

# AutoMobile



Artashes Asatryan 2°DAW

# Índice

<b>Forma de Acceso.....</b>	<b>Página 3</b>
<b>Documentación completa.....</b>	<b>Página 3</b>
<b>Inicio de sesión.....</b>	<b>Página 3</b>
<b>Usuarios.....</b>	<b>Página 3-4</b>
<b>Coches.....</b>	<b>Página 4</b>
<b>Imágenes.....</b>	<b>Página 4-5</b>
<b>Noticias.....</b>	<b>Página 5</b>
<b>Configuración.....</b>	<b>Página 6</b>
<b>Crear Cuenta.....</b>	<b>Página 6-7</b>
<b>Iniciar sesión.....</b>	<b>Página 7-9</b>
<b>Configuración tests.....</b>	<b>Página 9-12</b>

## Forma de acceso

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

## Documentación completa

<https://documenter.getpostman.com/view/31938682/2sA3JNaKt7>

### Acceso

Recurso	Descripción
/public	Es de acceso <b>público</b>
/adminuser	Pueden acceder <b>Administradores</b> y <b>Usuarios</b> registrados
/admin	Pueden acceder sólo los <b>Administradores</b> registrados
/user	Pueden acceder sólo los <b>Usuarios</b> registrados

### Inicio de sesión

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

### Usuarios

Recurso	Descripción
---------	-------------

GET admin/usuario	Devuelve todos los usuarios que hay en la base de datos.
GET admin/usuario/{id}	Devuelve el usuario que tenga el id pasado.
GET adminuser/usuario/actual	Devuelve el usuario que tiene la sesión iniciada
POST public/usuario	Crea el usuario pasado por body.
PUT admin/usuario/{id}	Actualiza el usuario con el id buscado con los datos pasados por el body.
PUT adminuser/usuario	Actualiza el perfil del usuario que está iniciado sesión.
DELETE admin/usuario/{id}	Borra el usuario con el id buscado.

## Coches

Recurso	Descripción
GET public/coche	Devuelve todos los coches que hay en la base de datos.
GET public/coche/{id}	Devuelve el coche que tenga el id pasado.
GET adminuser/coche/usuario/{id}	Devuelve los coches que tenga el usuario con id pasado.
GET adminuser/coche/favorito/usuario/{id}	Devuelve los coches favoritos del usuario con id pasado.
GET <a href="https://www.autoscout24.es/listing-form/models/C/{id}">https://www.autoscout24.es/listing-form/models/C/{id}</a>	Devuelve los modelos de coche por la marca (se pasa un id por cada marca)
POST adminuser/coche	Crea el coche pasado por body.

<b>PUT</b> adminuser/coche/{id}	Actualiza el coche con el id buscado con los datos pasados por el body.
<b>DELETE</b> adminuser/coche/{id}	Borra el coche con el id buscado.

## Imágenes

Recurso	Descripción
<b>GET</b> public/imagen	Devuelve todas las imágenes que hay en la base de datos.
<b>GET</b> public/imagen/{id}	Devuelve la imagen que tenga el id pasado.
<b>GET</b> public/imagen/coche/{id}	Devuelve las imágenes del coche con el id pasado.
<b>POST</b> adminuser/imagen	Crea la imagen pasada por body.
<b>PUT</b> adminuser/imagen/{id}	Actualiza la imagen con el id buscado con los datos pasados por el body.
<b>DELETE</b> adminuser/imagen/{id}	Borra la imagen con el id buscado.

## Noticias

Recurso	Descripción
<b>GET</b> adminuser/noticia	Devuelve todas las noticias que hay en la base de datos.
<b>GET</b> adminuser/noticia/{id}	Devuelve la noticia que tenga el id pasado.
<b>POST</b> admin/noticia	Crea la noticia pasada por body.
<b>PUT</b> admin/noticia/{id}	Actualiza la noticia con el id buscado con los datos pasados por el body.

