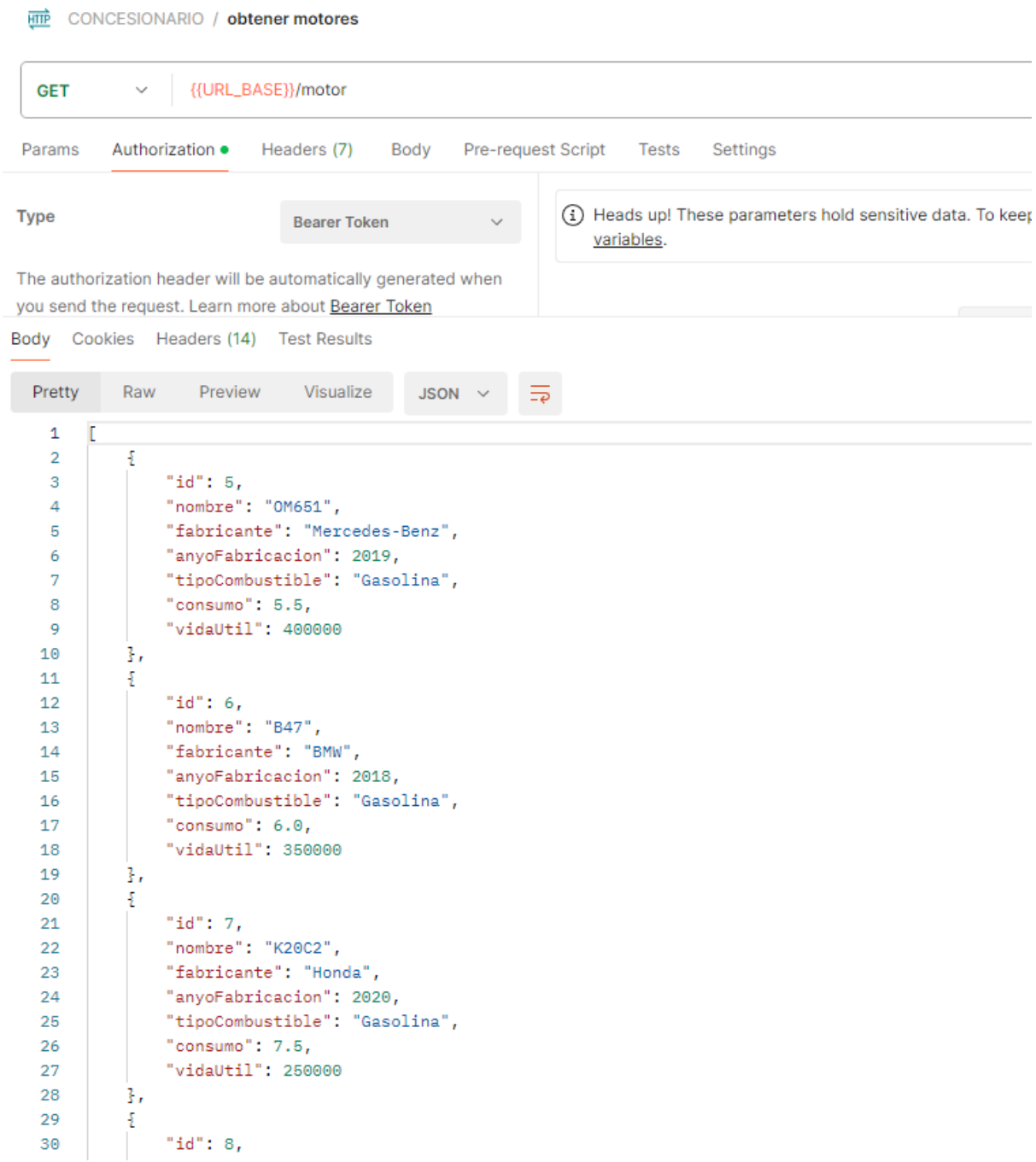
2. Obtener un coche por id

Nos dirigimos a “URL/coche/id” y realizamos un GET para recuperar el coche según el id.



