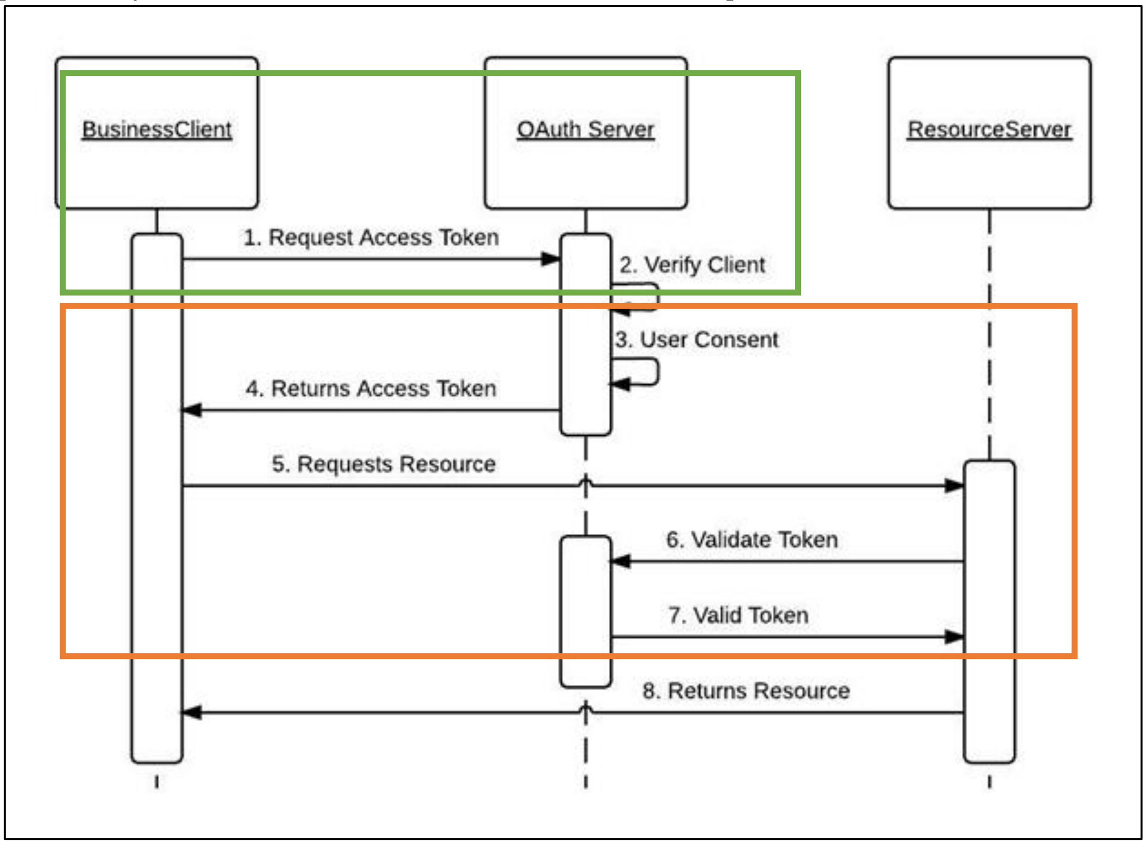
OAuth 2.0 in REST API Documentation

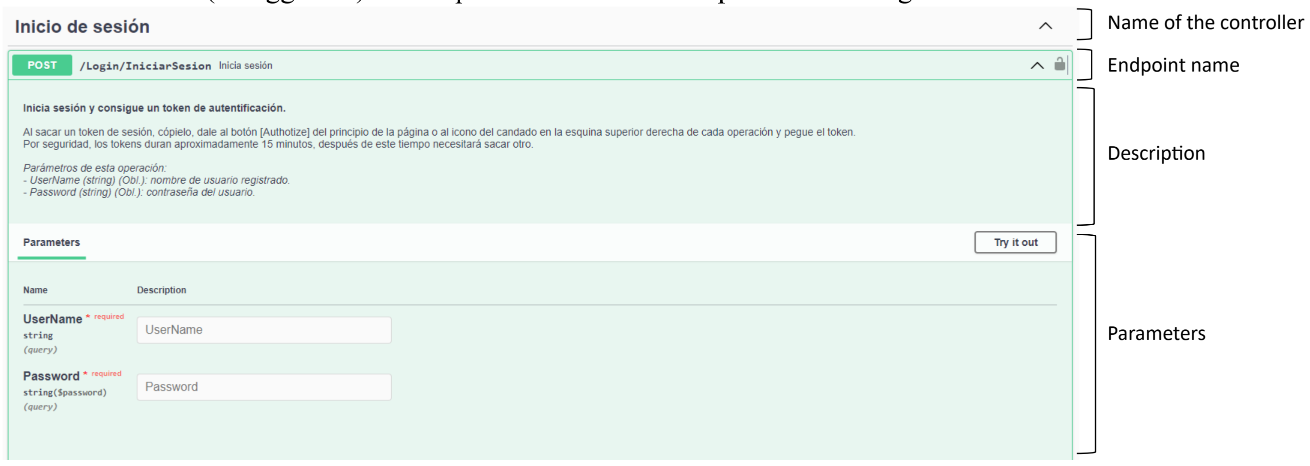
OAuth 2.0 is a token-based authentication protocol. It works by having the user request a token, which is verified by an authentication server where the different users are registered. If the validation is successful, the user is granted a token, which can then be used to request permission from the server that stores the resources to access protected resources.



OAuth 2.0 handles the process of validating the user, requesting the token, and protecting resources, while JWT Bearer takes care of creating and validating the token. JWT Bearer is a token format standard that signals OAuth 2.0 whether the token is valid and whether the resources should be unlocked.

These standards are implemented as follows:

There is a client (Swagger UI) that requests a username and password through a controller.



Inicio de sesión

