

Laboratorio 1 - Implementación de Monolito con Patrón MVC y DAO



Pontificia Universidad
JAVERIANA
Colombia

Manuel Ríos

Daniel Castellanos

Santiago Barbosa

Pontificia Universidad Javeriana

Arquitectura de Software

Abril De 2023

Tabla de contenido

Marco Conceptual.....	3
Diseño.....	4
Procedimiento.....	5
Resultados	28
Conclusiones y lecciones aprendidas	28
Referencias.....	29

Marco Conceptual

.NET 6: .NET es un framework de desarrollo de software que se utiliza para construir aplicaciones de escritorio, móviles y web. La última versión, .NET 6, es una plataforma de desarrollo multiplataforma que se enfoca en la mejora de la productividad y la velocidad.

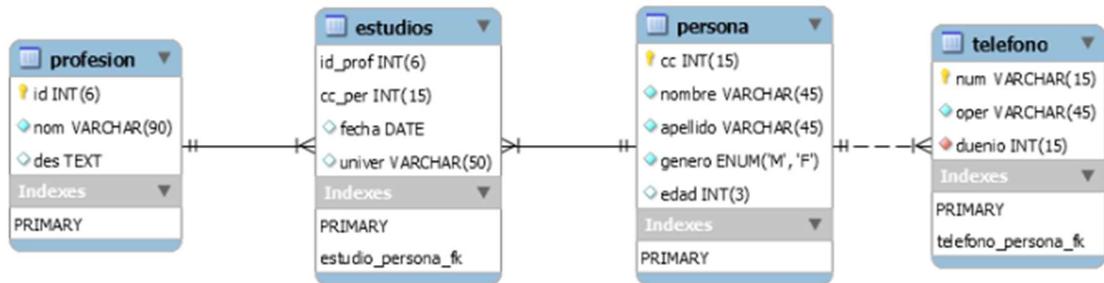
MS SQL Server 2019: MS SQL Server es un sistema de gestión de bases de datos relacional (RDBMS) desarrollado por Microsoft. La versión 2019 ofrece mejoras en la escalabilidad, el rendimiento y la seguridad, lo que la hace ideal para aplicaciones empresariales.

REST: REST (Representational State Transfer) es un estilo de arquitectura de software que se utiliza para construir servicios web. Se enfoca en la interacción entre clientes y servidores a través de HTTP, y permite una mayor escalabilidad y modularidad en las aplicaciones.

Swagger 3: Swagger es una herramienta de especificación y documentación de API. La versión 3 ofrece una serie de mejoras, como la compatibilidad con OpenAPI 3.0 y la integración con otras herramientas de desarrollo.

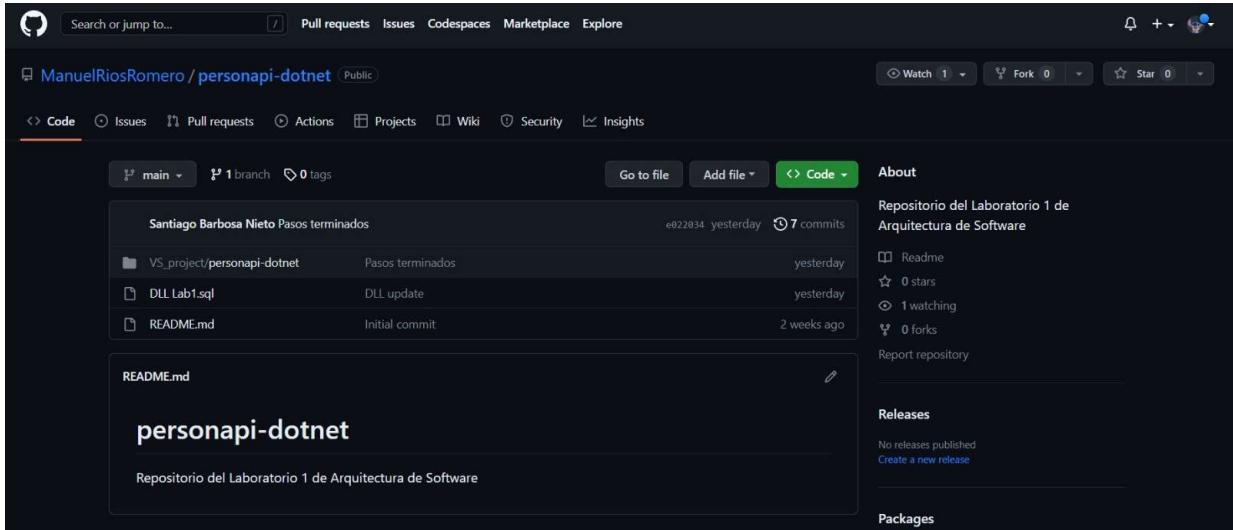
Diseño

Para el diseño de este taller desarrollamos un CRUD para un modelo de datos basado en 4 tablas que fueron distribuidas de esta manera



Procedimiento

Empezamos el procedimiento para la elaboración de este taller creando el repositorio personapi-dotnet. Que luego será utilizado para la elaboración del código de este taller.



The screenshot shows a GitHub repository page for 'ManuelRiosRomero / personapi-dotnet'. The repository has 1 watch, 0 forks, and 0 stars. The README.md file contains the text 'personapi-dotnet' and 'Repositorio del Laboratorio 1 de Arquitectura de Software'. The repository has 1 branch and 0 tags. The commit history shows 7 commits from Santiago Barbosa Nieto, with the latest being 'Pasos terminados' at e022034 yesterday. Other commits include 'VS_project/personapi-dotnet' (yesterday), 'DLL Lab1.sql' (yesterday), and 'Initial commit' (2 weeks ago).

Como segundo paso vamos a la página: <https://www.microsoft.com/en-us/sql-server/sql-server-downloads> donde podremos descargar SQL Server 2019, como observamos en la imagen, descargaremos la versión express.

Or, download a free specialized edition



Developer

SQL Server 2022 Developer is a full-featured free edition, licensed for use as a development and test database in a non-production environment.

[Download now](#)

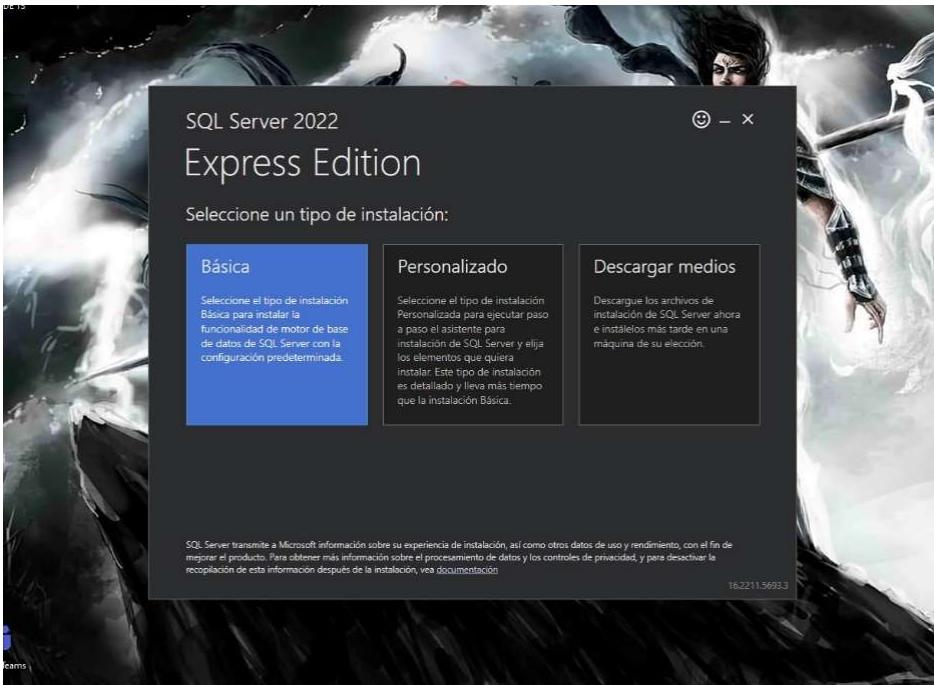


Express

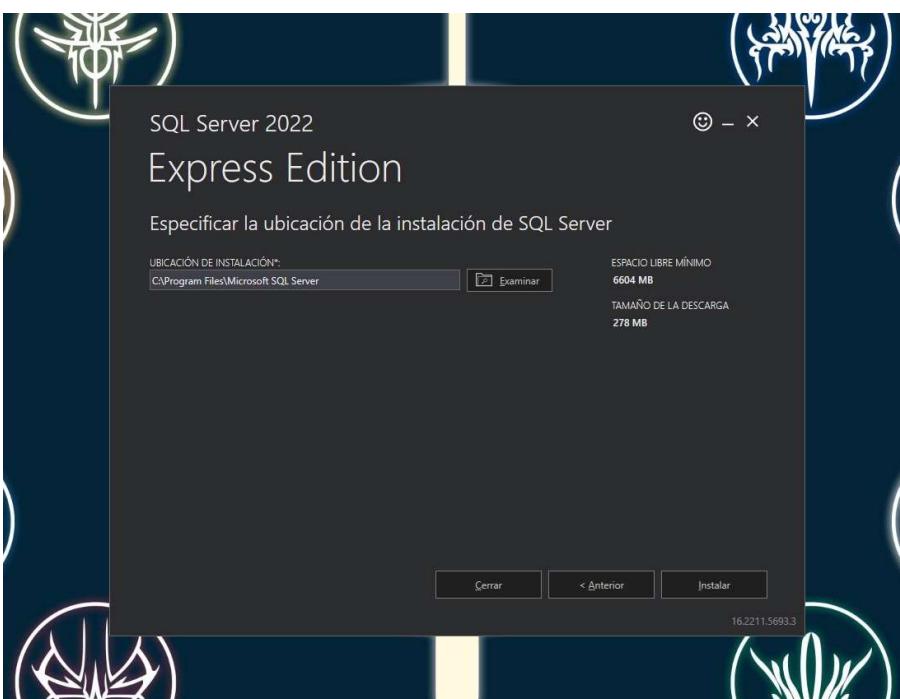
SQL Server 2022 Express is a free edition of SQL Server, ideal for development and production for desktop, web, and small server applications.

[Download now](#)

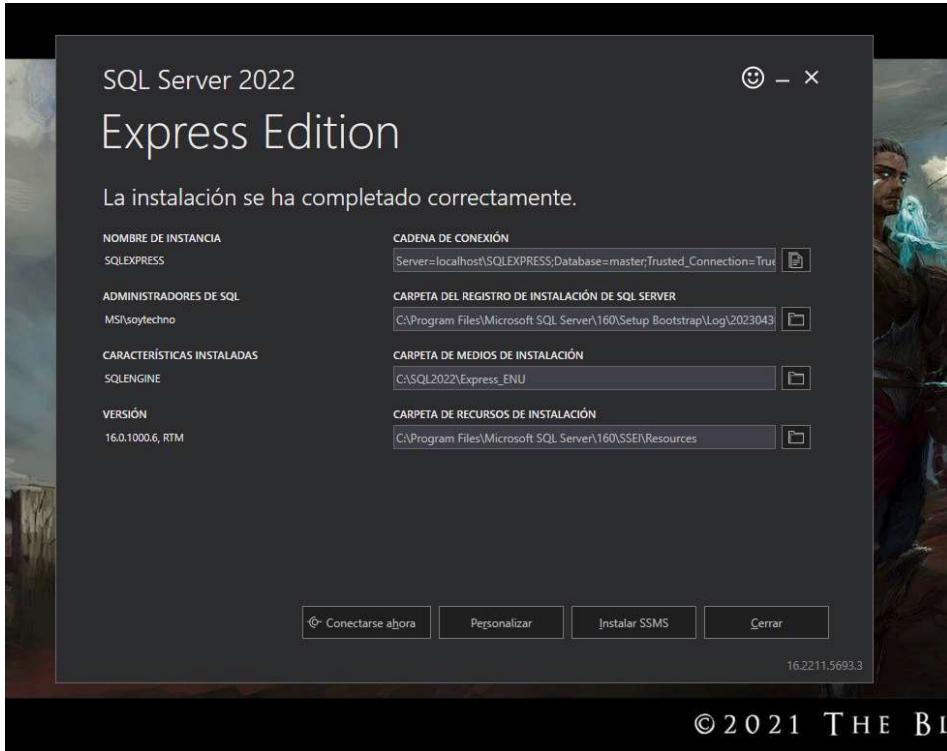
Una vez hallamos descargado el instalador de SQL Server escogeremos el tipo de instalación básica.