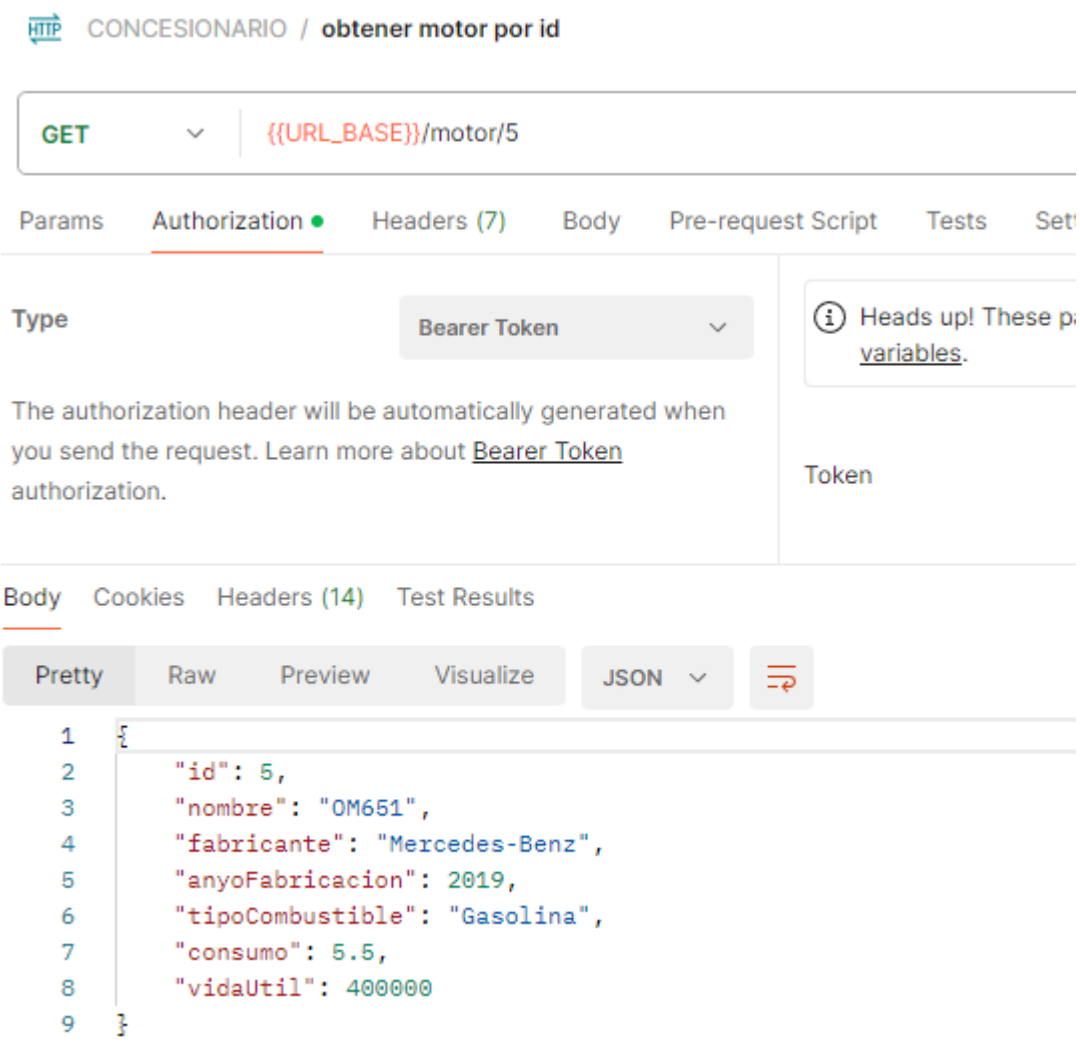
3. Obtener motores

Realizamos los mismos pasos que en el caso de los coches, pero cambiando coche por motor: "URL/motor"



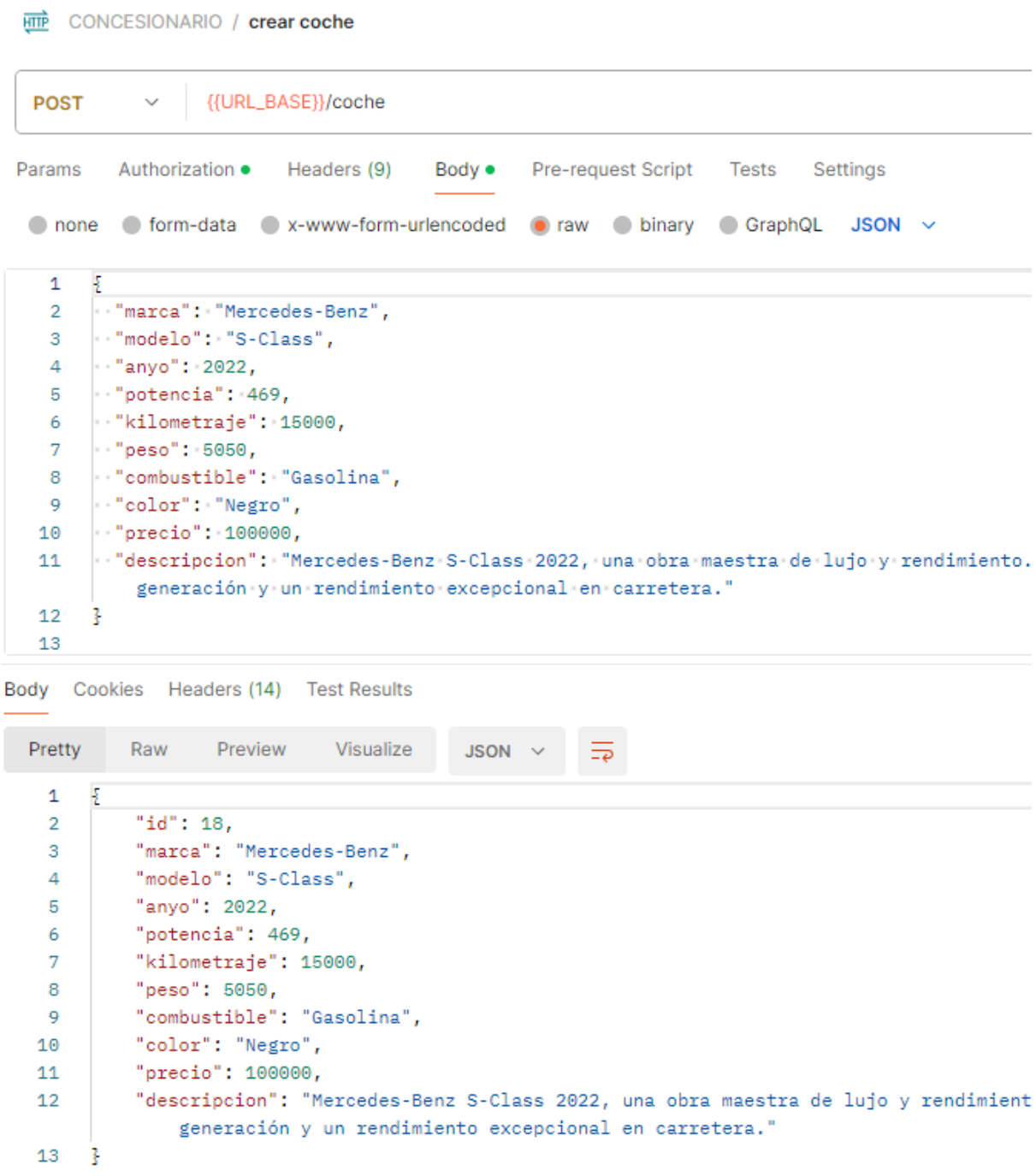
4. Obtener un motor por id

Nos dirigimos a “URL/motor/id” y realizamos un GET para recuperar el motor según el id.



