



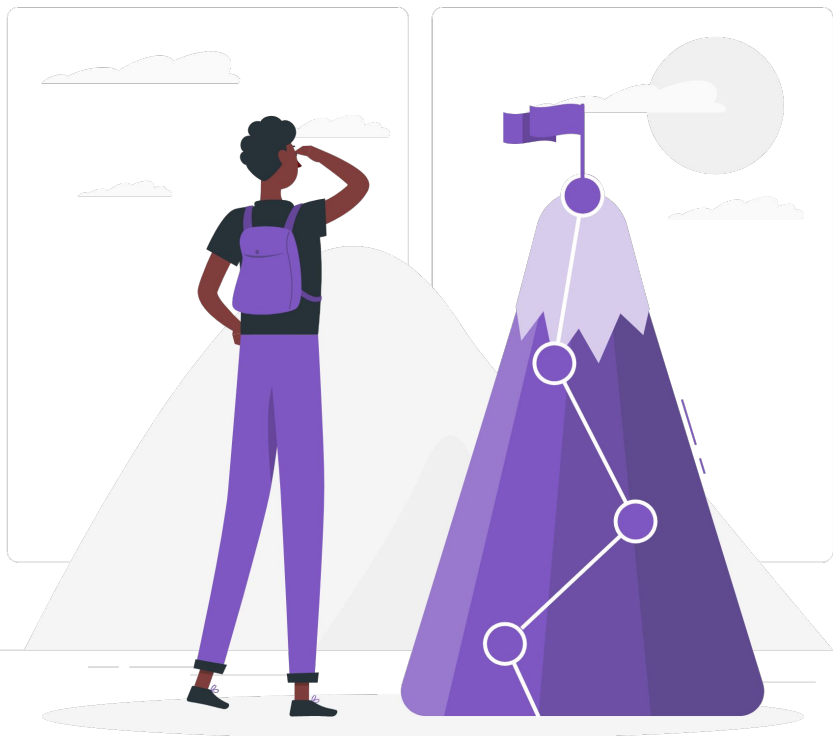
# Autenticación

**DEV.F.**  
DESARROLLAMOS(PERSONAS);

Elaborado por: César Guerra  
[www.cesarquerra.mx](http://www.cesarquerra.mx)



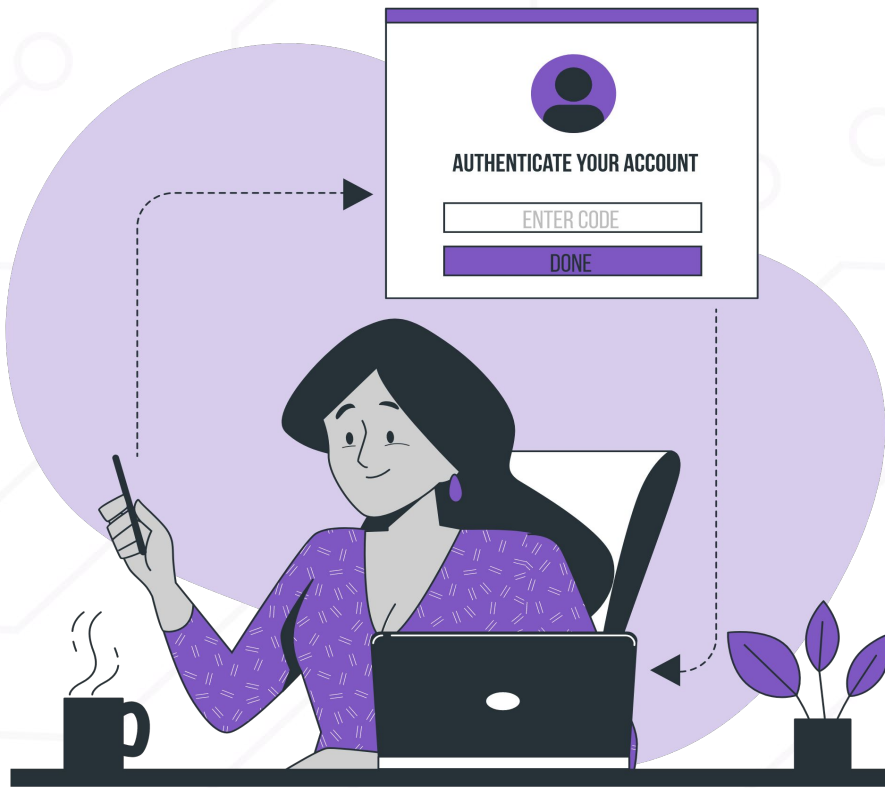
Buy me a coffee



## Objetivos

- Entender qué es la Autenticación y la Autorización.
- Entender el funcionamiento de JWT a nivel conceptual.
- Aprender a explorar una API y seguir el flujo de trabajo de autenticación con JWT
- Implementar un sistema de autenticación por JWT en React

