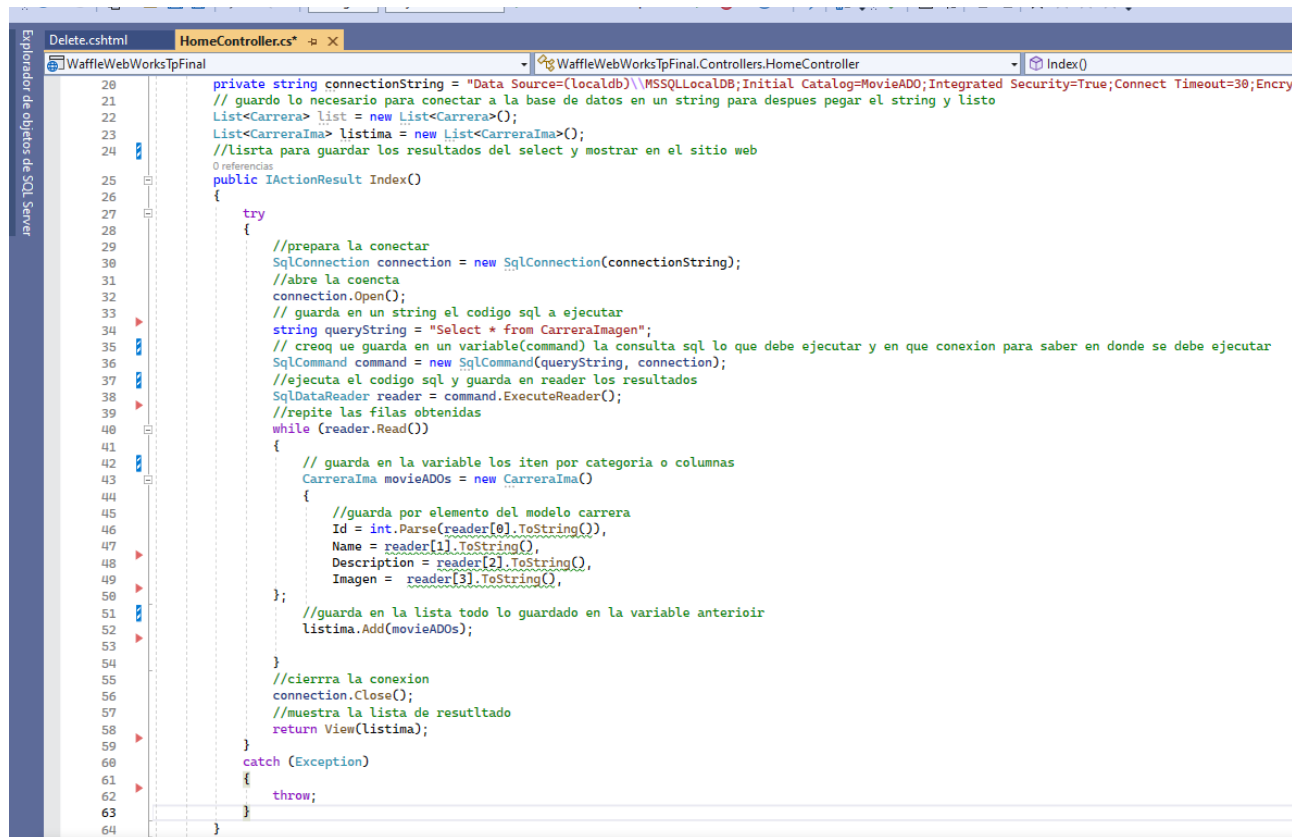


Guia



```
20 private string connectionString = "Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=MovieADO;Integrated Security=True;Connect Timeout=30;Encrypt=False;Trust Server Certificate=False;Application Intent=ReadWrite;Multi Subnet Failover=False";
21 // guardo lo necesario para conectar a la base de datos en un string para despues pegar el string y listo
22 List<Carrera> list = new List<Carrera>();
23 List<CarreraIma> listima = new List<CarreraIma>();
24 //lisrta para guardar los resultados del select y mostrar en el sitio web
25 public IActionResult Index()
26 {
27     try
28     {
29         //prepara la conectar
30         SqlConnection connection = new SqlConnection(connectionString);
31         //abre la coencta
32         connection.Open();
33         // guarda en un string el codigo sql a ejecutar
34         string queryString = "Select * from CarreraImagen";
35         // creo ue guarda en un variable(command) la consulta sql lo que debe ejecutar y en que conexion para saber en donde se debe ejecutar
36         SqlCommand command = new SqlCommand(queryString, connection);
37         //ejecuta el codigo sql y guarda en reader los resultados
38         SqlDataReader reader = command.ExecuteReader();
39         //repite las filas obtenidas
40         while (reader.Read())
41         {
42             // guarda en la variable los iten por categoria o columnas
43             CarreraIma movieADOs = new CarreraIma()
44             {
45                 //guarda por elemento del modelo carrera
46                 Id = int.Parse(reader[0].ToString()),
47                 Name = reader[1].ToString(),
48                 Description = reader[2].ToString(),
49                 Imagen = reader[3].ToString(),
50             };
51             //guarda en la lista todo lo guardado en la variable anterior
52             listima.Add(movieADOs);
53         }
54         //cierra la conexion
55         connection.Close();