Luego de escoger la instalación debemos escoger la ubicación en la cual queremos instalar SQL Server.



Después de un tiempo concluirá la instalación y nos presenta con la opción de instalar el SSMS es decir el SQL Server Management Studio.



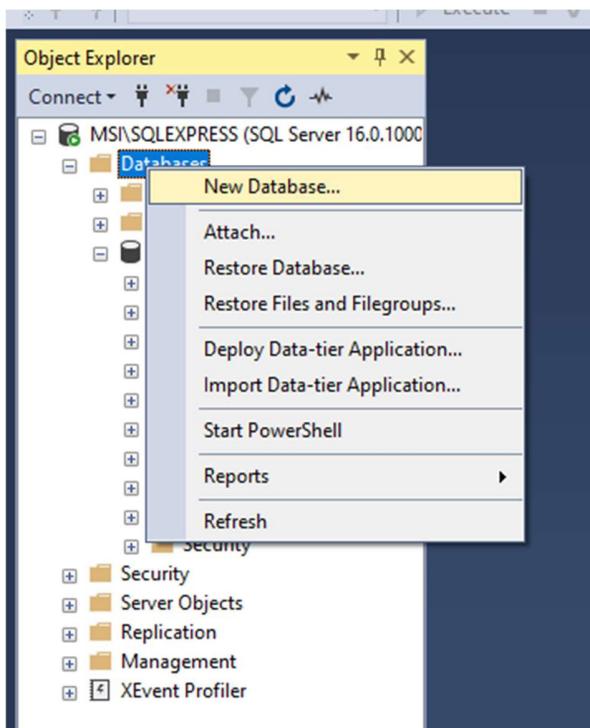
Al seleccionar la opción de instalar el SSMS nos dirigirá automáticamente al siguiente enlace <https://learn.microsoft.com/es-es/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16> seleccionamos la opción de descarga gratuita y esperamos a que se descargue el instalador.

The screenshot shows the Microsoft Learn page for downloading SQL Server Management Studio (SSMS). The main content is titled "Descarga de SQL Server Management Studio (SSMS)". It highlights that SSMS 19.0.2 is the latest general availability (GA) version. The page provides a download link for "Descargar SSMS" and includes a brief description of what SSMS is used for. A sidebar on the right is titled "En este artículo" and contains links to "Descargar SSMS", "Idiomas disponibles", "Noticias", and "Versiones anteriores".

Al abrir el instalador se ejecutará y se instalará automáticamente en la misma carpeta del SQL Server.



Una vez instalado el SSMS lo abriremos, crearemos la base de datos persona_db.



Una vez creada la base de datos crearemos las tablas que definimos con anterioridad en el diseño de este laboratorio.

```
USE [persona_db]
/******** Object: Table [dbo].[persona]    Script Date: 4/24/2023 4:35:12 PM *****/
CREATE TABLE [dbo].[persona] (
    [cc]      INT      NOT NULL,
    [nombre]  VARCHAR (45) NOT NULL,
    [apellido] VARCHAR (45) NOT NULL,
    [genero]  VARCHAR (1) NOT NULL,
    [edad]    INT      NULL,
    CONSTRAINT [PK_persona] PRIMARY KEY CLUSTERED ([cc] ASC),
    CONSTRAINT [CK_persona_genero] CHECK ([genero] LIKE 'M' OR [genero] LIKE 'F')
);

/******** Object: Table [dbo].[profesion]    Script Date: 4/24/2023 4:35:12 PM *****/
CREATE TABLE [dbo].[profesion] (
    [id]      INT      NOT NULL,
    [nom]    VARCHAR (60) NOT NULL,
    [des]    TEXT      NULL,
    CONSTRAINT [PK_profesion] PRIMARY KEY CLUSTERED ([id] ASC)
);

/******** Object: Table [dbo].[telefono]    Script Date: 4/24/2023 4:35:12 PM *****/
CREATE TABLE [dbo].[telefono] (
    [num]      VARCHAR (15) NOT NULL,
    [oper]    VARCHAR (45) NOT NULL,
    [duenio]   INT      NOT NULL,
    CONSTRAINT [PK_telefono] PRIMARY KEY CLUSTERED ([num] ASC),
    CONSTRAINT [FK_telefono_persona] FOREIGN KEY ([duenio]) REFERENCES [dbo].[persona] ([cc])
);

/******** Object: Table [dbo].[estudios]    Script Date: 4/24/2023 4:35:12 PM *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[estudios] (
    [id_prof]  INT      NOT NULL,
    [cc_per]   INT      NOT NULL,
    [fecha]    DATE     NULL,
    [univer]   VARCHAR (50) NULL,
    CONSTRAINT [PK_estudios] PRIMARY KEY ([cc_per] ASC, [id_prof] ASC),
    CONSTRAINT [FK_estudios_persona] FOREIGN KEY ([cc_per]) REFERENCES [dbo].[persona] ([cc]),
    CONSTRAINT [FK_estudios_profesion] FOREIGN KEY ([id_prof]) REFERENCES [dbo].[profesion] ([id])
);
```

Luego de haber creado las tablas en la base de datos deberemos instalar Visual Studio, lo podremos descargar desde el siguiente link <https://visualstudio.microsoft.com/es/free-developer-offers/>.



Visual Studio Community

El mejor IDE completo para desarrolladores de .NET y C++ en Windows. Completamente equipado con una buena matriz de herramientas y características para elevar y mejorar todas las etapas del desarrollo de software.

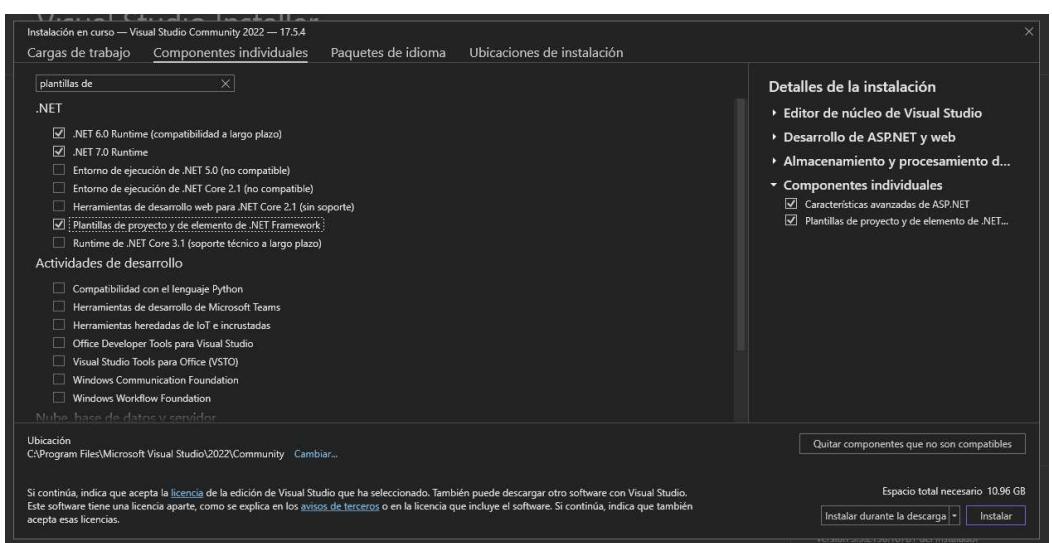
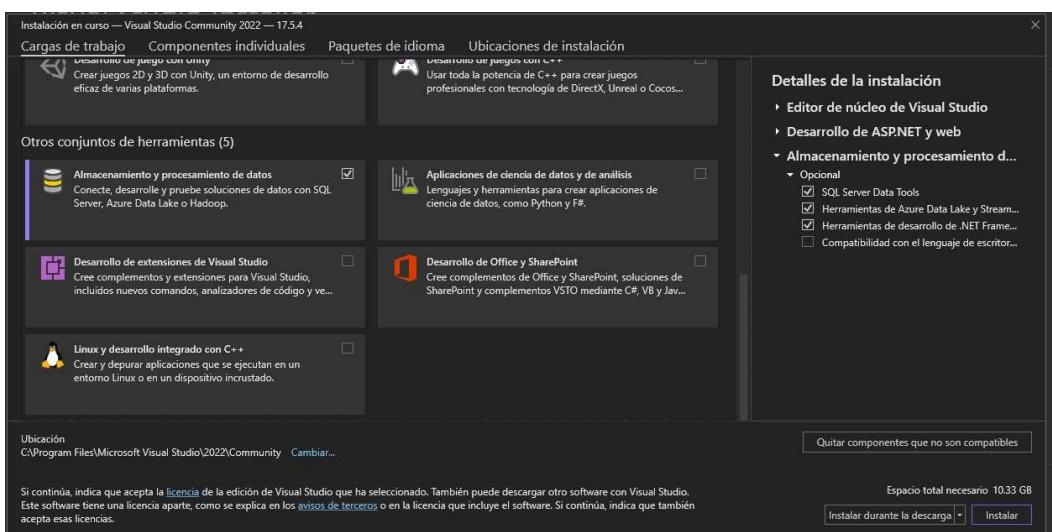
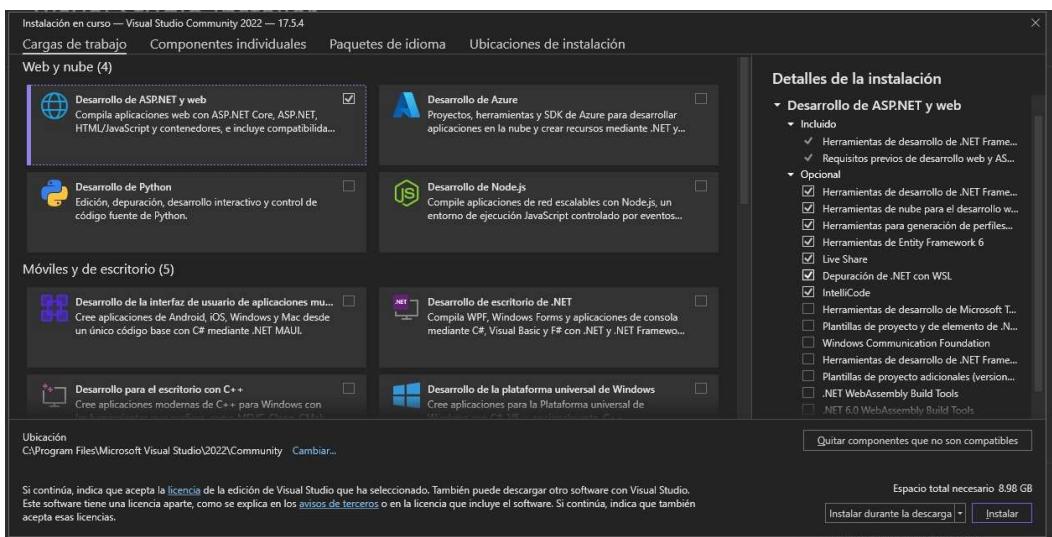
[Más información →](#)

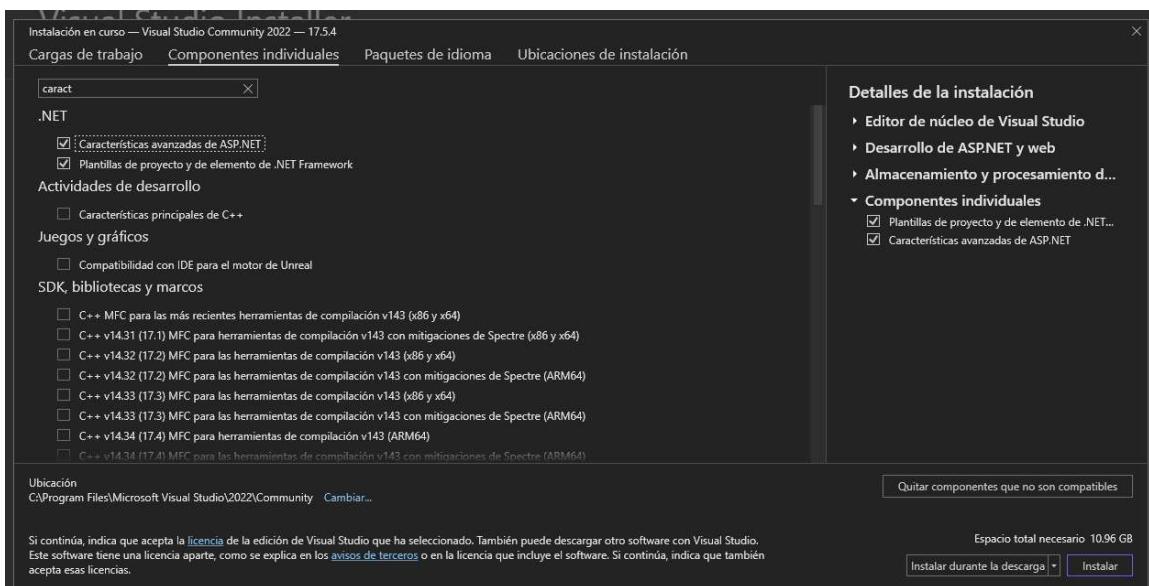
[Descarga gratuita](#)

Al descargar y ejecutar el instalador de Visual Studio, se nos presentaran diferentes opciones de complementos que podemos agregar a la instalación.