# ¿Qué entendemos por autenticación?

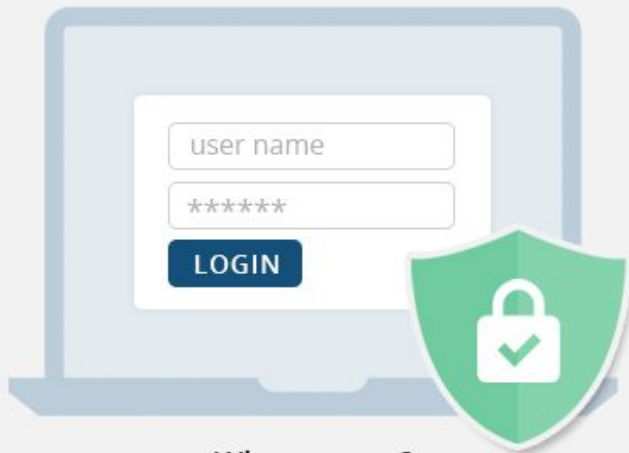


The logo consists of the text 'DEV.F.' in a bold, white, sans-serif font. The 'F' is stylized with three small squares at its base. The logo is centered within a dark blue diamond shape.

**DEV.F.**

**¿AUTENTICACIÓN  
Y AUTORIZACIÓN  
ES LO MISMO?**

## Authentication



**Who are you?**

Validate a system is accessing by the right person

La **autenticación** es la capacidad de demostrar que un usuario o una aplicación es realmente quién asegura ser.

## Authorization



**Are you allowed to do that?**

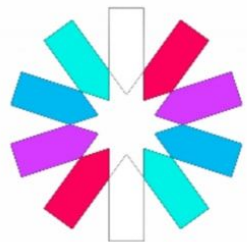
Check users' permissions to access data

La **autorización** son las acciones que tienes permitidas realizar de acuerdo quien demostraste ser en la autenticación.

# JWT (JSON Web Token)

**DEV.F**  
DESARROLLAMOS(PERSONAS);

dev

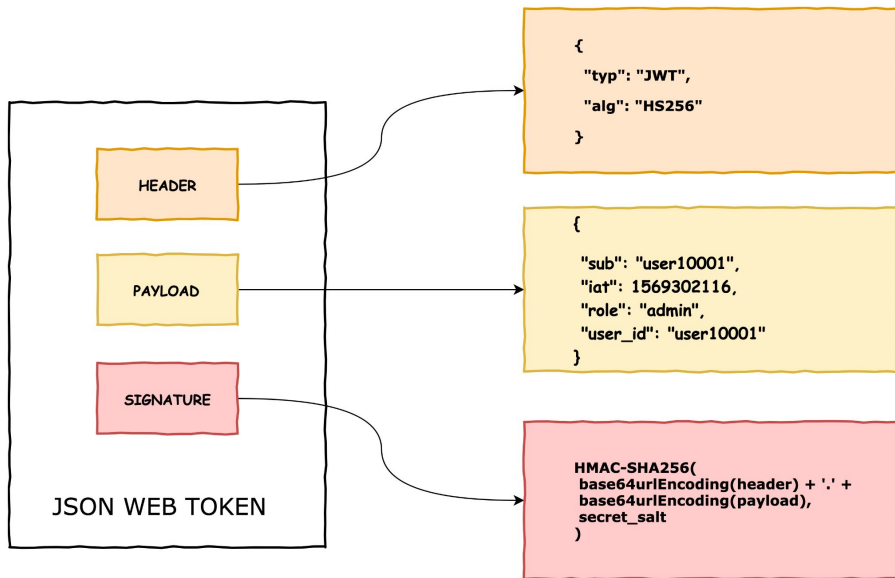


# JWT

## JWT

Un **JSON Web Token** estándar abierto (RFC-7519) basado en JSON para crear un token que sirva para enviar datos entre aplicaciones o servicios y garantizar que sean válidos y seguros.

