.NET y SWAGGER (2022)

[Servidor ¿Who is? 2](#_Toc189590111)

[Introducción Front (HTML) 2](#_Toc189590112)

[Tablas 3](#_Toc189590113)

[Botones 3](#_Toc189590114)

[Menú y elemento modal 6](#_Toc189590115)

[Servicios Web 9](#_Toc189590116)

[SOAP 9](#_Toc189590117)

[REST 10](#_Toc189590118)

[Estructura de una URL 10](#_Toc189590119)

[Métodos HTTP 11](#_Toc189590120)

[JSON y XML 11](#_Toc189590121)

[MVC y REST 12](#_Toc189590122)

[MVC 12](#_Toc189590123)

[REST 13](#_Toc189590124)

[Creación de un API Rest en .Net 13](#_Toc189590125)

[Pasos 13](#_Toc189590126)

[Validaciones 17](#_Toc189590127)

[Status code 17](#_Toc189590128)

[Xampp 19](#_Toc189590129)

[Conexión a la BBDD 20](#_Toc189590130)

[Herramienta para probar nuestra API 22](#_Toc189590131)

[Herramienta Documentación 22](#_Toc189590132)

[Solución a la creación de la API 22](#_Toc189590133)

[Conexión entre Back and Front-end 28](#_Toc189590134)

[1. CORS (Cross-Origin Resource Sharing) 28](#_Toc189590135)

[Cómo habilitar CORS en .NET 28](#_Toc189590136)

[Creación de nuestro Script 30](#_Toc189590137)

[Vista final del proyecto 35](#_Toc189590138)

# Servidor ¿Who is?

# Introducción Front (HTML)

Interfaz de usuario gráfica

Descripción generada automáticamente

**Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <b><u>Hola</u> mundo</b> <!-- b: negrita | u: subrayado -->

    <br>

    <br/> <!-- br: salto de línea -->

    <br/> <!-- br: salto de línea -->

    Esto es html!

    <br>

    <iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d46405.72502209565!2d-5.889857618056057!3d43.36954185097312!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0xd368c9a60ac1c67%3A0x3134440ecc5e6224!2sOviedo%2C%20Asturias!5e0!3m2!1ses!2ses!4v1738657338169!5m2!1ses!2ses" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy" referrerpolicy="no-referrer-when-downgrade"></iframe>

    <br>

    <iframe width="560" height="315" src="https://www.youtube.com/embed/pjg\_09tDltY?si=GUtZ3j0L-aIAn8AQ" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" referrerpolicy="strict-origin-when-cross-origin" allowfullscreen></iframe>

</body>

</html>

## Tablas

* Buscar en Google: style tables css tool
* Página de referencia (ejemplo): <https://www.w3schools.com/css/css_table.asp>

## Botones

* Buscar en Google: style button tool
* Página de ejemplo: <https://www.bestcssbuttongenerator.com/#/13>

Pinchas en los primeros enlaces (infinidad de herramientas)

**NOTA**: *Siempre que se trabaja en front-end, se usa nada más que HTML, CSS y JavaScript*

**Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Index</title>

</head>

<body>

    <h1>Gestión de Clientes</h1>

    <table id="customers">

        <tr>

            <th>Nombre</th>

            <th>Apellido</th>

            <th>Email</th>

            <th>Teléfono</th>

            <th>Acciones</th>

        </tr>

        <tr>

            <td>Germán</td>

            <td>Iglesias</td>

            <td>german@terra.es</td>

            <td>666999000</td>

            <td>

                <a href="#" class="myButton">Editar</a>

                <a href="#" class="myButtonDelete">Eliminar</a>

            </td>

        </tr>

        <tr>

            <td>José</td>

            <td>Mourinho</td>

            <td>jose@terra.es</td>

            <td>666111222</td>

            <td>

                <a href="#" class="myButton">Editar</a>

                <a href="#" class="myButtonDelete">Eliminar</a>

            </td>

        </tr>

    </table>

    <style>

        #customers {

            font-family: Arial, Helvetica, sans-serif;

            border-collapse: collapse;

            width: 100%;

        }

        #customers td,

        #customers th {

            border: 1px solid #ddd;

            padding: 8px;

        }

        #customers tr:nth-child(even) {

            background-color: #f2f2f2;

