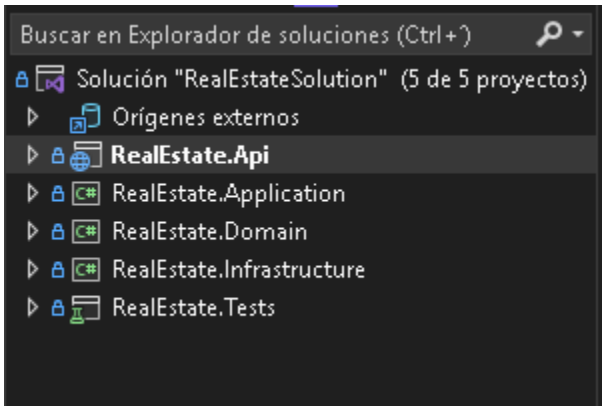
Manual de Usuario - RealEstate API (.NET 8)

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.



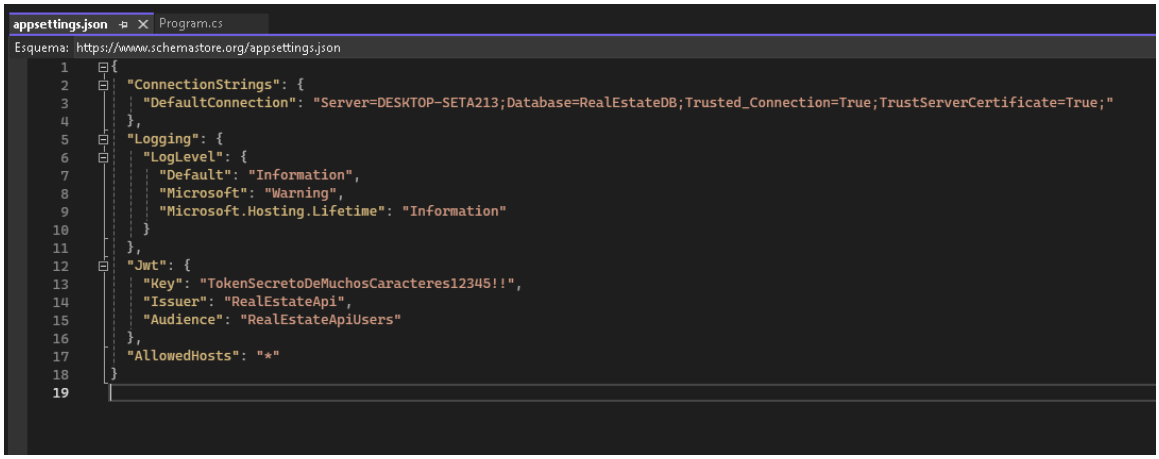
3. Requisitos Previos

- .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:

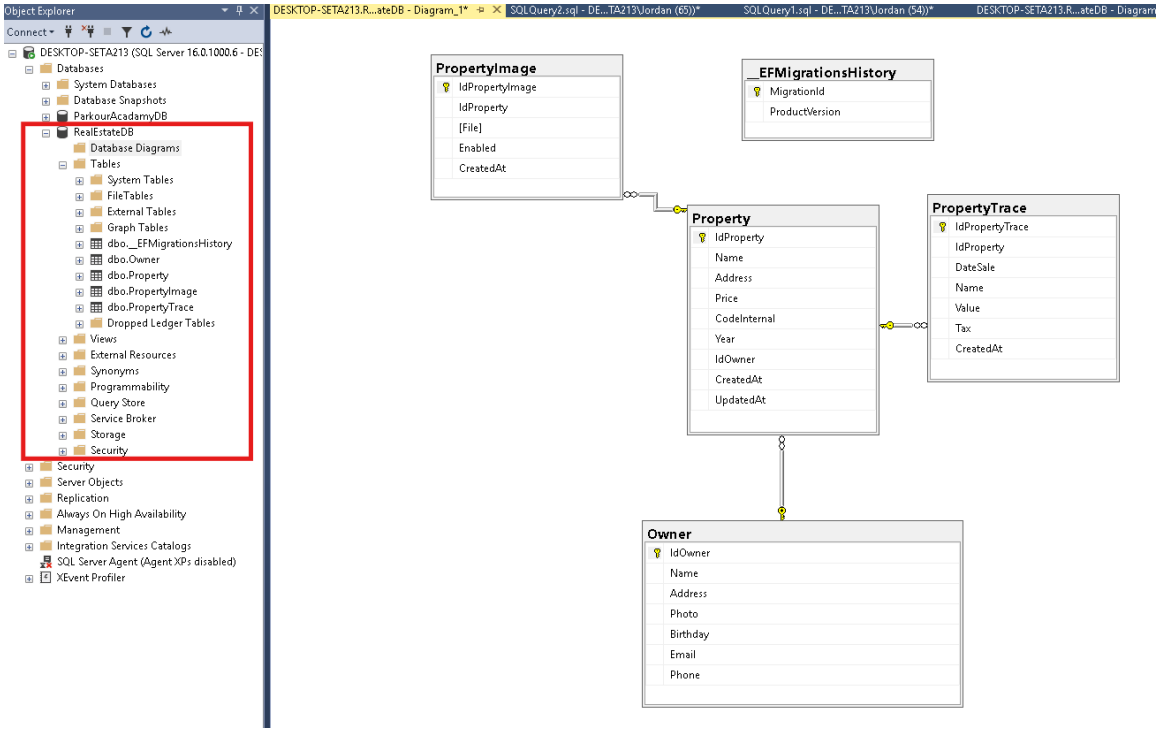
Cadena de conexión (SQL Server) y JWT (seguridad)



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.

