

Manual de Usuario - RealEstate API (.NET 8)

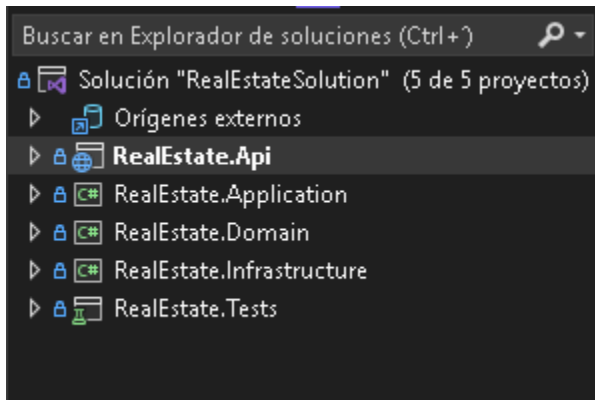
1. Introducción

Este documento describe la configuración, ejecución, autenticación, uso de endpoints y pruebas de la aplicación RealEstate API desarrollada en .NET 8. El objetivo es proporcionar al evaluador una guía completa para levantar la solución, probar cada funcionalidad y comprender las validaciones y medidas de seguridad implementadas.

2. Arquitectura del Proyecto

El proyecto está organizado en capas siguiendo buenas prácticas de arquitectura limpia:

- RealEstate.Api: controladores, Swagger, autenticación JWT, carpeta de imágenes.
- RealEstate.Application: DTOs, AutoMapper, servicios y lógica de negocio.
- RealEstate.Infrastructure: DbContext, repositorios EF Core, inyección de dependencias.
- RealEstate.Domain: entidades y contratos.
- RealEstate.Tests: pruebas unitarias y de integración con NUnit.



3. Requisitos Previos

- .NET SDK 8.0
- SQL Server (local o remoto)
- Visual Studio 2022 o Visual Studio Code
- EF Core Tools: `dotnet tool install --global dotnet-ef`

4. Configuración del Proyecto

En el archivo appsettings.json de RealEstate.Api se encuentran las configuraciones clave:

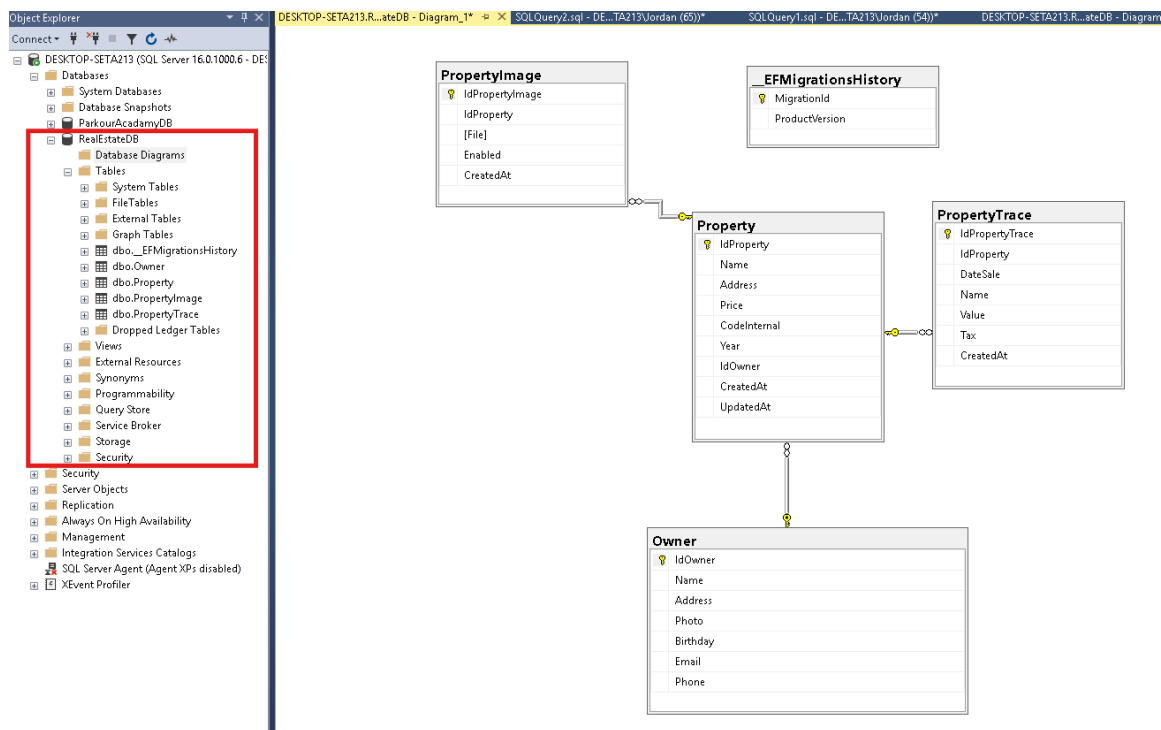
Cadena de conexión (SQL Server) y JWT (seguridad)

```
appsettings.json  Program.cs
Esquema: https://www.schemastore.org/appsettings.json
1 {
2   "ConnectionStrings": {
3     "DefaultConnection": "Server=DESKTOP-SETA213;Database=RealEstateDB;Trusted_Connection=True;TrustServerCertificate=True;"
4   },
5   "Logging": {
6     "LogLevel": {
7       "Default": "Information",
8       "Microsoft": "Warning",
9       "Microsoft.Hosting.Lifetime": "Information"
10    },
11  },
12  "Jwt": {
13    "Key": "TokenSecretoDeMuchosCaracteres12345!!",
14    "Issuer": "RealEstateApi",
15    "Audience": "RealEstateApiUsers"
16  },
17  "AllowedHosts": "*"
18 }
19
```