        }

        #customers tr:hover {

            background-color: #ddd;

        }

        #customers th {

            padding-top: 12px;

            padding-bottom: 12px;

            text-align: left;

            background-color: #04AA6D;

            color: white;

        }

        .myButton {

            box-shadow: 0px 1px 0px 0px #f0f7fa;

            background: linear-gradient(to bottom, #33bdef 5%, #019ad2 100%);

            background-color: #33bdef;

            border-radius: 6px;

            border: 1px solid #057fd0;

            display: inline-block;

            cursor: pointer;

            color: #ffffff;

            font-family: Arial;

            font-size: 15px;

            font-weight: bold;

            padding: 6px 24px;

            text-decoration: none;

            text-shadow: 0px -1px 0px #5b6178;

        }

        .myButton:hover {

            background: linear-gradient(to bottom, #019ad2 5%, #33bdef 100%);

            background-color: #019ad2;

        }

        .myButton:active {

            position: relative;

            top: 1px;

        }

        .myButtonDelete {

    box-shadow: 0px 1px 0px 0px #f0f7fa;

    background:linear-gradient(to bottom, #ed3434 5%, #d10214 100%);

    background-color:#ed3434;

    border-radius:6px;

    border:1px solid #057fd0;

    display:inline-block;

    cursor:pointer;

    color:#ffffff;

    font-family:Arial;

    font-size:15px;

    font-weight:bold;

    padding:6px 24px;

    text-decoration:none;

    text-shadow:0px -1px 0px #5b6178;

}

.myButtonDelete:hover {

    background:linear-gradient(to bottom, #d10214 5%, #ed3434 100%);

    background-color:#d10214;

}

.myButtonDelete:active {

    position:relative;

    top:1px;

}

    </style>

</body>

</html>

**Página web**

Tabla

Descripción generada automáticamente con confianza media

**NOTA***: Página de Referencia:* <https://codepen.io/>

## Menú y elemento modal

**Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="style.css"> <!-- Enlaza el archivo style.css -->

    <link rel="stylesheet" href="menu.css">

    <title>Index</title>

</head>

<nav id='menu'>

    <input type='checkbox' id='responsive-menu' onclick='updatemenu()'><label></label>

    <ul>

        <li><a href='http://'>Home</a></li>

        <li><a class='dropdown-arrow' href='http://'>Products</a>

            <ul class='sub-menus'>

                <li><a href='http://'>Products 1</a></li>

                <li><a href='http://'>Products 2</a></li>

                <li><a href='http://'>Products 3</a></li>

                <li><a href='http://'>Products 4</a></li>

            </ul>

        </li>

        <li><a href='http://'>About</a></li>

        <li><a class='dropdown-arrow' href='http://'>Services</a>

            <ul class='sub-menus'>

                <li><a href='http://'>Services 1</a></li>

                <li><a href='http://'>Services 2</a></li>

                <li><a href='http://'>Services 3</a></li>

            </ul>

        </li>

        <li><a href='http://'>Contact Us</a></li>

    </ul>

</nav>

<body>

    <h1>Gestión de Clientes</h1>

    <a onclick="agregarCliente()" href="#" class="myButton">Agregar Cliente</a>

    <table id="customers">

        <tr>

            <th>Nombre</th>

            <th>Apellido</th>

            <th>Email</th>

            <th>Teléfono</th>

            <th>Acciones</th>

        </tr>

        <tr>

            <td>Germán</td>

            <td>Iglesias</td>

            <td>german@terra.es</td>

            <td>666999000</td>

            <td>

                <a href="#" class="myButton">Editar</a>

                <a href="#" class="myButtonDelete">Eliminar</a>

            </td>

        </tr>

        <tr>

            <td>José</td>

            <td>Mourinho</td>

            <td>jose@terra.es</td>

            <td>666111222</td>

            <td>

                <a href="#" class="myButton">Editar</a>

                <a href="#" class="myButtonDelete">Eliminar</a>

            </td>

        </tr>

    </table>

    <!-- Modal -->

    <div id="modal" class="modale" aria-hidden="true">

        <div class="modal-dialog">

            <div class="modal-header">

                <h2>Datos del Cliente</h2>

                <a onclick="cerrarCliente()" href="#" class="btn-close closemodale" aria-hidden="true">&times;</a>

