



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



FACULTAD DE
Ingeniería

FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN SISTEMAS Y
COMPUTACION

Tema

Informe Servidor y Cliente Web (CRUD MVC)

Nombre del Estudiante

John Jairo Guaman Pineda

Fecha

11-06-2024

Código Estudiantil

57322



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Objetivo General:

- Describir y analizar la implementación de un sistema web basado en la arquitectura MVC (Modelo-Vista-Controlador) que permita realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar) tanto en el servidor como en el cliente.

Objetivos Específicos:

- Examinar la estructura y funcionamiento del patrón MVC en el contexto de aplicaciones web, destacando sus ventajas y desventajas en la gestión de operaciones CRUD.
- Desarrollar un prototipo funcional que demuestre la comunicación entre el cliente y el servidor, implementando operaciones CRUD mediante tecnologías web actuales.



Introducción

En la era digital actual, las aplicaciones web han adquirido una relevancia sin precedentes, facilitando la interacción entre usuarios y sistemas a través de plataformas accesibles desde cualquier lugar del mundo. La arquitectura MVC (Modelo-Vista-Controlador) se ha consolidado como un estándar en el desarrollo de estas aplicaciones, ofreciendo una separación clara entre la lógica de negocio, la interfaz de usuario y el control de flujo, lo que promueve un desarrollo más organizado y mantenible.

Este informe se centra en la implementación de un sistema web que permita realizar operaciones CRUD (Crear, Leer, Actualizar y Eliminar), esenciales para la gestión de datos en cualquier aplicación interactiva. Utilizando la arquitectura MVC, exploraremos cómo se puede diseñar y desarrollar tanto el lado del servidor como el del cliente para asegurar una comunicación eficiente y segura entre ambos.

A lo largo de este documento, se presentará una visión detallada del patrón MVC, sus componentes y cómo estos interactúan para llevar a cabo las operaciones CRUD. Además, se desarrollará un prototipo funcional que ejemplifica estos conceptos, utilizando tecnologías web modernas. El objetivo es proporcionar una comprensión profunda de los mecanismos que subyacen en las aplicaciones web actuales, y ofrecer una guía práctica para su implementación efectiva.

Mediante la combinación de teoría y práctica, este informe no solo busca explicar los principios fundamentales del desarrollo web con MVC y CRUD, sino también demostrar su aplicabilidad en escenarios del mundo real, contribuyendo así al conocimiento y habilidades necesarios para crear aplicaciones web robustas y escalables.



Marco Teórico

Arquitectura MVC (Modelo-Vista-Controlador)

La arquitectura MVC es un patrón de diseño que se utiliza ampliamente en el desarrollo de aplicaciones web. MVC divide una aplicación en tres componentes interrelacionados:

- **Modelo (Model):** Representa la lógica de negocio y la estructura de datos de la aplicación. Es responsable de manejar la lógica de la aplicación y los datos, y notificar a la Vista cuando hay cambios en los datos.
- **Vista (View):** Se encarga de la presentación de los datos. Es la interfaz de usuario que muestra los datos del Modelo. La Vista es responsable de la presentación de los datos en un formato comprensible para el usuario.
- **Controlador (Controller):** Actúa como un intermediario entre el Modelo y la Vista. El Controlador recibe las entradas del usuario, procesa esas entradas (interactuando con el Modelo si es necesario), y actualiza la Vista con los resultados de la operación. (Hernandez, 2015)

Este patrón promueve la separación de responsabilidades, facilitando la gestión y el mantenimiento del código. Cada componente tiene una función específica y no se solapan entre ellos, lo que permite modificar uno sin afectar directamente a los otros.

Tecnologías usada:

MySQLServer:

MySQL Server es un sistema de gestión de bases de datos relacionales (SGBDR) de código abierto y gratuito que goza de gran popularidad por su facilidad de uso, flexibilidad y escalabilidad. Es ampliamente utilizado en aplicaciones web y de escritorio, tanto para pequeños proyectos como para grandes empresas.

Las principales características de MySQL Server incluyen:

- **Arquitectura cliente-servidor:** Separa la lógica de la base de datos en dos componentes: el servidor, que reside en un ordenador, y el cliente, que se ejecuta en la aplicación del usuario.
- **Lenguaje de consulta SQL:** Permite interactuar con la base de datos de manera eficiente mediante comandos sencillos y comprensibles.
- **Soporte para múltiples plataformas:** Funciona en diversos sistemas operativos, incluyendo Windows, Linux y macOS.
- **Alto rendimiento:** Es capaz de manejar grandes volúmenes de datos con rapidez y eficiencia.
- **Amplia comunidad:** Cuenta con una comunidad grande y activa que proporciona soporte y recursos para los usuarios. (De TechTarget, 2021)



Postman:

Postman es una herramienta de plataforma de desarrollo de API de colaboración que facilita el proceso de diseño, prueba y uso de APIs. Es ampliamente utilizada por desarrolladores, testers y gestores de productos para crear, documentar y consumir APIs de manera eficiente.