5. Base de Datos

El repositorio incluye un archivo RealEstateDB.bak para restaurar la base con datos de prueba.

Restaurar DB: Restaurar el backup .bak en SQL Server.



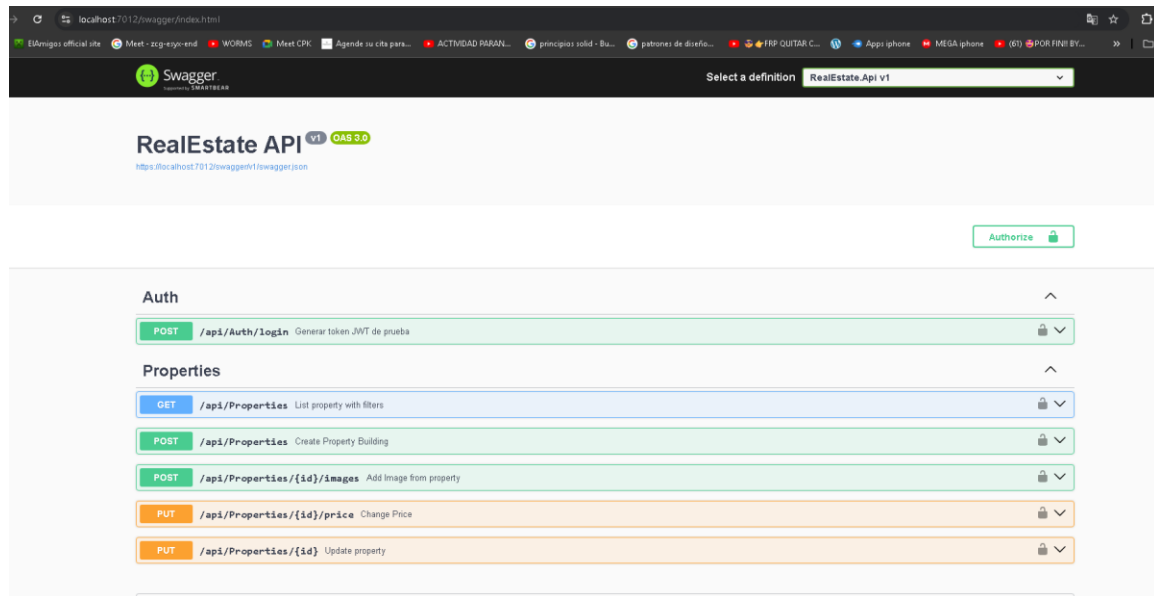
Una vez restaurada o creada, la base contiene datos en las tablas Owner, Property, PropertyImage y PropertyTrace.

