

CRUD REACT-NET

Creación del proyecto: => dotnet new weapi --o NombreProyecto

Navegar hacia la carpeta: => cd nombreProyecto

Crear los archivos de configuración => run -> genera la carpeta .vscode

- 1- En la carpeta principal crear la carpeta Entidades, esta carpeta guardará todas las clases para cada tabla de la base de datos. **Entidades=>Libro.cs**

```
WebApiLibro > Entidades > C# Libro.cs > Libro
1 namespace WebApiLibro.Entidades;
2 public class Libro//nombre de la tabla
3 {
4
5     0 references
6     public int Id { get; set; }
7
8     0 references
9     public string? Titulo { get; set; }
10
11     0 references
12     public string? Autor { get; set; }
13 }
```

```
WebApiLibro > Entidades > C# Libro.cs > Libro
1 namespace WebApiLibro.Entidades;
2 public class Libro//nombre de la tabla
3 {
4
5     0 references
6     public int Id { get; set; }
7
8     0 references
9     public string? Titulo { get; set; }
10
11     0 references
12     public string? Autor { get; set; }
13 }
```

usando
nombre del proyecto
carpeta donde están todas las clases de las tablas

```
--CREATE TABLE Libro
(
  Id INT PRIMARY KEY IDENTITY(1,1),
  titulo NVARCHAR(1000) NOT NULL,
  Autor NVARCHAR(1000) NOT NULL,
);
```

Campos de la tabla: La inicial en mayúscula,
no importa como estén en la tabla.

- 2- En la carpeta principal crear la carpeta **Context=>dataContext.cs**

```
WebApiLibro > context > C# dataContext.cs > DataContext
1 using Microsoft.EntityFrameworkCore;
2 using WebApiLibro.Entidades;
3
4 public class DataContext:DbContext
5 {
6     5 references
7     public DataContext(DbContextOptions<DataContext>options):base(options){}
8     //Entidad que mapea en .net Libro.cs que contiene la clase Libro
9     public DbSet<Libro> Libro{get;set;}//public DbSet<Clase que mapea> nomTabla{get;set;}
10
11     //si hay más tablas se van añadiendo aquí
12 }
```

Framework:
EntityFrameworkCore

```
WebApiLibro > context > C# dataContext.cs > DataContext
1 using Microsoft.EntityFrameworkCore;
2 using WebApiLibro.Entidades;
3
4 public class DataContext:DbContext
5 {
6     5 references
7     public DataContext(DbContextOptions<DataContext>options):base(options){}
8     //Entidad que mapea en .net Libro.cs que contiene la clase Libro
9     public DbSet<Libro> Libro{get;set;}//public DbSet<Clase que mapea> nomTabla{get;set;}
10
11     //si hay más tablas se van añadiendo aquí
12 }
```

usando
nombre del proyecto
carpeta donde están todas las clases de las tablas

Entidades
C# Libro.cs

```
--CREATE TABLE Libro
(
  Id INT PRIMARY KEY IDENTITY(1,1),
  titulo NVARCHAR(1000) NOT NULL,
  Autor NVARCHAR(1000) NOT NULL,
);
```

3- En la carpeta **program.cs**

```
WebApiLibro > C# Program.cs
1 using Microsoft.EntityFrameworkCore;
2
3
4 var builder = WebApplication.CreateBuilder(args);
5 //---Cadena de conexión
6 builder.Services.AddDbContext<DataContext>
7 (options=>options.UseSqlServer
8 (@"Data Source=PORTATIL\SQLEXPRESS;Initial Catalog=biblioteca;User Id=sa;Password=rootadmin;Encrypt=false"));
9
10 // Add services to the container.
11
12 builder.Services.AddControllers();
13 // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
14 builder.Services.AddEndpointsApiExplorer();
15 builder.Services.AddSwaggerGen();
16 //---distinto dominio habilitar las cors
17 builder.Services.AddCors(options=>
18 {
19     options.AddPolicy("AllowAll", builder =>{
20         builder.AllowAnyOrigin()
21             .AllowAnyMethod()
22             .AllowAnyHeader();
23     });
24 });
25
26
27 var app = builder.Build();
28 //---añadir para que no bloquee en el front
29 app.UseCors("AllowAll");
30
```

```
1 using Microsoft.EntityFrameworkCore;
2
3
4 var builder = WebApplication.CreateBuilder(args);
5 //---Cadena de conexión
6 builder.Services.AddDbContext<DataContext>
7 (options=>options.UseSqlServer
8 (@"Data Source=PORTATIL\SQLEXPRESS;Initial Catalog=biblioteca;User Id=sa;Password=rootadmin;Encrypt=false"));
9
10 // Add services to the container.
11
12 builder.Services.AddControllers();
13 // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
14 builder.Services.AddEndpointsApiExplorer();
15 builder.Services.AddSwaggerGen();
16 //---distinto dominio habilitar las cors
17 builder.Services.AddCors(options=>
18 {
19     options.AddPolicy("AllowAll", builder =>{
20         builder.AllowAnyOrigin()
21             .AllowAnyMethod()
22             .AllowAnyHeader();
23     });
24 });
25
26
27 var app = builder.Build();
28 //---añadir para que no bloquee en el front
29 app.UseCors("AllowAll");
30
```

Autenticación para acceder a la Base de datos

Nombre Base de datos

Usuario y contraseña

Escapar

Cuando hay distinto dominio para el front y para el back habilitar las cors

Activa

Lo demás en el archivo se queda tal cual.

```
31 // Configure the HTTP request pipeline.
32 if (app.Environment.IsDevelopment())
33 {
34     app.UseSwagger();
35     app.UseSwaggerUI();
36 }
37
38 app.UseHttpsRedirection();
39
40 app.UseAuthorization();
41
42 app.MapControllers();
43
44 app.Run();
```