Contiene toda la información importante sobre una entidad, lo que implica que no hace falta consultar una base de datos ni que la sesión tenga que guardarse en el servidor (sesión sin estado).



Ejemplo de un JWT:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6IjEiLCJ1c2VybmFtZSI6InNlcmdpb2R4YSJ9.Qu7rv5wqk6zGjiMU8ZixwvKQGBNW9hhj55DbSP50b1g
```

Header

Payload

Signature

# Estructura de un JWT

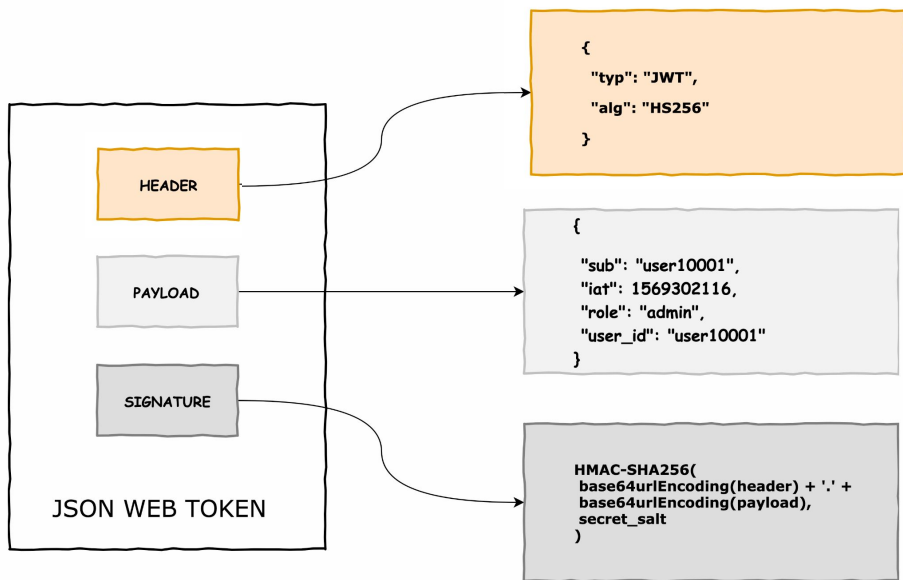
Los JWT tienen una estructura definida y estándar basada en tres partes:

1. header
2. payload
3. signature

Todas las partes se concatenan con un punto quedando de la siguiente manera:

```
header.payload.signature
```

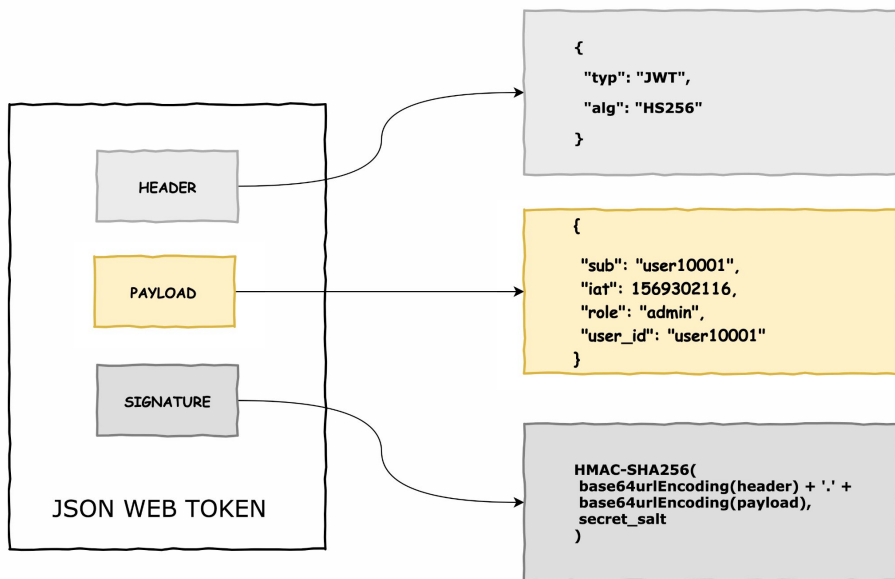




# JWT - Header

Json con Metadata del JWT el cual especifica el tipo de token y algoritmo de encriptación.

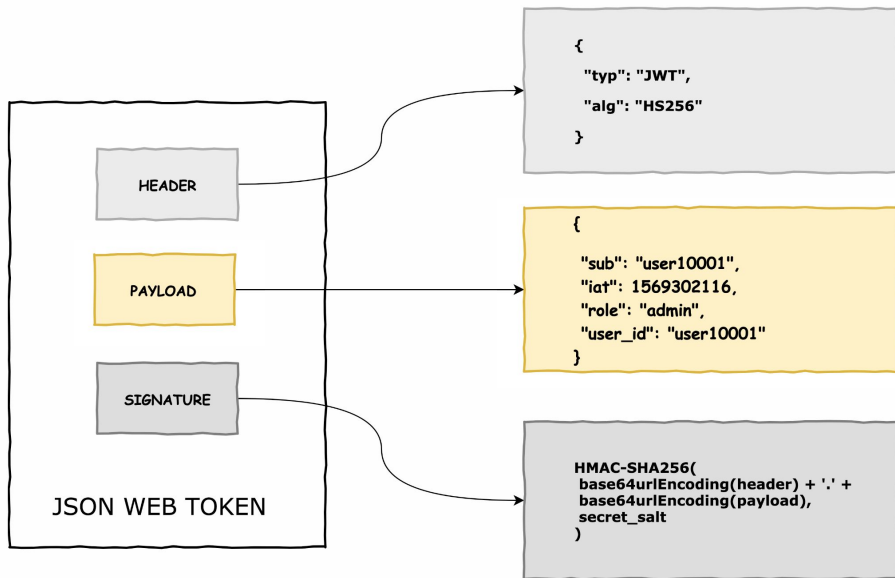
- **Tipo de token (typ):** Identifica el tipo de token. (siempre debe ser JWT)
- **Algoritmo de firmado y/o cifrado (alg):** Indica que tipo de algoritmo fue usado para firmar el token.



# JWT - Payload <sup>(1)</sup>

Es un JSON que contiene la información real que se transmitirá a la aplicación. Esta información se proporciona como pares clave/valor; las claves se denominan “claims” en JWT. Hay tres tipos diferentes de claims:

- **Claims registrados:** Son los que figuran en el IANA JSON Web Token Claim Register y cuyo propósito se establece en un estándar.
- **Claims públicos:** Son los que pueden ser definidos a voluntad por quienes usan JWTs.
- **Claims privados:** Si bien también son propiamente públicos, contienen información más concreta como datos de identificación de usuario o nombre de departamento.

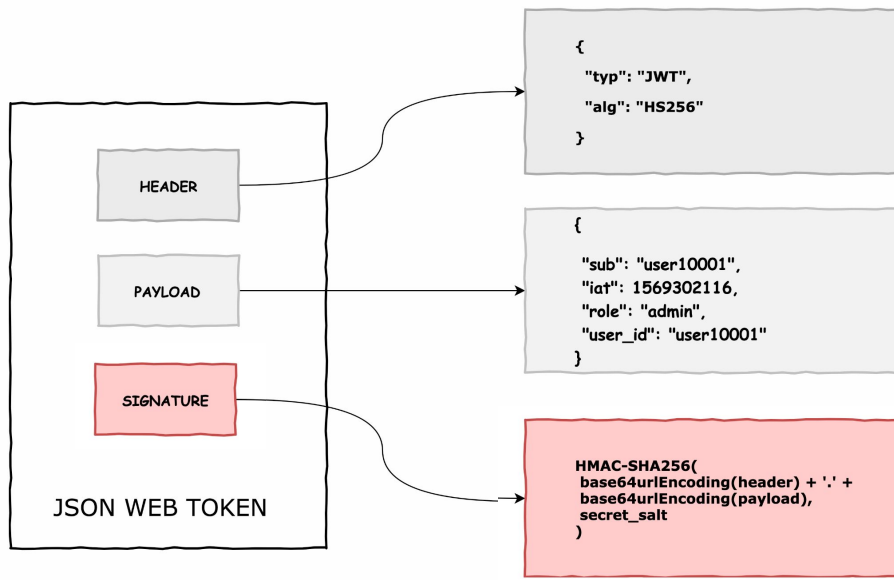


## JWT - Payload (2)

Entre los claims registrados en el IANA JSON Web Token con fines específicos encontramos:

- **Creador (iss):** Identifica a quien creo el JWT
- **Razón (sub):** Identifica la razón del JWT, se puede usar para limitar su uso a ciertos casos.
- **Tiempo de expiración (exp):** Una fecha que sirva para verificar si el JWT está vencido y obligar al usuario a volver a autenticarse.
- **No antes (nbf):** Indica desde qué momento se va a empezar a aceptar un JWT.
- **Creado (iat):** Indica cuando fue creado el JWT.

Nota: Todos los claims son opcionales, por lo que no es obligatorio utilizar todos los claims registrados. Y se recomienda usar el mínimo necesario de claims.



## JWT - Signature (2)

Es la parte que garantiza la integridad y autenticidad del token. Se genera encriptando el contenido del encabezado (header) y el payload, además de una clave secreta que solo conoce el servidor. **La firma se construye de tal forma que se puede verificar que el remitente es quien dice ser, y que el mensaje no se ha modificado por el camino.**

Es como los Pollos con Receta Secreta de KFC

Sabes que es pollo, que está empanizado crujiente, pero solo el coronel sabe cual es su receta secreta que da el sabor característico. Sin embargo nuestro paladar puede identificar si es pollo de KFC o no.

# Explorando JWT Debugger

<https://jwt.io/>

Algorithm

## Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_adQssw5c
```