6. Ejecución de la Aplicación

Para ejecutar el proyecto:

- En Visual Studio: establecer RealEstate.Api como proyecto de inicio y presionar F5.
- Por CLI: ubicarse en la carpeta RealEstate.Api y ejecutar dotnet run.

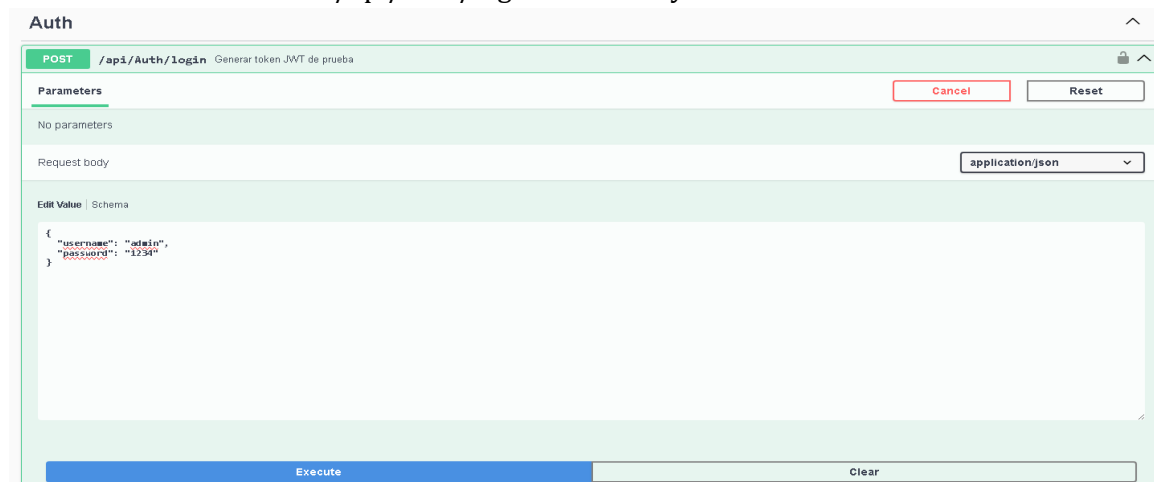
La API expone Swagger en `https://localhost:{puerto}/swagger`.



7. Autenticación JWT

Todos los endpoints de PropertiesController requieren autenticación.

1. Generar token en POST /api/auth/login con el body:



Cada endpoint valida datos y responde con los códigos adecuados (200, 201, 400, 401, 404, 409).

GET /api/properties

GET /api/Properties List property with filters

Parameters

Cancel

Name	Description
PropertyId <small>integer(\$int32) (query)</small>	<input type="text" value="PropertyId"/>
City <small>string (query)</small>	<input type="text" value="Cartagena"/>
State <small>string (query)</small>	<input type="text" value="State"/>
MinPrice <small>number(\$double) (query)</small>	<input type="text" value="MinPrice"/>
MaxPrice <small>number(\$double) (query)</small>	<input type="text" value="MaxPrice"/>
Year <small>integer(\$int32) (query)</small>	<input type="text" value="2013"/>