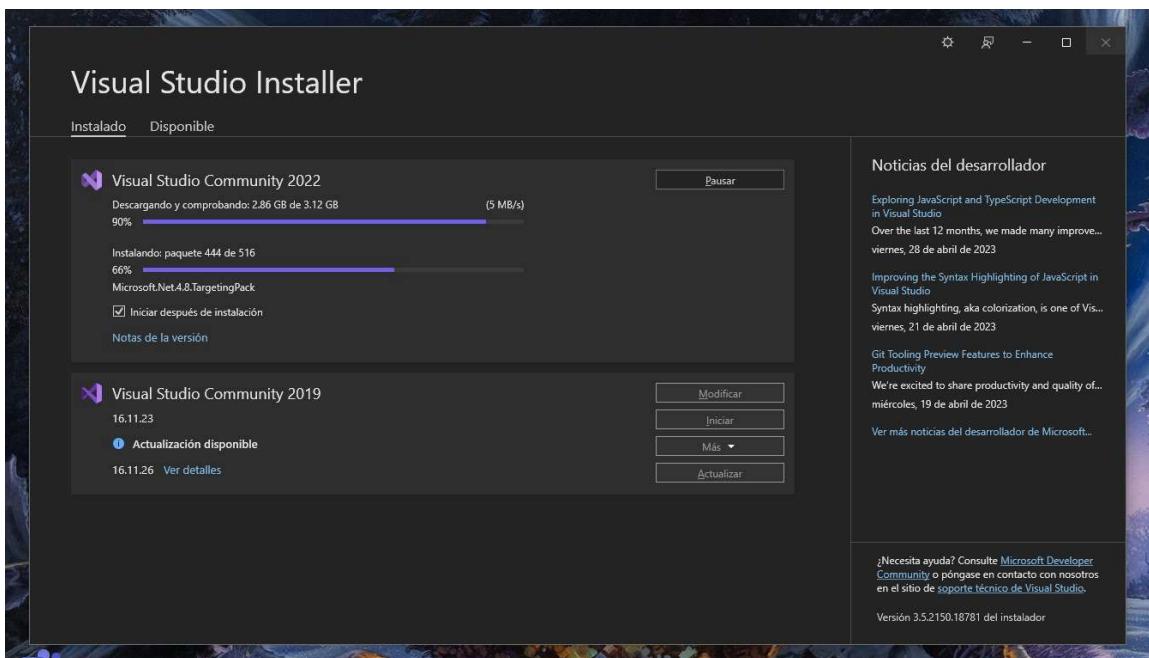
Como podremos ver en las siguientes imágenes escogeremos los siguientes complementos:

- Desarrollo ASP.NET y web
- Almacenamiento y procesamiento de datos
- Plantillas de proyecto y elementos de .NET Framework
- Características avanzadas de ASP.NET

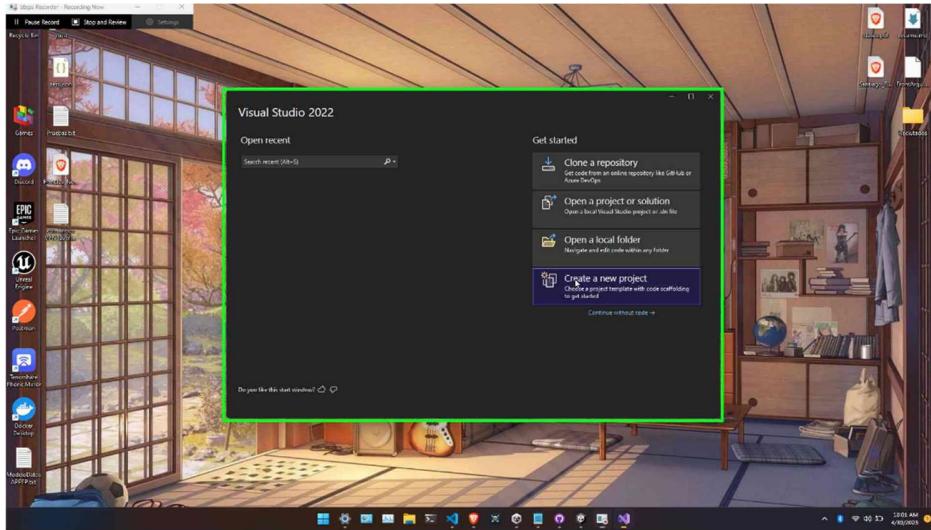




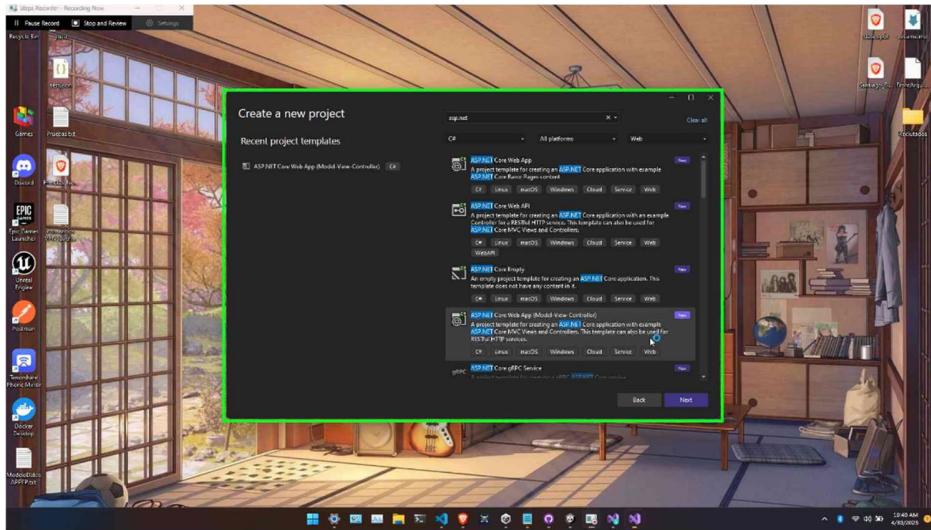
Esperamos a que se termine la instalación de Visual Studio y sus componentes.



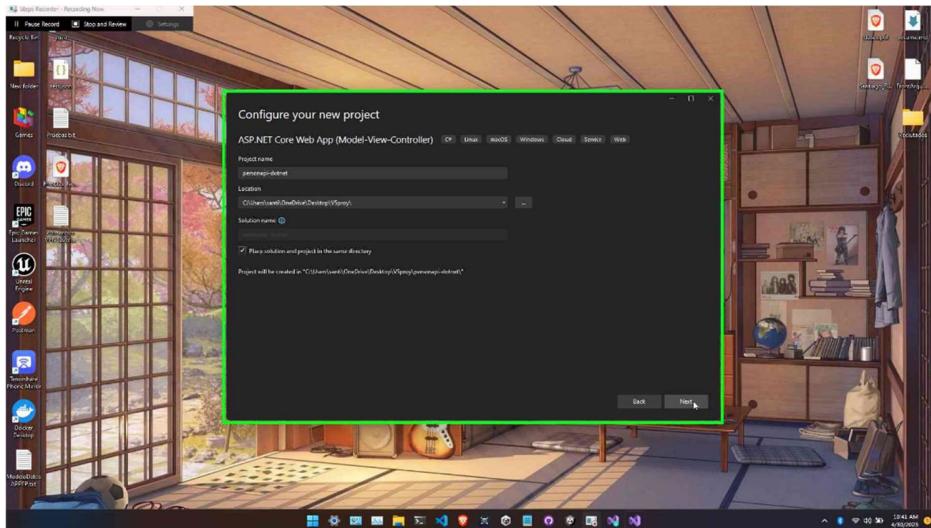
Para la creación del proyecto seguiremos los pasos tal cual como lo indican las siguientes imágenes.



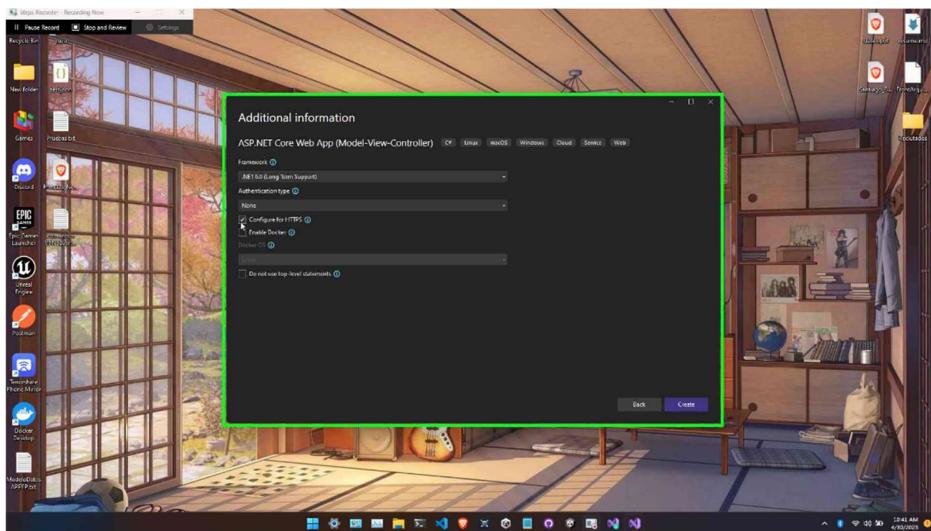
Seleccionaremos la plantilla Aplicación web de ASP.NET Core (Modelo-Vista-Controlador).



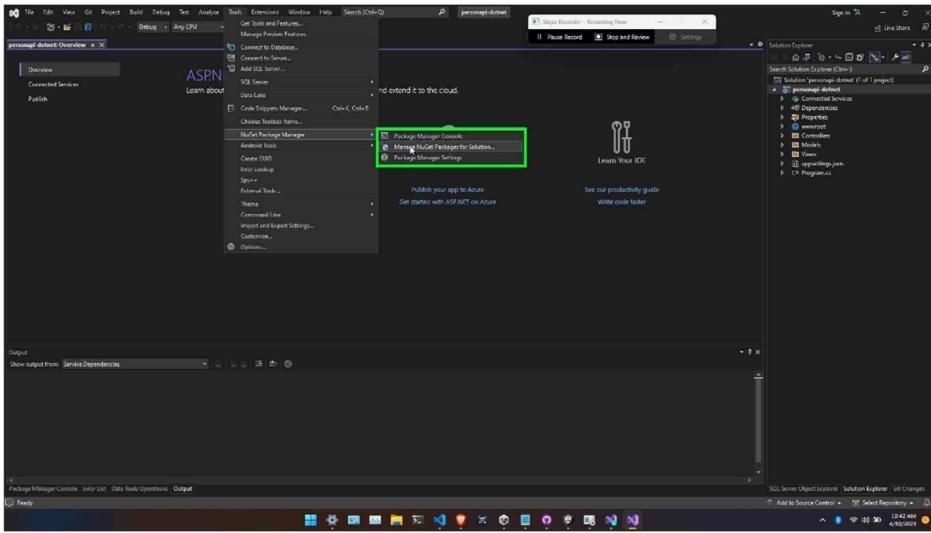
Al configurar el proyecto le pondremos el mismo nombre que al repositorio anteriormente creado que en este caso fue personapi-dotnet.



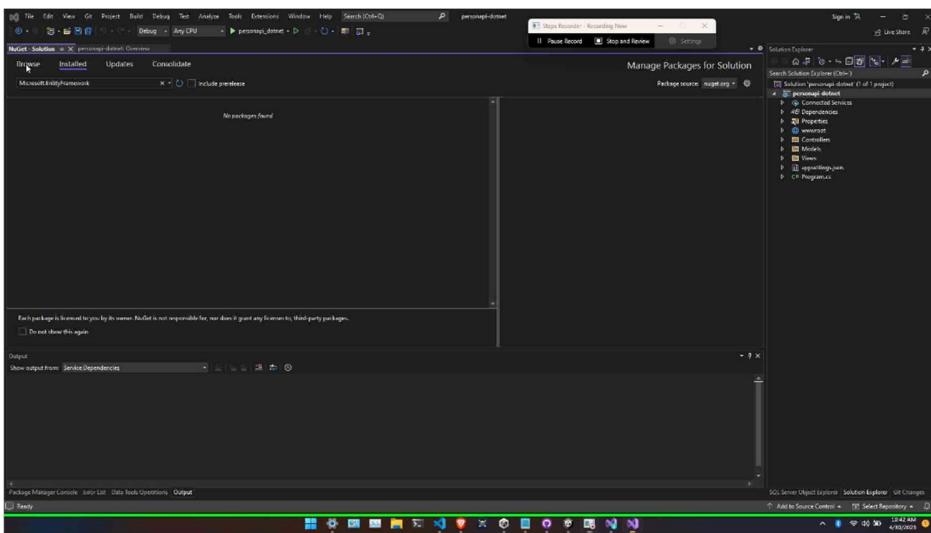
Otro punto en el cual se tiene que estar pendiente es asegurarse de que el Framework .NET 6.0 quede sin autenticación y sin configuración HTTPS.



Luego de crear el proyecto debemos realizar la instalación de los paquetes requeridos para la elaboración de este laboratorio. Para ello iremos al menu ver, y activaremos la vista de explorador de objetos de SQL Server.



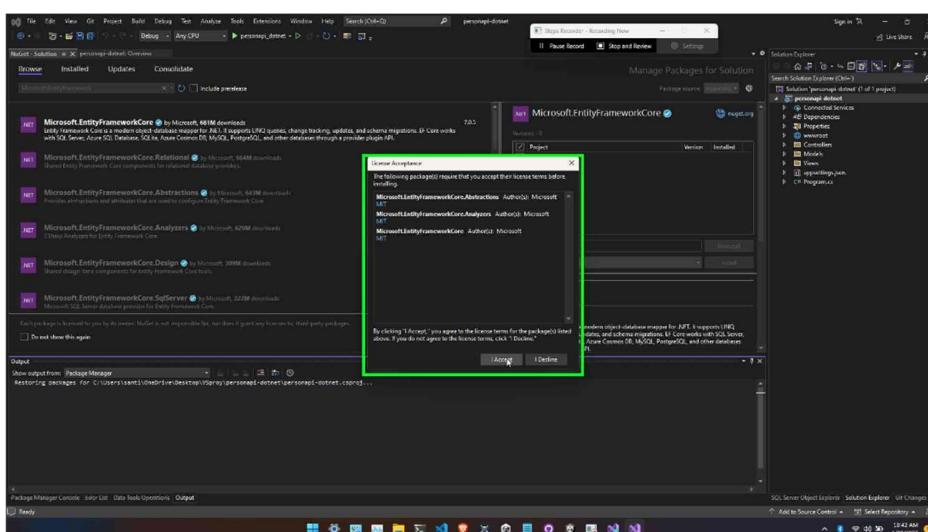
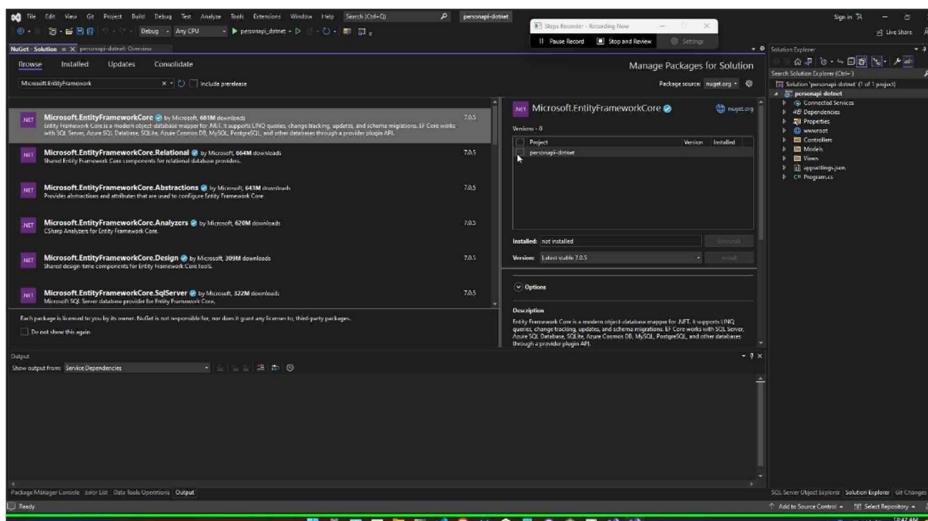
Luego de esto probaremos la conexión de tipo local express

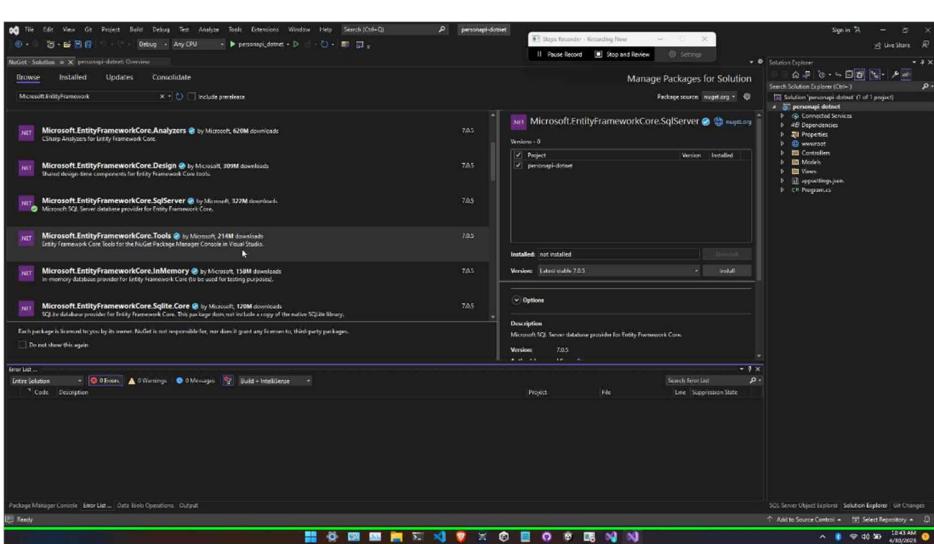
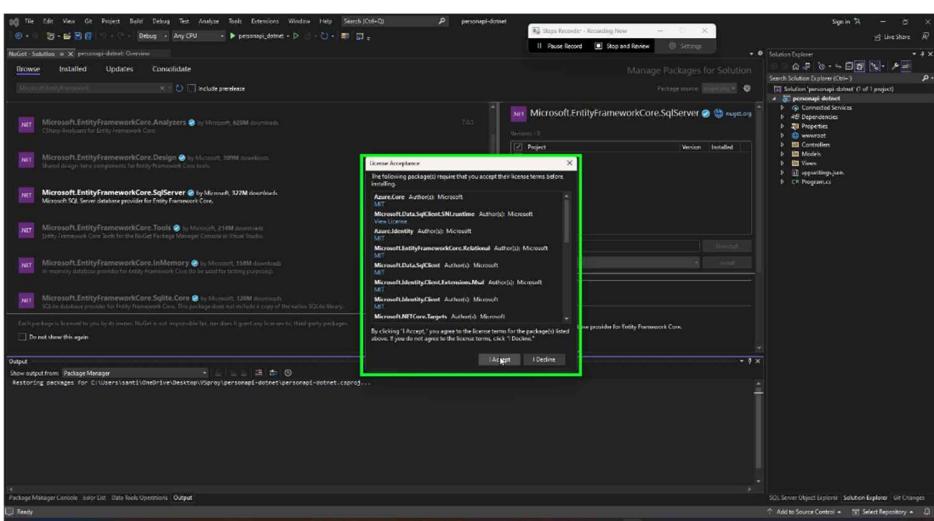
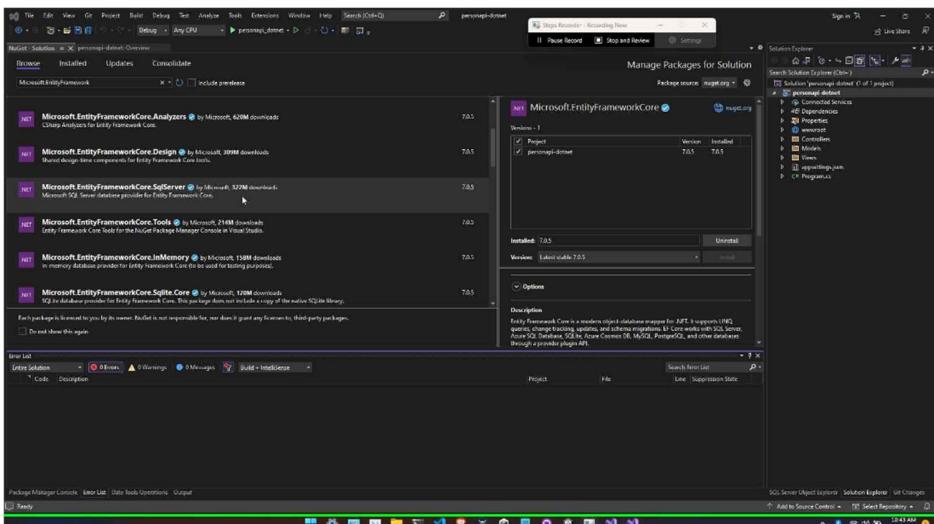


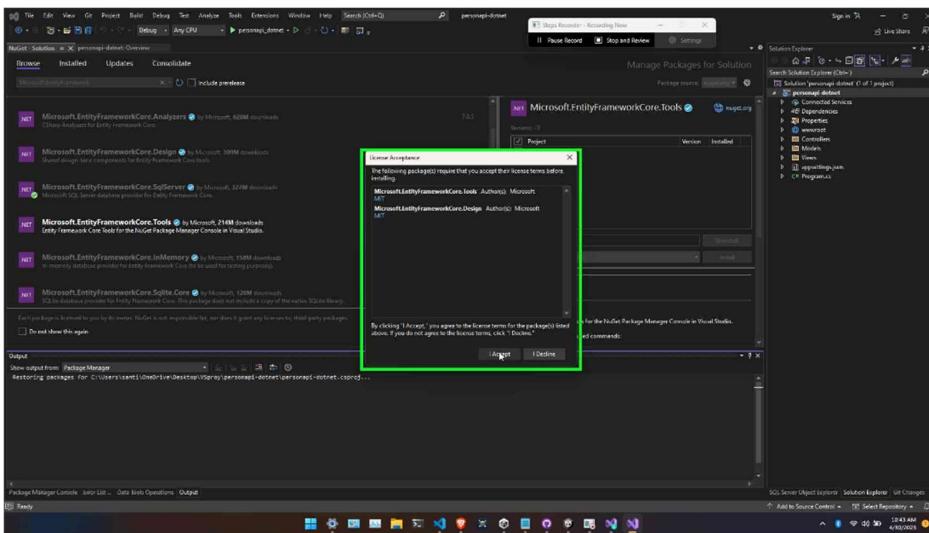
Debemos instalar diferentes paquetes para esto nos dirigimos al menu de herramientas, de aca a administrador de paquetes NuGet y por ultimo a Consola del Administrador de paquetes

En dicho explorador instalaremos estos paquetes como se ve en las siguientes imagenes

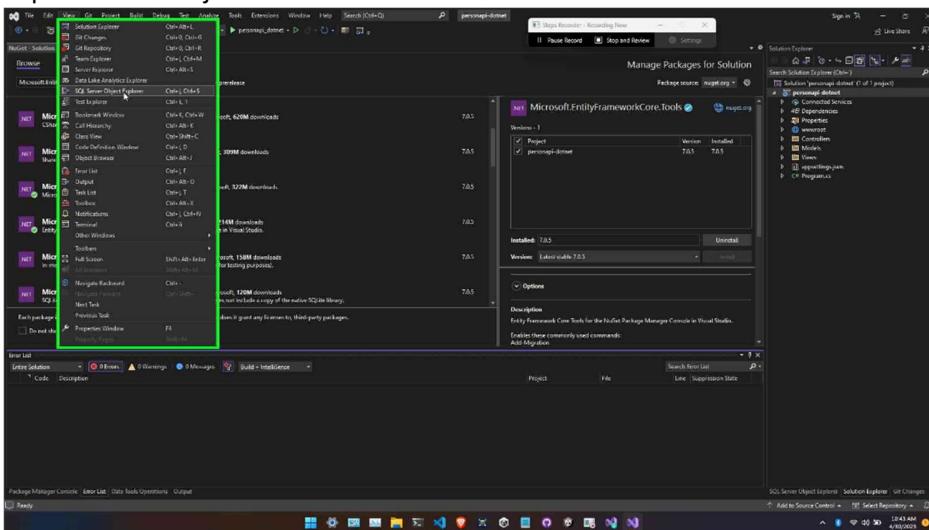
- Microsoft.EntityFrameworkCore
- Microsoft.EntityFrameworkCore.SqlServer
- Microsoft.EntityFrameworkCore.Tools

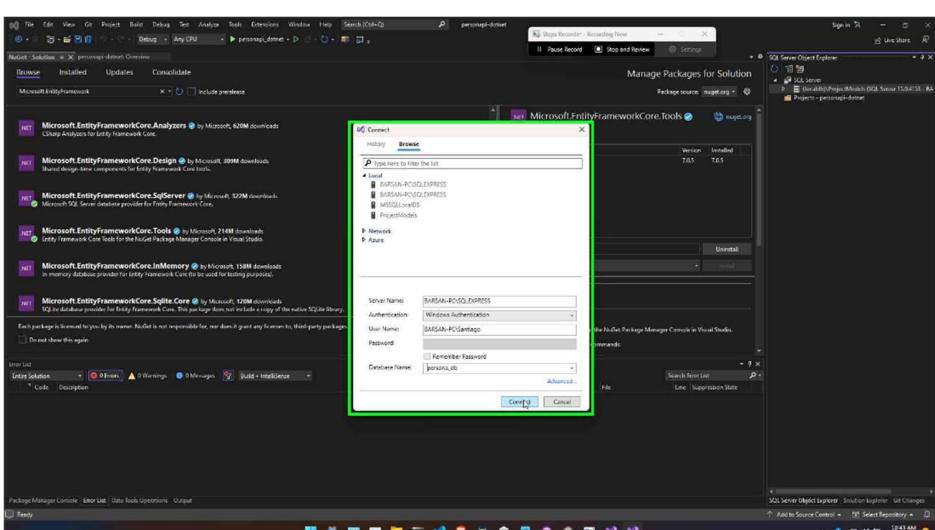
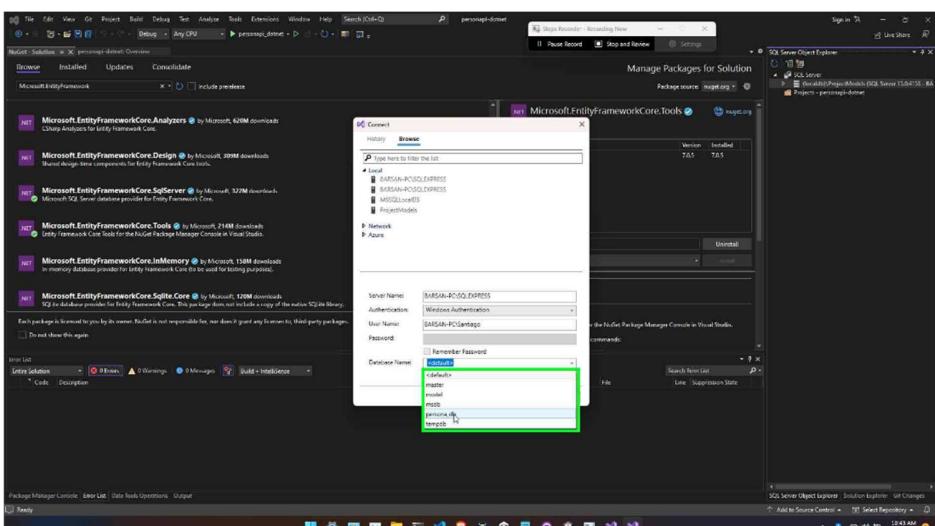
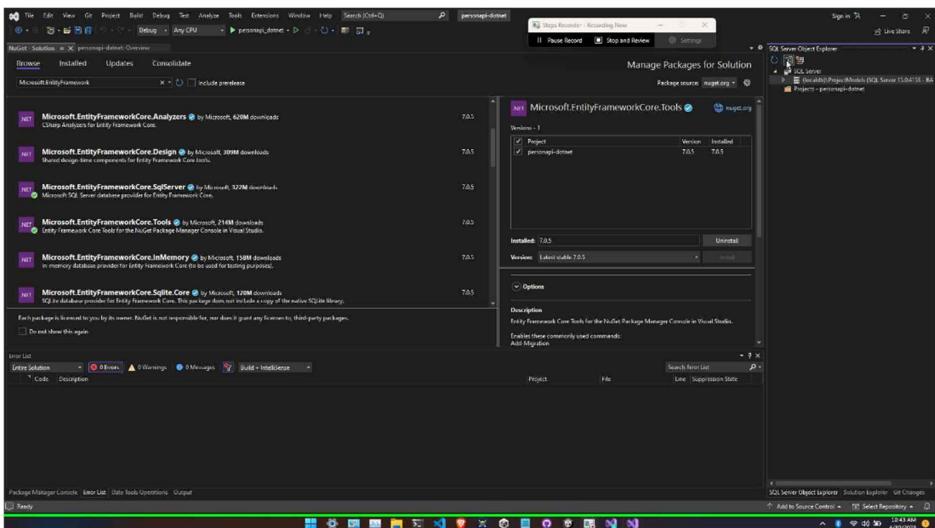


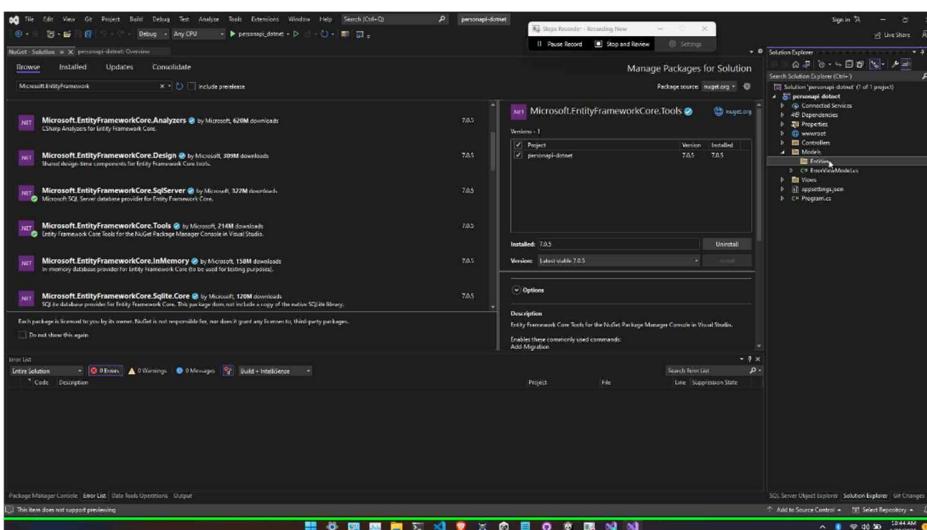
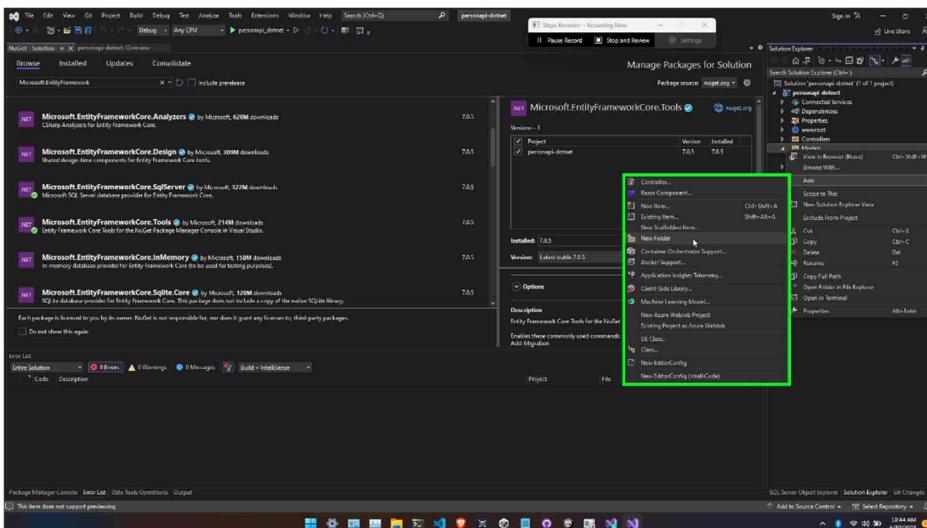




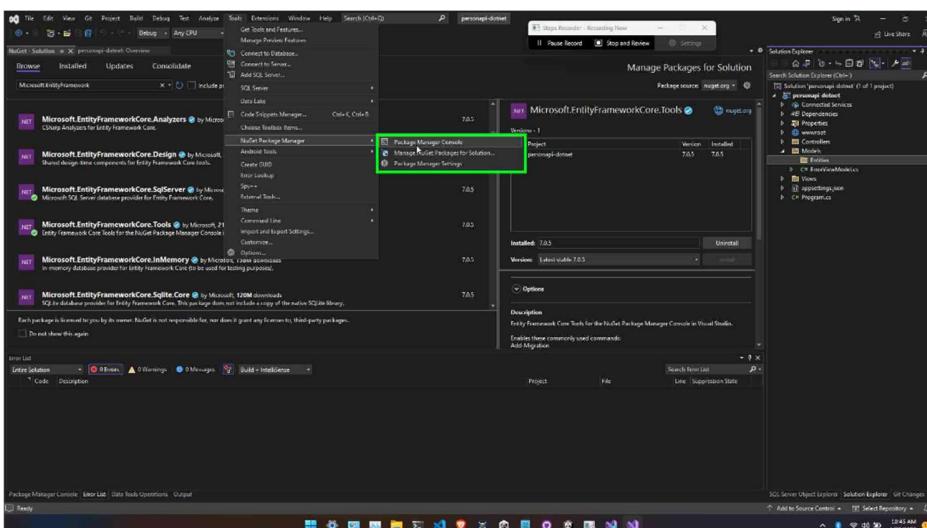
Para la conexión a la base de datos debemos acceder a vista y seleccionar la opción SQL Server explorador de objetos.

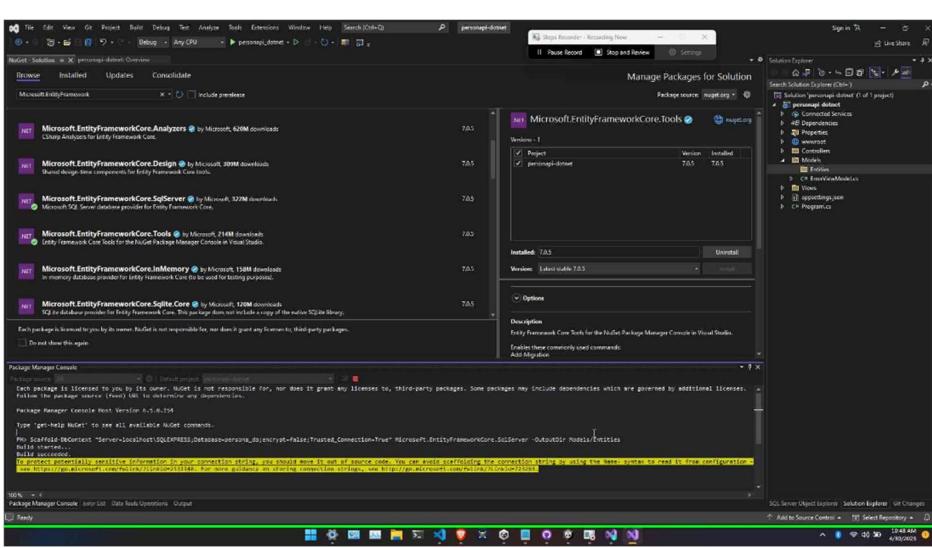
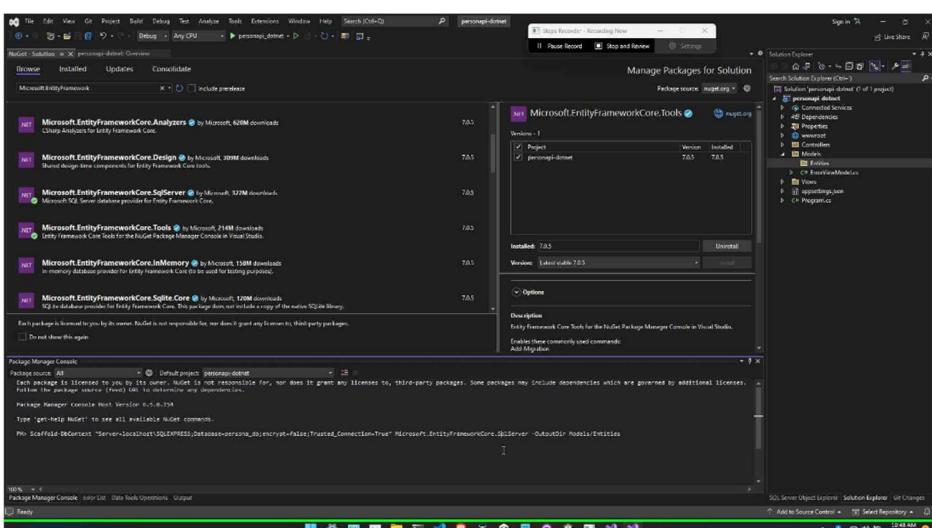
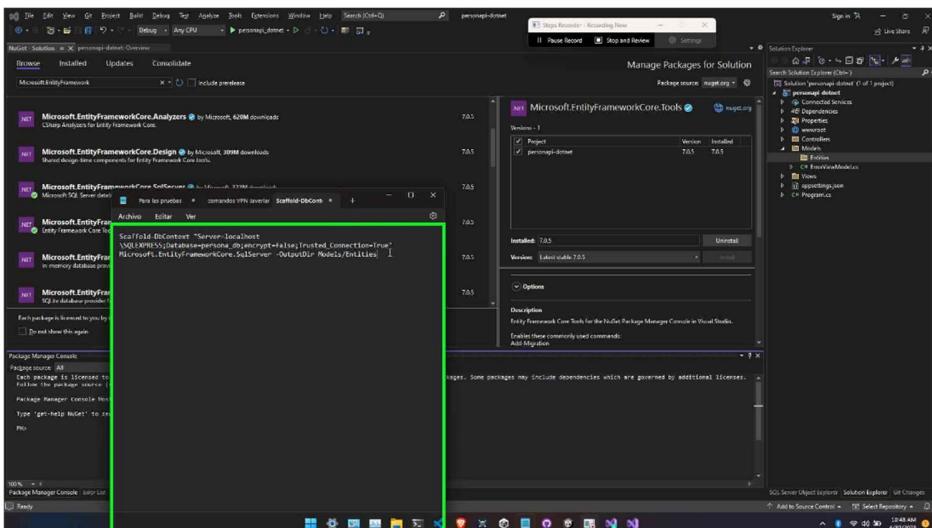






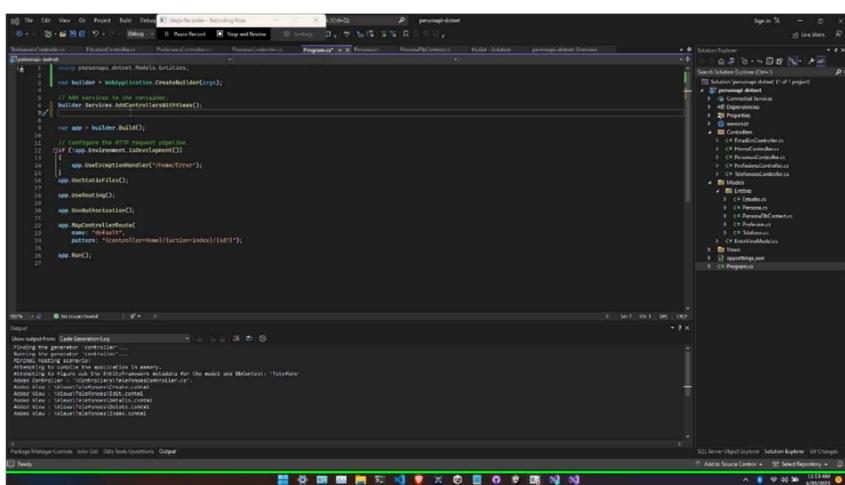
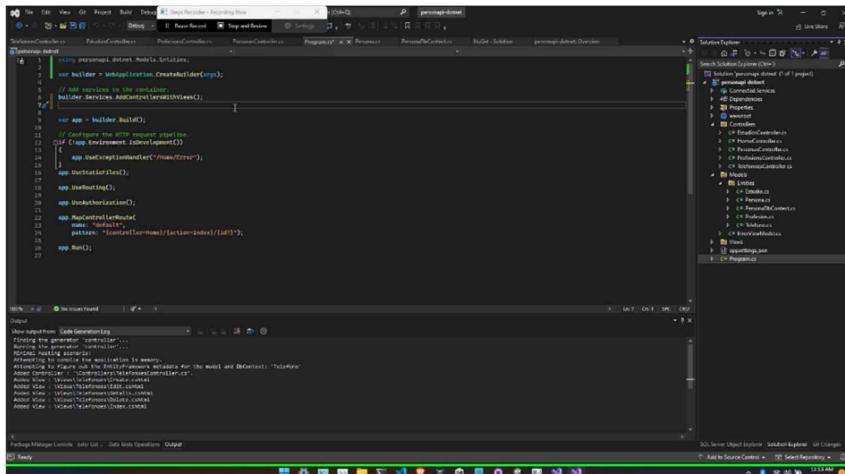
Como siguiente paso pasamos a la creación de entidades crearemos una carpeta llamada Entities.



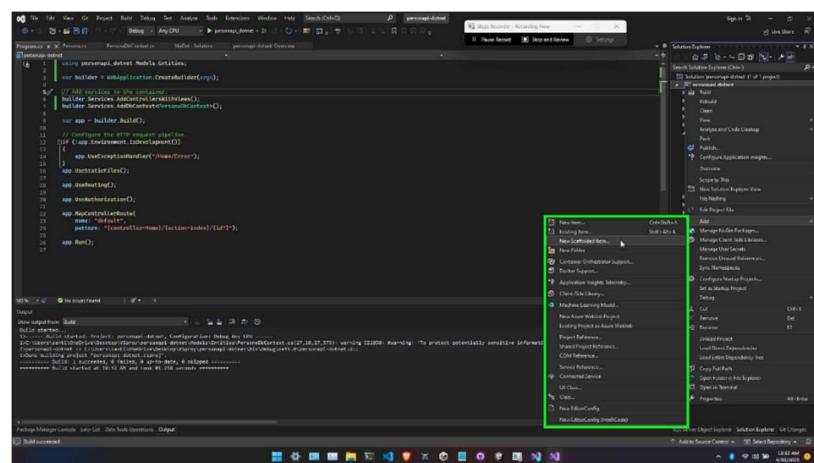


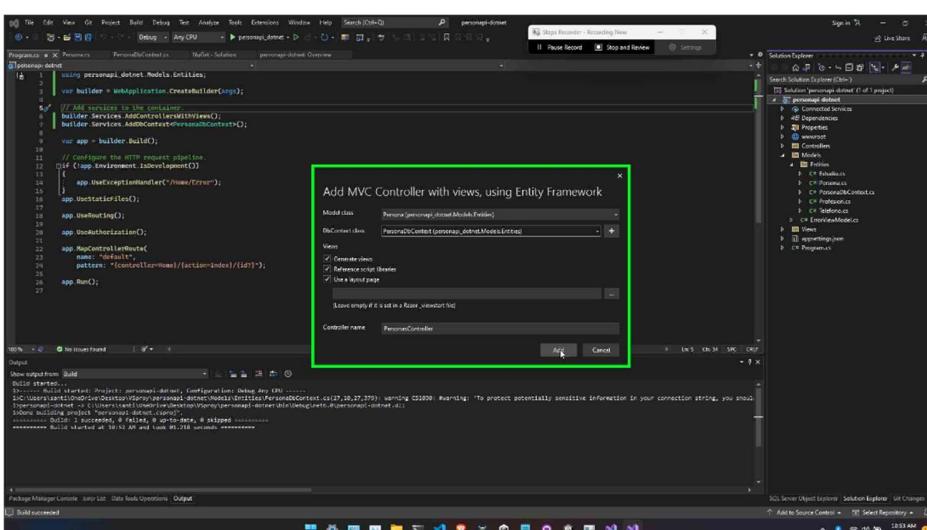
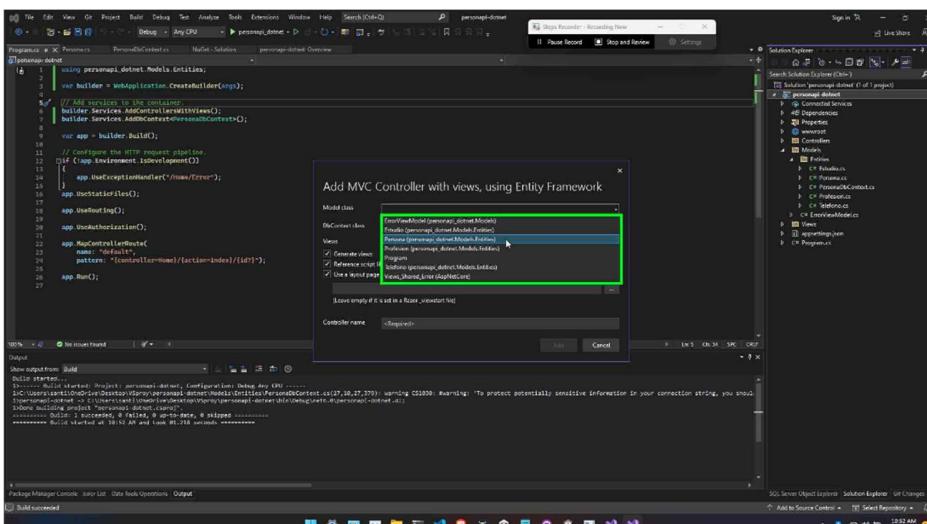
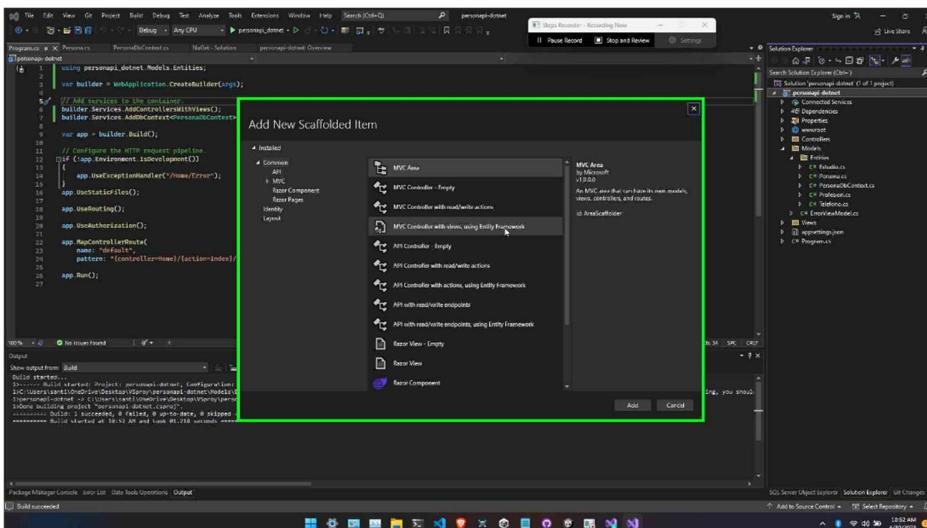
Luego de esto en el archivo Program.cs, añadir:

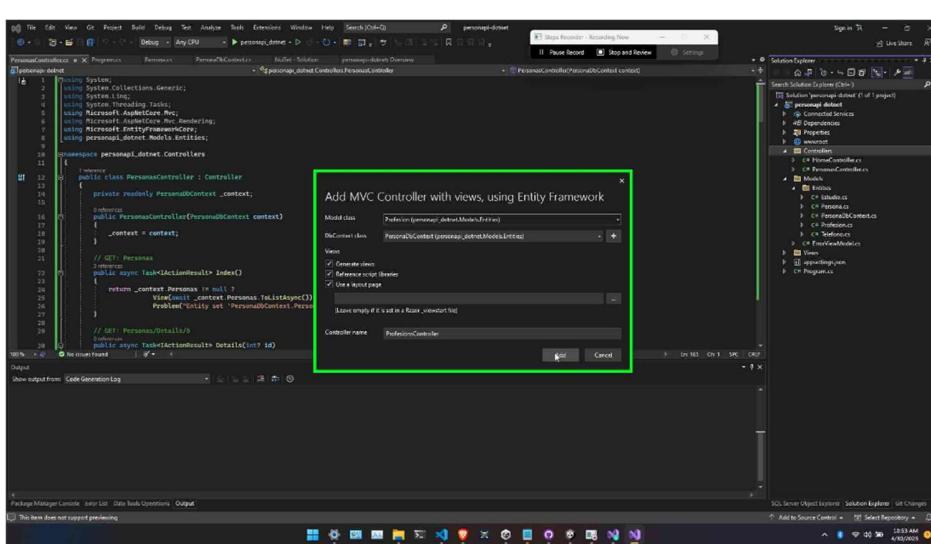
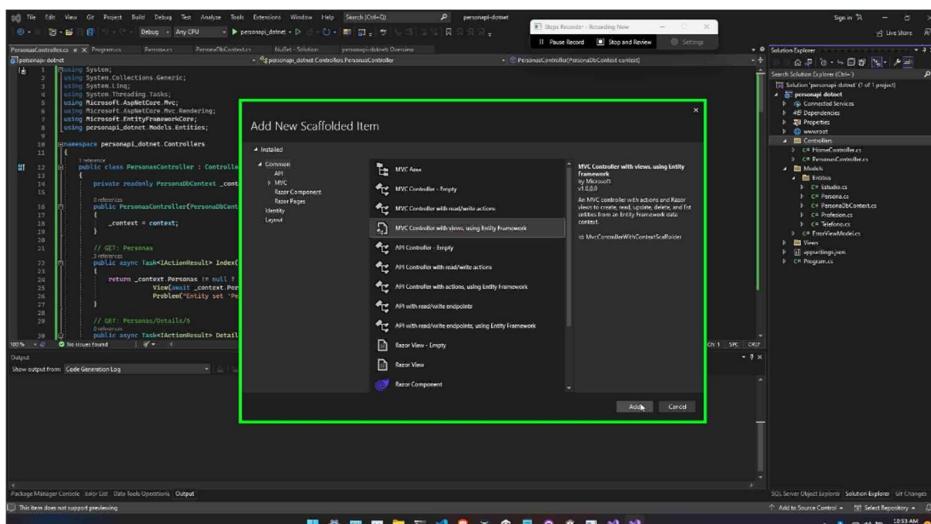
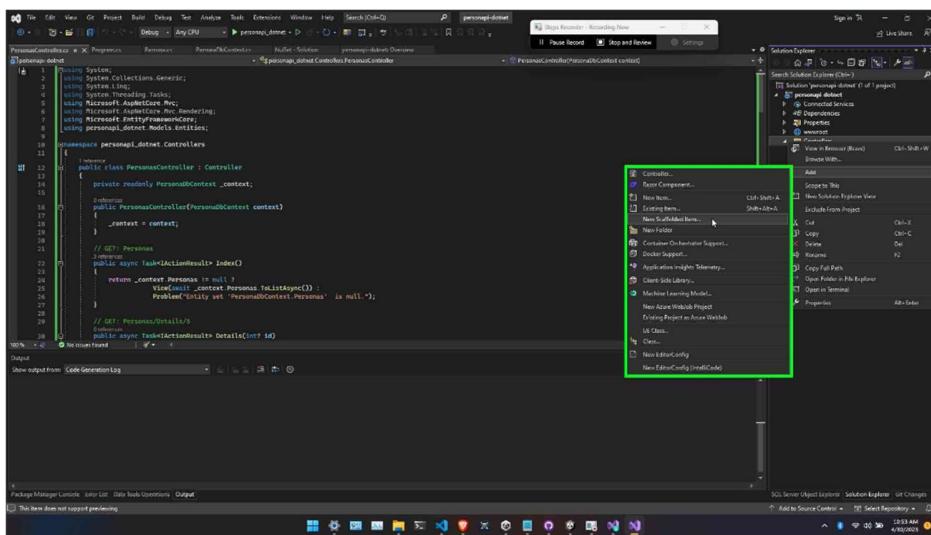
```
builder.Services.AddDbContext<PersonaDbContext>();
```

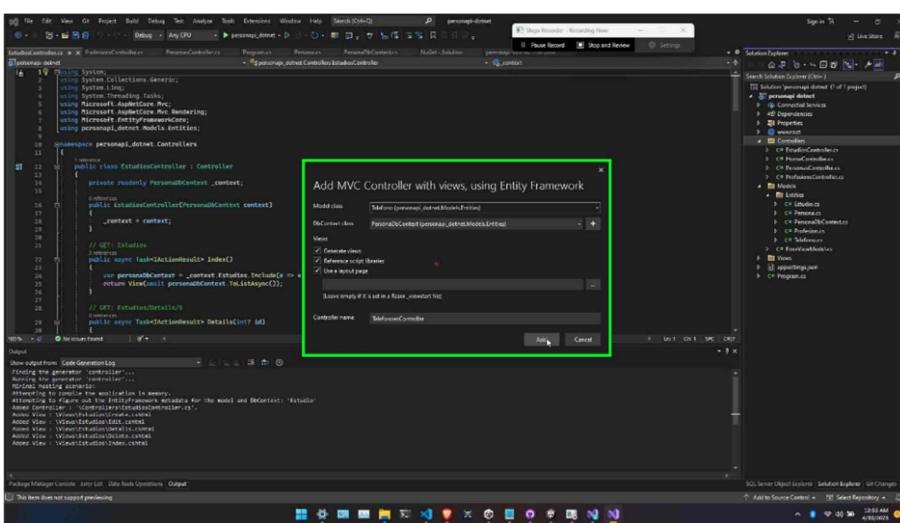
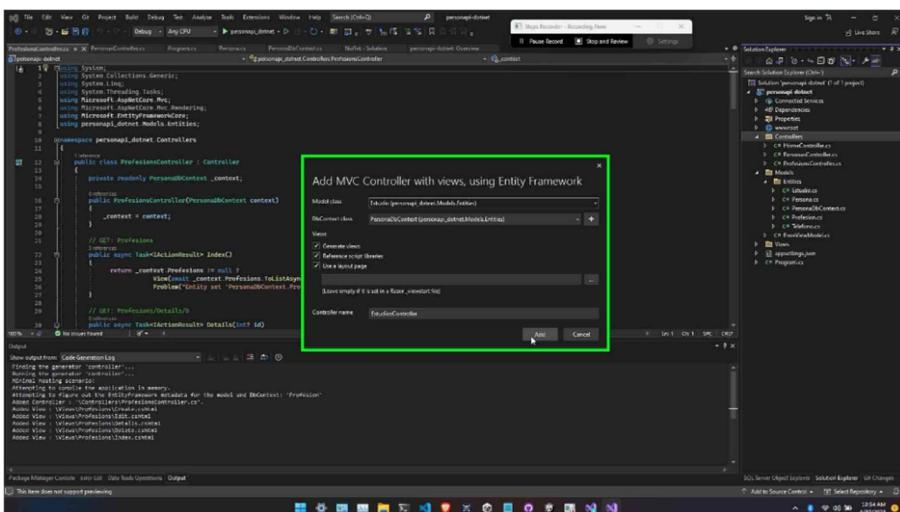


Como siguiente paso debemos crear los controladores y views como se ve en las siguientes imágenes.

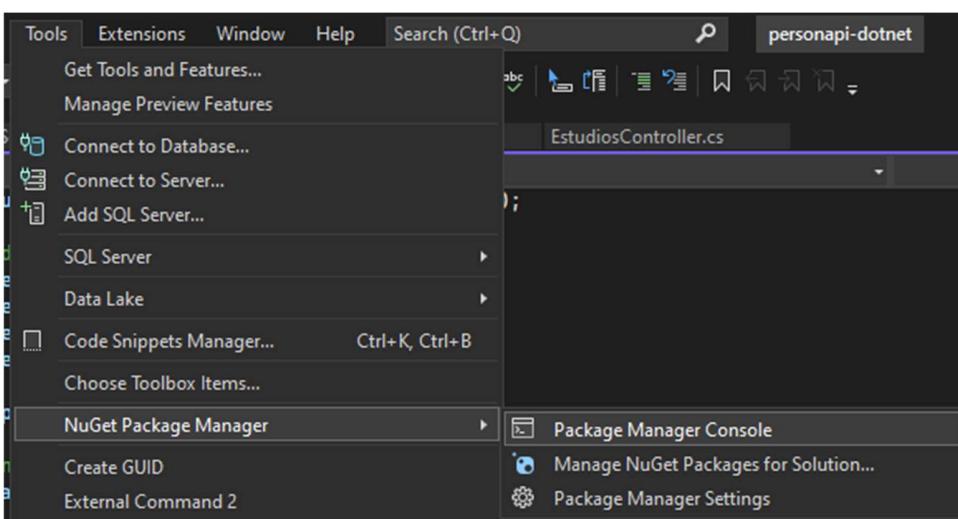








Para la integracion con swagger, abre la terminal de Package Manager Console e ingresa el comando “Install-Package Swashbuckle.AspNetCore -Version 6.2.3”