56         //muestra la lista de resultado
57         return View(listima);
58     }
59     catch (Exception)
60     {
61         throw;
62     }
63 }
64 }
```

```
private string connectionString = "Data Source=(localdb)\\MSSQLLocalDB;Initial Catalog=MovieADO;Integrated Security=True;Connect Timeout=30;Encrypt=False;Trust Server Certificate=False;Application Intent=ReadWrite;Multi Subnet Failover=False";
```

// guardo lo necesario para conectar a la base de datos en un string para despues pegar el string y listo

```
List<Carrera> list = new List<Carrera>();
```

```
List<CarreraIma> listima = new List<CarreraIma>();
```

//lisrta para guardar los resultados del select y mostrar en el sitio web

```
public IActionResult Index()
```

```
{
```

```
    try
```

```
    {
```

```
        //prepara la conectar
```

```
        SqlConnection connection = new SqlConnection(connectionString);
```

```
        //abre la coencta
```

```
        connection.Open();
```

```
        // guarda en un string el codigo sql a ejecutar
```

```
        string queryString = "Select * from CarreraImagen";
```

// creo que guarda en un variable(command) la consulta sql lo que debe ejecutar y en que conexion para saber en donde se debe ejecutar

```
SqlCommand command = new SqlCommand(queryString, connection);
```

//ejecuta el codigo sql y guarda en reader los resultados (ejecuta y lee el resultado)

```
SqlDataReader reader = command.ExecuteReader();
```

//repite las filas obtenidas

```
while (reader.Read())
```

```
{
```

// guarda en la variable los item por categoria o columnas

```
Carreralma movieADOs = new Carreralma()
```

```
{
```

//guarda por elemento del modelo carrera

```
Id = int.Parse(reader[0].ToString()),
```

```
Name = reader[1].ToString(),
```

```
Description = reader[2].ToString(),
```

```
Imagen = reader[3].ToString(),
```

```
};
```

//guarda en la lista todo lo guardado en la variable anterior

```
listima.Add(movieADOs);
```

```
}
```