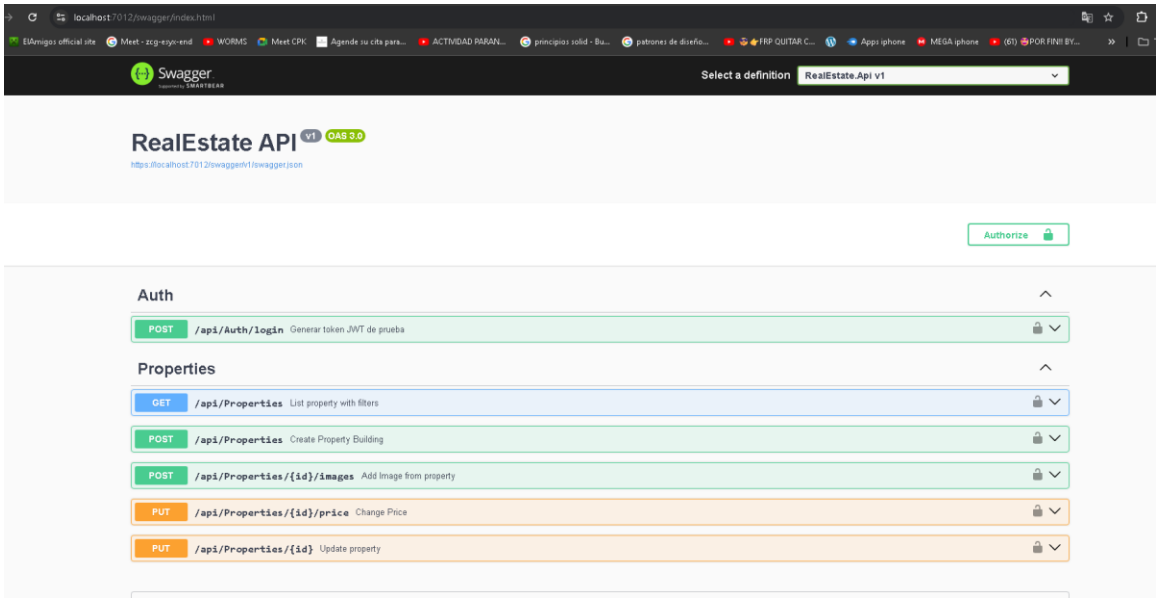


Una vez restaurada o creada, la base contiene datos en las tablas Owner, Property, PropertyImage y PropertyTrace.

6. Ejecución de la Aplicación

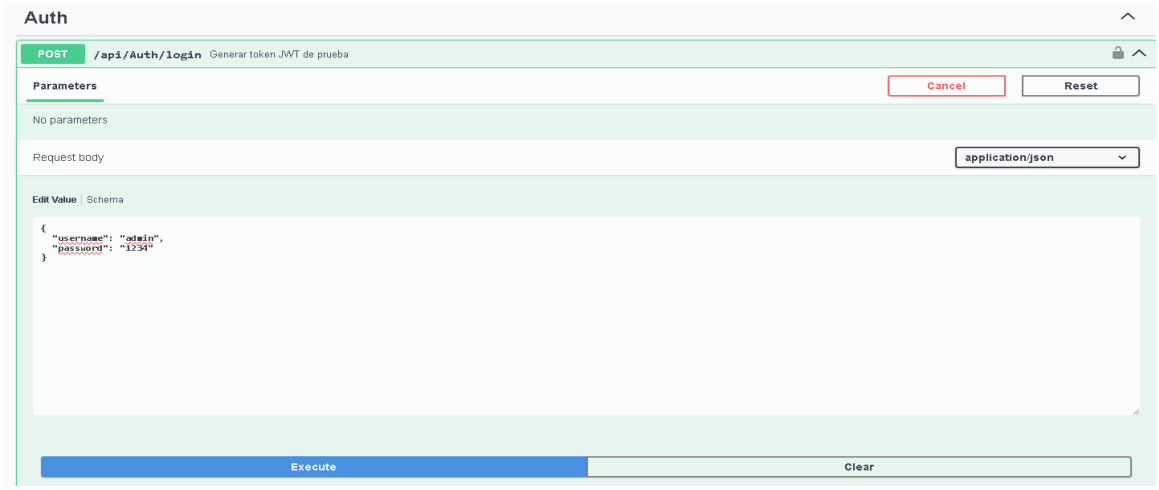
Para ejecutar el proyecto:
- En Visual Studio: establecer RealEstate.Api como proyecto de inicio y presionar F5.
- Por CLI: ubicarse en la carpeta RealEstate.Api y ejecutar dotnet run.