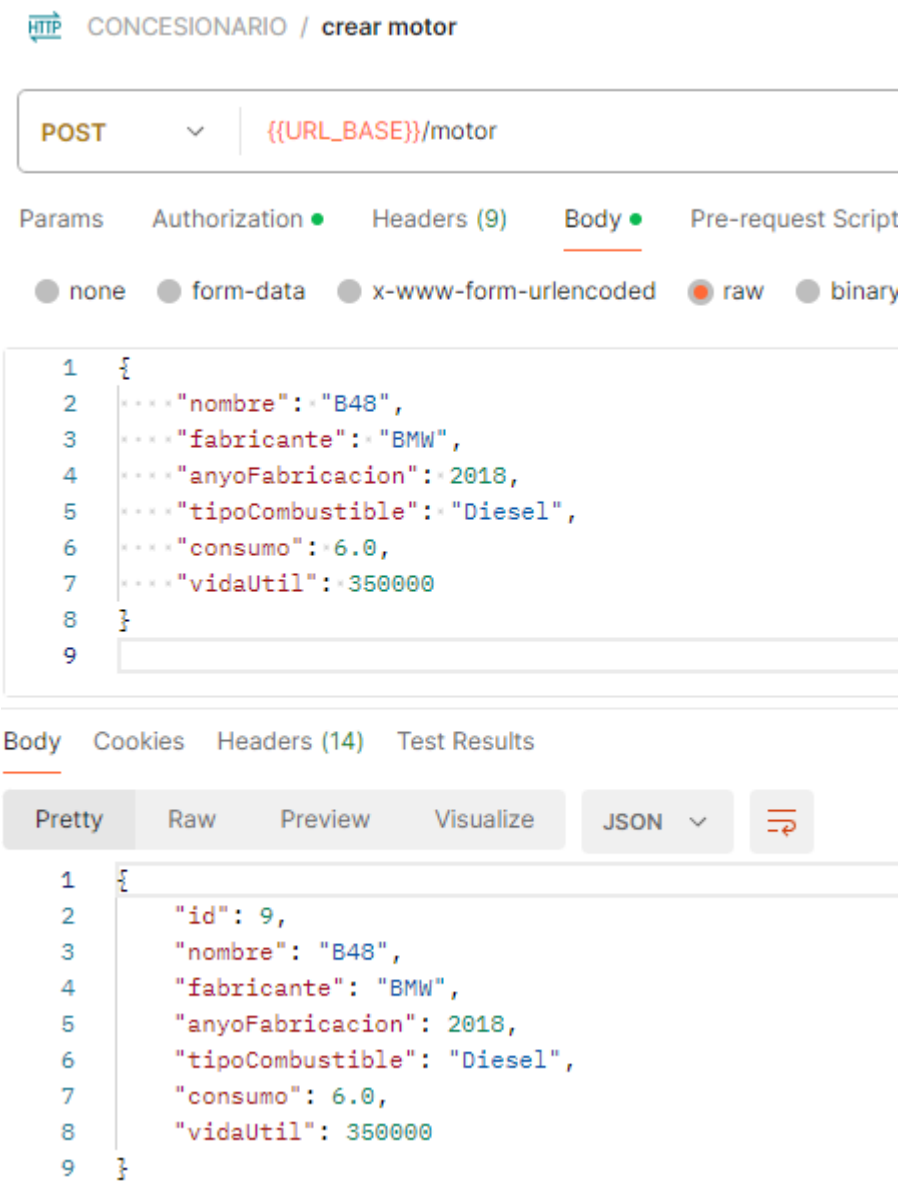
5. Crear coche

Nos dirigimos a “URL/coche”, introducimos el token de un admin, pasamos el coche por body, y realizamos un POST.



6. Crear motor

Nos dirigimos a “URL/motor”, introducimos el token de un admin, pasamos el motor por body, y realizamos un POST.



