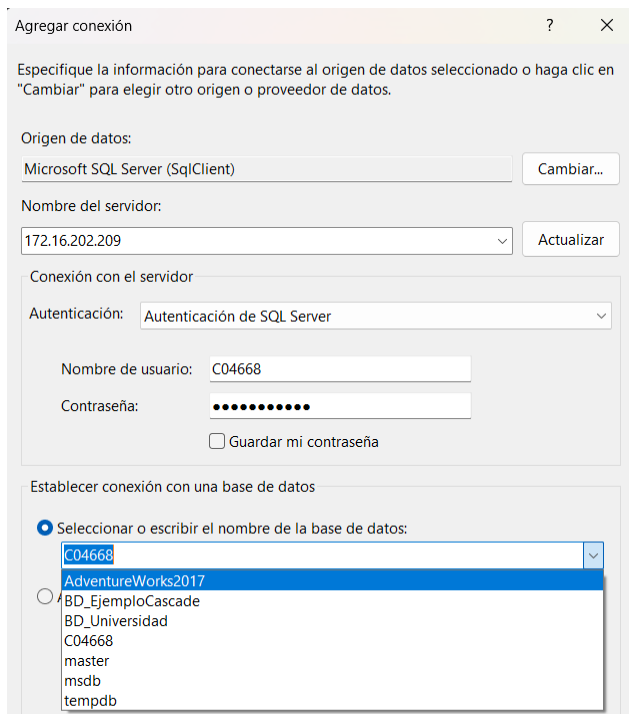


**Jose Andres Matarrita Miranda**

## Screenshots

### 1. Conexión con la base de datos remota (sección 1.1)



Agregar conexión

Especifique la información para conectarse al origen de datos seleccionado o haga clic en "Cambiar" para elegir otro origen o proveedor de datos.

Origen de datos:  
Microsoft SQL Server (SqlClient) Cambiar...

Nombre del servidor:  
172.16.202.209 Actualizar

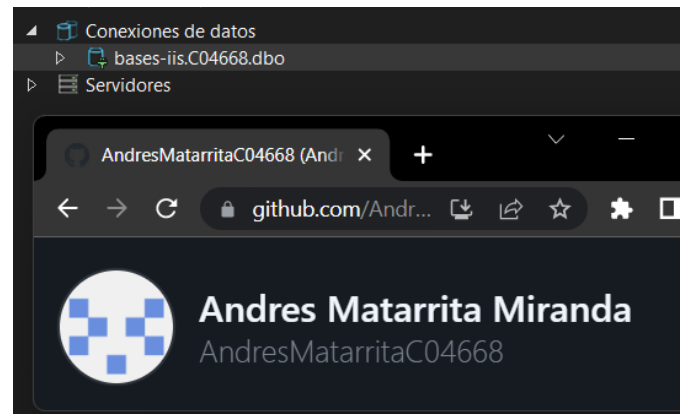
Conexión con el servidor  
Autenticación: Autenticación de SQL Server