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



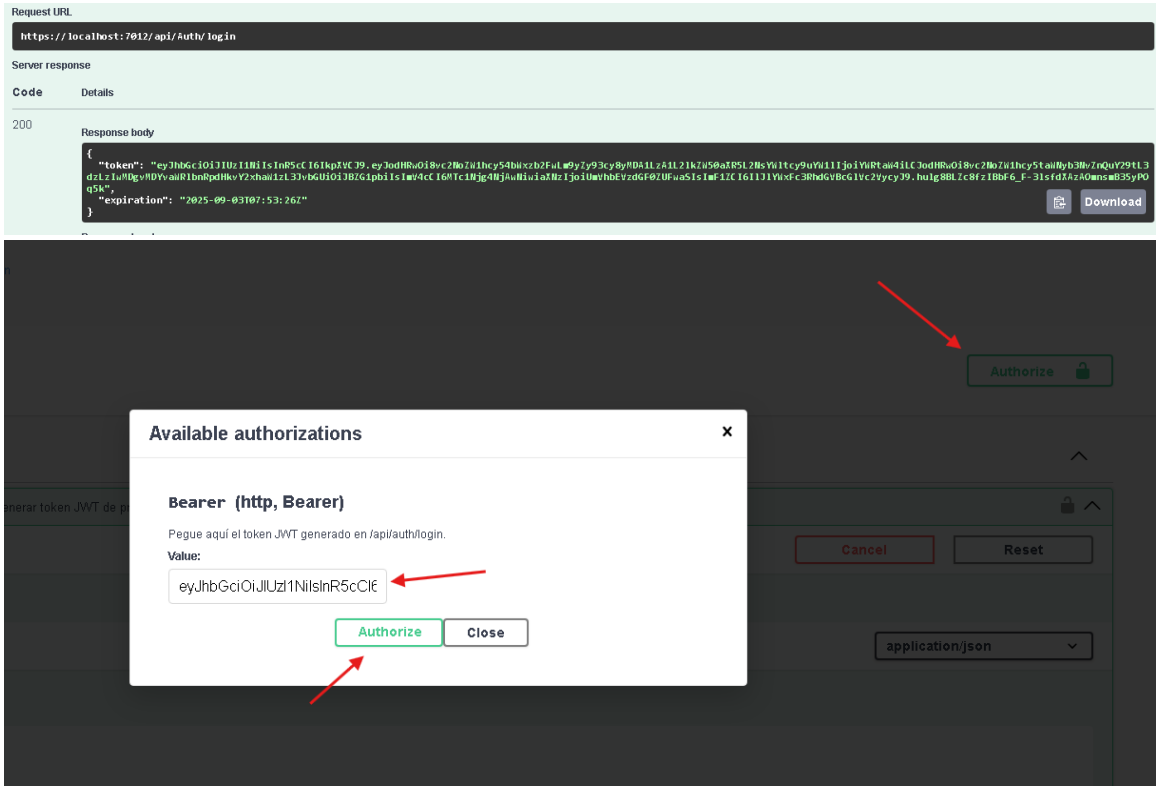
7. Autenticación JWT

Todos los endpoints de PropertiesController requieren autenticación.
1. Generar token en POST /api/auth/login con el body:



2. Copiar el token devuelto y autorizar en Swagger con el botón Authorize.

Notas:
- No incluir 'Bearer' manualmente, Swagger lo añade.
- La clave de JWT en appsettings debe tener mínimo 32 caracteres.

