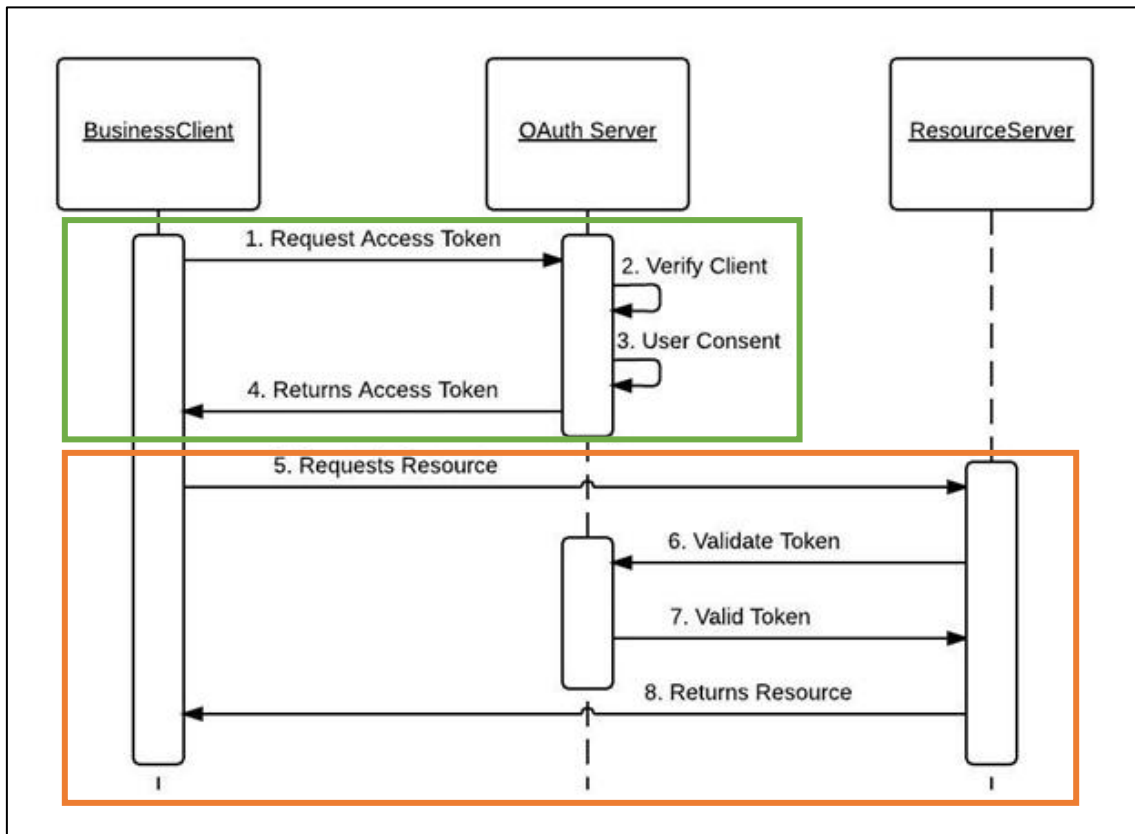


Documentación sobre OAuth 2.0 en REST API

OAuth 2.0 es un protocolo de autenticación por tokens, este se basa en que el usuario solicite un token, solicitud que verifica un servidor de autenticación donde están registrados los diferentes usuarios, en caso de que la validación sea exitosa, se le dará el token al usuario, que podrá usar para solicitarle al servidor en el que se almacenan los recursos que le de permisos para acceder a recursos protegidos.



La parte de validar al usuario, solicitar el token y mantener protegidos los recursos se encarga OAuth 2.0, mientras que la creación del token y validación de este se encarga *JWT Bearer*, un estándar de formato de tokens que le da la “señal” a OAuth 2.0 de si es válido o no y si debe desbloquear los recursos.

Estos estándares están implementados de la siguiente manera:

Hay un cliente (Swagger UI) que pide un usuario y una contraseña mediante un *controlador*.

Inicio de sesión

POST /LogIn/IniciarSesión Inicia sesión

Inicia sesión y consigue un token de autenticación.

Al sacar un token de sesión, cópielo, dale al botón [Authorize] 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

Nombre del controlador

Nombre del recurso

Descripción

Parámetros

(Correspondiente a la parte marcada de verde en el esquema)

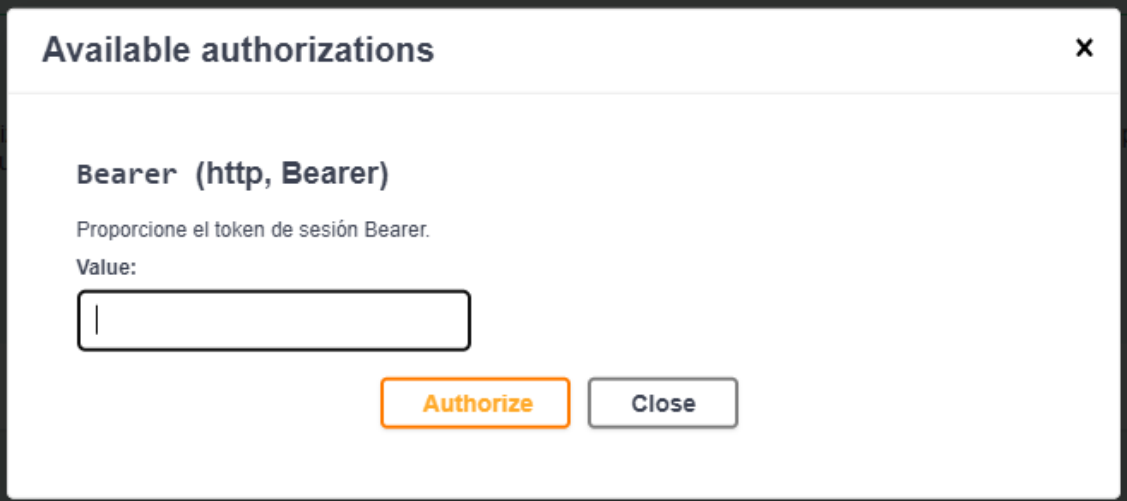
Estos datos se validan en el servidor de autenticación, en este caso se trata de una clase dentro del mismo proyecto (*/Schemas/Login/User.cs*) en las que hay diferentes usuarios registrados.

Cada usuario tiene su nombre, contraseña *hashseada* y especificación de si se trata de una cuenta de administrador o no.

Es entonces cuando se crea el token de acceso para el usuario usando *JWT Bearer* con los siguientes parámetros:

- Firma (Issuer Signing Key): verifica que el token es válido.
- Rol (Claim): los permisos otorgados al usuario.
- Emisor (Issuer): cliente que ha solicitado el token.
- Receptor (Audience): servicio que va a recibir el token.
- Fecha de caducidad: por seguridad, los tokens caducan al poco tiempo (~15 minutos).

Ahora que el usuario ya tiene su token al haber iniciado sesión, OAuth 2.0 (asistido por Swagger UI) solicita el token tipo *Bearer* de acceso para permitir el uso de recursos protegidos.



(Correspondiente a la parte marcada de naranja en el esquema)

El usuario entonces envía su recién obtenido token al servidor donde se almacenan los recursos para que lo valide usando como esquema el formato de los tokens JWT Bearer.

Si el token recibido es válido, (tiene la misma firma, tiene los permisos necesarios y la fecha de caducidad aún no se ha alcanzado) el servidor de recursos le da acceso al recurso solicitado.

Actualmente los recursos tienen 3 niveles de seguridad:

- Anónimos: no se requiere de un token para acceder al recurso (funciones básicas).
- Normal: se requiere un token de acceso al iniciar sesión para acceder al recurso (puede ver pero no tocar la base de datos).
- Administrador: se requiere un token de acceso al iniciar sesión como administrador para acceder al recurso (tiene permiso para ver, añadir, modificar y eliminar registros de la base datos).

En caso que el token no sea válido, la operación devolverá los códigos de resultado 401 o 403, significando que necesitan un token de acceso o un token de administrador respectivamente.

Guía paso a paso

Informe al encargado de la administración de usuarios de la API REST que le registre un usuario o le dé las credenciales de uno ya existente.

Con estos datos, vaya al *controlador* ‘Inicio de sesión’ e ingrese su nombre de usuario y contraseña.

Inicio de sesión

POST /Login/IniciarSesion Inicia sesión

Inicia sesión y consigue un token de autenticación.

Al sacar un token de sesión, cópielo, dale al botón [Authorize] 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) (Obj.): nombre de usuario registrado.
- Password (string) (Obj.): contraseña del usuario.

Parameters Cancel

Name	Description
Username * required string (query)	<input type="text" value="adv"/>
Password * required string(\$password) (query)	<input type="password" value="***"/>

Execute

Si estos son correctos, el recurso le devolverá un token con formato ‘eyJ’ seguido de una cadena aleatoria de caracteres.

Code **Details**

200

Response body

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ3c2VyIjoiaXNlLnR5cC...
```

Download

Respuesta

Presione el botón [Authorize] en el principio de la página o en los iconos de candados abiertos que hay en cada recurso.

Pegue el token conseguido en el campo que le solicitan y presione Authorize.

Available authorizations ×

Bearer (http, Bearer)

Proporcione el token de sesión Bearer.

Value:

Authorize **Close**

Los candados estarán ahora cerrados y podrá acceder a los recursos protegidos, siempre y cuando el recurso se encuentre en el mismo o inferior nivel de seguridad que su cuenta, cosa que puede verificar revisando la descripción de cada recurso o viendo el código de resultado devuelto.