//cierra la conexion

```
connection.Close();
```

//muestra la lista de resultado

```
return View(listima);
```

```
}
```

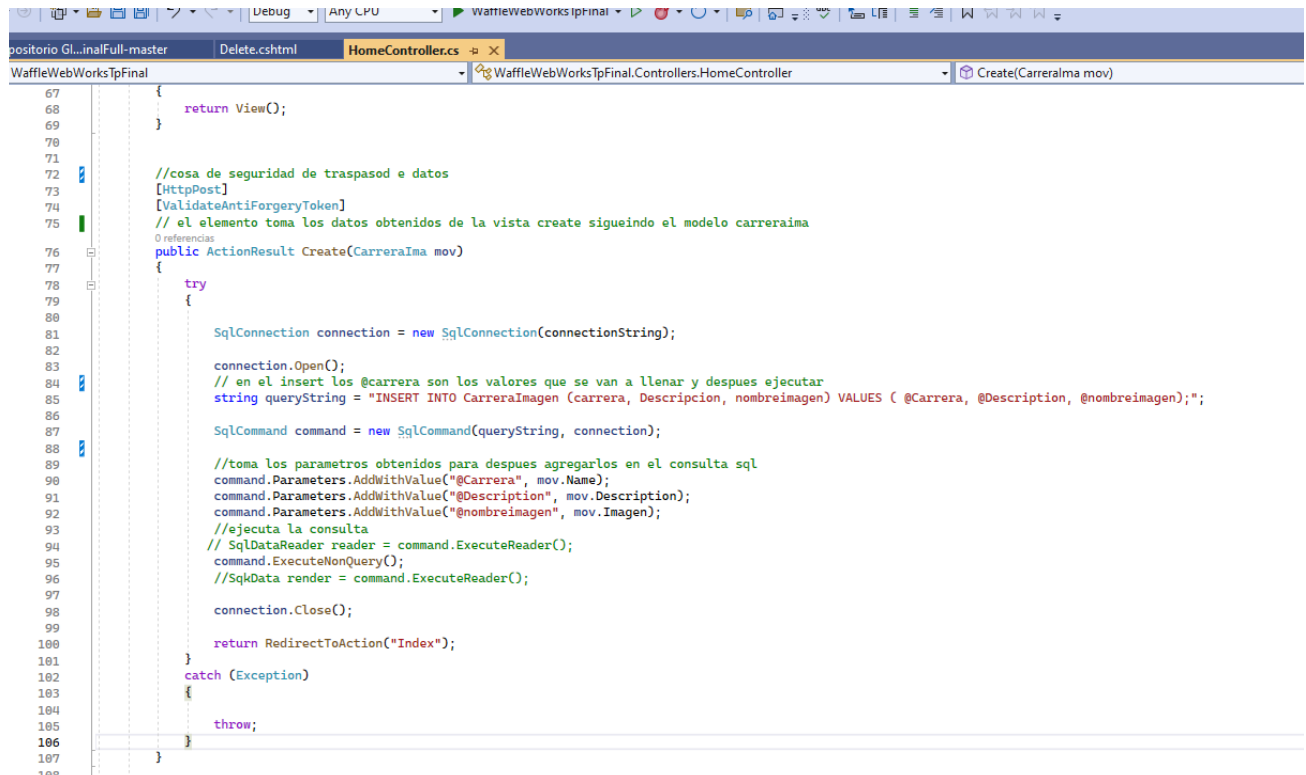
```
catch (Exception)
```

```
{
```

```
throw;
```

```
}
```

```
}
```



```
67 {
68     return View();
69 }
70
71
72 //cosa de seguridad de traspasod e datos
73 [HttpPost]
74 [ValidateAntiForgeryToken]
75 // el elemento toma los datos obtenidos de la vista create siguiendo el modelo carreraima
76 public ActionResult Create(CarreraIma mov)
77 {
78     try
79     {
80
81         SqlConnection connection = new SqlConnection(connectionString);
82
83         connection.Open();
84         // en el insert los @carrera son los valores que se van a llenar y despues ejecutar
85         string queryString = "INSERT INTO CarreraImagen (carrera, Descripcion, nombreimagen) VALUES ( @Carrera, @Descripcion, @nombreimagen);";
86
87         SqlCommand command = new SqlCommand(queryString, connection);
88
89         //toma los parametros obtenidos para despues agregarlos en el consulta sql
90         command.Parameters.AddWithValue("@Carrera", mov.Name);
91         command.Parameters.AddWithValue("@Descripcion", mov.Description);
92         command.Parameters.AddWithValue("@nombreimagen", mov.Imagen);
93         //ejecuta la consulta
94         // SqlDataReader reader = command.ExecuteReader();
95         command.ExecuteNonQuery();
96         //SqlDataReader reader = command.ExecuteReader();
97
98         connection.Close();
99
100         return RedirectToAction("Index");
101     }
102     catch (Exception)
103     {
104     }
105     throw;
106 }
107 }
```

//cosa de seguridad de traspasod e datos

[HttpPost]

[ValidateAntiForgeryToken]

// el elemento toma los datos obtenidos de la vista create siguiendo el modelo carreraima

public ActionResult Create(CarreraIma mov)

{

try

{

SqlConnection connection = new SqlConnection(connectionString);

connection.Open();

// en el insert los @carrera son los valores que se van a llenar y despues ejecutar
string queryString = "INSERT INTO CarreraImagen (carrera, Descripcion,
nombreimagen) VALUES (@Carrera, @Descripcion, @nombreimagen);";

SqlCommand command = new SqlCommand(queryString, connection);

//toma los parametros obtenidos para despues agregarlos en el consulta sql

command.Parameters.AddWithValue("@Carrera", mov.Name);

command.Parameters.AddWithValue("@Descripcion", mov.Description);

command.Parameters.AddWithValue("@nombreimagen", mov.Imagen);

//ejecuta la consulta

// ejecuta la consulta una sola vez no lee los resultados

command.ExecuteNonQuery();