            </div>

            <div class="modal-body">

                <input type="text" name="u" placeholder="Nombre" size="20" /><br/>

                <input type="text" name="u" placeholder="Apellido" size="20" /><br/>

                <input type="text" name="u" placeholder="Teléfono" size="20" /><br/>

                <input type="text" name="u" placeholder="Email" size="20" /><br/>

            </div>

            <div class="modal-footer">

                <a href="#" class="btn" id="btn\_ingresar">Guardar</a>

            </div>

        </div>

    </div>

    <!-- /Modal -->

</body>

<script>

    //alert(1);//Muestra un mensaje de alerta

    function agregarCliente() {

        //alert("Agregar Cliente");//Muestra un mensaje de alerta

        htmlModal = document.getElementById("modal"); //Obtiene el elemento con el id modal

        htmlModal.setAttribute("class", "modale opened"); //Agrega la clase open al elemento modal

    }

    function cerrarCliente() {

        //alert("Agregar Cliente");//Muestra un mensaje de alerta

        htmlModal = document.getElementById("modal"); //Obtiene el elemento con el id modal

        htmlModal.setAttribute("class", "modale"); //Agrega la clase open al elemento modal

    }

</script>

</html>

**Vista.html**

Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente

# Servicios Web

## SOAP

Interfaz de usuario gráfica

Descripción generada automáticamente

## REST

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

# Estructura de una URL

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Enlace** 🡪 <https://jarroba.com/cliente-servidor-peticion-del-cliente/>

# Métodos HTTP

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

# JSON y XML

Comunicación entre Servicios web

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

# MVC y REST

## MVC

Diagrama

Descripción generada automáticamente

Diagrama

Descripción generada automáticamente

**NOTA**: uso de Ajax (envió de emails), cientos de miles de proyectos, etc

## REST



# Creación de un API Rest en .Net

## Pasos

1. Creas un proyecto API core
2. versión 6
3. Habilita los controladores y OpenIA

Una captura de pantalla de una computadora

Descripción generada automáticamente

1. Agrega un nuevo controlador

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Ejecutando el código donde se construye el swagger…

**NOTA**: ¡pon de explorador a *Microsoft Edge*, con *Google* tendrás problemas!

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Para definir o modificar y especificar otro número del puerto en tu proyecto ve a launchsettings.json de la carpeta Properties

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

{

"$schema": "https://json.schemastore.org/launchsettings.json",

"iisSettings": {

"windowsAuthentication": false,

"anonymousAuthentication": true,

"iisExpress": {

"applicationUrl": "http://localhost:43668",

"sslPort": 44304

}

},

"profiles": {

"CustomersApi": {

"commandName": "Project",

"dotnetRunMessages": true,

"launchBrowser": true,

"launchUrl": "swagger",

"applicationUrl": "https://localhost:7163;http://localhost:5168",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

},

"IIS Express": {

"commandName": "IISExpress",

"launchBrowser": true,

"launchUrl": "swagger",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

}

}

}

## Validaciones

**CreateCustomerDto.cs**

using System.ComponentModel.DataAnnotations;

namespace CustomersApi.Dtos

{

public class CreateCustomerDto

{

[Required (ErrorMessage = "El nombre propio tiene que estar especificado")]

public string FirstName { get; set; }

[Required(ErrorMessage = "El apellido propio tiene que estar especificado")]

public string LastName { get; set; }

[RegularExpression(@"^[^@\s]+@[^@\s]+\.[^@\s]+$", ErrorMessage = "El formato del email no es válido.")]

public string Email { get; set; }

public string Address { get; set; }

public string Phone { get; set; }

}

}

## Status code

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

* **200** OK
* **400** mala respuesta
* **401** no autorización
* **403** el cliente no posee los permisos. El cliente si está identificado, pero no tiene permisos
* **404** el recurso no lo pudo encontrar
* **500** el servidor se encuentra con un problema
* **503** el servidor no esta listo para manejar la petición. El servidor esta sobrecargado o caído

**CustomerController.cs**

using CustomersApi.Dtos;

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace CustomersApi.Controllers

{

[ApiController]

[Route("api/[controller]")]

public class CustomerController : Controller

{

//api/customer/

[HttpGet]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(CustomerDto))]

public async Task<IActionResult> GetCustomers()