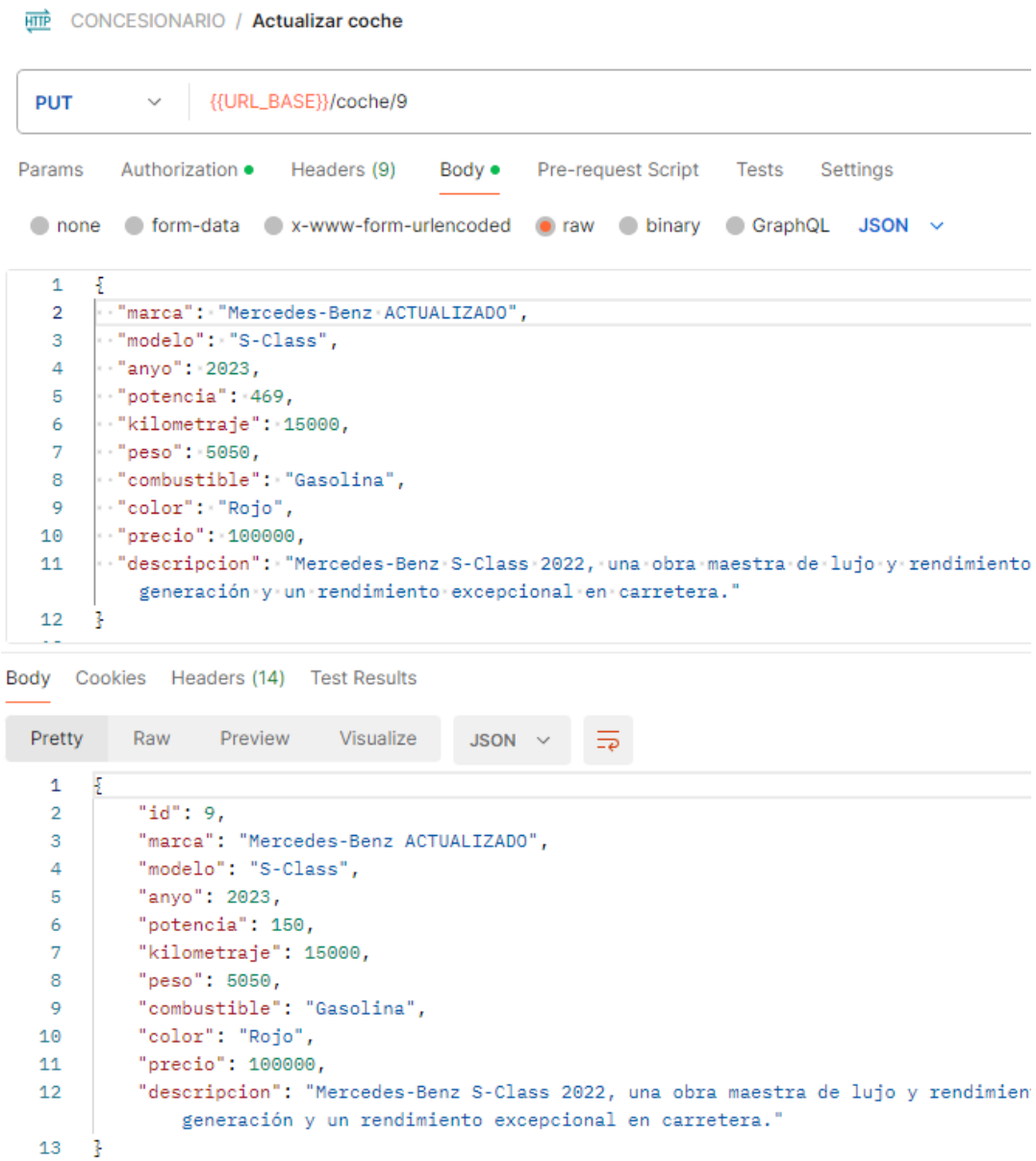
7. Actualizar coche

Nos dirigimos a coche y tenemos que introducir el id del coche que queremos actualizar “URL/coche/id”.

En este caso vamos a modificar el siguiente:

```
{
    "id": 18,
    "marca": "Mercedes-Benz",
    "modelo": "S-Class",
    "anyo": 2022,
    "potencia": 469,
    "kilometraje": 15000,
    "peso": 5050,
    "combustible": "Gasolina",
    "color": "Negro",
    "precio": 100000,
    "descripcion": "Mercedes-Benz S-Class 2022, una obra maestra d
        generación y un rendimiento excepcional en carretera."
}
```


Pasamos el coche actualizado por body y tiene que dar un status: 200 OK.



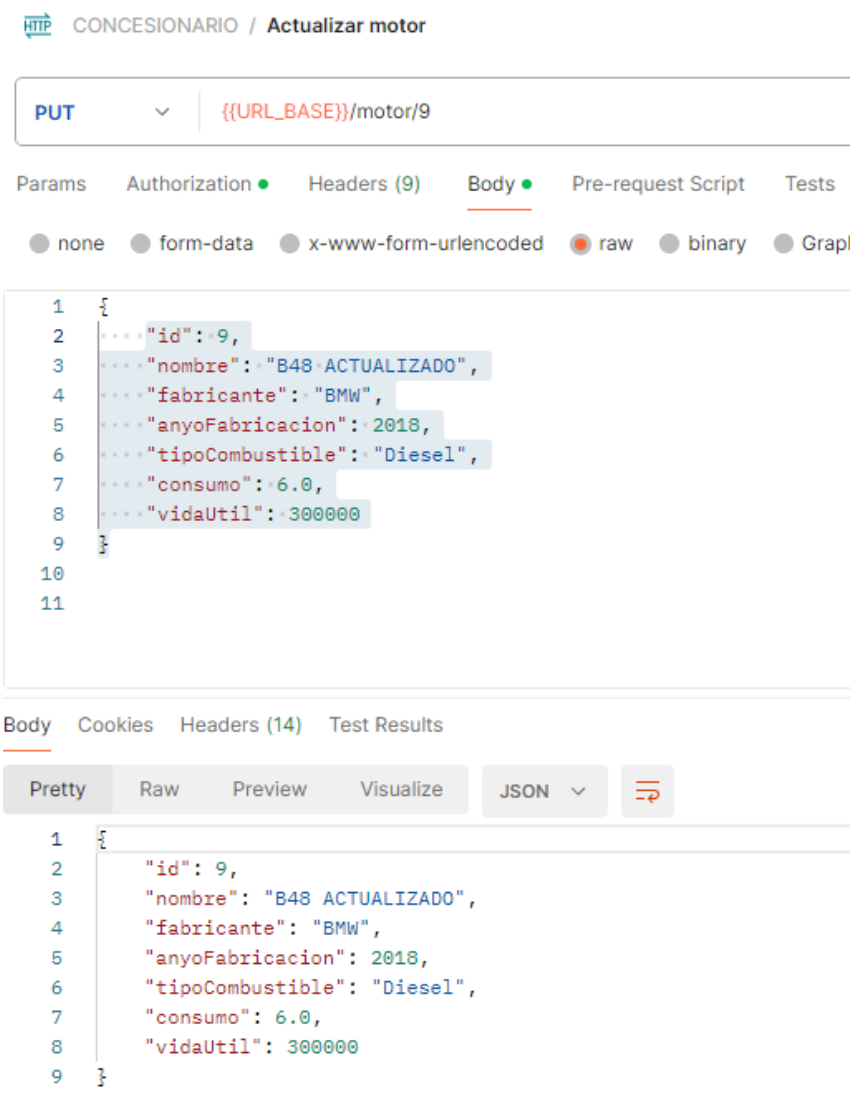
8. Actualizar motor

Para actualizar un motor, nos dirigimos a motor y tenemos que introducir el id del motor que queremos actualizar "URL/motor/id".