<b>DELETE</b> admin/noticia/{id}	Borra la noticia con el id buscado.
----------------------------------	-------------------------------------

## Favoritos

Recurso	Descripción
<b>GET</b> adminuser/favorito	Devuelve todos los favoritos que hay en la base de datos.
<b>GET</b> adminuser/favorito/{id}	Devuelve el favorito que tenga el id pasado.
<b>GET</b> adminuser/favorito/usuario/{id}	Devuelve los favoritos que tiene el usuario con el id pasado.
<b>POST</b> adminuser/favorito	Crea el favorito pasado por body.
<b>PUT</b> adminuser/favorito/{id}	Actualiza el favorito con el id buscado con los datos pasados por el body.
<b>DELETE</b> adminuser/favorito/{id}	Borra el favorito con el id buscado.

## Configuración

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

AutoMobile Save Fork 0 Share ...

	Variable	Type	Initial value	Current value	...
<input checked="" type="checkbox"/>	URL_BASE	default <span>▼</span>	http://localhost:8080/api/v1	http://localhost:8080/api/v1	
	Add new variable				

## Crear cuenta

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

Por defecto, el usuario que se crea tiene rol de usuario.

## AutoMobile

The screenshot shows a REST client interface for a project named "AutoMobile". The active tab is "Signup". The request method is "POST" and the URL is "{{URL\_BASE}}/auth/signup". The request body is a JSON object with the following fields: "nombre\_usuario" (usuarioPrueba), "email" (usuarioPrueba@gmail.com), and "password" (prueba). The response status is "200 OK" with a response time of "141 ms" and a body size of "546 B". The response body is a JSON object with the following fields: "headers" ({}), "body" (Usuario usuarioPrueba ha sido registrado correctamente), "statusCode" (CREATED), and "statusCodeValue" (201).

HTTP AutoMobile / Signup

POST {{URL\_BASE}}/auth/signup

Send

Params Authorization Headers (8) Body Scripts Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "nombre_usuario": "usuarioPrueba",
3   "email": "usuarioPrueba@gmail.com",
4   "password": "prueba"
5 }
6
```

Body Cookies Headers (14) Test Results 200 OK 141 ms 546 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "headers": {},
3   "body": "Usuario usuarioPrueba ha sido registrado correctamente",
4   "statusCode": "CREATED",
5   "statusCodeValue": 201
6 }
```