Las principales características de Postman incluyen:

- Interfaz gráfica intuitiva: Permite crear y enviar solicitudes HTTP, visualizar respuestas y analizar datos en diferentes formatos de forma sencilla.
- Entorno de prueba integrado: Facilita la prueba de APIs en diferentes escenarios y condiciones.
- Colecciones y carpetas: Permite organizar las solicitudes y respuestas en colecciones y carpetas para una mejor gestión.
- Colaboración en equipo: Facilita el trabajo en equipo al permitir compartir colecciones y entornos de prueba con otros usuarios.
- Amplia biblioteca de extensiones: Ofrece una amplia biblioteca de extensiones que amplían las funcionalidades de Postman. (Sydle, 2024)

Visual Studio 2022:

Visual Studio 2022 es un entorno de desarrollo integrado (IDE) desarrollado por Microsoft para crear aplicaciones web, de escritorio y móviles. Es una herramienta completa que ofrece una amplia gama de funcionalidades para el desarrollo de software.

Las principales características de Visual Studio 2022 incluyen:

- Editor de código potente: Permite escribir, depurar y probar código de manera eficiente.
- Herramientas de diseño visual: Facilita el diseño de interfaces gráficas de usuario (UI) y experiencias de usuario (UX).
- Soporte para múltiples lenguajes de programación: Soporta una amplia gama de lenguajes de programación, incluyendo C#, Visual Basic .NET, F#, Python, JavaScript y C++.
- Integración con herramientas de control de versiones: Permite trabajar con sistemas de control de versiones como Git y Subversion.
- Depuración integrada: Facilita la depuración de código para identificar y corregir errores.
- Publicación de aplicaciones: Permite publicar aplicaciones en diferentes plataformas, incluyendo Windows, macOS, Linux, Android e iOS. (Anandmeg, 2023)



Navegador Edge:

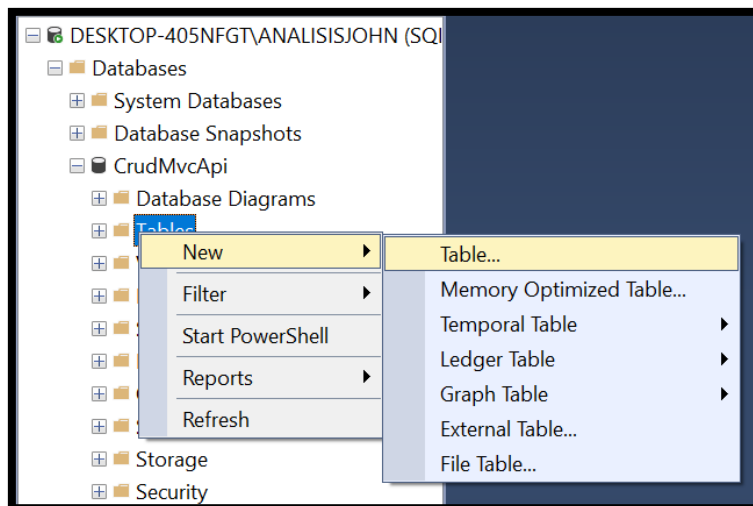
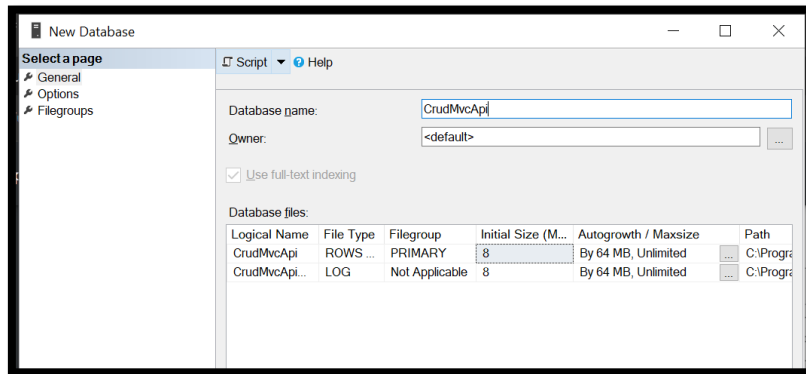
Microsoft Edge es un navegador web desarrollado por Microsoft para Windows 10 y 11. Es el navegador predeterminado en estos sistemas operativos y ofrece una interfaz moderna, compatibilidad con estándares web y herramientas para mejorar la privacidad y seguridad del usuario.