En este caso vamos a modificar el siguiente:


```
{
    "id": 9,
    "nombre": "B48",
    "fabricante": "BMW",
    "anyoFabricacion": 2018,
    "tipoCombustible": "Diesel",
    "consumo": 6.0,
    "vidaUtil": 350000
}
```

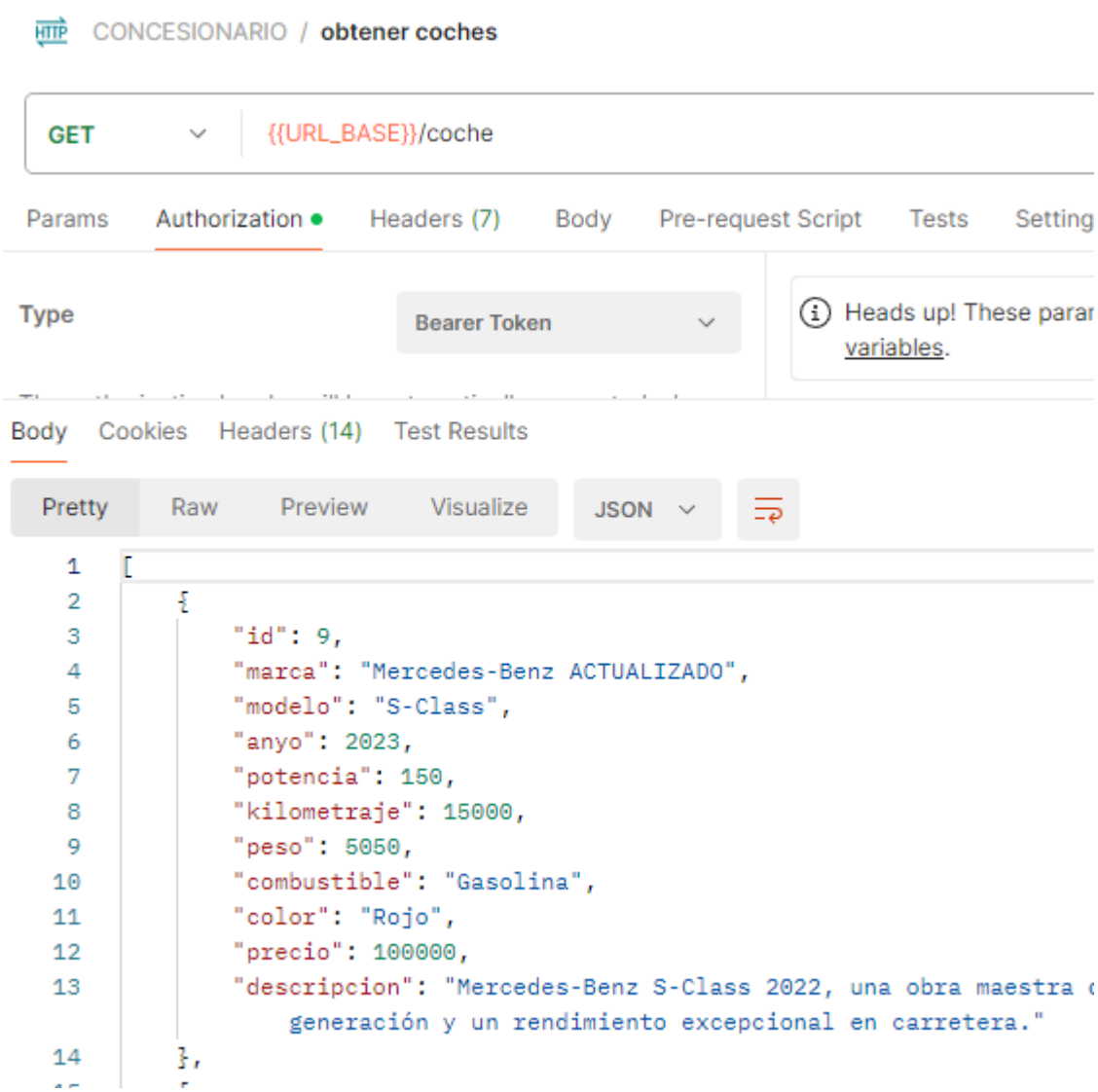

Pasamos el motor actualizado por body y tiene que dar un status: 200 OK.