{

throw new NotImplementedException();

}

//api/customer/{id}

[HttpGet("{id}")]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(CustomerDto))]

[ProducesResponseType(StatusCodes.Status404NotFound)]

public async Task<IActionResult> GetCustomer(long id)

{

var vacio = new CustomerDto();

return new OkObjectResult(vacio);

}

[HttpDelete("{id}")]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(bool))]

public async Task<IActionResult> DeleteCustomer(long id)

{

throw new NotImplementedException();

}

[HttpPost]

[ProducesResponseType(StatusCodes.Status201Created, Type = typeof(CustomerDto))]

public async Task<IActionResult> CreateCustomer(CreateCustomerDto customer)

{

var vacio = new CustomerDto();

return new CreatedResult($"https://localhost:7163/api/customer/{vacio.Id}", null);

}

[HttpPut]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(CustomerDto))]

[ProducesResponseType(StatusCodes.Status404NotFound)]

public async Task<IActionResult> UpdateCustomer(CustomerDto customer)

{

throw new NotImplementedException();

}

}

}

## Xampp

Bien corre tanto php(Apache) como MySql

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Ve al navegador y pon: <http://localhost/phpmyadmin/>

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Crear BBDD

Nombre y UTF8\_bin

Crear la tabla

**Nota**: acuérdate en el id de poner autoIncrementar

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Inserta valores

## Conexión a la BBDD

**EntityFramework**: librería dentro de **Nugget**

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Instala entityFrameworkCore

**NOTA**: ¡cuidado con la versión!, no tiene por qué ser la última, mira que sea lo mas cercana y compatible a tu versión inicial (en tu caso fue la seis)

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Y creas una clase nuevo “DataContext”

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

namespace CustomersApi.Repositories

{

public class CustomerDatabaseContext : DbContext

{

}

}

Importante añadir una clase parecida al CustomerDto, pero con otro nombre. Nos resulta útil, aunque sean iguales para que se puede acceder de dos sitios (Acuérdate de RI, dtos para bussines y para persistence)

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

namespace CustomersApi.Repositories

{

public class CustomerDatabaseContext : DbContext

{

}

public class CustomerEntity

{

public long Id { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Address { get; set; }

public string Phone { get; set; }

}

}

Accede a la BBDD y establece una conexión asíncrona (await) 🡪 Buenas practicas

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

namespace CustomersApi.Repositories

{

public class CustomerDatabaseContext : DbContext

{

public DbSet<CustomerEntity> Customer { get; set; } // acceso a la BBDD

public async Task<CustomerEntity> Get(long id)

{

return await Customer.FirstAsync(x => x.Id == id); // comunica asincrona

}

}

public class CustomerEntity

{

public long Id { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Address { get; set; }

public string Phone { get; set; }

}

}

Necesitaras volver a Nuget e instalar otro paquete en este caso mysql.entityCore

Una captura de pantalla de una computadora

Descripción generada automáticamente

## Herramienta para probar nuestra API

**Postman**

## Herramienta Documentación

**Swagger** es un conjunto de herramientas y especificaciones que facilita la creación, documentación, diseño, consumo y pruebas de APIs RESTful. Es ampliamente utilizado en el desarrollo de servicios web para comunicar de manera clara cómo funcionan las APIs y cómo los clientes pueden interactuar con ellas.

## Solución a la creación de la API

**Program.cs**

using CustomersApi.CasosDeUso;

using CustomersApi.Repositories;

using Microsoft.EntityFrameworkCore;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers(); // añade el controlador que hemos creado

// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

builder.Services.AddRouting(routing => routing.LowercaseUrls = true); // minuscula en la ruta . De https://localhost:7163/api/Customer a https://localhost:7163/api/customer

builder.Services.AddDbContext<CustomerDatabaseContext>(mysqlBuilder =>

{

//builder.UseMySQL("Server=localhost;Port=3306;Database=system;Uid=root;pwd="); // conexion string a nuestra bbdd

mysqlBuilder.UseMySQL(builder.Configuration.GetConnectionString("Connection1"));

});

builder.Services.AddScoped<IUpdateCustomerUseCase, UpdateCustomerUseCase>();

var app = builder.Build(); // construye una app

// Configure the HTTP request pipeline.