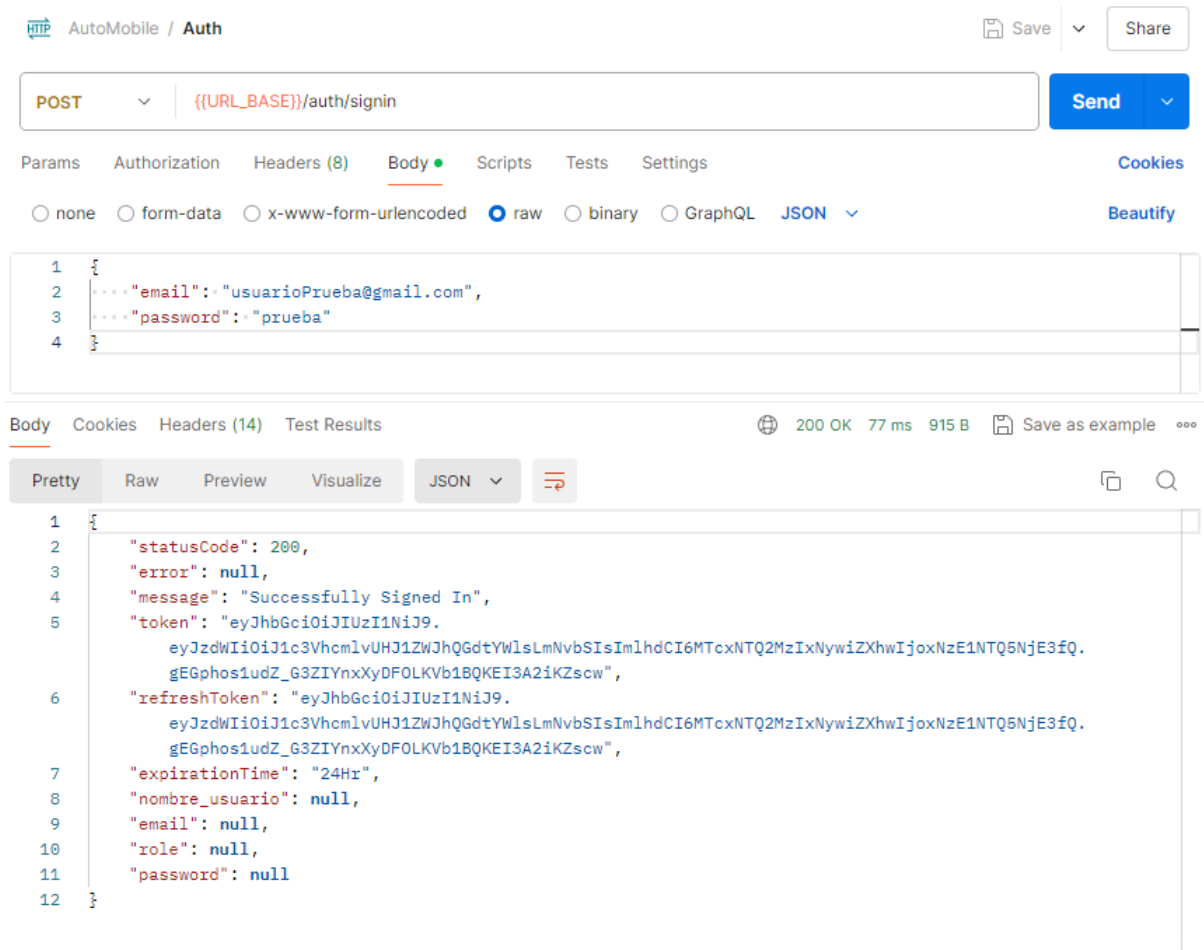
## Iniciar sesión

Ya podemos iniciar sesión con esta cuenta:

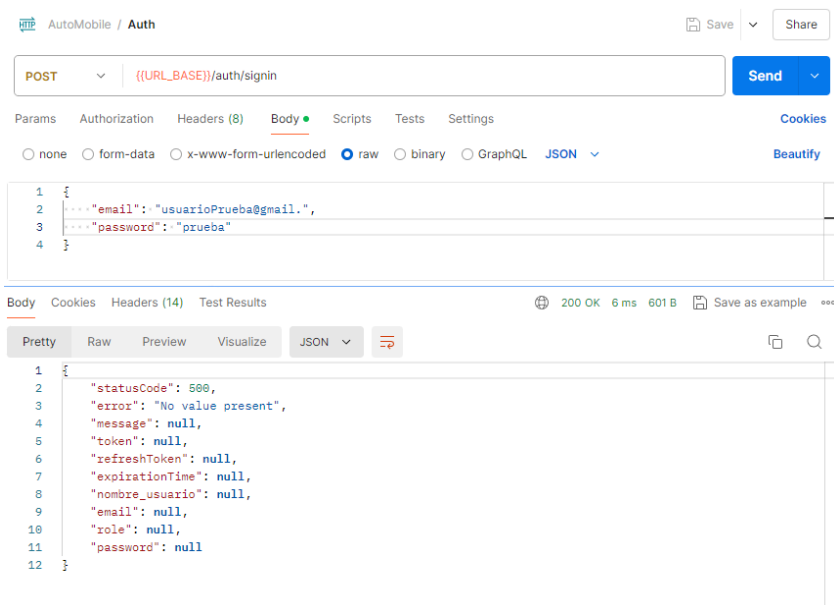
Nos dirigimos al signin y hacemos un POST con el json del usuario creado en el body:



## AutoMobile



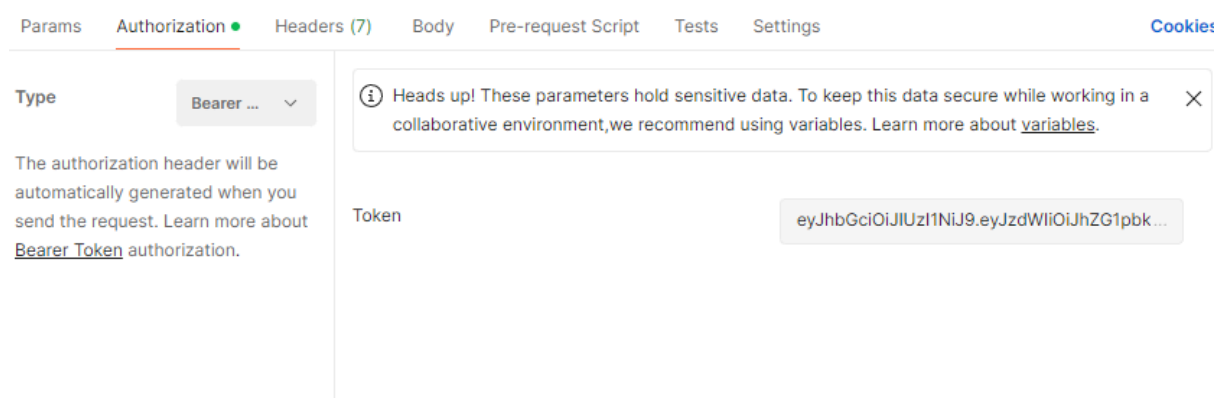
Nos devuelve el mensaje de que se ha iniciado sesión y nos da un token.



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/public/coche" y realizamos un GET para recuperar todos los coches. En este caso no necesitamos token por ser un endpoint público

## AutoMobile

The screenshot shows a REST client interface for a project named "AutoMobile / Coche". The selected endpoint is "Get coches" with a GET method and the URL template "{{URL\_BASE}}/public/coche". The "Authorization" tab is active, showing "No Auth". The response body is displayed in JSON format, showing details for a Toyota Corolla.

```
{
  "id": 1,
  "marca": "Toyota",
  "modelo": "Corolla",
  "imagen_principal": "https://cdn.motor1.com/images/mgl/P3gQyK/s1/corolla-grs-1.jpg",
  "precio": 20000,
  "anyo": 2018,
  "potencia": 120,
  "kilometraje": 50000,
  "combustible": "Gasolina",
  "consumo": "7.5L/100km",
  "tipoCambio": "Manual",
  "categoria": "Compacto",
  "tipoVehiculo": "Sedán",
  "traccion": "Delantera",
  "plazas": 5,
  "puertas": 4,
  "garantia": "1 año",
  "peso": 1200,
  "color": "Blanco",
```

## 2. Obtener un coche por id

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

## AutoMobile

HTTP AutoMobile / Coche / Obtener coche por id Save Share

GET {{URL\_BASE}}/public/coche/6 Send

Params Authorization Headers (6) Body Scripts Tests Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (14) Test Results 200 OK 7 ms 1.28 KB Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 6,
3   "marca": "Mercedes-Benz",
4   "modelo": "E-Class",
5   "imagen_principal": "https://assets.newatlas.com/dims4/default/6f61b09/2147483647/strip/true/crop/4032x2272+0+0/resize/2880x1623!/quality/90/?url=http%3A%2F%2Fnewatlas-brightspot.s3.amazonaws.com%2F%2Fda%2F65700dd24d619365b4270fa796ce%2F2021-mercedes-benz-e450-1.jpg",
6   "precio": 40000,
7   "anyo": 2021,
8   "potencia": 200,
9   "kilometraje": 15000,
10  "combustible": "Gasolina",
11  "consumo": "8.5L/100km",
12  "tipoCambio": "Automático",
13  "categoria": "Sedán",
14  "tipoVehiculo": "Sedán",
15  "traccion": "Trasera",
16  "plazas": 5,
17  "puertas": 4,
```

### 3. Crear coche

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

### 4. Actualizar coche

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

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

## 5. Borrar coche

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

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

The screenshot shows a REST client interface for a project named "AutoMobile". The current view is for a "DELETE" request to the endpoint `{{URL_BASE}}/admin/coche/9`. The "Authorization" tab is selected, showing a "Bearer Token" type. A warning message states: "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](#)." The token value is `eyJhbGciOiJIUzI1NiJ9.eyJzdWUiOiJhZG1pbk...`. The "Body" tab is active, displaying the response: `1 Coche eliminado correctamente`. The status bar at the bottom indicates a successful response: `200 OK` with a response time of `32 ms` and a size of `452 B`. The response is formatted as "Pretty" text.

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