- 4- Se crea el controlador para la tabla, en la carpeta '**controllers**' se crea el archivo, ya viene uno de ejemplo por lo que se copia y se le cambia el nombre.

```
WebApiLibro > Controllers > C# LibrosController.cs > LibrosController
1 using Microsoft.AspNetCore.Mvc;
2
3 namespace WebApiLibro.Controllers;
4
5 [ApiController]
6 [Route("[controller]")]
7 public class LibrosController : ControllerBase
8 {
9     private static readonly string[] Summaries = new[]
10     {
11         "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"
12     };
13
14     private readonly ILogger<LibrosController> _logger;
15     private readonly DataContext _dataContext; //accede a la base de datos
16
17     public LibrosController(ILogger<LibrosController> logger, DataContext dataContext) //pasa el acceso a la bse de datos
18     {
19         _logger = logger;
20         _dataContext = dataContext;
21     }
22 }
```

```
WebApiLibro > Controllers > C# LibrosController.cs > LibrosController
1 using Microsoft.AspNetCore.Mvc;
2
3 namespace WebApiLibro.Controllers;
4
5 [ApiController]
6 [Route("[controller]")]
7 public class LibrosController : ControllerBase
8 {
9     private static readonly string[] Summaries = new[]
10     {
11         "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"
12     };
13
14     private readonly ILogger<LibrosController> _logger;
15     private readonly DataContext _dataContext; //accede a la base de datos
16
17     public LibrosController(ILogger<LibrosController> logger, DataContext dataContext) //pasa el acceso a la bse de datos
18     {
19         _logger = logger;
20         _dataContext = dataContext;
21     }
22 }
```

Se crea una variable para acceder a la base de datos

Se pasa el acceso por parámetro en el constructor y se declara

```
[HttpGet]//obtener los datos de la tabla libro
0 references
public ActionResult Get()
{
    return Ok(_dataContext.Libro);
}

[HttpPost]//añadir registro
0 references
public ActionResult Add(Libro element)
{
    _dataContext.Libro.Add(element); //añade una fila nueva con el nuevo registro en la tabla Libro
    _dataContext.SaveChanges(); //guarda los cambios
    return Ok(element); //devuelve un ok del elemento
}
}
```

```
[HttpDelete("{id}")]//Eliminar registro
0 references
public IActionResult Delete(int id)
{
    var itemToDelete = _dataContext.Libro.FirstOrDefault(w => w.Id == id);
    if(itemToDelete == null)
    {
        return NotFound();
    }
    _dataContext.Libro.Remove(itemToDelete);
    _dataContext.SaveChanges(); //guarda los cambios
    return NoContent(); //devuelve un ok del elemento
}
}
```

```

[HttpPut]//Modificar registro
0 references
public IActionResult UpDate(Libro updatedItem)
{
    var existingItem = _dataContext.Libro.FirstOrDefault(w => w.Id == updatedItem.Id);
    if(existingItem == null)
    {
        return NotFound();//devuelve un 404 si el elemento no se encuentra
    }
    // Actualiza los campos necesarios del elemento existente
    existingItem.Título = updatedItem.Título;
    existingItem.Autor = updatedItem.Autor;

    _dataContext.SaveChanges();//guarda los cambios
    return NoContent();//Devuelve un 204 despues de actualizar el elemento
}

```

[HttpGet]//obtener los datos de la tabla libro

```

public ActionResult Get()

{

    return Ok(_dataContext.Libro);

}

```

[HttpPost]//añadir registro

```

public ActionResult Add(Libro element)
{

    _dataContext.Libro.Add(element);//añade una fila nueva con el nuevo registro en la tabla Libro

    _dataContext.SaveChanges(); //guarda los cambios

    return Ok(element);//devuelve un ok del elemento

}

```

[HttpDelete("{id}")]//Eliminar registro

```

public IActionResult Delete(int id)
{

    var itemToDelete = _dataContext.Libro.FirstOrDefault(w => w.Id == id);

    if(itemToDelete == null)

    {

        return NotFound();

    }

    _dataContext.Libro.Remove(itemToDelete);

    _dataContext.SaveChanges();//guarda los cambios

    return NoContent();//devuelve un ok del elemento
}

```

```

}

[HttpPut]//Modificar registro
public IActionResult UpDate(Libro updatedItem)
{
    var existingItem = _dataContext.Libro.FirstOrDefault(w => w.Id == updatedItem.Id);
    if(existingItem == null)
    {
        return NotFound();//devuelve un 404 si el elemento no se encuentra
    }

    // Actualiza los campos necesarios del elemento existente
    existingItem.Título = updatedItem.Título;
    existingItem.Autor = updatedItem.Autor;

    _dataContext.SaveChanges();//guarda los cambios
    return NoContent();//Devuelve un 204 despues de actualizar el elemento
}

```

Añadir los entityFramework, en el archivo con extension => **csproj**

```

<ItemGroup>
  <PackageReference Include="Microsoft.AspNetCore.OpenApi" Version="7.0.12" />
  <PackageReference Include="Microsoft.EntityFrameworkCore" Version="7.0.12" />
  <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="7.0.12">
    <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
    <PrivateAssets>all</PrivateAssets>
  </PackageReference>
  <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="7.0.12" />
  <PackageReference Include="Swashbuckle.AspNetCore" Version="6.5.0" />
</ItemGroup>

```