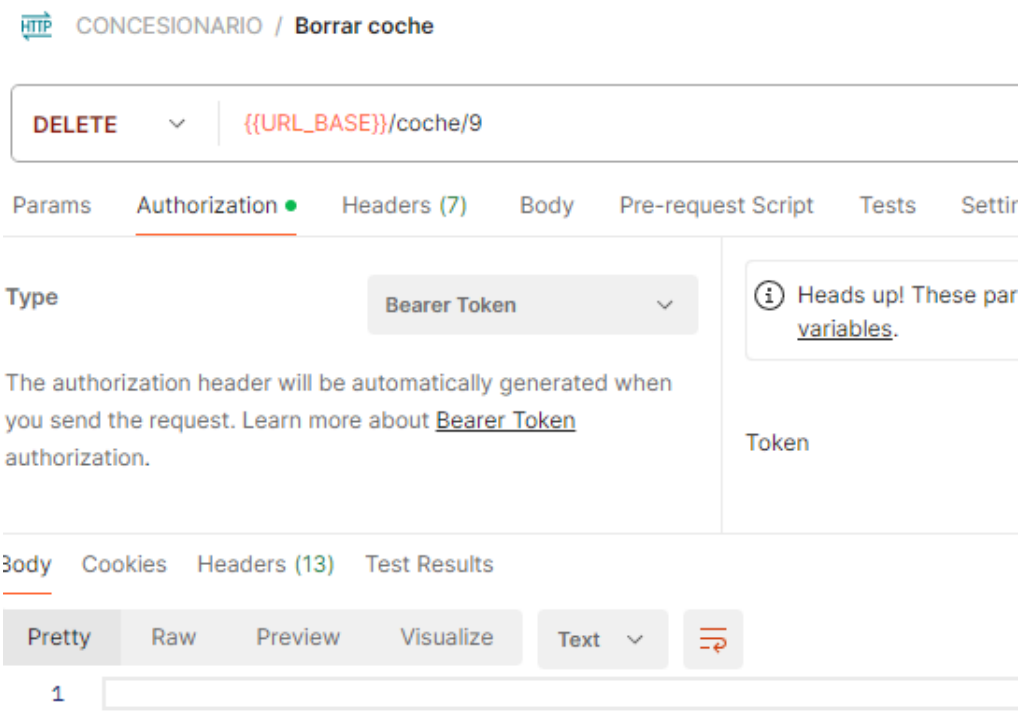


9. Borrar coche

Para borrar un coche, nos dirigimos a coche y tenemos que introducir el id del coche que queremos borrar “URL/coche/id”.

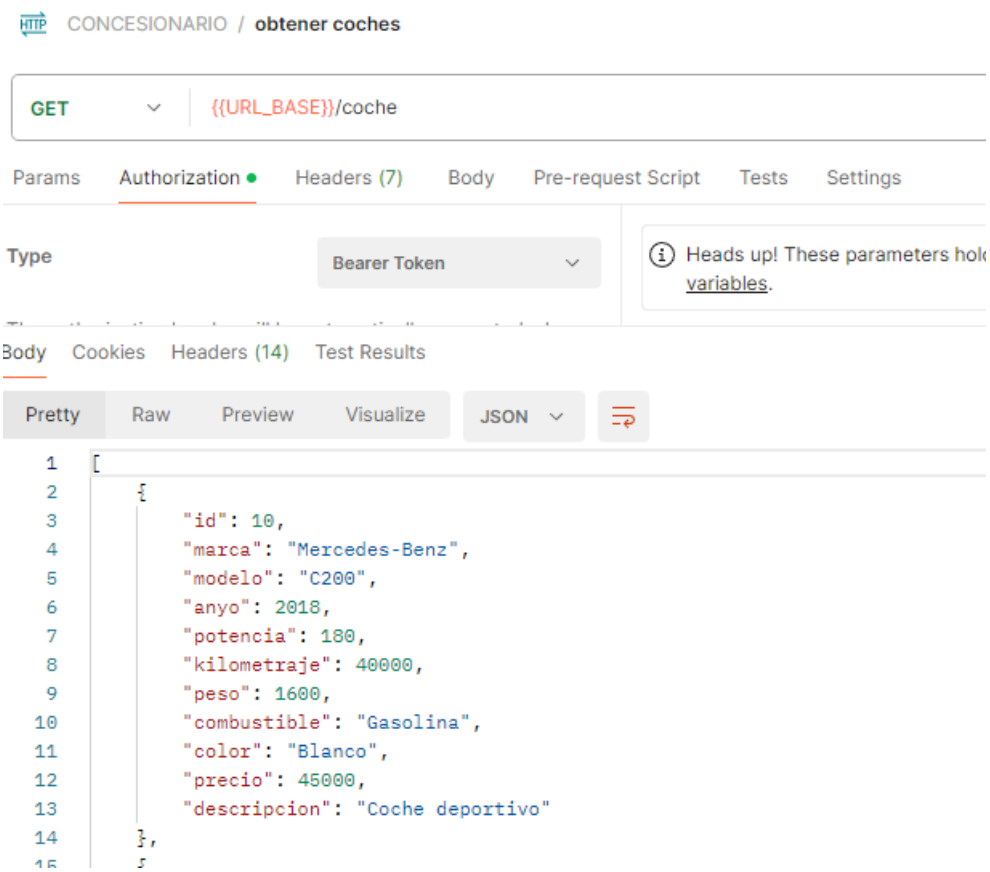
En este caso, vamos a borrar el que tiene id=9


CONCESIONARIO / obtener coches
GET
{{URL_BASE}}/coche
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Setting
Type
Bearer Token
Heads up! These parar
variables.
Body
Cookies
Headers (14)
Test Results
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 9,
"marca": "Mercedes-Benz ACTUALIZADO",
"modelo": "S-Class",
"anyo": 2023,
"potencia": 150,
"kilometraje": 15000,
"peso": 5050,
"combustible": "Gasolina",
"color": "Rojo",
"precio": 100000,
"descripcion": "Mercedes-Benz S-Class 2022, una obra maestra
generación y un rendimiento excepcional en carretera."
},



Si nos devuelve un Status: 200 OK, es que se ha realizado correctamente.