Execute

Clear

Responses

Request URL

https://localhost:7012/api/Properties?City=Cartagena&Year=2013

Server response

Code

Details

200

Response body

```
{
  "total": 1,
  "items": [
    {
      "id": 10,
      "name": "Casa Jardines",
      "address": "Cra 33 #70-15, Cartagena",
      "price": 380000000,
      "year": 2013,
      "ownerName": "Andrés Ramírez",
      "images": [
        {
          "id": 19,
          "file": "house19.jpg",
          "url": "https://localhost:7012/images/house19.jpg",
          "enabled": true,
          "propertyId": 10
        },
        {
          "id": 20,
          "file": "house20.jpg",
          "url": "https://localhost:7012/images/house20.jpg",
          "enabled": true,
          "propertyId": 10
        },
        {
          "id": 36,
          "file": "house36.jpg",
          "url": "https://localhost:7012/images/house36.jpg",
          "enabled": true,

```

POST /api/properties

POST

/api/Properties

Create Property Building

Parameters

Cancel

Reset

No parameters

Request body

application/json

Edit Value

Schema

```
{
  "name": "Propiedad Prueba",
  "address": "direccion prueba",
  "price": 2800000,
  "codeInternal": "PROP00312",
  "year": 2019,
  "ownerId": 2
}
```

Execute

Request URL	
https://localhost:7012/api/Properties	
Server response	
Code	Details
201	<div><div>Response body</div><div><pre>{ "id": 13, "name": "Propiedad Prueba", "address": "direccion prueba", "price": 2800000, "year": 2019, "ownerName": null, "images": [], "traces": [] }</pre></div></div>

POST /api/properties/{id}/images

POST

/api/Properties/{id}/images

Add Image from property

Parameters

Name	Description
id ★ required	
integer(\$int32)	3
(path)	

Request body

File

string(\$binary)

Seleccionar archivo

house52.jpg

☐ Send empty value

Execute

Clear

Request URL

https://localhost:7012/api/Properties/3/images

Server response

Code	Details
200	<div>Response body<div><pre>{ "message": "Imagen cargada correctamente", "fileName": "house52.jpg"}</pre></div><div>Response headers<div><pre>content-type: application/json; charset=utf-8date: Wed, 03 Sep 2025 07:02:58 GMTserver: Kestrel</pre></div></div></div>

PUT /api/properties/{id}/Price

PUT

/api/Properties/{id}/price

Change Price

Parameters

Name	Description
id * required integer(\$int32) (path)	<div>1</div>

Request body

Edit Value

Schema

347000000

Request URL	
https://localhost:7012/api/Properties/1/price	
Server response	
Code	Details
204	<div>Response headers</div> <div>date: Wed, 03 Sep 2025 07:05:07 GMT server: Kestrel</div>
Responses	
Code	Description
204	No Content

PUT /api/properties/{id}

PUT

/api/Properties/{id}

Update property

Parameters

Name	Description
id * required integer(\$int32) (path)	<div>4</div>

Request body

Edit Value | Schema

```
{  
  "name": "Propiedad Actualizada",  
  "address": "Direccion actualizada prueba",  
  "price": 4589000000,  
  "year": 2025  
}
```

Request URL

https://localhost:7012/api/Properties/4

Server response

Code	Details
204	<div><div>Response headers</div><div><div>date: Wed, 03 Sep 2025 07:07:38 GMT</div><div>server: Kestrel</div></div></div>

Responses

Code	Description
------	-------------

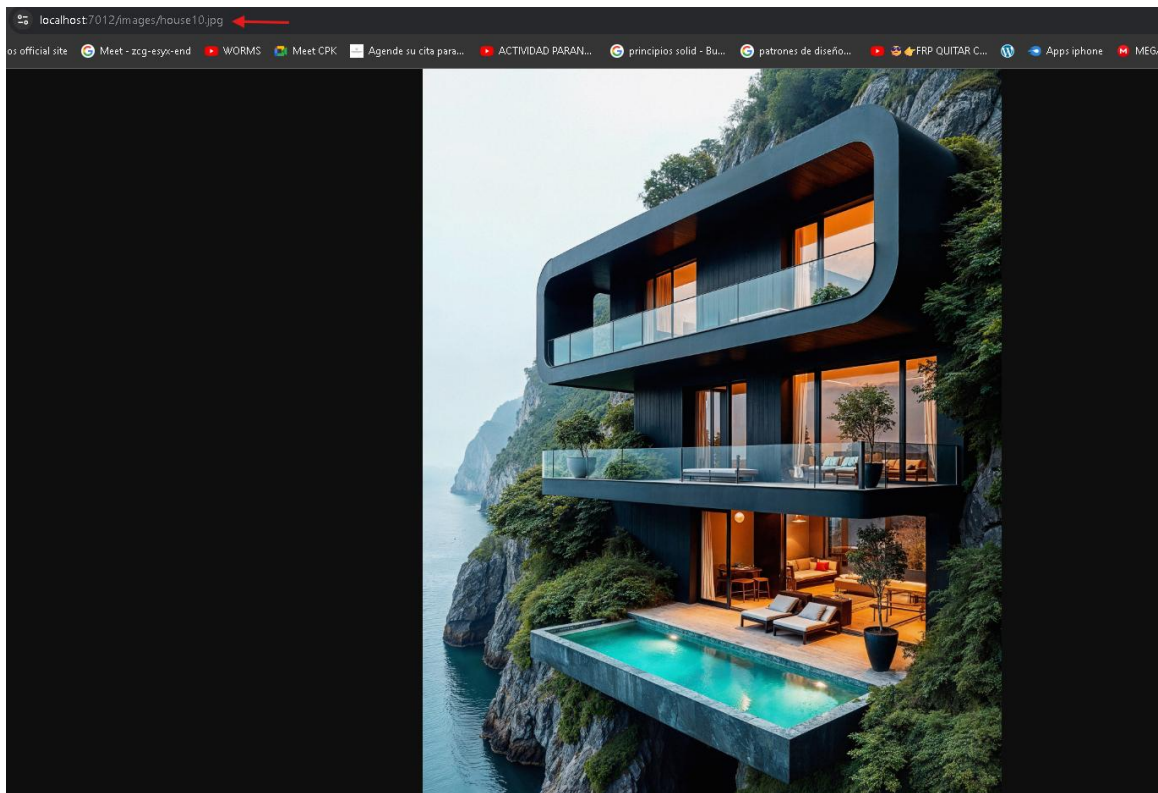
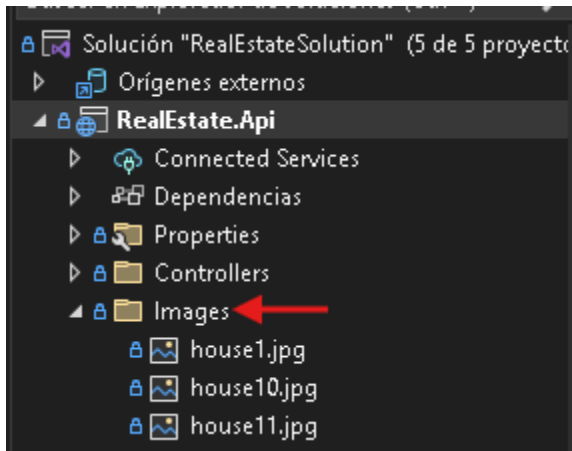
]

9. Manejo de Imágenes

La API expone la carpeta 'images' en la raíz de RealEstate.Api. Las imágenes subidas se guardan allí y son accesibles públicamente desde <https://localhost:{puerto}/images/{archivo}>.

Validaciones:

- Solo se permiten extensiones .jpg y .png.
- Se valida que la propiedad exista.
- Se valida que no haya nombres duplicados de imágenes para la misma propiedad.
- Existe un índice único (PropertyId, File) en BD para reforzar la restricción.



10. Pruebas con NUnit

El proyecto RealEstate.Tests incluye:

- Pruebas unitarias de servicios, usando TransactionScope para rollback automático.
- Pruebas de integración con WebApplicationFactory, probando endpoints reales con JWT.

Ejecución:

- Visual Studio: Test Explorer → Run All.
- CLI: `dotnet test RealEstate.Tests/RealEstate.Tests.csproj`

Todas las pruebas pasaron exitosamente.