## Decoded

EDIT THE PAYLOAD AND SECRET

### HEADER: ALGORITHM & TOKEN TYPE

Type of token {  
"alg": "HS256",  
"typ": "JWT"  
}

### PAYLOAD: DATA

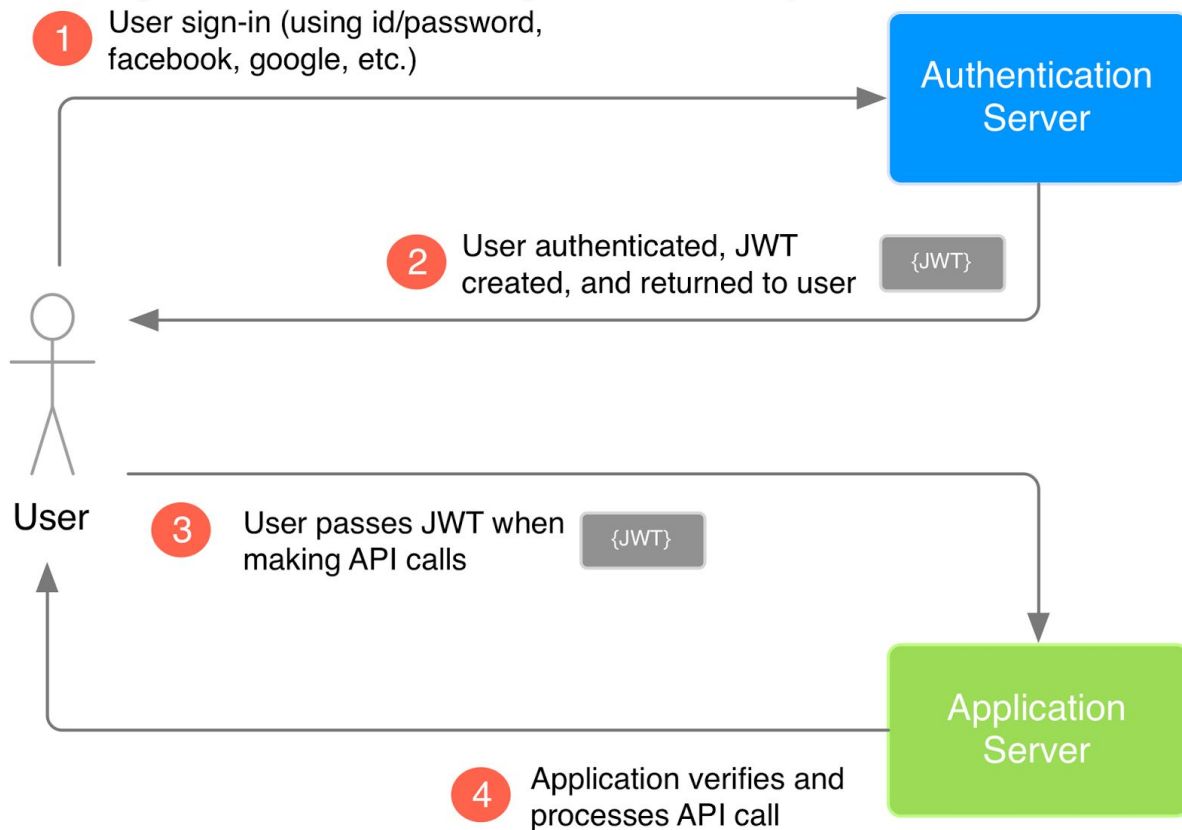
```
{  
  "sub": "1234567890",  
  "name": "John Doe",  
  "iat": 1516239022  
}
```

### VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  your-256-bit-secret  
) ☐ secret base64 encoded
```

Herramienta auxiliar: <https://www.base64encode.org/>

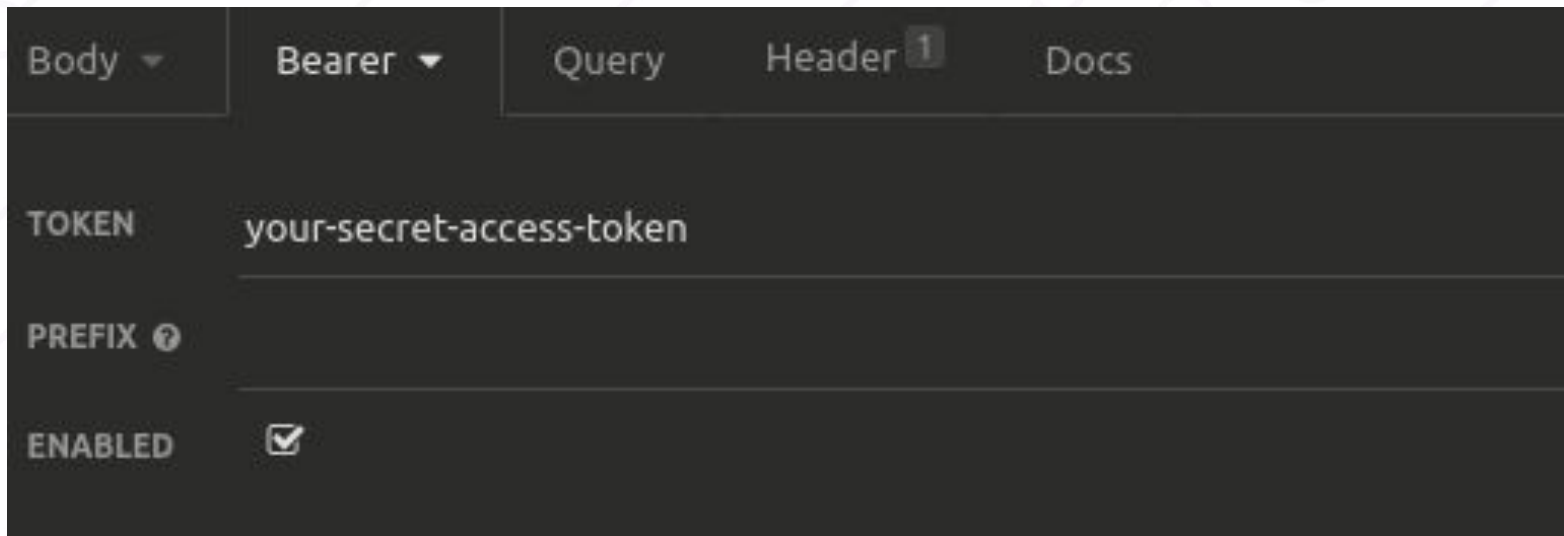
# Esquema de Authentication con JWT



# Demostración en Insomnia / Post Man

Utilizando la API del Proyecto de eCommerce:

<https://github.com/warderer/json-server-jwt>



The image shows a dark-themed interface for configuring Bearer authentication. At the top, there are tabs: 'Body', 'Bearer', 'Query', 'Header', and 'Docs'. The 'Bearer' tab is selected. Below the tabs, there are three rows of configuration:

Label	Value
TOKEN	your-secret-access-token
PREFIX ?	
ENABLED	<input checked="" type="checkbox"/>





# ¡ GRACIAS !



CésarGuerra



[www.cesarguerra.mx](http://www.cesarguerra.mx)