if (app.Environment.IsDevelopment()) // swagger solo desarrollo

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseHttpsRedirection();

app.UseAuthorization();

// ESTO es lógica. No se recomienda ponerlo aqui

//app.MapGet("/customer/{id}", (long id) =>

//{

// return "net 6";

//});

app.MapControllers();

app.Run();

**appsettings.json**

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning"

}

},

"ConnectionStrings": {

"Connection1": "Server=localhost;Port=3306;Database=system;Uid=root;pwd="

},

"AllowedHosts": "\*"

}

**CustomerDatabaseContext.cs**

using CustomersApi.Dtos;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Microsoft.EntityFrameworkCore.ChangeTracking;

namespace CustomersApi.Repositories

{

public class CustomerDatabaseContext : DbContext

{

public CustomerDatabaseContext(DbContextOptions<CustomerDatabaseContext> options) : base(options)

{

}

public DbSet<CustomerEntity> Customer { get; set; } // acceso a la BBDD

public async Task<CustomerEntity?> Get(long id)

{

return await Customer.FirstOrDefaultAsync(x => x.Id == id); // comunica asincrona

}

/// <summary>

/// Añade un customer a la BBDD

/// </summary>

/// <param name="customerDto"></param>

/// <returns></returns>

/// <exception cref="Exception"></exception>

public async Task<CustomerEntity> Add(CreateCustomerDto customerDto)

{

CustomerEntity entity = new CustomerEntity()

{

Id = null,

Address = customerDto.Address,

Email = customerDto.Email,

FirstName = customerDto.FirstName,

LastName = customerDto.LastName,

Phone = customerDto.Phone

};

EntityEntry<CustomerEntity> response = await Customer.AddAsync(entity);

await SaveChangesAsync();

return await Get(response.Entity.Id ?? throw new Exception("no se ha podido añadir"));

}

/// <summary>

/// Borra un customer

/// </summary>

/// <param name="id">Ident del customer</param>

/// <returns></returns>

public async Task<bool> Delete(long id)

{

var entity = await Get(id);

Customer.Remove(entity);

SaveChanges();

return true;

}

public async Task<bool> Actualizar(CustomerEntity customerEntity)

{

Customer.Update(customerEntity);

await SaveChangesAsync();

return true;

}

}

public class CustomerEntity

{

public long? Id { get; set; } // puede ser null

public string FirstName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Address { get; set; }

public string Phone { get; set; }

public CustomerDto ToDto()

{

return new CustomerDto

{

Address = Address,

Email = Email,

FirstName = FirstName,

LastName = LastName,

Phone = Phone,

Id = Id ?? throw new Exception("El id no puede ser null")

};

}

}

}

**CustomerDto.cs**

namespace CustomersApi.Dtos

{

public class CustomerDto

{

public long Id { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Address { get; set; }

public string Phone { get; set; }

}

}

**CreateCustomerDto.cs**

using System.ComponentModel.DataAnnotations;

namespace CustomersApi.Dtos

{

public class CreateCustomerDto

{

[Required (ErrorMessage = "El nombre propio tiene que estar especificado")]

public string FirstName { get; set; }

[Required(ErrorMessage = "El apellido propio tiene que estar especificado")]

public string LastName { get; set; }

[RegularExpression(@"^[^@\s]+@[^@\s]+\.[^@\s]+$", ErrorMessage = "El formato del email no es válido.")]

public string Email { get; set; }

public string Address { get; set; }

public string Phone { get; set; }

}

}

**CustomerController.cs**

using CustomersApi.CasosDeUso;

using CustomersApi.Dtos;

using CustomersApi.Repositories;

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace CustomersApi.Controllers

{

[ApiController]

[Route("api/[controller]")]

public class CustomerController : Controller

{

private readonly CustomerDatabaseContext \_customerDatabaseContext;

private readonly IUpdateCustomerUseCase \_updateCustomerUseCase;

public CustomerController(CustomerDatabaseContext customerDatabaseContext,

IUpdateCustomerUseCase updateCustomerUseCase)

{

this.\_customerDatabaseContext = customerDatabaseContext;

this.\_updateCustomerUseCase = updateCustomerUseCase;

}

//api/customer/

[HttpGet]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(List<CustomerDto>))]

public async Task<IActionResult> GetCustomers()