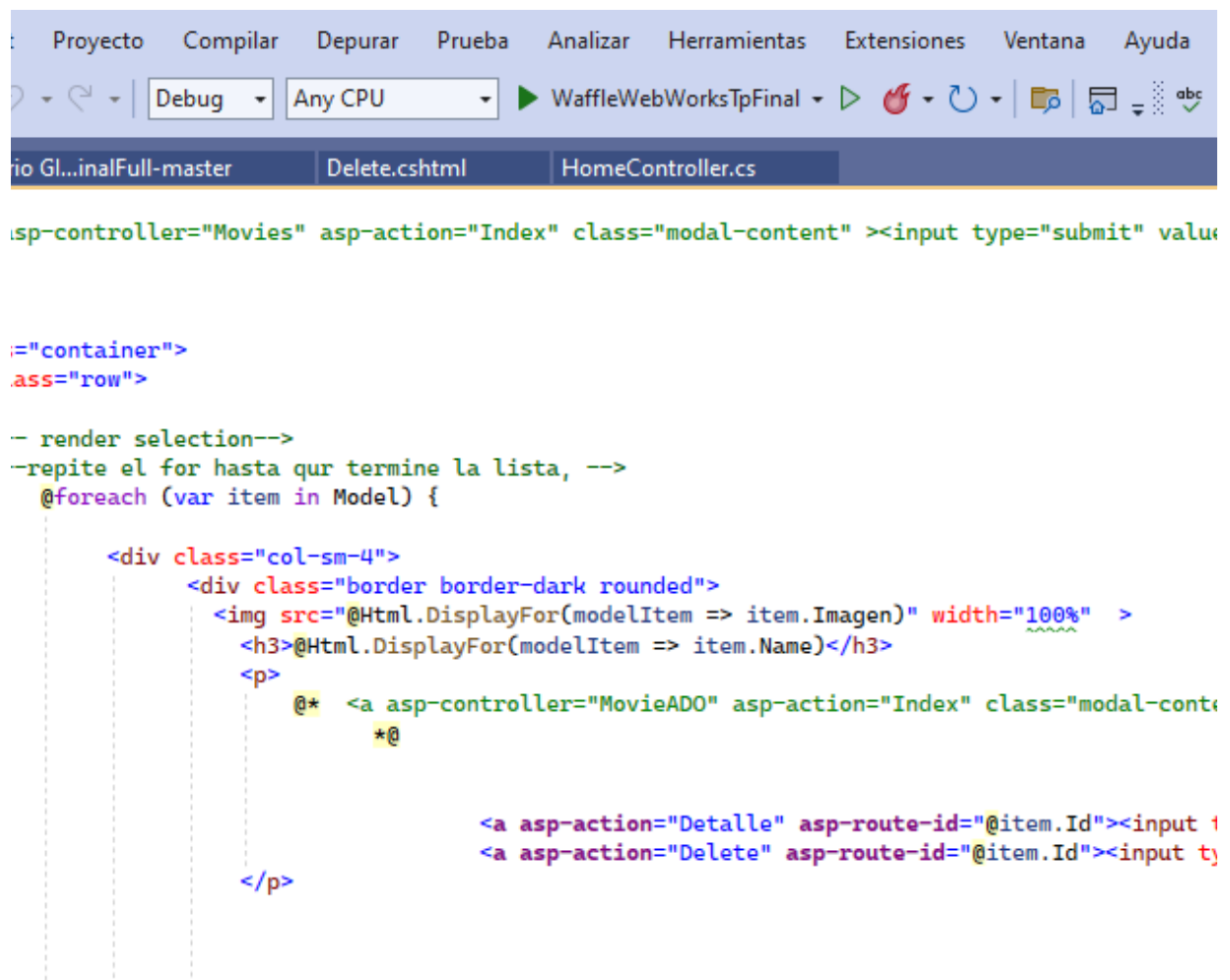
```

        connection.Close();

        return RedirectToAction("Index");
    }
    catch (Exception)
    {

        throw;
    }
}

```



```

asp-controller="Movies" asp-action="Index" class="modal-content" ><input type="submit" value=

:= "container">
.class="row">

-- render selection-->
-- repite el for hasta que termine la lista, -->
@foreach (var item in Model) {

    <div class="col-sm-4">
        <div class="border border-dark rounded">
            
            <h3>@Html.DisplayFor(modelItem => item.Name)</h3>
            <p>
                @* <a asp-controller="MovieADO" asp-action="Index" class="modal-conta
                *@

                <a asp-action="Detalle" asp-route-id="@item.Id"><input type="button" value="Detalle" />
                <a asp-action="Delete" asp-route-id="@item.Id"><input type="button" value="Delete" />

            </p>
        </div>
    </div>
}

```

@model IEnumerable<WaffleWebWorksTpFinal.Models.CarreraAlma>

El @Html.DisplayFor(modelItem => item. imagen)
 Es el valor es el valor almacenado en la base de datos item.Name hace referencia a la columna nombre o carrera de la base de datos con su valor determinado

<a asp-action="Detalle"(redirecciona a la vista detalle) asp-route-id="@item.Id" (manda el id de un solo elemento a la vista detalle)