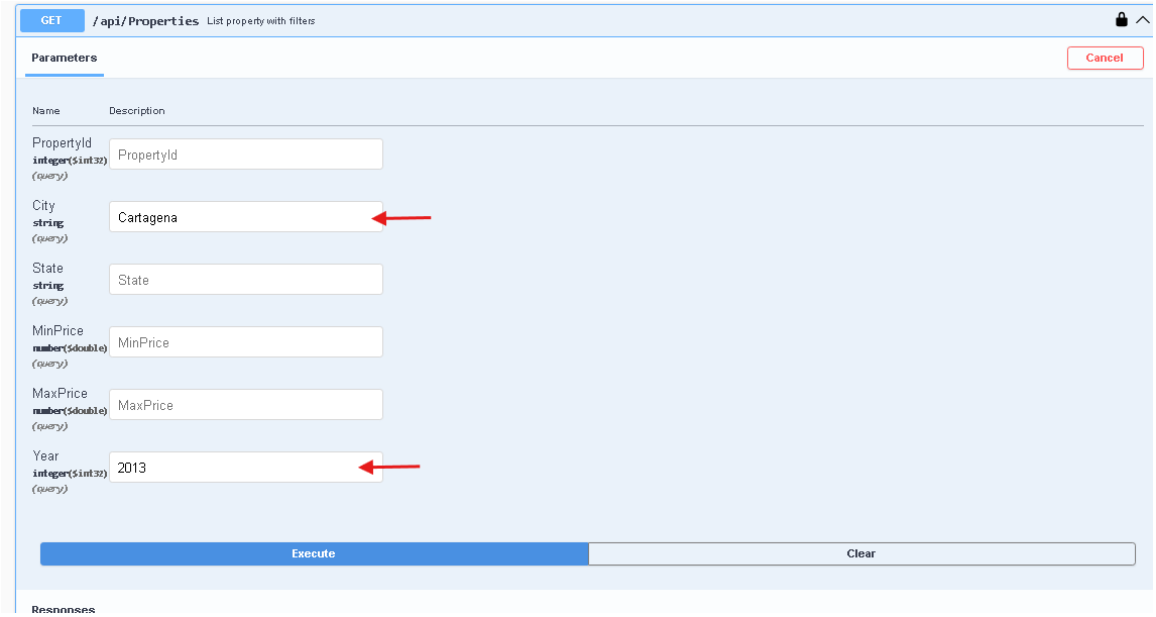


8. Endpoints Principales

Los endpoints implementados según la prueba son:
- GET /api/properties → Listar propiedades con filtros.
- POST /api/properties → Crear nueva propiedad.
- POST /api/properties/{id}/images → Subir imagen a propiedad.
- PUT /api/properties/{id}/price → Cambiar precio de propiedad.
- PUT /api/properties/{id} → Actualizar datos de propiedad.

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

GET /api/properties



Request URL

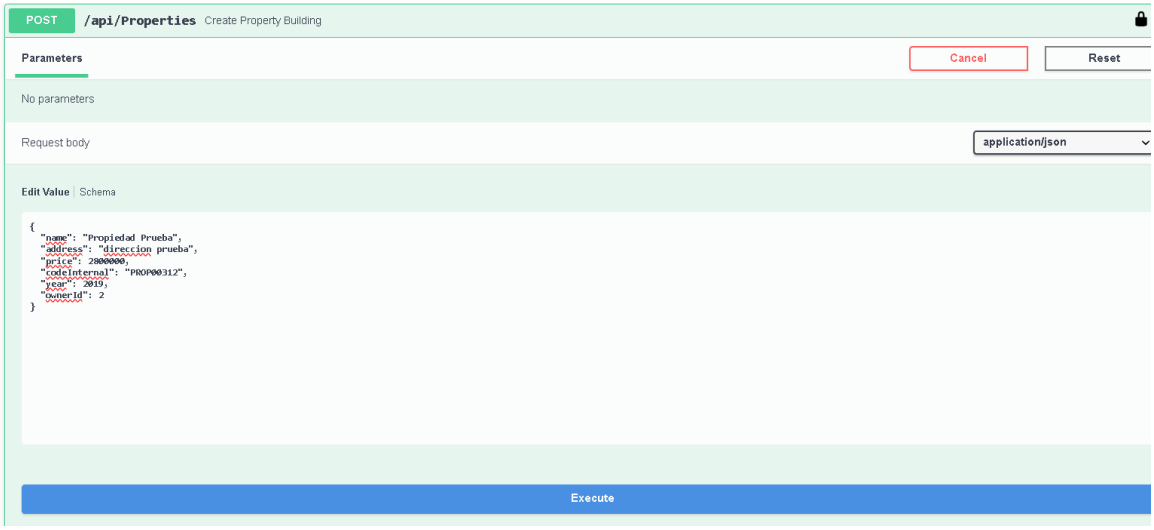
```
https://localhost:7012/api/Properties?City=Cartagena&Year=2013
```

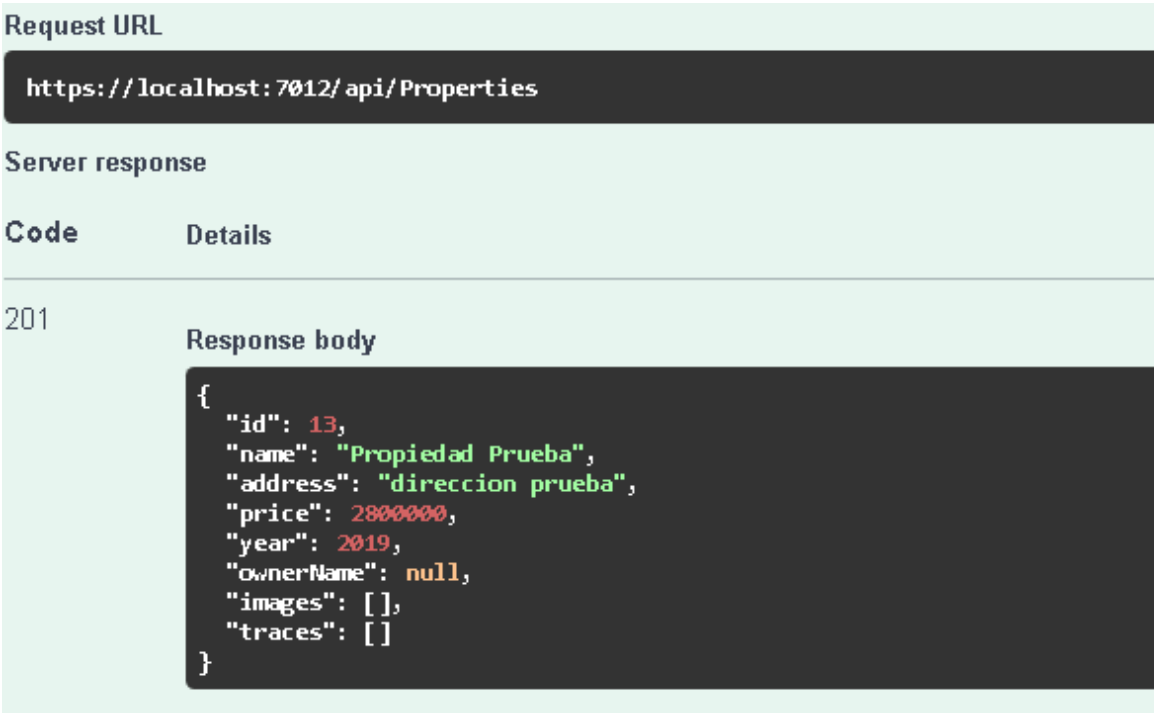
Server response

Code	Details
200	**Response body**

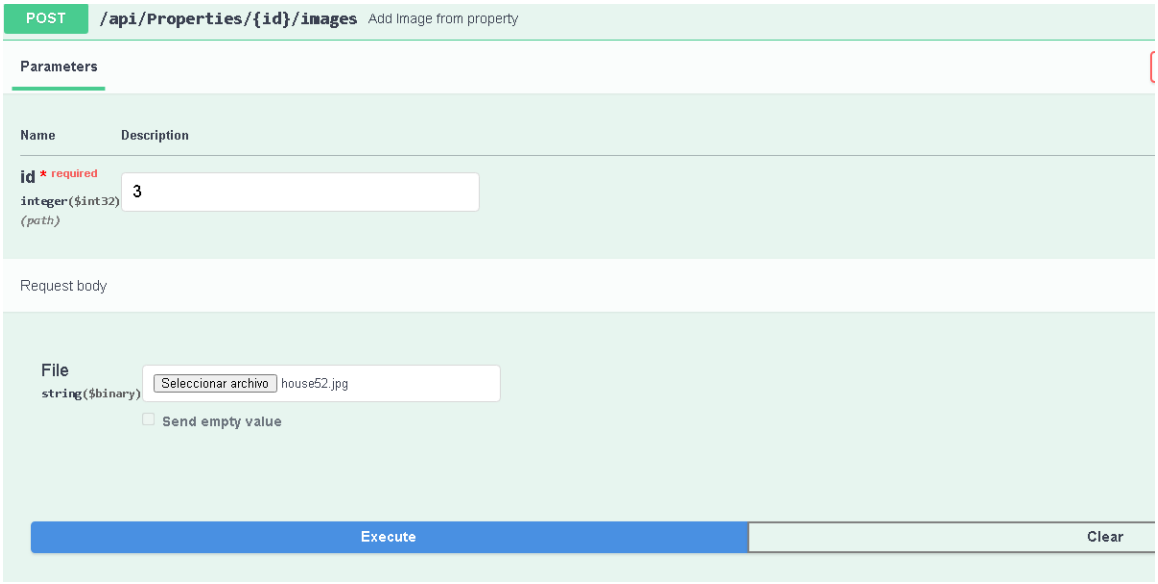

```
  "total": 1,
  "items": [
    {
      "id": 10,
      "name": "Casa Jardines",
      "address": "Cra 33 #70-15, Cartagena",
      "price": 380000000,
      "year": 2013,
      "ownerName": "Andrés Ramírez",
      "images": [
        {
          "id": 19,
          "file": "house19.jpg",
          "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,
```

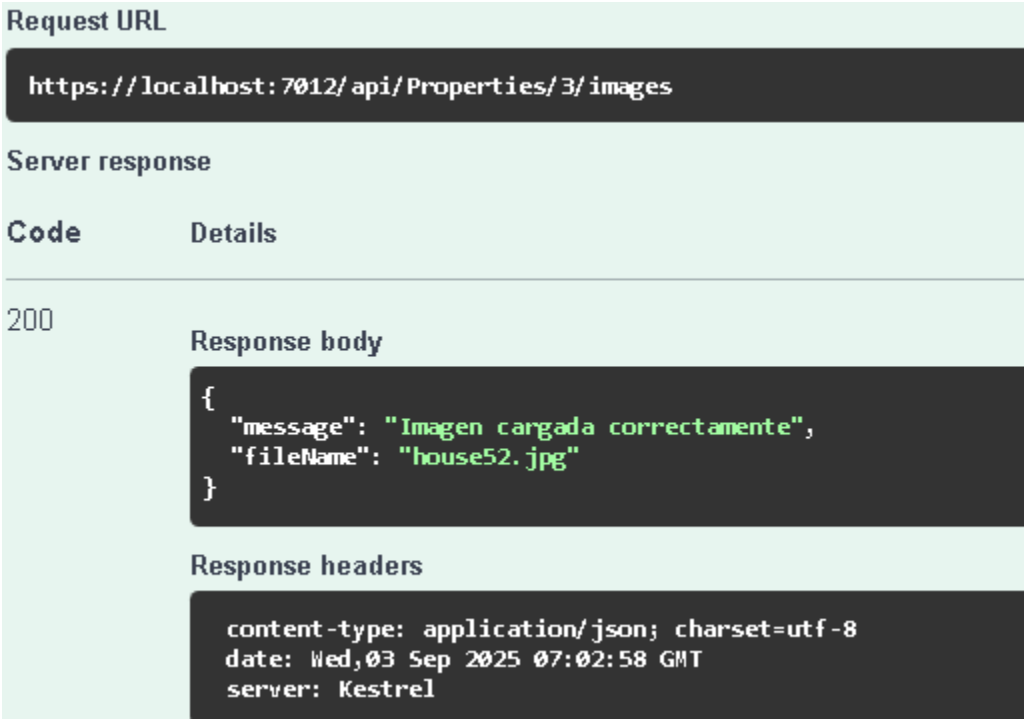
POST /api/properties



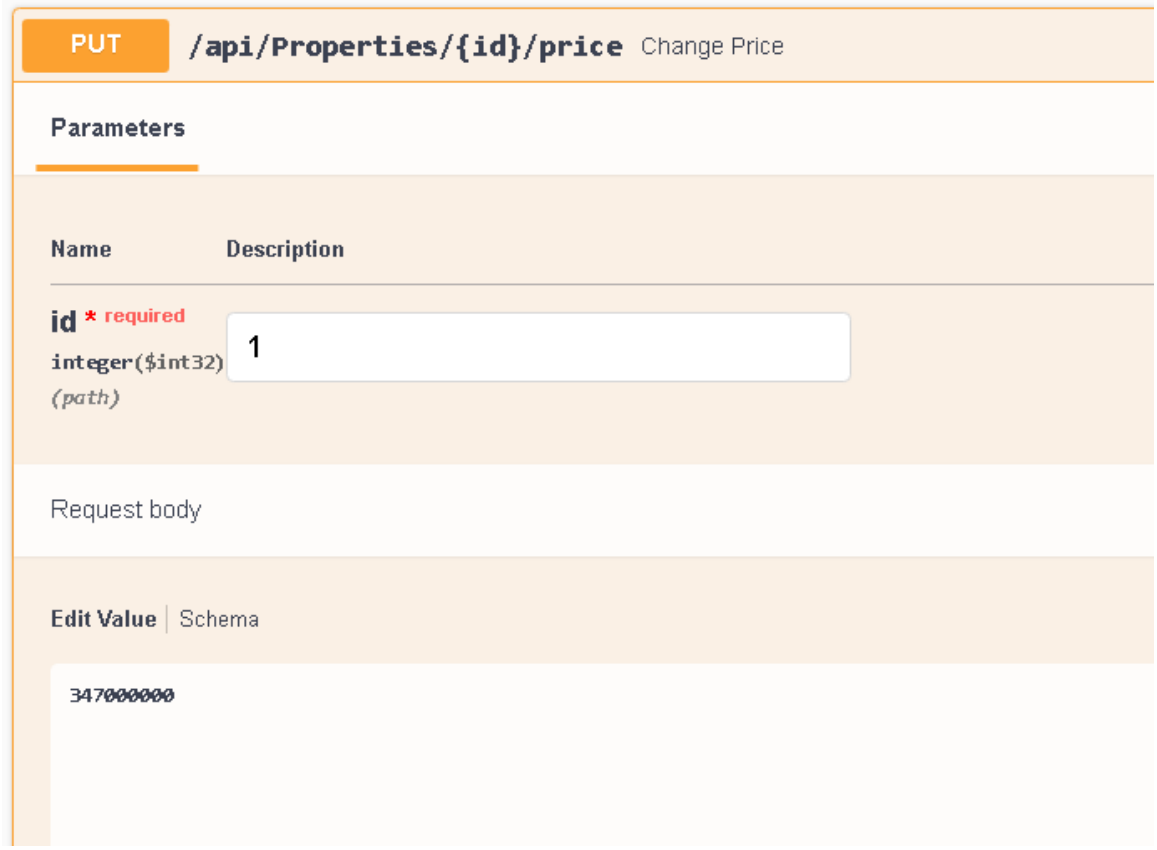


POST /api/properties/{id}/images

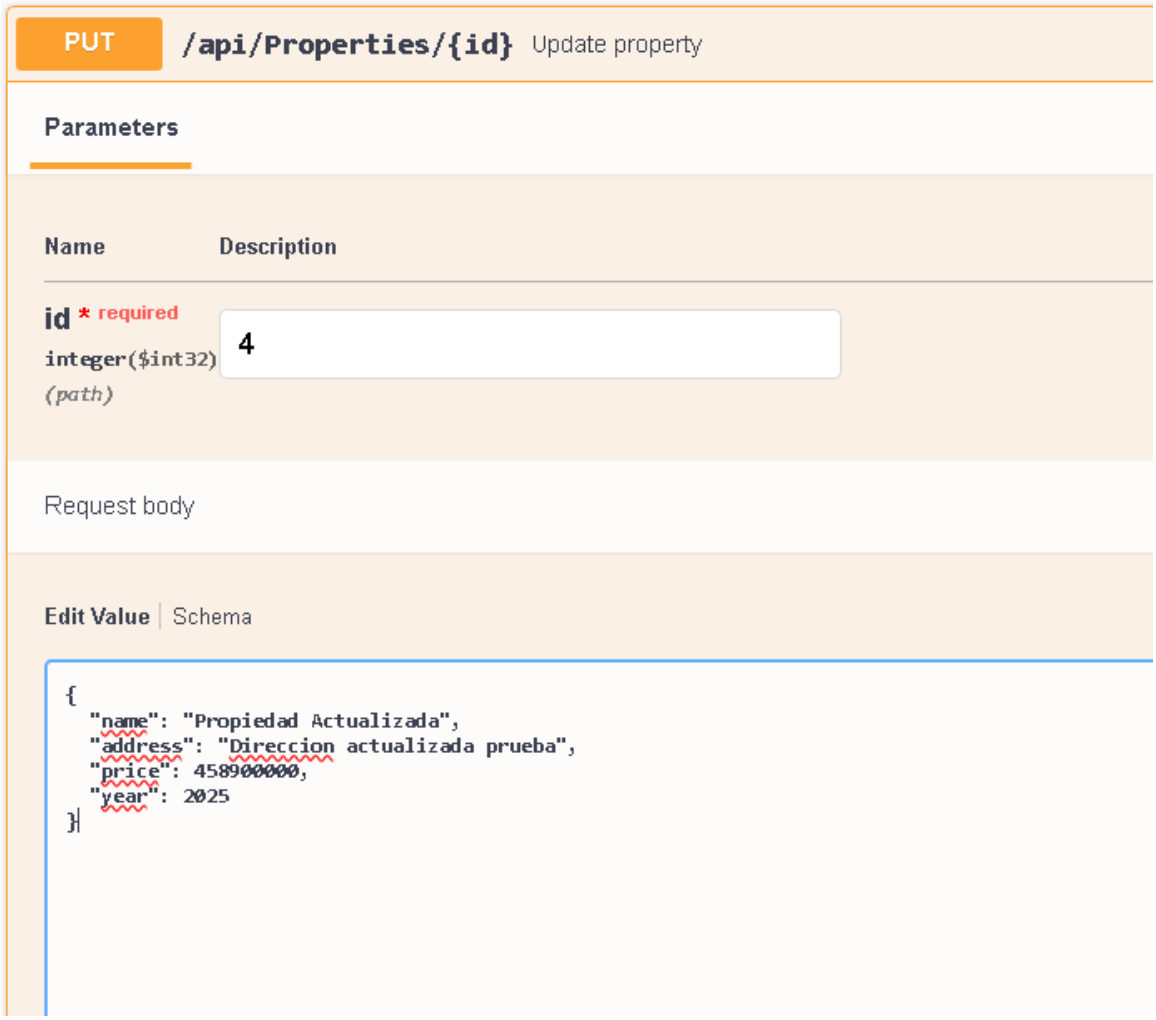


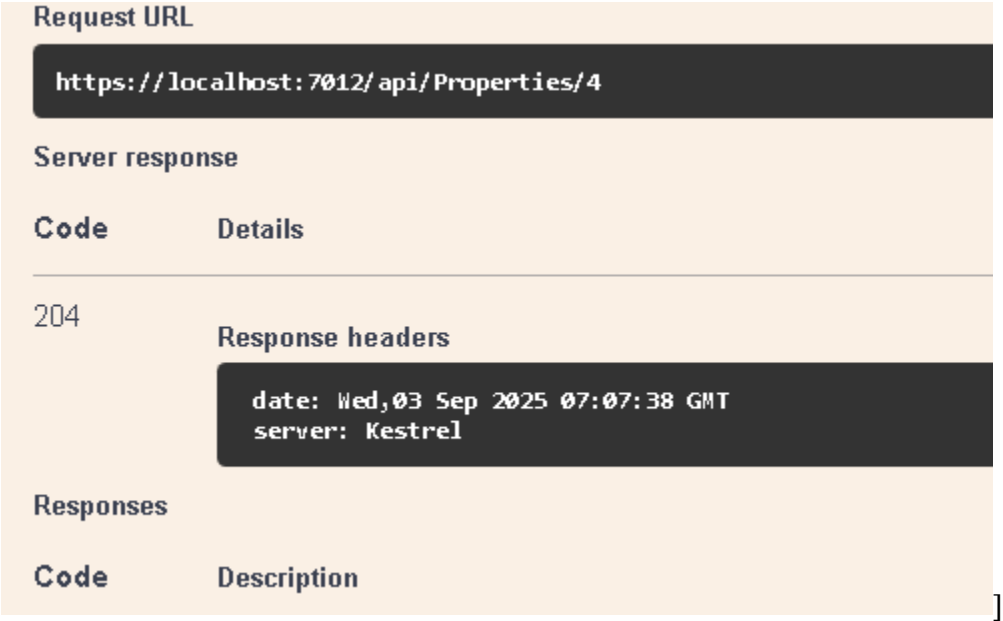


PUT /api/properties/{id}/Price



PUT /api/properties/{id}



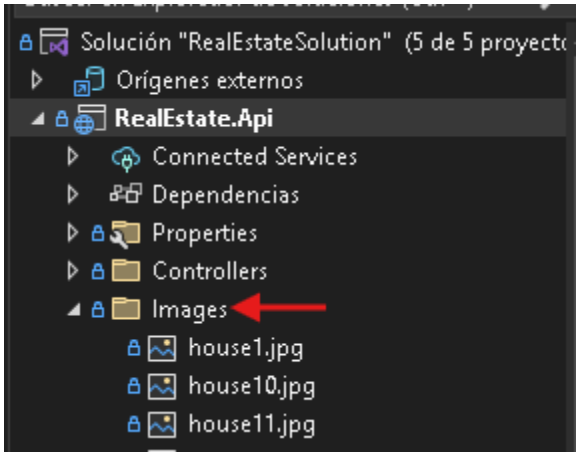


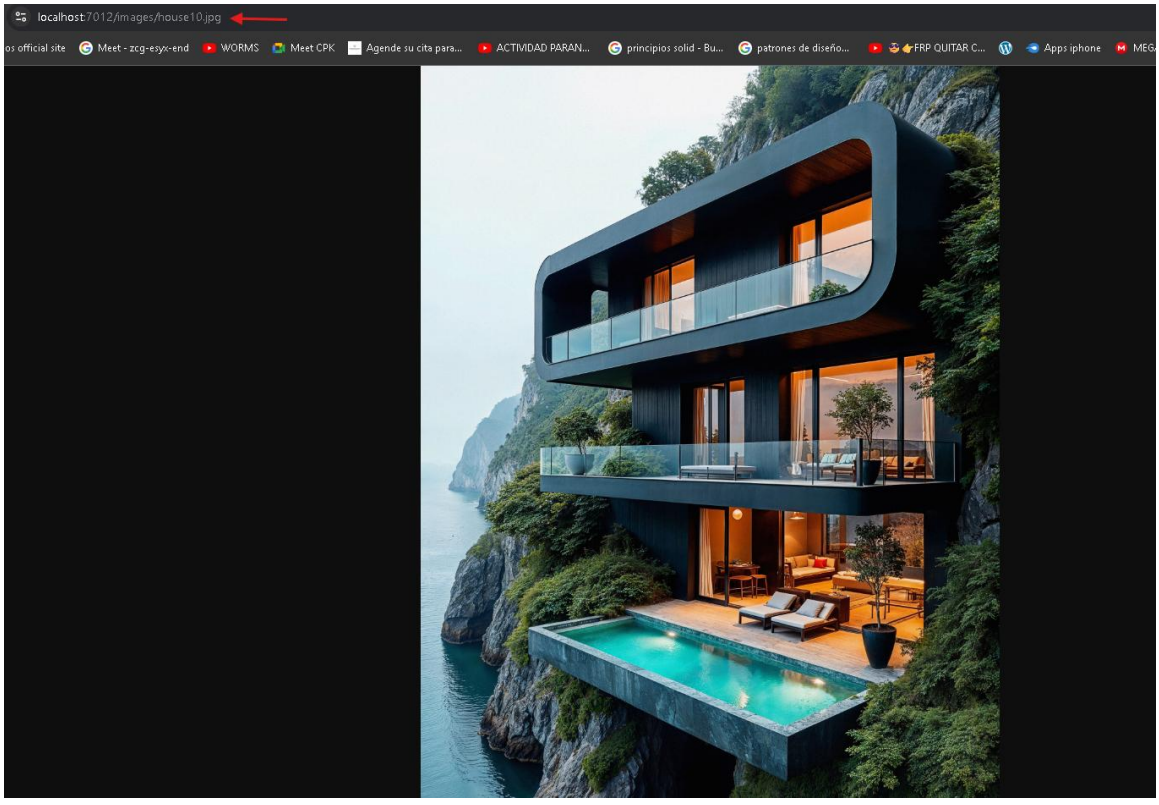
]

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.