POST `/Login/IniciarSesion` Inicia sesión

Inicia sesión y consigue un token de autentificación.

Al sacar un token de sesión, cópielo, dale al botón [Authotize] del principio de la página o al icono del candado en la esquina superior derecha de cada operación y pegue el token.
Por seguridad, los tokens duran aproximadamente 15 minutos, después de este tiempo necesitará sacar otro.

Parámetros de esta operación:
- UserName (string) (Obl.): nombre de usuario registrado.
- Password (string) (Obl.): contraseña del usuario.

Parameters

Name	Description
UserName * required string *(query)*	UserName
Password * required string($password) *(query)*	Password

(It corresponds to the green part of the image above)

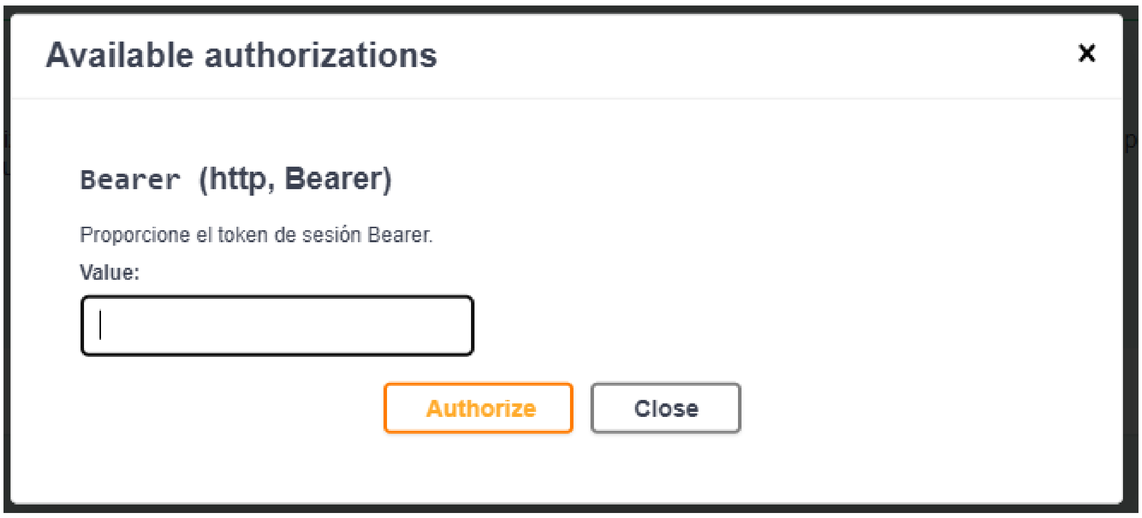
These credentials are validated on the authentication server, which in this case is a class within the same project (/Schemas/Login/User.cs) where various users are registered.