Nombre de usuario: C04668  
Contraseña: .....  
☐ Guardar mi contraseña

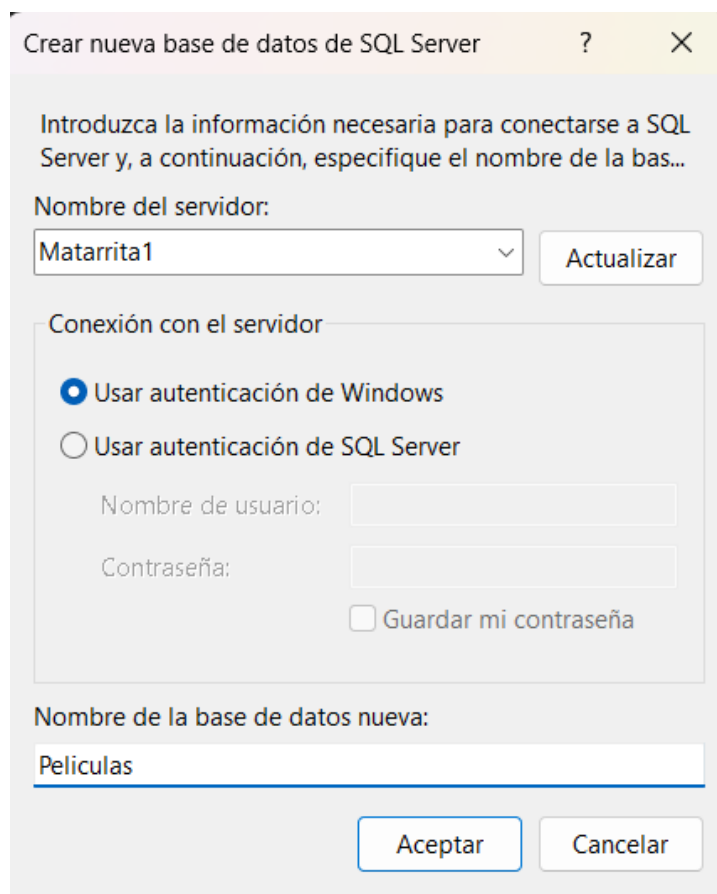
Establecer conexión con una base de datos

☒ Seleccionar o escribir el nombre de la base de datos:

- C04668
- AdventureWorks2017
- BD\_EjemploCascade
- BD\_Universidad
- C04668
- master
- msdb
- tempdb



### 2. Conexión con la base de datos local (sección 1.2)



Crear nueva base de datos de SQL Server

Introduzca la información necesaria para conectarse a SQL Server y, a continuación, especifique el nombre de la bas...

Nombre del servidor:  
Matarrita1 Actualizar

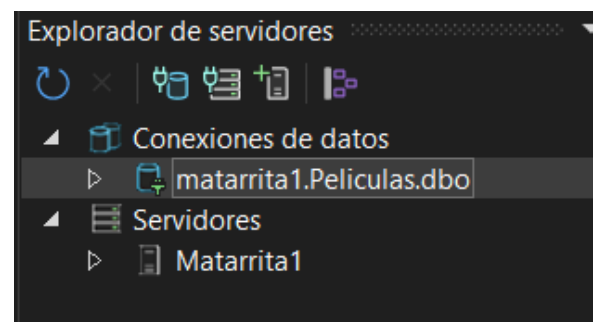
Conexión con el servidor

☒ Usar autenticación de Windows  
☐ Usar autenticación de SQL Server

Nombre de usuario:   
Contraseña:   
☐ Guardar mi contraseña

Nombre de la base de datos nueva:  
Películas

Aceptar Cancelar



### 3. Cadena de conexión donde se vea nombre de base de datos local (sección 1.2.1)

matarrita1.Peliculas.dbo Conexión	
	
(Identidad)	
(Nombre)	Peliculas
Conexión	
Cadena de conexión	Data Source=Matarrita1;Initial Catalog=Peliculas;Integrated Security=True;Pooling=False
Estado	Abrir
Proveedor	Proveedor de datos .NET Framework para servidor SQL Server
Tipo	Microsoft SQL Server
Versión	16.00.1000
Varios	
Distinguir mayúsculas de minúsculas	False
Propietario	MATARRITA1\andre

```
appsettings.json
Esquema: https://json.schemastore.org/appsettings.json
1  {
2    "Logging": {
3      "LogLevel": {
4        "Default": "Information",
5        "Microsoft.AspNetCore": "Warning"
6      }
7    },
8    "AllowedHosts": "*",
9  },
10  "ConnectionStrings": {
11    "ContextoDePeliculas": "Data Source=Matarrita1;Initial Catalog=Peliculas;Integrated Security=True;Pooling=False"
12  }
13  }
14  }
```

### 4. Captura de la tabla Película sobre la base de datos local (sección 1.2.2)

SQLQuery1.sql \* appsettings.json


1 CREATE TABLE Pelicula(  
2 Id INTEGER PRIMARY KEY IDENTITY,  
3 Nombre VARCHAR(200) NOT NULL,  
4 Año INT NOT NULL  
5 )  
6 GO

GO

Peliculas

Laborato x AndresM x +

github.com...