{

var result = \_customerDatabaseContext.Customer.Select(c => c.ToDto()).ToList();

return new OkObjectResult(result);

}

//api/customer/{id}

[HttpGet("{id}")]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(CustomerDto))]

[ProducesResponseType(StatusCodes.Status404NotFound)]

public async Task<IActionResult> GetCustomer(long id)

{

//var vacio = new CustomerDto();

CustomerEntity result = await \_customerDatabaseContext.Get(id);

return new OkObjectResult(result.ToDto());

}

[HttpDelete("{id}")]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(bool))]

public async Task<IActionResult> DeleteCustomer(long id)

{

var result = await \_customerDatabaseContext.Delete(id);

return new OkObjectResult(result);

}

[HttpPost]

[ProducesResponseType(StatusCodes.Status201Created, Type = typeof(CustomerDto))]

public async Task<IActionResult> CreateCustomer(CreateCustomerDto customer)

{

CustomerEntity result = await \_customerDatabaseContext.Add(customer);

//var vacio = new CustomerDto();

return new CreatedResult($"https://localhost:7163/api/customer/{result.Id}", null);

}

[HttpPut]

[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(CustomerDto))]

[ProducesResponseType(StatusCodes.Status404NotFound)]

public async Task<IActionResult> UpdateCustomer(CustomerDto customer)

{

CustomerDto? result = await \_updateCustomerUseCase.Execute(customer);

if (result == null)

{

return new NotFoundResult();

}

return new OkObjectResult(result);

}

}

}

**UpdateCustomerUseCase.cs**

using CustomersApi.Dtos;

using CustomersApi.Repositories;

namespace CustomersApi.CasosDeUso

{

public interface IUpdateCustomerUseCase

{

Task<CustomerDto> Execute(CustomerDto customer);

}

public class UpdateCustomerUseCase: IUpdateCustomerUseCase

{

private readonly CustomerDatabaseContext \_customerDatabaseContext;

public UpdateCustomerUseCase(CustomerDatabaseContext customerDatabaseContext)

{

\_customerDatabaseContext = customerDatabaseContext;

}

public async Task<CustomerDto> Execute(CustomerDto customer)

{

//llamaar a la BBDD

var entity = await \_customerDatabaseContext.Get(customer.Id);

if (entity == null)

{

return null;

}

entity.FirstName = customer.FirstName;

entity.LastName = customer.LastName;

entity.Address = customer.Address;

entity.Phone = customer.Phone;

entity.Email = customer.Email;

await \_customerDatabaseContext.Actualizar(entity);

return entity.ToDto();

}

}

}

# Conexión entre Back and Front-end

´Pasos:

1. Crear .js
2. Del servidor vamos a recibir la info para convertir en HTML
3. El código HTML debemos de ponerlo en el index

**NOTA**: Mucho cuidado con la url, no te salía!. Tuviste que modificar el back añadiendo CORS

¿cómo queda ahora program.cs BackEnd API?

## 1. CORS (Cross-Origin Resource Sharing)

Dado que tu frontend está en http://127.0.0.1:5500 y el backend en http://localhost:5168, se considera una solicitud de **origen cruzado**, y el navegador la puede bloquear. Necesitas asegurarte de que el backend permita CORS.

### Cómo habilitar CORS en .NET

En Program.cs o Startup.cs, agrega algo así:

csharp

CopiarEditar

app.UseCors(builder =>

builder.WithOrigins("http://127.0.0.1:5500")

.AllowAnyHeader()

.AllowAnyMethod());

También debes asegurarte de registrar CORS en los servicios:

csharp

CopiarEditar

builder.Services.AddCors();

Si deseas una política más abierta durante el desarrollo, puedes permitir cualquier origen:

csharp

CopiarEditar

app.UseCors(builder =>

builder.AllowAnyOrigin()

.AllowAnyHeader()

.AllowAnyMethod());

**Enlace** 🡪 <https://learn.microsoft.com/es-es/aspnet/core/security/cors?view=aspnetcore-9.0>

**Program.cs**

using CustomersApi.CasosDeUso;

using CustomersApi.Repositories;

using Microsoft.EntityFrameworkCore;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers(); // añade el controlador que hemos creado

// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

builder.Services.AddRouting(routing => routing.LowercaseUrls = true); // minuscula en la ruta . De https://localhost:7163/api/Customer a https://localhost:7163/api/customer

builder.Services.AddDbContext<CustomerDatabaseContext>(mysqlBuilder =>

{

//builder.UseMySQL("Server=localhost;Port=3306;Database=system;Uid=root;pwd="); // conexion string a nuestra bbdd

mysqlBuilder.UseMySQL(builder.Configuration.GetConnectionString("Connection1"));

});

builder.Services.AddScoped<IUpdateCustomerUseCase, UpdateCustomerUseCase>();

builder.Services.AddCors(); // añadido CORSSSSSSSS

var app = builder.Build(); // construye una app

// Configure the HTTP request pipeline.

if (app.Environment.IsDevelopment()) // swagger solo desarrollo

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseHttpsRedirection();

app.UseAuthorization();

// ESTO es lógica. No se recomienda ponerlo aqui

//app.MapGet("/customer/{id}", (long id) =>

//{

// return "net 6";

//});

// Añadida instruccion para que admita y sea mas flexible en el origen url

app.UseCors(builder =>

builder.AllowAnyOrigin()

.AllowAnyHeader()

.AllowAnyMethod());

app.MapControllers();

app.Run();

## Creación de nuestro Script

**Script,js**

document.addEventListener("DOMContentLoaded", init); // Cuando se cargue la página, se ejecuta la función search

const URL\_API = 'https://localhost:7163/api/customer' // URL de la API

var customers = [] // Array de objetos con los datos de los clientes

function init() { search() } // Función que se ejecuta al cargar la página

//alert(1);//Muestra un mensaje de alerta

function abrirFormulario() {

    //alert("Agregar Cliente");//Muestra un mensaje de alerta

    htmlModal = document.getElementById("modal"); //Obtiene el elemento con el id modal

    htmlModal.setAttribute("class", "modale opened"); //Agrega la clase open al elemento modal

}

function cerrarCliente() {

    //alert("Agregar Cliente");//Muestra un mensaje de alerta

    htmlModal = document.getElementById("modal"); //Obtiene el elemento con el id modal

    htmlModal.setAttribute("class", "modale"); //Agrega la clase open al elemento modal

}

function clean() { // Función para limpiar el formulario

    document.getElementById('txtId').value = ''

    document.getElementById('txtFirstName').value = ''

    document.getElementById('txtLastName').value = ''

    document.getElementById('txtPhone').value = ''

    document.getElementById('txtEmail').value = ''

    document.getElementById('txtAddress').value = ''

}

function agregar() {// Función para agregar un cliente

    clean() // Se limpia el formulario

    abrirFormulario() // Se abre el formulario

}

// Función que se ejecuta al cargar la página

async function search() {

    var url = URL\_API // URL de la API

    var response = await fetch(url, {

        "method": 'GET',

        "headers": {

            "Content-Type": 'application/json'

        }

    })

    customers = await response.json();

    var html = ''

    for (customer of customers) {

        var row = `<tr>

        <td>${customer.firstName}</td>

        <td>${customer.lastName}</td>

        <td>${customer.email}</td>

        <td>${customer.phone}</td>

        <td>

            <a href="#" onclick="edit(${customer.id})" class="myButton">Editar</a>

            <a href="#" onclick="remove(${customer.id})" class="myButtonDelete">Eliminar</a>

        </td>

    </tr>`

        html = html + row;

    }

    // Array de objetos con los datos de los clientes

    // Se añade la tabla al HTML

    document.querySelector("#customers > tbody").innerHTML = html

}

// Función para eliminar un cliente

async function remove(id) {

    // Se muestra un mensaje de confirmación

    var respuesta = confirm("¿Estás seguro de que quieres eliminar este cliente?")

