# Índice

Contenido

[Índice 1](#_Toc143771806)

[Modelo de datos 2](#_Toc143771807)

[Arquitectura del proyecto 2](#_Toc143771808)

[Crear la base de datos con EF 8](#_Toc143771809)

[Clase DataContext 9](#_Toc143771810)

[Appsettings 10](#_Toc143771811)

[Inyección de dependencias Servicio SQlServer 11](#_Toc143771812)

[Ejecutar proyectos de forma múltiple 11](#_Toc143771813)

[Migración de entidades hacia la base de datos SQL SERVER 12](#_Toc143771814)

[Crear un nuevo Branch en GIT para publicar la nueva versión del proyecto en el repositorio GIT. 12](#_Toc143771815)

# Modelo de datos

Vamos a crear un sencillo sistema de ventas que va a utilizar el siguiente modelo de datos:Diagrama

Descripción generada automáticamente

# Arquitectura del proyecto

SQL Server on Azure

.NET Core 7 API

Blazor WEB Application

MAUI Mobile Application

Windows

Android

iOS

Mac



Crear proyecto

Vamos a crear esta estructura en Visual Studio (asegúrese de poner todos los proyectos en el mismo directorio C://Projects

Crear un nuevo repositorio GITHUB, usar gitignore, copiar ruta, repositorio público

Clonar proyecto git desde Visual Studio C://Projects/Market

* Nuevo proyecto Solution Blank llamado **Market**. Dentro de C://Projects al final la .sln queda el ícono dentro de Projects🡪(Market.sln)

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

* Sobre el ícono Market.sln del explorador de soluciones oprimimos click derecho y presionamos Open
* Click derecho sobre la solución y agregamos un nuevo proyecto tipo: **Class Library**, llamado **Market.Shared**

Ubicación: C:\Projects/Market (Borramos la class1.cs que se genera)

* Click derecho sobre la solución y agregamos un nuevo proyecto tipo: **ASP.NET Core Web API**, llamado **Market.API** Ubicación: C:\Projects/Market
* Click derecho sobre la solución y agregamos un nuevo proyecto tipo: **Blazor WebAssembly App**, llamado **Market.WEB** Ubicación: C:\Projects/Market

* Click derecho sobre la solución y agregamos un nuevo proyecto tipo: **.NET MAUI App**, llamado **Market.Mobile**. Ubicación: C:\Projects/Market

Después de la creación del proyecto Market.Mobile , debemos aceptar la licencia del SDK de Android:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Así debe verse al final , la estructura de los proyectos en el Solution Explorer:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Hacemos el primer commit en nuestro repositorio. Pestaña Git Changes

\*(Si en Git Changes no se visualiza el árbol de carpetas de los proyectos, será necesario cerrar la solución, y abrirla de nuevamente) Commit All and Sync

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

# Crear la base de datos con EF

Diagrama

Descripción generada automáticamente

Code First y Database First. En este curso trabajaremos con EF Code First,

Documentación: <https://docs.microsoft.com/en-us/ef/core/get-started/aspnetcore/existing-db>

1. Empecemos creando en el proyecto **Market.Shared** la carpeta **Entities** y dentro de esta carpeta la entidad **Country**:

using System.ComponentModel.DataAnnotations;

namespace Market.Shared.Entities

{

public class Country

{

public int Id { get; set; }

[Display(Name = "País")]

[MaxLength(100, ErrorMessage = "El campo {0} debe tener máximo {1} caractéres.")]

[Required(ErrorMessage = "El campo {0} es obligatorio.")]

public string Name { get; set; } = null!;

}

}

# Clase DataContext

1. En el proyecto **API** creamos la carpeta **Data** y dentro de esta la clase **DataContext**:

using Microsoft.EntityFrameworkCore;

using Market.Shared.Entities;

namespace Market.API.Data

{

public class DataContext : DbContext

{

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

{

}

public DbSet<Country> Countries { get; set; }

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

base.OnModelCreating(modelBuilder);

modelBuilder.Entity<Country>().HasIndex(c => c.Name).IsUnique();

}

}

}

# Appsettings

1. Configurar el string de conexión en el **appsettings.Development.json** y en **appsettings.json** del proyecto **API**:

{

"ConnectionStrings": {

"DefaultConnection": "Server= MyServer;Database=Market;Encrypt=False;User Id=dba;Password=Abcd1234\*;"

},

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning"

}

},

"AllowedHosts": "\*"

}

1. Agregar/verificar los paquetes al proyecto **API**:

Microsoft.EntityFrameworkCore.SqlServer

Microsoft.EntityFrameworkCore.Tools

# Inyección de dependencias Servicio SQlServer

1. Configurar la inyección del DataContext en la clase **Program** del proyecto **API**:

builder.Services.AddSwaggerGen();

builder.Services.AddDbContext<DataContext>(x => x.UseSqlServer("name=DefaultConnection"));

var app = builder.Build();

# Ejecutar proyectos de forma múltiple

1. En el desplegable Startup Projects seleccionar Market.API como proyecto de inicio, abrir Package Manager Console(Tool) , e igualmente elegir Market.API ,como Default Project

# Migración de entidades hacia la base de datos SQL SERVER

1. Correr los siguientes comandos en Package Manager Console:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

add-migration InitialDb

update-database

# Crear un nuevo Branch en GIT para publicar la nueva versión del proyecto en el repositorio GIT.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

Una captura de pantalla de una computadora

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. Hacemos nuestro segundo **Commit**. All and Sync y luego el Push en una nuevo branch(Si en Git Changes no se visualiza el árbol de carpetas de los proyectos, será necesario cerrar la solución, y abrirla de nuevamente)