

INSTRUCCIONES

- Se debe acceder a la URL <https://testgoalsystems.azurewebsites.net/swagger/index.html>.
- Se debe Obtener un token.

Podemos ver los usuarios con acceso disponibles en el archivo del código fuente TestGoalSystems.Identity.Configurations. Se han creado con migrations. También se puede registrar uno nuevo con el servicio Register. La contraseña debe ser alfanumérico, con al menos una mayúscula y un carácter especial. Ejemplo:

```
{
  "name": "Luis",
  "surname": "Garcia",
  "email": "luis@gmail.com",
  "username": "Luis",
  "password": "Luis123$"
}

builder.HasData(
    new ApplicationUser
    {
        Id = "462d5cc0-3ff8-4e1f-a8c0-5105bce99b6e",
        Email = "eugenio.malfeito@gmail.com",
        NormalizedEmail = "eugenio.malfeito@gmail.com",
        Name = "Eugenio",
        Surname = "Malfeito",
        UserName = "emalfeito",
        NormalizedUserName = "emalfeito",
        PasswordHash = hasher.HashPassword(null, "euge5656$"),
        EmailConfirmed = true
    },
    new ApplicationUser
    {
        Id = "b0a52ab6-e5ba-4def-ba8b-0aaa1a38c3ee",
        Email = "eugenio.empleo@gmail.com",
        NormalizedEmail = "eugenio.empleo@gmail.com",
        Name = "Eugenio",
        Surname = "Malfeito",
        UserName = "emalfeitoempleo",
        NormalizedUserName = "emalfeitoempleo",
        PasswordHash = hasher.HashPassword(null, "euge5656$"),
        EmailConfirmed = true
    }
);
```

Se obtiene con el email y password.

Account

POST

/api/v1/Account/Login

Parameters

No parameters

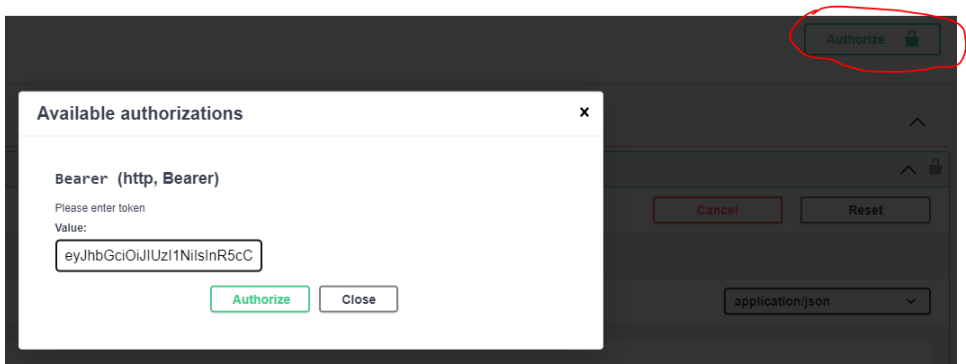
Request body

```
{  "email": "eugenio.malfeito@gmail.com",  "password": "euge5656$"}

```

[illegible]

- Introducimos el Token en el Authorize.



- El controlador Inventory está protegido. Con lo cual hay que introducir el Token para que funcionen los servicios.
- Se compone de 4 servicios:

Inventory	
GET	/api/v1/Inventory/{username}
POST	/api/v1/Inventory
PUT	/api/v1/Inventory
DELETE	/api/v1/Inventory/{id}

- 1- Obtiene los inventarios filtrados por su propietario.
- 2- Creación de un nuevo inventario.
- 3- Actualización de un inventario.
- 4- Eliminación de un inventario.

NOTA: En la creación y actualización se ha creado un evento para avisar que el inventario ha caducado. Lo mismo para avisar de la eliminación de un inventario. Se ha realizado mediante el ILogger. Este evento lo podemos ver en la consola del Visual Studio.

```
"TestGoalSystemsAPI.exe" (CoreCLR: clrhost): 'C:\Program Files\dotnet\shared\Microsoft.NETCore.App\6.0.0\System.Net.NetworkInformation.dll' cargado. Se omitió la carga de s
Microsoft.EntityFrameworkCore.Infrastructure: Information: Entity Framework Core 6.0.4 initialized 'InventoryDbContext' using provider 'Microsoft.EntityFrameworkCore.SqlSer
El subproceso 0x337c terminó con código 0 (0x0).
Microsoft.EntityFrameworkCore.Database.Command: Information: Executed DbCommand (70ms) [Parameters=[@p0='?' (Size = 4000), @p1='?' (DbType = DateTime2), @p2='?' (DbType = D
SET NOCOUNT ON;
INSERT INTO [Inventory] ([CreatedBy], [CreatedDate], [Expiration], [InventoryTypeId], [LastModifiedBy], [LastModifiedDate], [Name])
VALUES (@p0, @p1, @p2, @p3, @p4, @p5, @p6);
SELECT [Id]
FROM [Inventory]
WHERE @@ROWCOUNT = 1 AND [Id] = scope_identity();
TestGoalSystems.Application.Features.Inventories.Commands.CreateInventory.CreateInventoryCommandHandler: Information: Inventory 10 was created successfully.
TestGoalSystems.Application.Features.Inventories.Commands.CreateInventory.CreateInventoryCommandHandler: Information: Inventory 10 has expired.
El subproceso 0x4cb4 terminó con código 0 (0x0).
El subproceso 0x27bc terminó con código 0 (0x0).
El subproceso 0x1b38 terminó con código 0 (0x0).
```

Create

El inventoryTypeId debe ser 1 ya que se ha creado en la base de datos este tipo de inventario relacionado con el inventario.

```
{
  "name": "Prueba",
  "expiration": "2022-04-28T15:42:53.026Z",
  "inventoryTypeId": 1
}
```

En caso de que la respuesta sea OK(200) nos devuelve el identificador del inventario creado.

Se ha creado una validación con FluentValidator para que no se pueda dejar vacío el nombre. En caso de hacerlo aparece una respuesta indicando el error:

```
{
  "Details": "{ \"Name\": \"{Name} cannot be blank.\" }",
  "StatusCode": 0,
  "Message": ""
}
```

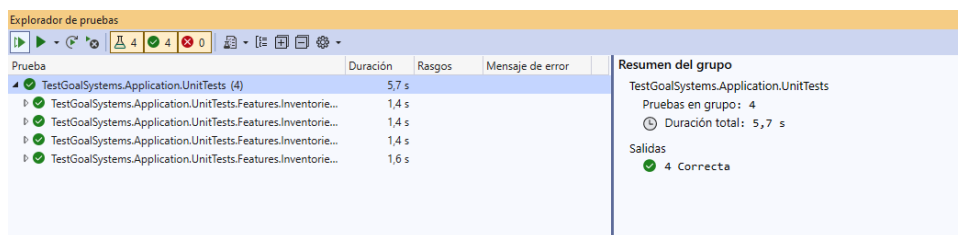
Delete

<https://testgoalsystems.azurewebsites.net/api/v1/Inventory/16>

La respuesta en caso de que la eliminación sea correcta será un 204 (No Content).

Importante: Si se quiere probar desde el Visual Studio me tenéis que informar de la IP Pública para darle acceso a la base de datos en Azure.

Tests



Prueba	Duración	Rasgos	Mensaje de error
TestGoalSystems.Application.UnitTests (4)	5,7 s		
TestGoalSystems.Application.UnitTests.Features.Inventory...	1,4 s		
TestGoalSystems.Application.UnitTests.Features.Inventory...	1,4 s		
TestGoalSystems.Application.UnitTests.Features.Inventory...	1,4 s		
TestGoalSystems.Application.UnitTests.Features.Inventory...	1,6 s		

Resumen del grupo
TestGoalSystems.Application.UnitTests
Pruebas en grupo: 4
Duración total: 5,7 s
Salidas
4 Correcta