Package Manager Console

Package source: All Default project: personapi-dotnet

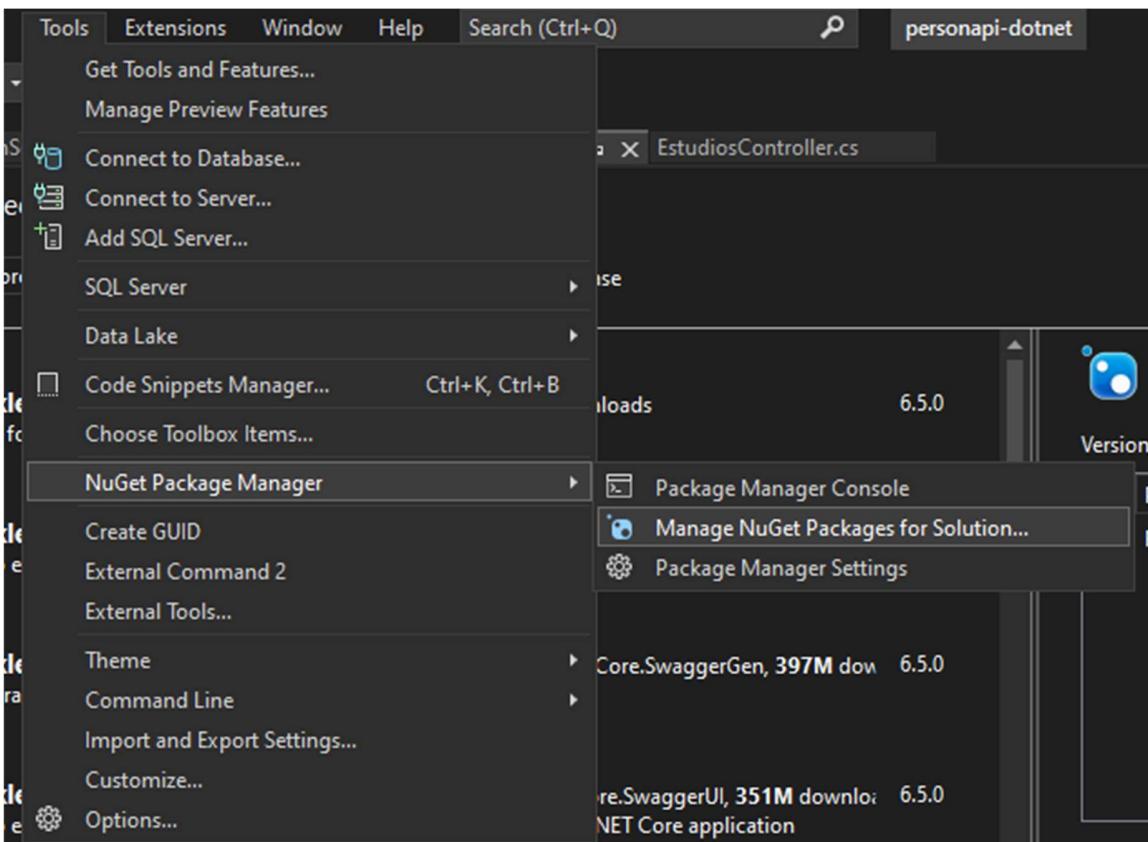
Each package is licensed to you by its owner. NuGet is not responsible for, nor does it grant any licenses to, third-party packages. Some packages may include dependencies which are governed by additional licenses. Follow the package source (feed) URL to determine any dependencies.

Package Manager Console Host Version 6.5.0.154

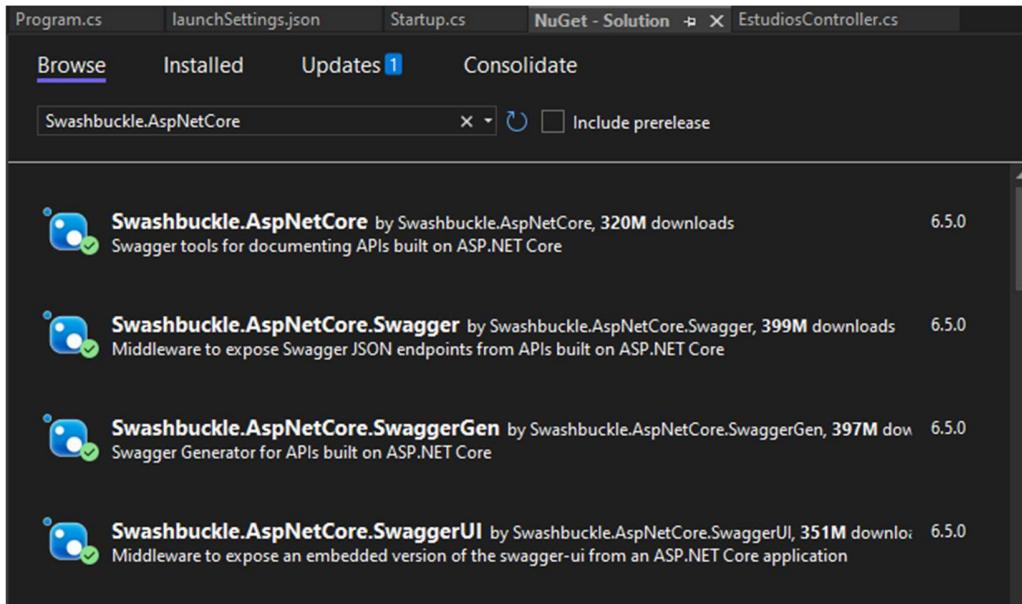
Type 'get-help NuGet' to see all available NuGet commands.

```
PM> Install-Package Swashbuckle.AspNetCore -Version 6.2.3
Restoring packages for C:\Users\USUARIO\Documents\GitHub\Laboratorios Arqui\personapi-dotnet\VS_project\personapi-dotnet\personapi-dotnet.csproj...
GET https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore/index.json 379ms
OK https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore/index.json 379ms
GET https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore/6.2.3/swashbuckle.aspnetcore.6.2.3.nupkg
OK https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore/6.2.3/swashbuckle.aspnetcore.6.2.3.nupkg 10ms
GET https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore.swaggergen/index.json
OK https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore.swaggergen/index.json 375ms
GET https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore.swaggergen/6.2.3/swashbuckle.aspnetcore.swaggergen.6.2.3.nupkg
OK https://api.nuget.org/v3-flatcontainer/swashbuckle.aspnetcore.swaggergen/6.2.3/swashbuckle.aspnetcore.swaggergen.6.2.3.nupkg 13ms
Installed Swashbuckle.AspNetCore 6.2.3 from https://api.nuget.org/v3/index.json with content hash cnzQ0n0Le+hInsw2SYw10hOCPXpYi/szcvnyqZJ12v+QvrLBwAMwXBg6RiyhB18s/mLeyC+C+g209ndz0IUNyQ==.
Installed Swashbuckle.AspNetCore.SwaggerGen 6.2.3 from https://api.nuget.org/v3/index.json with content hash +Xq7wdlCCfcxlnlbJVFNng8ITdP2TRYIpb6IKzDw5FwFxd191BfNDtcT+wkKwX70iBBFmXldnd02/V072A==.
Installing NuGet package Swashbuckle.AspNetCore 6.2.3.
Generating MSBuild file C:\Users\USUARIO\Documents\GitHub\Laboratorios Arqui\personapi-dotnet\VS_project\personapi-dotnet\obj\personapi-dotnet.csproj.nuget.g.props.
Generating MSBuild file C:\Users\USUARIO\Documents\GitHub\Laboratorios Arqui\personapi-dotnet\VS_project\personapi-dotnet\obj\personapi-dotnet.csproj.nuget.g.targets.
Writing assets file to disk. Path: C:\Users\USUARIO\Documents\GitHub\Laboratorios Arqui\personapi-dotnet\VS_project\personapi-dotnet\obj\project.assets.json
Restored C:\Users\USUARIO\Documents\GitHub\Laboratorios Arqui\personapi-dotnet\VS_project\personapi-dotnet\personapi-dotnet.csproj (in 1.14 sec).
Successfully uninstalled 'Microsoft.Extensions.ApiDescription.Server 6.0.5' from personapi-dotnet
Successfully uninstalled 'Swashbuckle.AspNetCore 6.5.0' from personapi-dotnet
Successfully uninstalled 'Swashbuckle.AspNetCore.SwaggerGen 6.5.0' from personapi-dotnet
Successfully installed 'Microsoft.Extensions.ApiDescription.Server 3.0.0' to personapi-dotnet
Successfully installed 'Swashbuckle.AspNetCore 6.2.3' to personapi-dotnet
Successfully installed 'Swashbuckle.AspNetCore.SwaggerGen 6.2.3' to personapi-dotnet
Executing nuget actions took 468 ms
Time Elapsed: 00:00:01.0991238
PM> |
```

Alternativa parte 1: Abre el Nuget Packages for Solutions mediante Tools>NuGet Package Manager>Manage NuGet Packages for Solutions



Alternativa parte 2: Instalar las siguientes extensiones mediante el Nuget Package for solutions en las versiones más recientes.



Una vez instalados, en el archivo de "Program.cs" agrega las siguientes líneas de código para que el programa ejecute el swagger usando el /swagger.

```
var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();
builder.Services.AddControllersWithViews();

var app = builder.Build();

// Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Home/Error");
}

app.UseSwagger();
app.UseSwaggerUI();

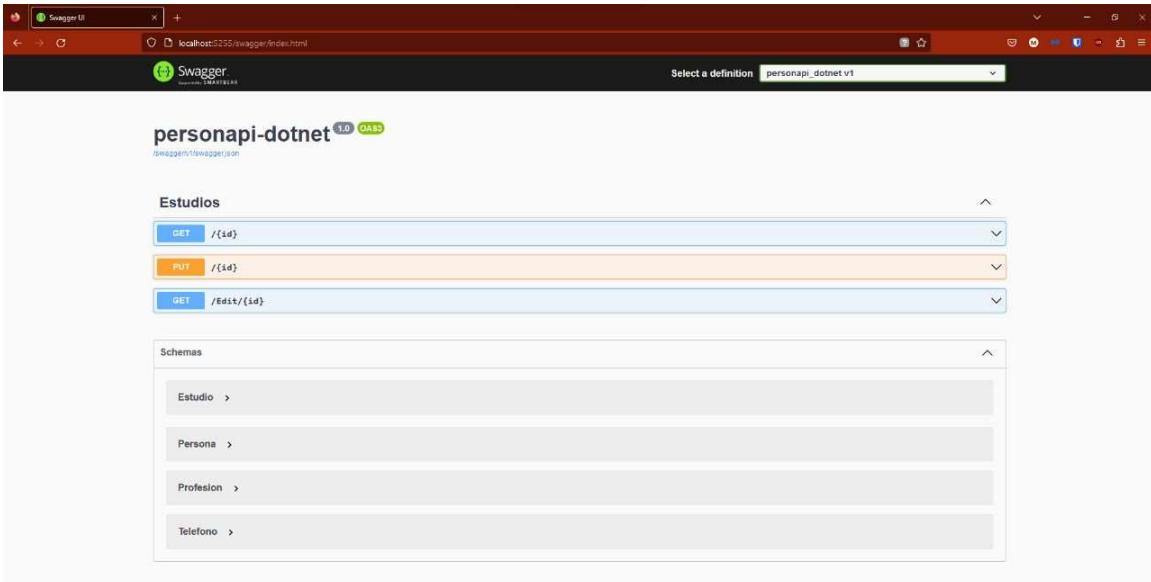
app.UseStaticFiles();
app.UseRouting();
app.UseAuthorization();

app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();
```

Realiza el build y ejecuta el API para poder acceder a la UI de swagger.

Resultados



The screenshot shows the Swagger UI interface for the 'personapi-dotnet' API. At the top, there's a navigation bar with a back button, forward button, refresh button, and a search bar labeled 'Select a definition' with the value 'personapi_dotnet v1'. Below the header, the title 'personapi-dotnet' is displayed along with its version '0.0' and a 'GATE' badge. The main content area is divided into two sections: 'Estudios' and 'Schemas'. The 'Estudios' section contains three operation boxes: a blue 'GET /{id}', an orange 'PUT /{id}', and a blue 'GET /Edit/{id}'. The 'Schemas' section lists four entities: 'Estudio >', 'Persona >', 'Profesion >', and 'Telefono >'. Each entity has a small arrow icon next to it.

Como podemos observar desde el swagger se ve representado nuestro laboratorio, aunque tuvimos problemas con el swagger debido a una super posición de algunas direcciones, por lo cual no se muestran todas las operaciones del CRUD.

Conclusiones y lecciones aprendidas

Después de realizar el procedimiento descrito para el laboratorio podemos decir que entre las lecciones aprendidas se encuentran que:

- Es importante planificar adecuadamente el proceso de configuración del entorno de desarrollo, incluyendo la instalación y configuración de todas las herramientas necesarias.
- La documentación y organización del código es clave para garantizar la legibilidad y la mantenibilidad del proyecto a largo plazo. En particular, la documentación de la API utilizando Swagger 3 puede facilitar el trabajo de los desarrolladores que trabajen con el código en el futuro.
- La implementación de patrones de diseño como la separación de responsabilidades entre las entidades, los repositorios y los controladores puede facilitar el desarrollo de aplicaciones escalables y mantenibles.
- Es importante realizar pruebas y depuraciones rigurosas para asegurar la calidad del código antes de implementarlo en producción.
- La implementación de un sistema de control de versiones como Git puede facilitar el trabajo colaborativo y el mantenimiento del proyecto a largo plazo.

Referencias

SQL Server <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>

SQL Server Manager Studio <https://learn.microsoft.com/es-es/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver16>

Visual Studio <https://visualstudio.microsoft.com/es/free-developer-offers/>