Each user has a hashed password and a specification of whether it is an administrator account or not.

At this point, an access token is generated for the user using JWT Bearer with the following parameters:

- Issuer Signing Key: verifies that the token is valid.
- Claim (Role): permissions granted to the user.
- Issuer: the client that requested the token.
- Audience: the service that will receive the token.
- Expiration date: for security reasons, tokens expire shortly after being issued (~15 minutes).

Once the user has their token after logging in, OAuth 2.0 (assisted by Swagger UI) requests the Bearer access token to allow the use of protected resources.


Available authorizations

Bearer (http, Bearer)

Proporcione el token de sesión Bearer.

Value:

(Corresponds to the Orange parto f the image)

The user then sends their newly obtained token to the server where the resources are stored so that it can be validated using the JWT Bearer token format.

If the token is valid (has the same signature, contains the necessary permissions, and the expiration date has not been reached), the resource server grants access to the requested resource.

Currently, the resources are secured by three levels of security:

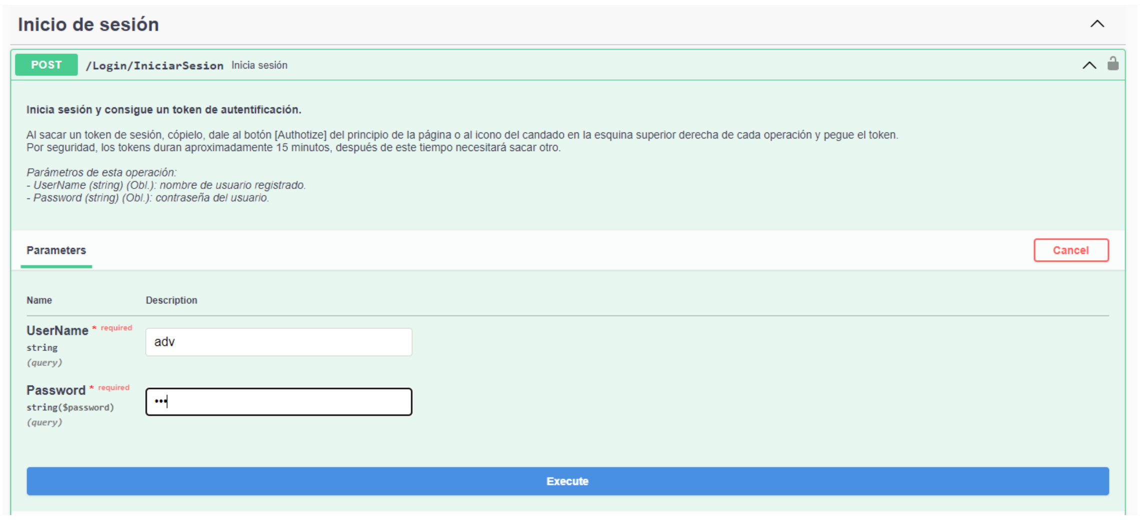
- Anonymous: no token is required to access the resource (basic functions).
- Normal: an access token is required after logging in to access the resource (can view but not modify the database).
- Administrator: an access token is required after logging in as an administrator to access the resource (has permission to view, add, modify, and delete records from the database).

If the token is invalid, the operation will return status codes 401 or 403, indicating that either an access token or an administrator token is required, respectively.

Step-by-Step Guide

Notify the person in charge of API REST user management to register a user for you or provide you with the credentials of an existing user.

With these credentials, go to the 'Login' controller and enter your username and password.