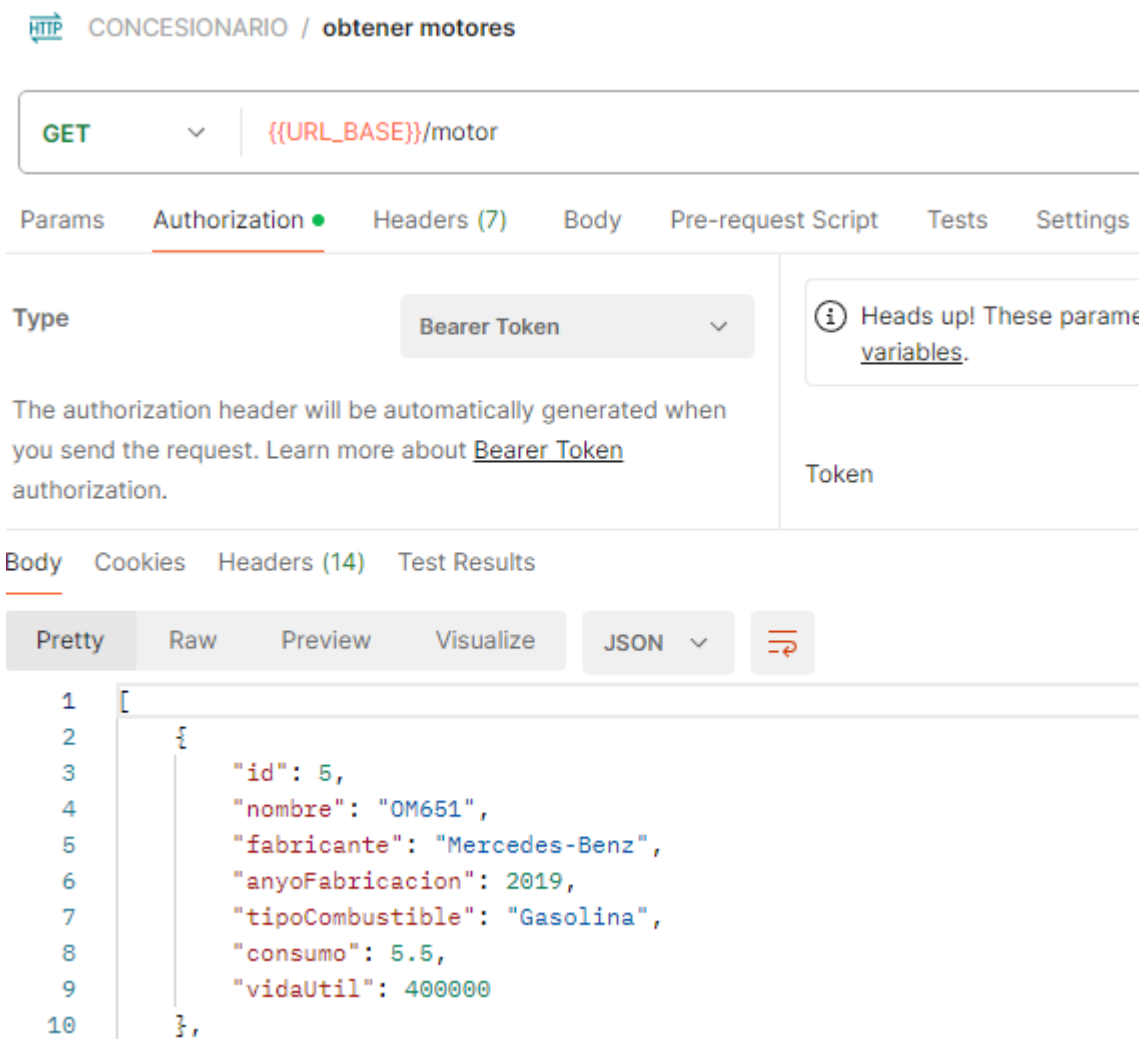
Si volvemos a comprobar todos los coches, vemos que no aparece:

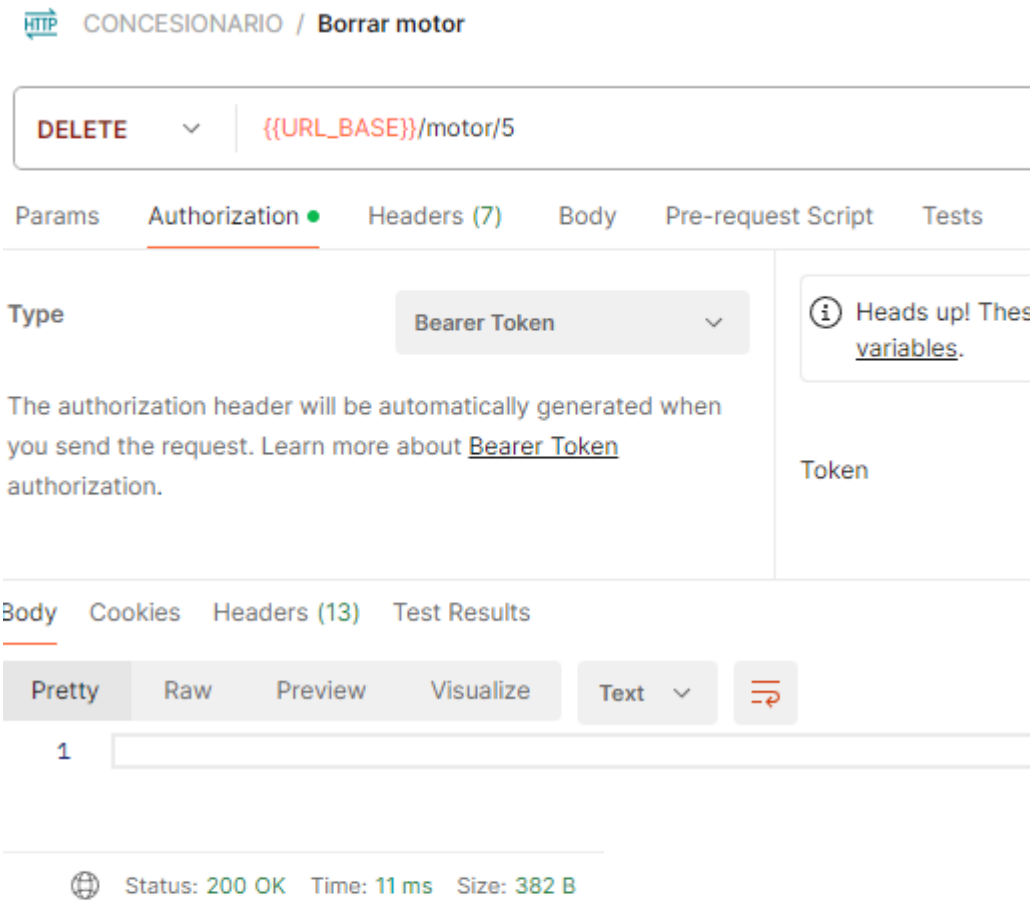


10. Borrar motor

Para borrar un motor, nos dirigimos al motor y tenemos que introducir el id del motor que queremos borrar “URL/motor/id”.

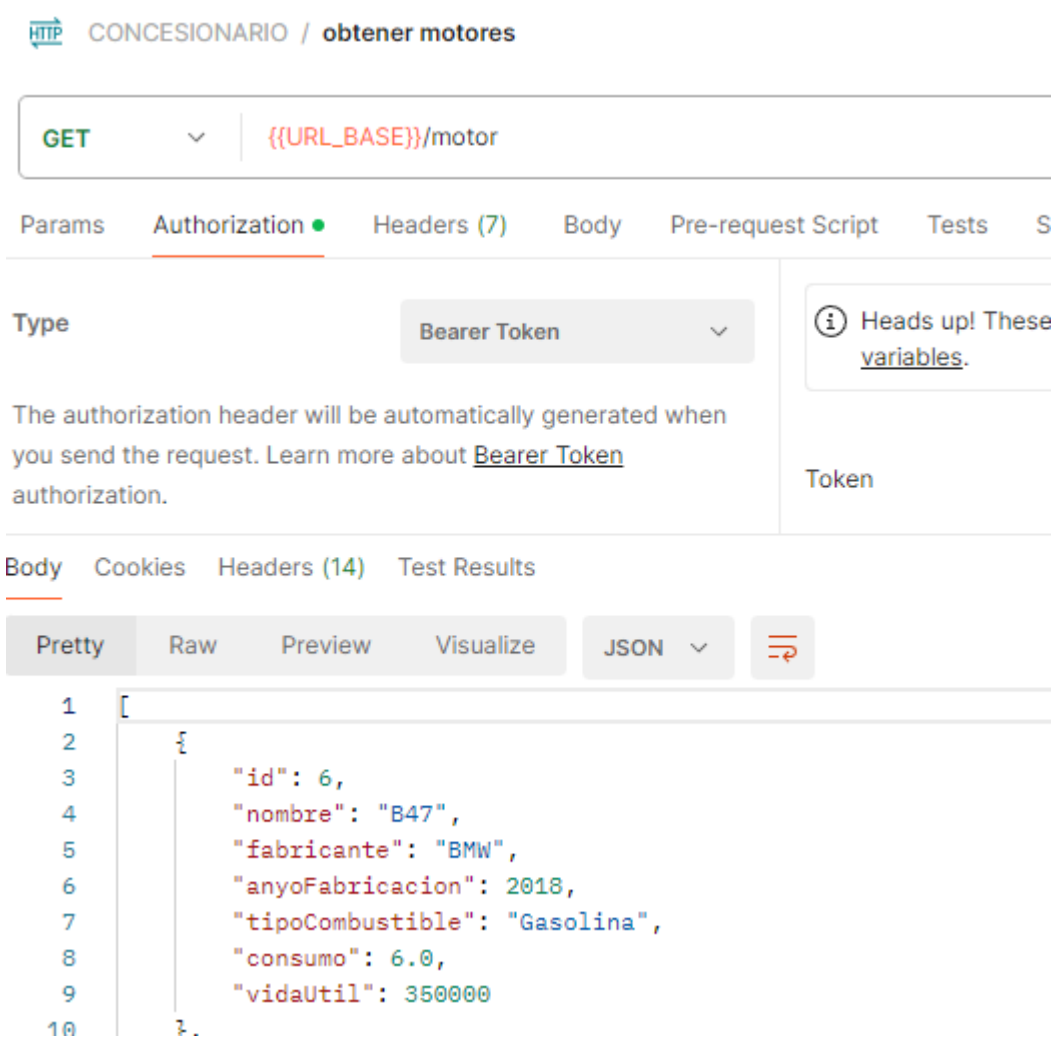
En este caso, vamos a borrar el que tiene id=5


CONCESIONARIO / obtener motores
GET
{{URL_BASE}}/motor
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Type
Bearer Token
The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.
Heads up! These parame
variables.
Token
Body
Cookies
Headers (14)
Test Results
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 5,
"nombre": "OM651",
"fabricante": "Mercedes-Benz",
"anyoFabricacion": 2019,
"tipoCombustible": "Gasolina",
"consumo": 5.5,
"vidaUtil": 400000
},



Si nos devuelve un Status: 200 OK, es que se ha realizado correctamente.

Si volvemos a comprobar todos los motores, vemos que no aparece:


CONCESIONARIO / obtener motores
GET
{{URL_BASE}}/motor
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Type
Bearer Token
Heads up! These
variables.
The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.
Token
Body
Cookies
Headers (14)
Test Results
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": 6,
"nombre": "B47",
"fabricante": "BMW",
"anyoFabricacion": 2018,
"tipoCombustible": "Gasolina",
"consumo": 6.0,
"vidaUtil": 350000