    // Si la respuesta es afirmativa

    if (respuesta) {

        var url = URL\_API + '/' + id // URL de la API

        await fetch(url, {

            method: 'DELETE',

            headers: {

                "Content-Type": 'application/json'

            }

        })

        // 1 version. Se actualiza

        window.location.reload() // Se recarga la página

        // 2 version. Se actualiza

        // .then(response => response.json()) // Se convierte la respuesta a JSON

        // .then(data => { // Se ejecuta la función search

        //     search()

        // })

        alert("Cliente eliminado") // Se muestra un mensaje de confirmación

    }

}

async function save() {

    // Se obtienen los datos del formulario

    var data = {

        "firstName": document.getElementById('txtFirstName').value,

        "lastName": document.getElementById('txtLastName').value,

        "phone": document.getElementById('txtPhone').value,

        "email": document.getElementById('txtEmail').value,

        "address": document.getElementById('txtAddress').value

    }

    var id = document.getElementById('txtId').value // Se obtiene el id del cliente

    // Si el id no está vacío, se añade al objeto

    if (id != '') {

        data.id = id

    }

    var url = URL\_API // URL de la API

    await fetch(url, {

        "method": id != '' ? 'PUT' : 'POST', // Si el id no está vacío, se actualiza, si no, se crea

        "body": JSON.stringify(data), // Se convierte el objeto a un string

        "headers": {

            "Content-Type": 'application/json' // Se especifica que el contenido es JSON

        }

    })

    // 1 version. Se actualiza

    window.location.reload() // Se recarga la página

}

//

function edit(id) { // Función para editar un cliente

    // Se busca el cliente con el id seleccionado

    var customer = customers.find(x => x.id == id)

    // Se muestran los datos del cliente en el formulario

    document.getElementById('txtId').value = customer.id

    document.getElementById('txtFirstName').value = customer.firstName

    document.getElementById('txtLastName').value = customer.lastName

    document.getElementById('txtPhone').value = customer.phone

    document.getElementById('txtEmail').value = customer.email

    document.getElementById('txtAddress').value = customer.address

    // Se abre el formulario

    abrirFormulario()

}

De esta “guisa” quedó el **index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="style.css"> <!-- Enlaza el archivo style.css -->

    <link rel="stylesheet" href="menu.css">

    <script src="script.js"></script>

    <title>Index</title>

</head>

<nav id='menu'>

    <input type='checkbox' id='responsive-menu' onclick='updatemenu()'><label></label>

    <ul>

        <li><a href='http://'>Home</a></li>

        <li><a class='dropdown-arrow' href='http://'>Products</a>

            <ul class='sub-menus'>

                <li><a href='http://'>Products 1</a></li>

                <li><a href='http://'>Products 2</a></li>

                <li><a href='http://'>Products 3</a></li>

                <li><a href='http://'>Products 4</a></li>

            </ul>

        </li>

        <li><a href='http://'>About</a></li>

        <li><a class='dropdown-arrow' href='http://'>Services</a>

            <ul class='sub-menus'>

                <li><a href='http://'>Services 1</a></li>

                <li><a href='http://'>Services 2</a></li>

                <li><a href='http://'>Services 3</a></li>

            </ul>

        </li>

        <li><a href='http://'>Contact Us</a></li>

    </ul>

</nav>

<body>

    <h1>Gestión de Clientes</h1>

    <a onclick="agregar()" href="#" class="myButton">Agregar Cliente</a>

    <table id="customers">

        <thead>

        <tr>

            <th>Nombre</th>

            <th>Apellido</th>

            <th>Email</th>

            <th>Teléfono</th>

            <th>Acciones</th>

        </tr>

    </thead>

        <tbody>

        </tbody>

    </table>

    <!-- Modal -->

    <div id="modal" class="modale" aria-hidden="true">

        <div class="modal-dialog">

            <div class="modal-header">

                <h2>Datos del Cliente</h2>

                <a onclick="cerrarCliente()" href="#" class="btn-close closemodale" aria-hidden="true">&times;</a>

            </div>

            <div class="modal-body">

                <input id="txtId" type="hidden" placeholder="Id" size="20" /><br/>

                <input id="txtFirstName" type="text"  placeholder="Nombre" size="20" /><br/>

                <input id="txtLastName" type="text"  placeholder="Apellido" size="20" /><br/>

                <input id="txtPhone" type="text"  placeholder="Teléfono" size="20" /><br/>

                <input id="txtEmail" type="text"  placeholder="Email" size="20" /><br/>

                <input id="txtAddress" type="text"  placeholder="Address" size="20" /><br/>

            </div>

            <div class="modal-footer">

                <a onclick="save()" href="#" class="btn" id="btn\_ingresar">Guardar</a>

            </div>

        </div>

    </div>

    <!-- /Modal -->

</body>

<script>

</script>

</html>

# Vista final del proyecto

Tabla

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente