**MANUAL DE USO DE LA API**

**DAVID ALEJANDRO CHAPARRO GALVIS**

Contenido

[Instrucciones de instalación 1](#_Toc153914596)

[Instrucciones para prueba de Api 8](#_Toc153914597)

# Instrucciones de instalación

Para el uso de la API se debe realizar varios pasos ya que como se encuentra localmente se debe crear las tablas de la base de datos, mapearlas y generar el string de conexión

**Crear base de datos**

Para la creación de la base de datos debe ejecutar las siguientes sentencias.

**CREATE database reportes;**

**USE reportes;**

**CREATE TABLE REGISTRO\_NOMBRE (**

**NOMBRE VARCHAR(255) PRIMARY KEY,**

**ESTADO BIT**

**);**

**Ejemplo**

**Interfaz de usuario gráfica, Texto, Aplicación

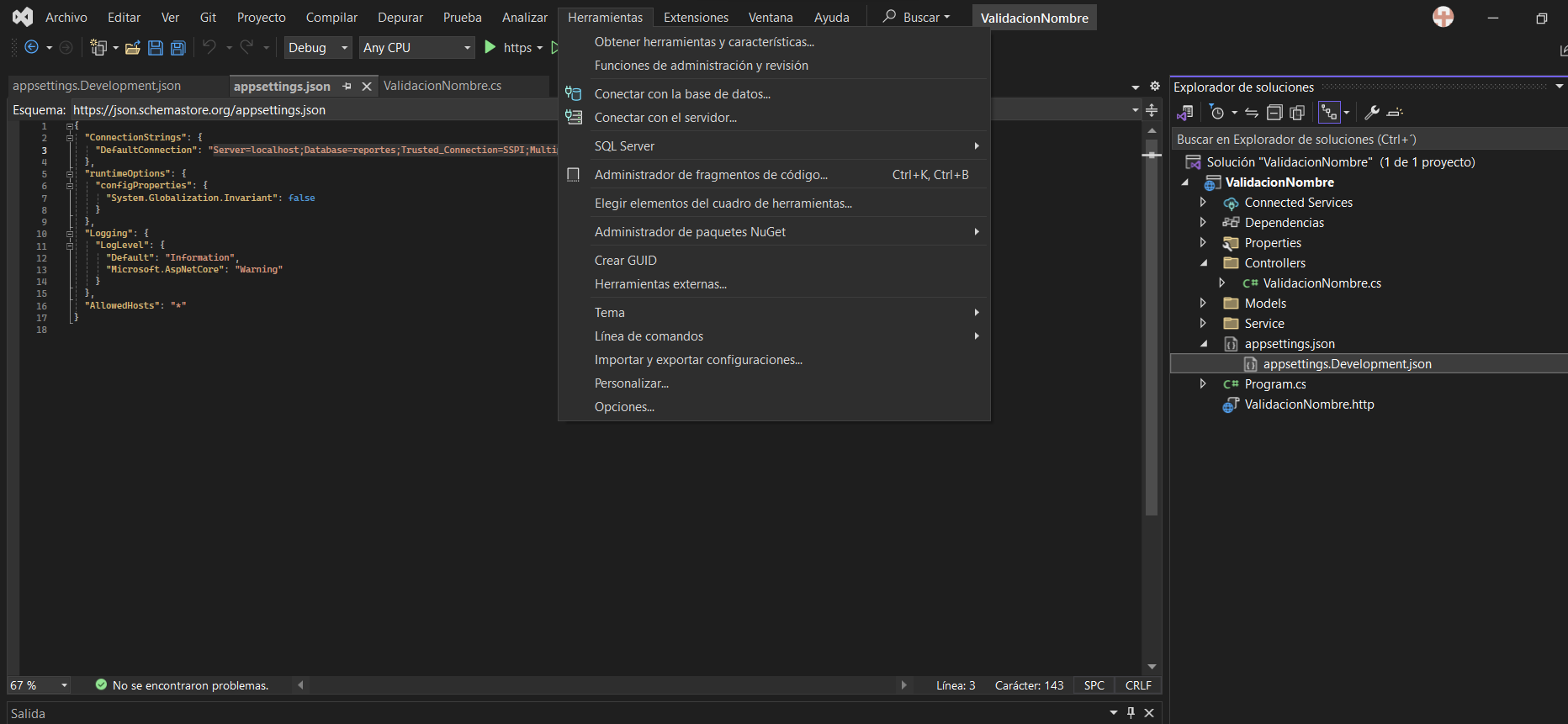
Descripción generada automáticamente**

Ya después de generar la base de datos, se puede ejecutar el api, **SI LA API GENERA ERROR SE DEBE VOLVER A MAPEAR LAS ENTIDADES Y GENERAR EL STRING DE CONEXIÓN**

**SOLO REALIZAR ESTOS PASOS SI SE GENERA ERROR:**

**Mapeo de entidades**

Para el mapeo de las entidades esto se realizará con entity framework, ejecutando el siguiente comando en la ventana de nuggets en Visual studio:

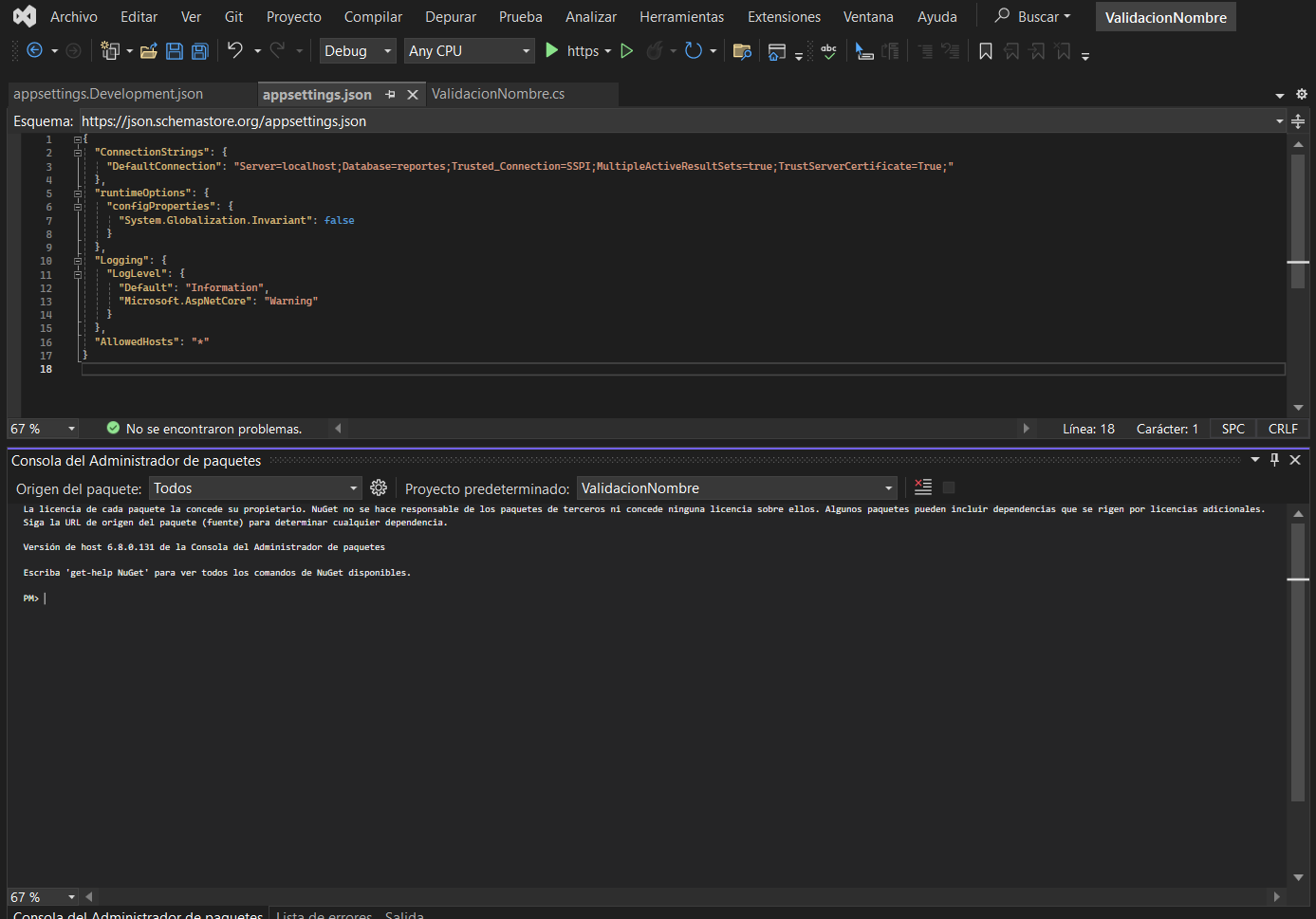
Vamos en la ventana superior a herramientas  


Posteriormente Administrador de paquetes NuGet > Consola de administrador de paquetes.

Captura de pantalla de computadora

Descripción generada automáticamente

Ya tendremos la consola



Al abrir la ventana de Nuget ejecutaremos el siguiente comando si estamos autenticados por autenticación de windows, en sql server.

**Autenticación de windows**

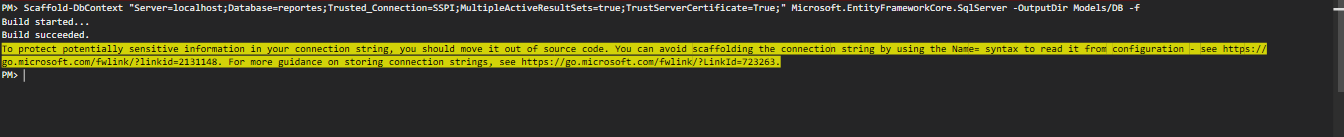
Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Comando**

Scaffold-DbContext "Server=localhost;Database=reportes;Trusted\_Connection=SSPI;MultipleActiveResultSets=true;TrustServerCertificate=True;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models/DB -f

**Ejemplo**

****

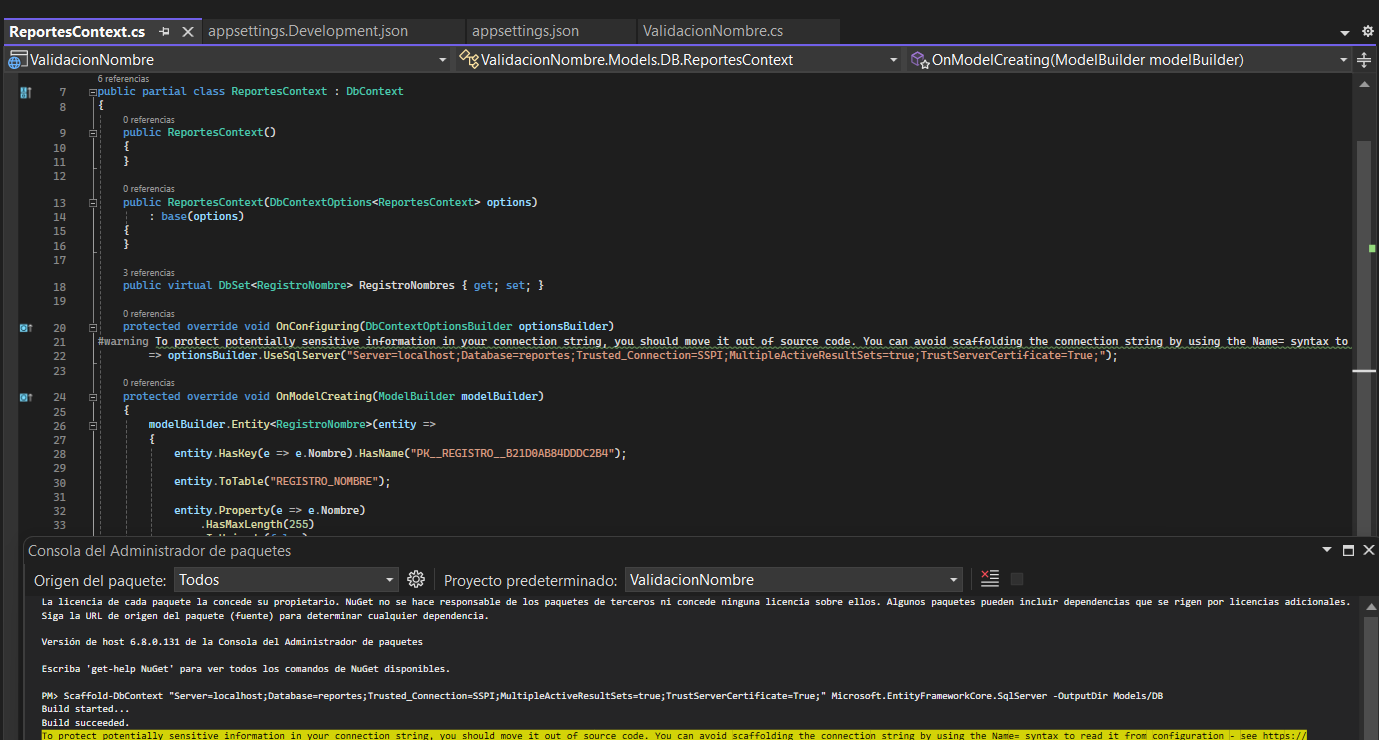
Pero si estamos autenticados por usuario y contraseña debemos usar la siguiente línea de código

Scaffold-DbContext "Server=localhost;Database=reportes;User Id=tu\_usuario;Password=tu\_contraseña;" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models/DB -f

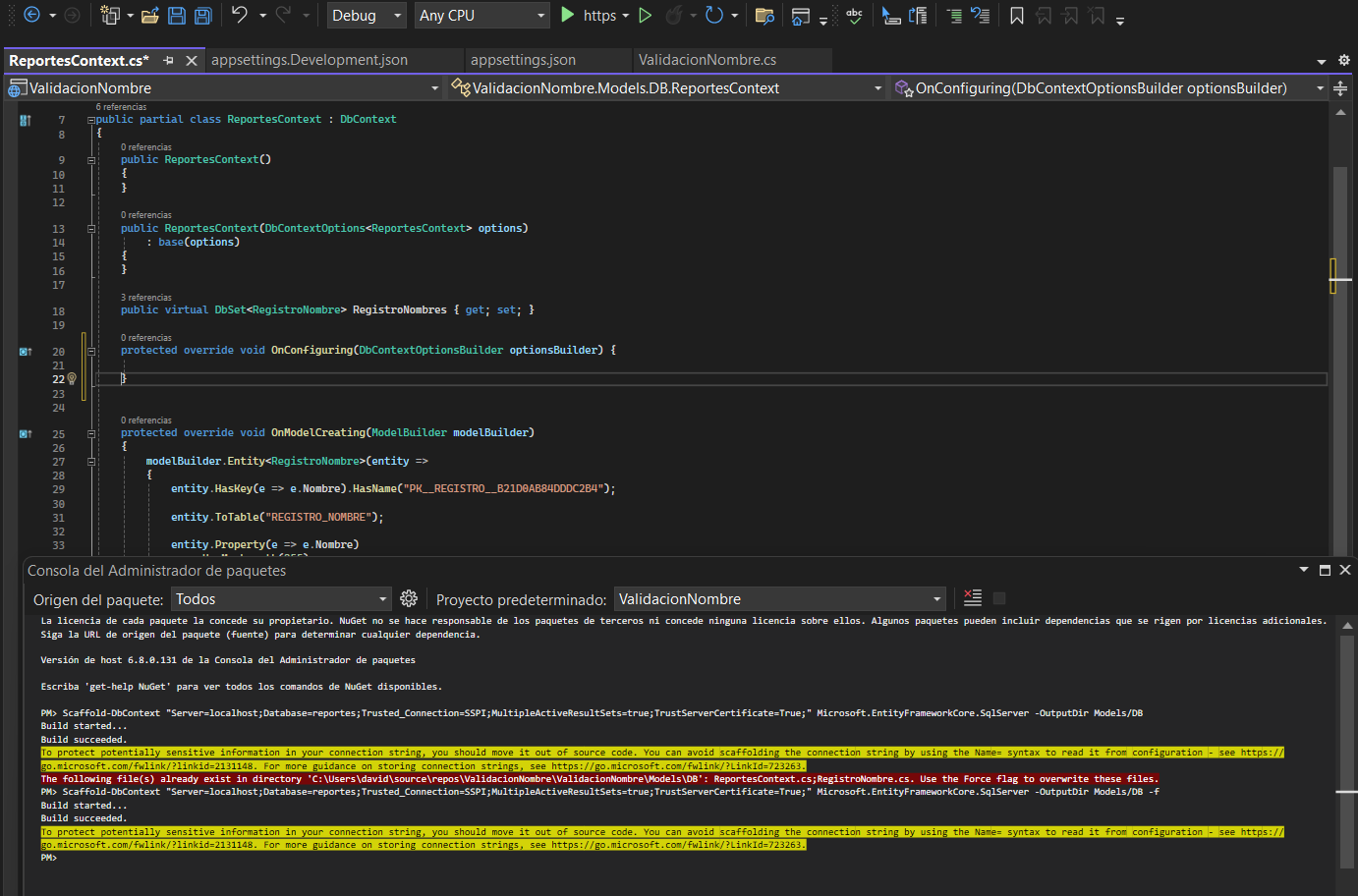
**Modificación de String de conexión**

Ya después de generado las entidades se debe hacer una modificación del proyecto cambiando la ubicación del string de conexión, si vamos al contexto del proyecto que en este caso en particular se llama **“ReportesContext”,** se observa que en el string de conexión se encuentra con un warning este lo debemos copiar y pegarlo en **appsetting.json.**

El string de conexión que se debe llevar es el que se encuentra en la línea 22 de este ejemplo

****

lo cortamos quedando el codigo de la siguiente manera:



Ya después vamos al archivo **appsetting.json** y lo pegamos de la siguiente manera:  
Texto

Descripción generada automáticamente

**Ejemplo**

La cadena del DefaultConnection debe cambiarse por la generada.

"ConnectionStrings": {

"DefaultConnection": "Server=localhost;Database=reportes;Trusted\_Connection=SSPI;MultipleActiveResultSets=true;TrustServerCertificate=True;"

}

Ya por último en el **Program.cs**, se debe poner la siguiente línea de código como la que se ve en la línea 15 a la 17.

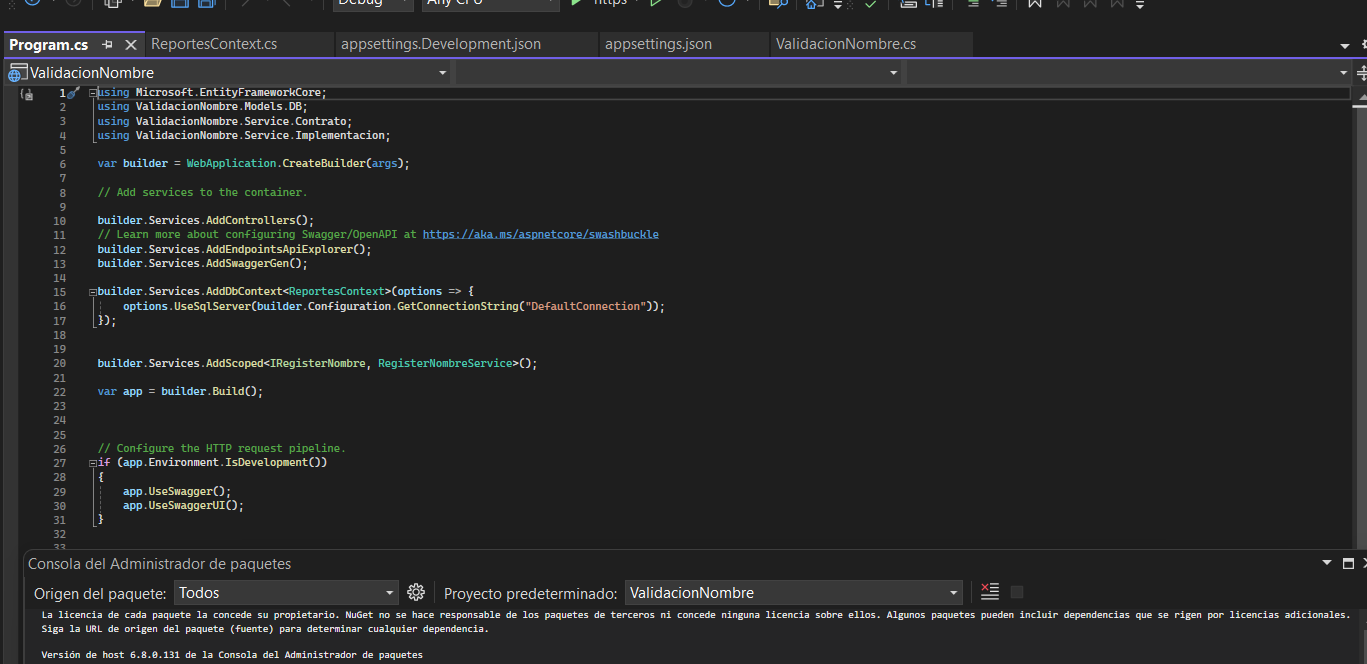
**Codigo**

builder.Services.AddDbContext<ReportesContext>(options => {

options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection"));

});

**Ejemplo**



Ya con estos ajustes el api corre correctamente.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

# Instrucciones para prueba de Api

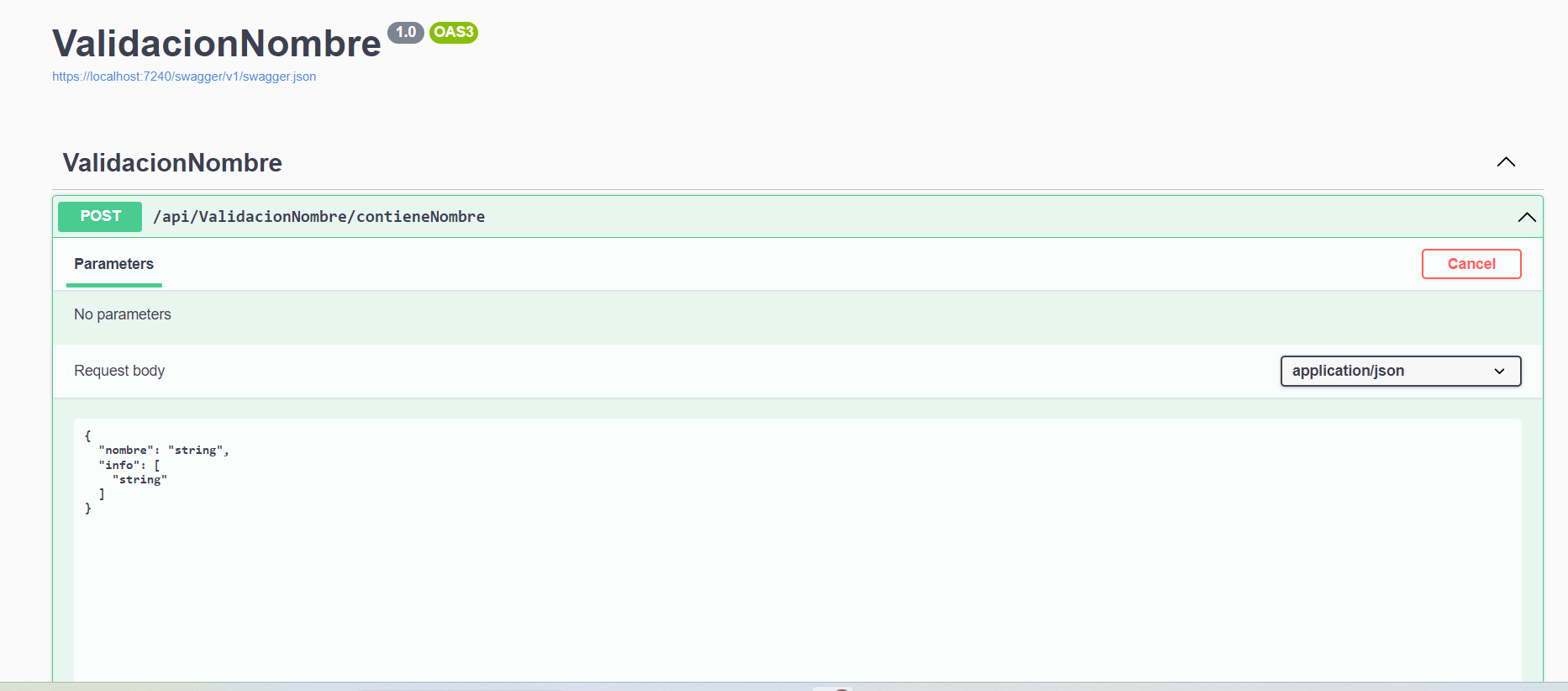
La API presentada se encuentra de forma local, por esto se debe realizar varios procesos para ejecutarla los cuales se van a exponer en este documento.

**Ejecución de la API**

La API cuenta con dos endpoints, que ejemplifican lo pedido en el primer punto del documento

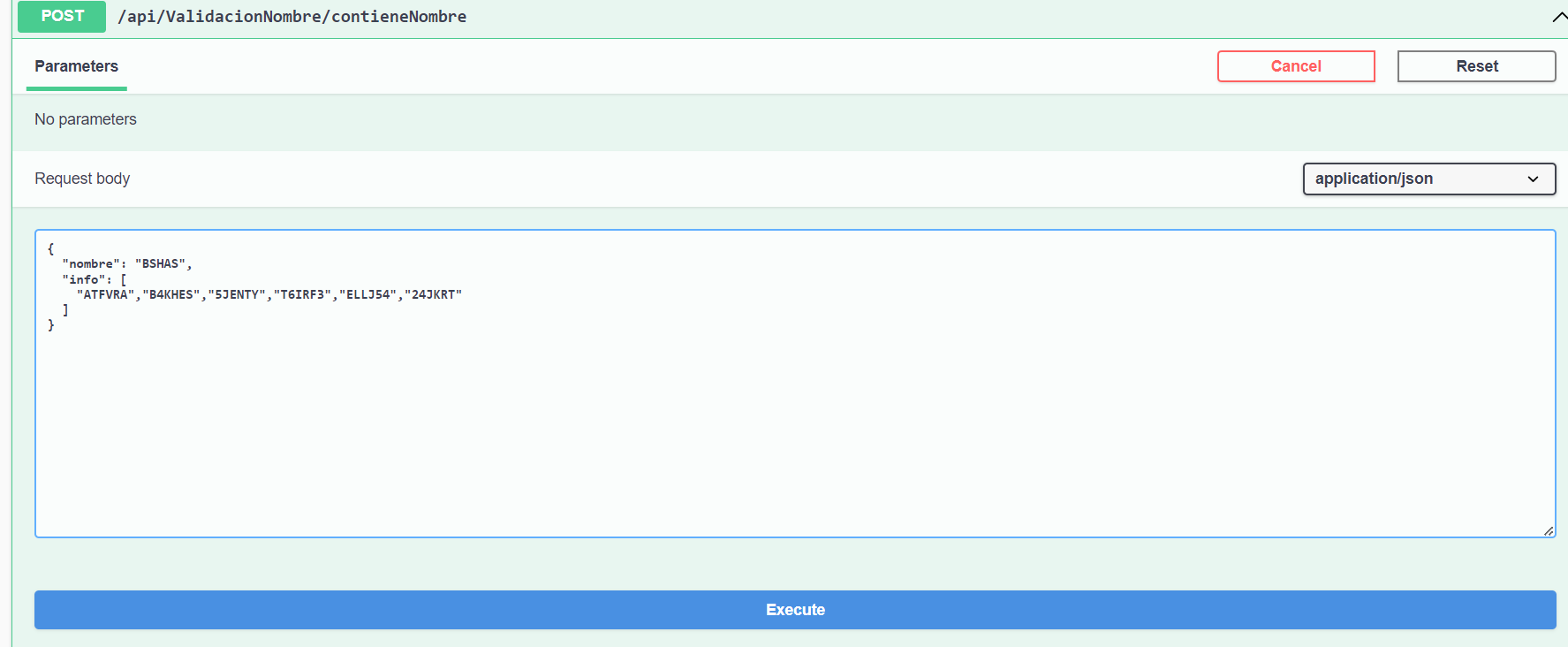
**Método POST → / contieneNombre/**

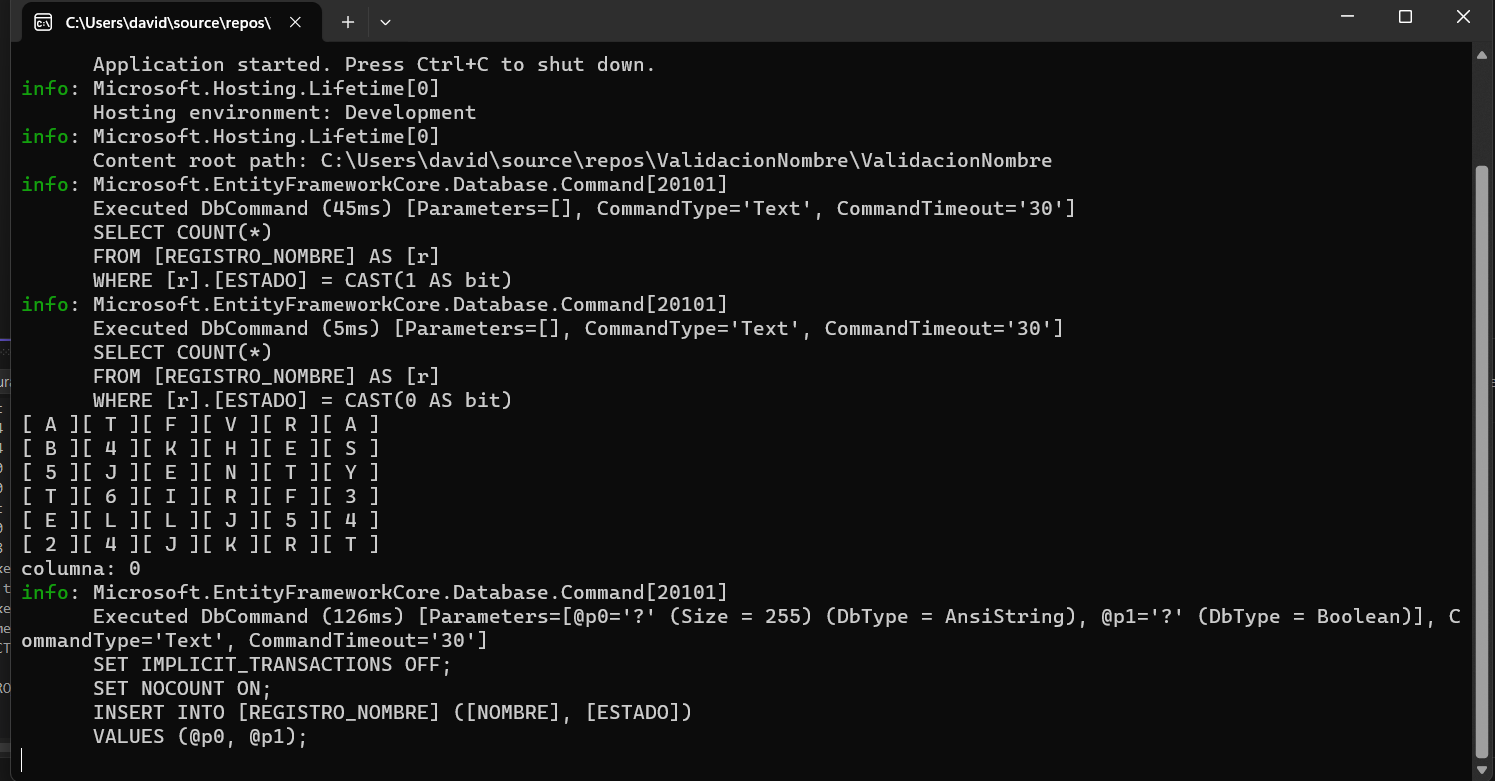
Al ejecutar el proyecto se va a abrir en el navegador Swagger que es un aplicativo integrado directamente en c# para realizar pruebas de APIS, para la ejecución correcta del endpoint **contieneNombre** se debe llenar dos campos, el primero es nombre (palabra a buscar):, el segundo es un con el que se va a llenar la sopa de letras propuesta.



**Ejemplo**

Al darle al botón ejecutar tendremos una salida por consola que nos muestra la sopa de letras para una mejor validación.

****

****

También se puede observar la respuesta, de si estaba o no en la sopa de letras la palabra ingresada por el enpoint.

Patrón de fondo

Descripción generada automáticamente

**Método GET → /reporte**

El siguiente método devuelve un json con tres campos:  
**cuenta\_contieneNombre** (Columnas de la base de datos que si estaban en la sopa de letras)

**cuenta\_noContieneNombre** (Columnas de la base de datos que no estaban en la sopa de letras)

**relación** (diferencia de las dos columnas cuenta\_contieneNombre/cuenta\_noContieneNombre)