**Andres Matarrita Miranda**  
AndresMatarritaC04668

Peliculas

Tablas

Tablas del sistema

Tablas de archivos


Tablas externas

dbo.Pelicula

Vistas

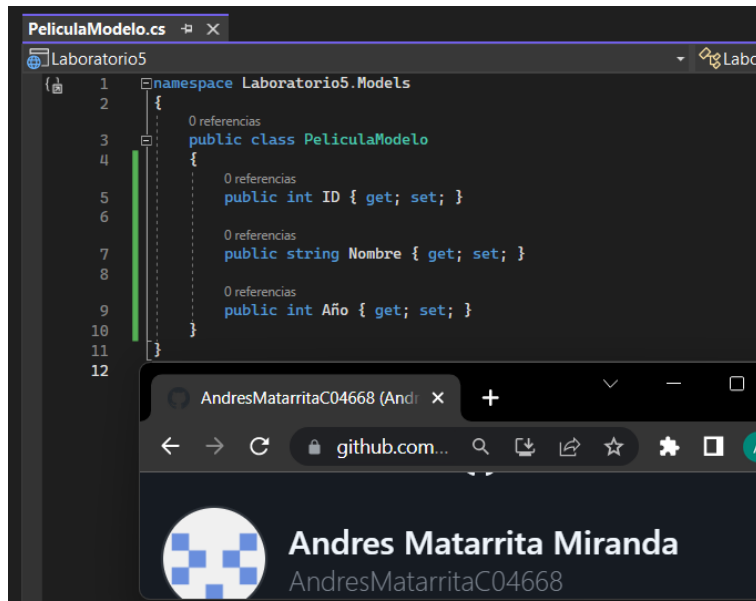
Sinónimos

github.com...



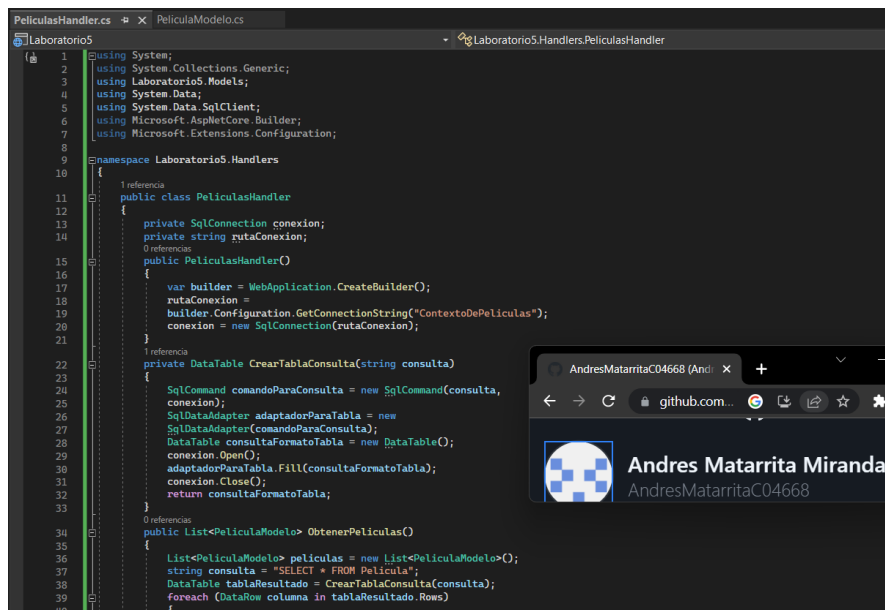
**Andres Matarrita Miranda**  
AndresMatarritaC04668

## 5. Código del modelo generado (sección 2.1)



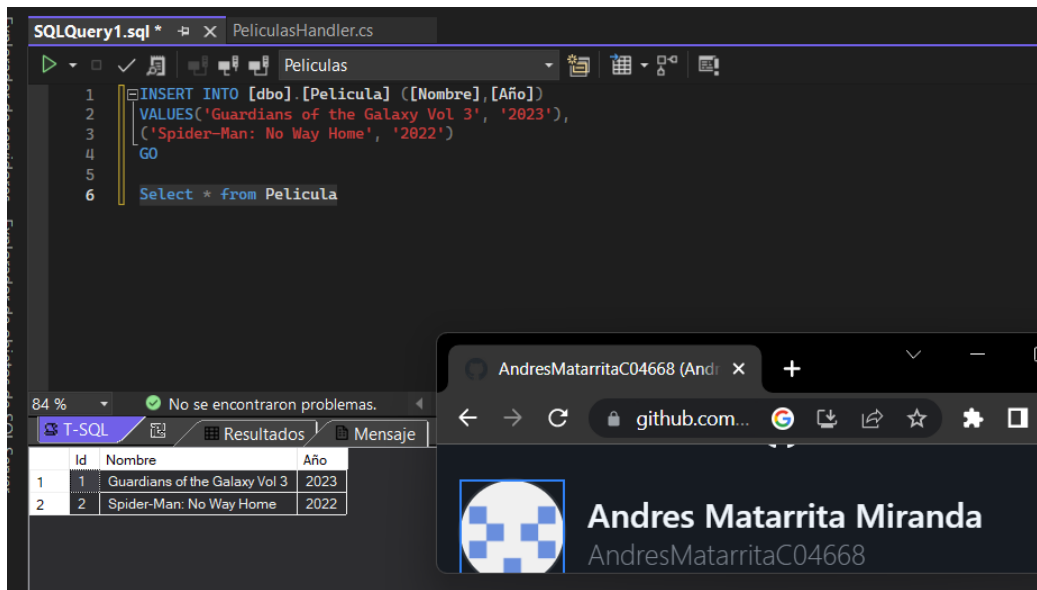
```
1 namespace Laboratorio5.Models
2 {
3     0 referencias
4     public class PeliculaModelo
5     {
6         0 referencias
7         public int ID { get; set; }
8
9         0 referencias
10        public string Nombre { get; set; }
11
12        0 referencias
13        public int Año { get; set; }
14    }
15 }
```

## 6. Captura de una porción del Handler (sección 2.2)



```
1 using System;
2 using System.Collections.Generic;
3 using Laboratorio5.Models;
4 using System.Data;
5 using System.Data.SqlClient;
6 using Microsoft.AspNetCore.Builder;
7 using Microsoft.Extensions.Configuration;
8
9 namespace Laboratorio5.Handlers
10 {
11     1 referencia
12     public class PeliculasHandler
13     {
14         private SqlConnection conexion;
15         private string rutaConexion;
16         0 referencias
17         public PeliculasHandler()
18         {
19             var builder = WebApplication.CreateBuilder();
20             rutaConexion =
21             builder.Configuration.GetConnectionString("ContextoDePeliculas");
22             conexion = new SqlConnection(rutaConexion);
23         }
24         1 referencia
25         private DataTable CrearTablaConsulta(string consulta)
26         {
27             SqlCommand comandoParaConsulta = new SqlCommand(consulta,
28             conexion);
29             SqlDataAdapter adaptadorParaTabla = new
30             SqlDataAdapter(comandoParaConsulta);
31             DataTable consultaFormatoTabla = new DataTable();
32             adaptadorParaTabla.Fill(consultaFormatoTabla);
33             conexion.Close();
34             return consultaFormatoTabla;
35         }
36         0 referencias
37         public List<PeliculaModelo> ObtenerPeliculas()
38         {
39             List<PeliculaModelo> peliculas = new List<PeliculaModelo>();
40             string consulta = "SELECT * FROM Pelicula";
41             DataTable tablaResultado = CrearTablaConsulta(consulta);
42             foreach (DataRow columna in tablaResultado.Rows)
43             {
44             }
```

## 7. Captura de la tabla Pelicula con los datos ingresados (sección 2.3)



The screenshot shows the SQL Server Enterprise Manager interface. The top pane displays a T-SQL query in the 'PelículasHandler.cs' file:

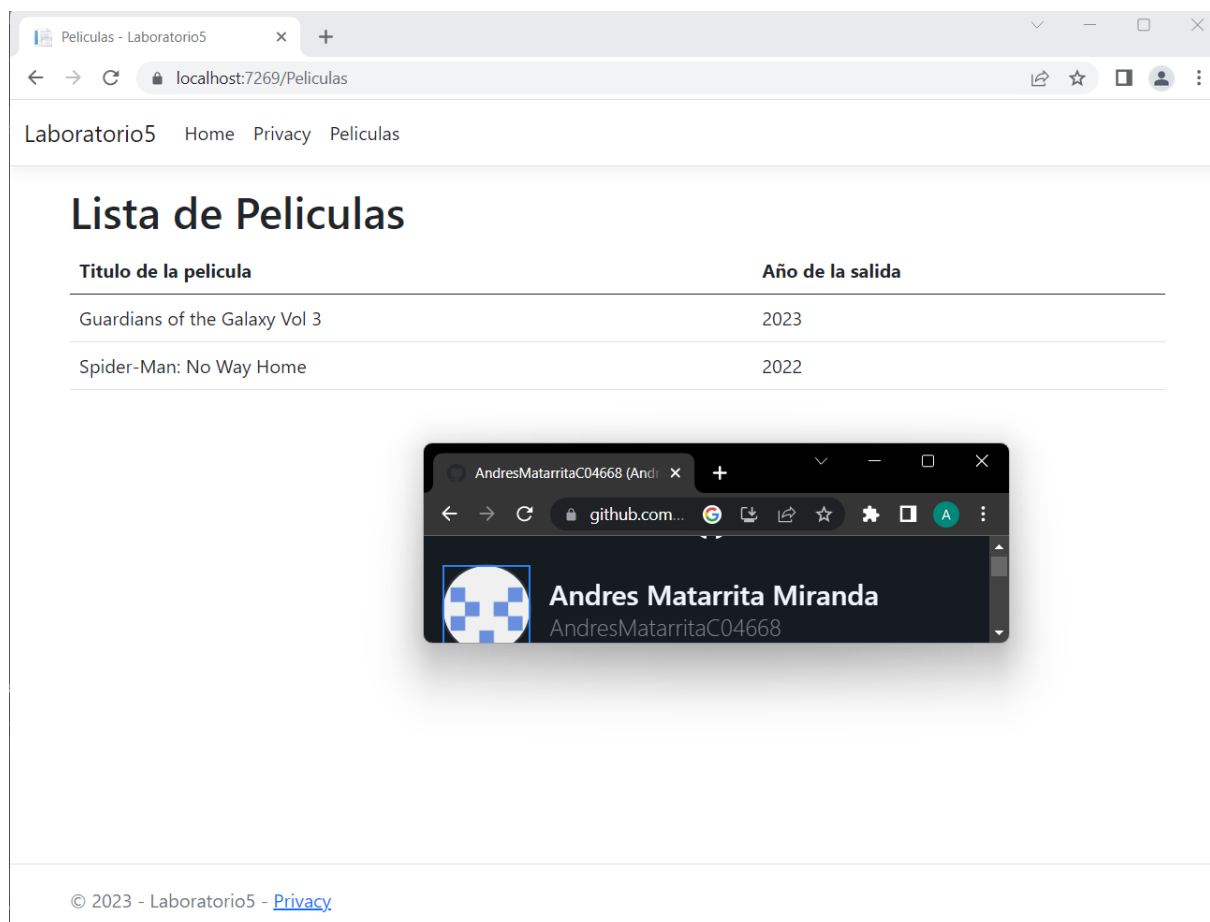
```
1 INSERT INTO [dbo].[Pelicula] ([Nombre],[Año])
2 VALUES('Guardians of the Galaxy Vol 3', '2023'),
3 ('Spider-Man: No Way Home', '2022')
4 GO
5
6 Select * from Pelicula
```

The bottom pane shows the results of the query, displaying a table with two rows:

Id	Nombre	Año
1	Guardians of the Galaxy Vol 3	2023
2	Spider-Man: No Way Home	2022

A browser window is also visible in the foreground, showing the GitHub profile of Andres Matarrita Miranda (AndresMatarritaC04668).

## 8. Despliegue de la información en la página web (sección 2.3)



The screenshot shows a web application running on a browser. The browser address bar indicates the URL is `localhost:7269/Peliculas`. The page title is "Laboratorio5". The main content area displays a list of movies under the heading "Lista de Peliculas".

Titulo de la pelicula	Año de la salida
Guardians of the Galaxy Vol 3	2023
Spider-Man: No Way Home	2022

A browser window is also visible in the foreground, showing the GitHub profile of Andres Matarrita Miranda (AndresMatarritaC04668).

© 2023 - Laboratorio5 - [Privacy](#)

**Enlace del repositorio de github:**

**[https://github.com/AndresMatarritaC04668/c04668\\_ci0126\\_23a.git](https://github.com/AndresMatarritaC04668/c04668_ci0126_23a.git)**

**Agregue un pequeño resumen, no más de 200 palabras indicando claramente**

- 1. Dos cosas que no sabía y aprendió en el laboratorio**
- 2. Una cosa que se le hizo difícil de realizar y explique por qué fue difícil.**
- 3. Una cosa que se le hizo fácil de realizar y explique por qué fue fácil.**
- 4. Indique cuánto tiempo tardó en realizar el laboratorio.**

En el laboratorio 5 de Ingeniería de Software, se aprenden los conceptos básicos para conectar y manipular datos en una base de datos con un proyecto de ASP.NET MVC usando visual community, ya sea con una base de datos local o remota. Se aprenden nuevos conceptos como cadenas de conexión, creación de tablas en una base de datos y manipulación de datos haciéndolo desde la herramienta visual.

Dos cosas que aprendí en el laboratorio son la conexión con una base de datos remota a través de una VPN y la creación de una conexión con una base de datos local utilizando cadenas de conexión.

Se me hizo difícil conectar el proyecto a una base de datos local debido a que en mi computadora no tenía descargado SQL server por lo que me tomó mucho tiempo darme cuenta de que eso era lo que me producía errores.

Una cosa que se me hizo fácil de realizar es la creación de la conexión con una base de datos local utilizando cadenas de conexión, ya que se explica detalladamente los pasos necesarios para establecer la conexión.

El tiempo de realización que me tomó para realizar el laboratorio fue de 4 horas debido a que no tenía descargado algunas extensiones del SQL Client en visual por lo que daba problemas al crear el Handler de las películas.

**Responder a la pregunta que se realiza en la sección 3.2.1 en relación con los connection strings, con una extensión máxima de 300 palabras.**

La conexión mediante cadena de conexión ofrece varias ventajas en comparación con otros métodos de conexión a una base de datos. En primer lugar, este enfoque simplifica el proceso de conexión, ya que permite configurar los parámetros de conexión en un formato estándar y fácil de entender. Esto significa que los desarrolladores no tienen que preocuparse por los detalles de la conexión, lo que reduce la posibilidad de errores y acelera el proceso de desarrollo.

Además, el uso de la cadena de conexión facilita el mantenimiento y la actualización de la información de conexión. Al tener toda la información de conexión en un solo lugar, se puede actualizar fácilmente sin tener que buscar en el código fuente. Esto también mejora la seguridad, ya que los datos de conexión pueden ser encriptados y protegidos de manera más efectiva.

Otra ventaja de la conexión mediante cadena de conexión es que permite una mayor portabilidad del código. Al separar la información de conexión del código fuente, es posible mover el código a diferentes entornos sin tener que modificar el código en sí. Esto puede ser especialmente útil cuando se trabaja con múltiples bases de datos o cuando se cambia de entorno de desarrollo a producción.

En resumen, la conexión mediante cadena de conexión simplifica y estandariza el proceso de conexión a una base de datos, lo que reduce la posibilidad de errores, mejora la seguridad, facilita el mantenimiento y la actualización, y permite una mayor portabilidad del código.