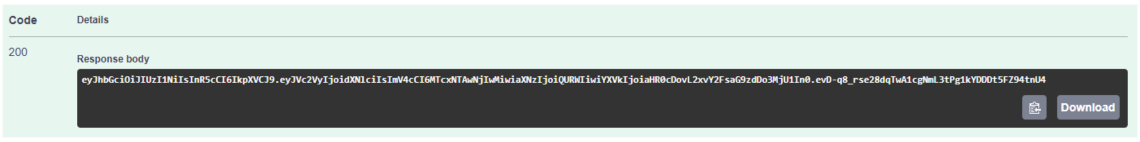
Inicia sesión y consigue un token de autentificación.

Al sacar un token de sesión, cópielo, dale al botón [Authotize] del principio de la página o al icono del candado en la esquina superior derecha de cada operación y pegue el token.
Por seguridad, los tokens duran aproximadamente 15 minutos, después de este tiempo necesitará sacar otro.

Parámetros de esta operación:
- UserName (string) (Obl.): nombre de usuario registrado.
- Password (string) (Obl.): contraseña del usuario.

Name	Description
UserName * required string (query)	adv
Password * required string($password) (query)	•••

If the credentials are correct, the resource will return a token in the format ‘eyJ’ followed by a random string of characters.



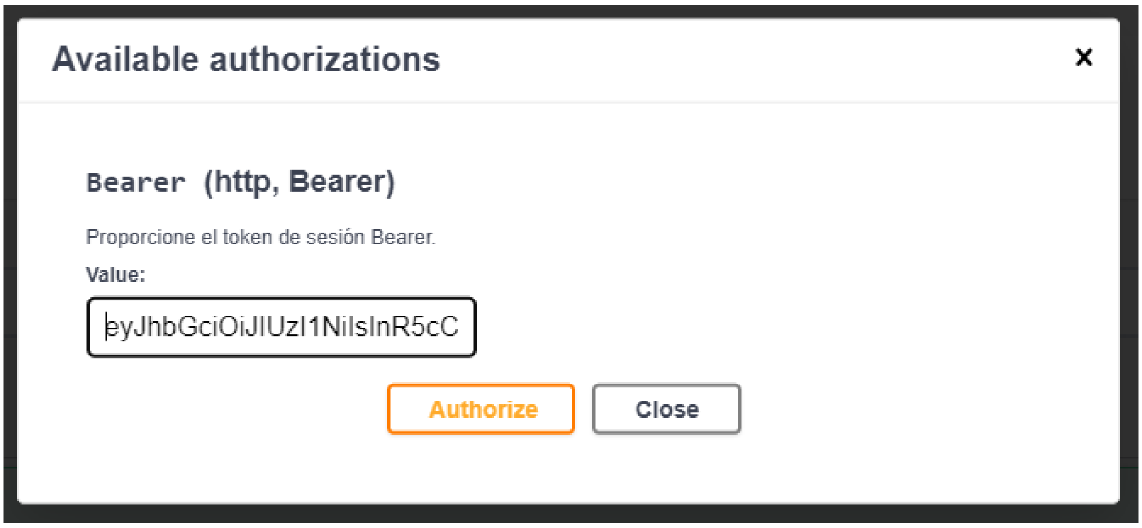
Code	Details
200	Response body

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJVc2VyIjoidXNlciIsImV4cCI6MTcxNTAwNjIwMiwiaXNzIjoiQURWIiwiYXVkIjoiaHR0cDovL2xvY2FsaG9zdDo3MjU1In0.evD-q8_rse28dqTwA1cgNmL3tPg1kYDDDt5FZ94tnU4
```

Respuesta

Press the [Authorize] button at the top of the page or on the open padlock icons next to each resource.

Paste the token you obtained into the requested field and press Authorize.



Bearer (http, Bearer)

Proporcione el token de sesión Bearer.

Value: eyJhbGciOiJIUzI1NiIsInR5cC

The padlocks will now be closed, and you will be able to access protected resources, provided that the resource is at the same or a lower security level than your account, which can be verified by reviewing the description of each resource or by checking the returned status code.