Las principales características de Edge incluyen:

- **Interfaz moderna y minimalista:** Ofrece una interfaz limpia y sencilla de usar.
- **Compatibilidad con estándares web:** Soporta los últimos estándares web, lo que garantiza una experiencia de navegación consistente y segura.
- **Motor de renderizado rápido:** Utiliza el motor de renderizado Chromium, que ofrece un rendimiento rápido y eficiente.
- **Herramientas de privacidad y seguridad:** Incluye herramientas para proteger la privacidad del usuario, como la prevención de seguimiento y la gestión de cookies.
- **Integración con Windows:** Se integra con otras aplicaciones de Windows, como el Calendario y las Notas.

Desarrollo

Crear la base de datos con sus tablas correspondientes



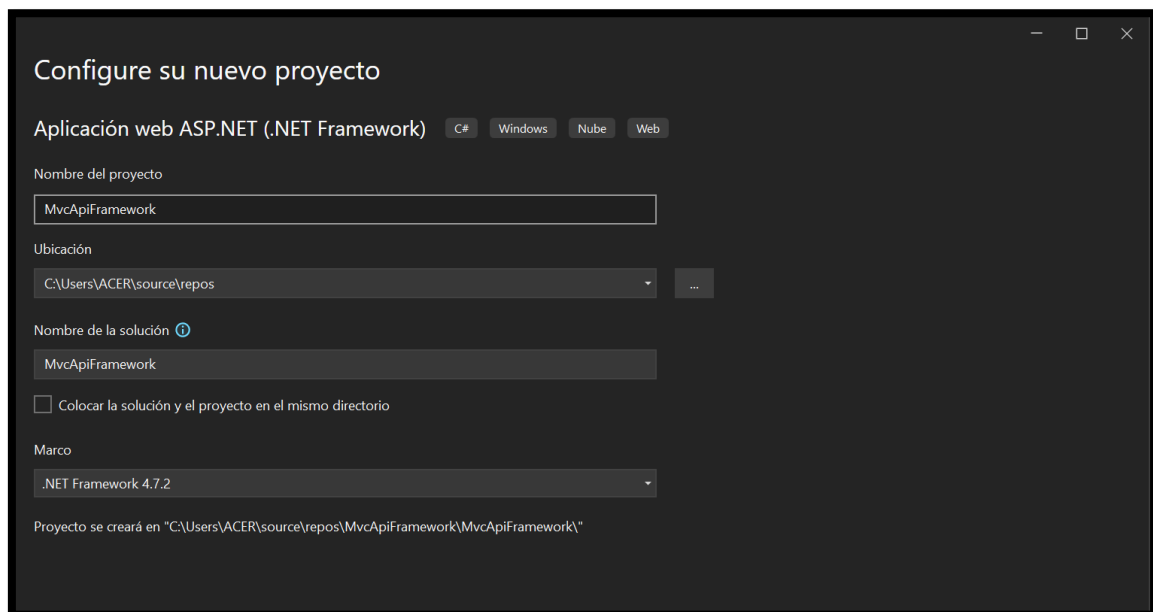
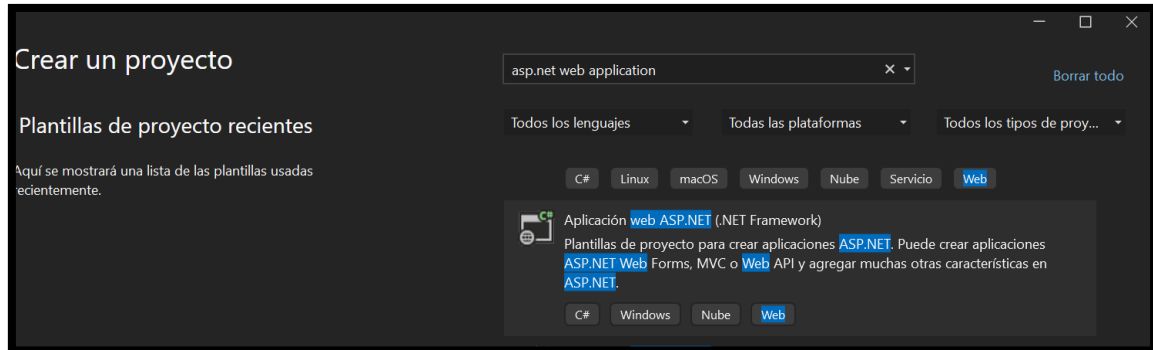
DESKTOP-405NFGT\...cApi - dbo.people			
	Column Name	Data Type	Allow Nulls
PK	id	int	<input type="checkbox"/>
	name	varchar(50)	<input type="checkbox"/>
	age	int	<input type="checkbox"/>
			<input type="checkbox"/>



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Crear un proyecto entity framework

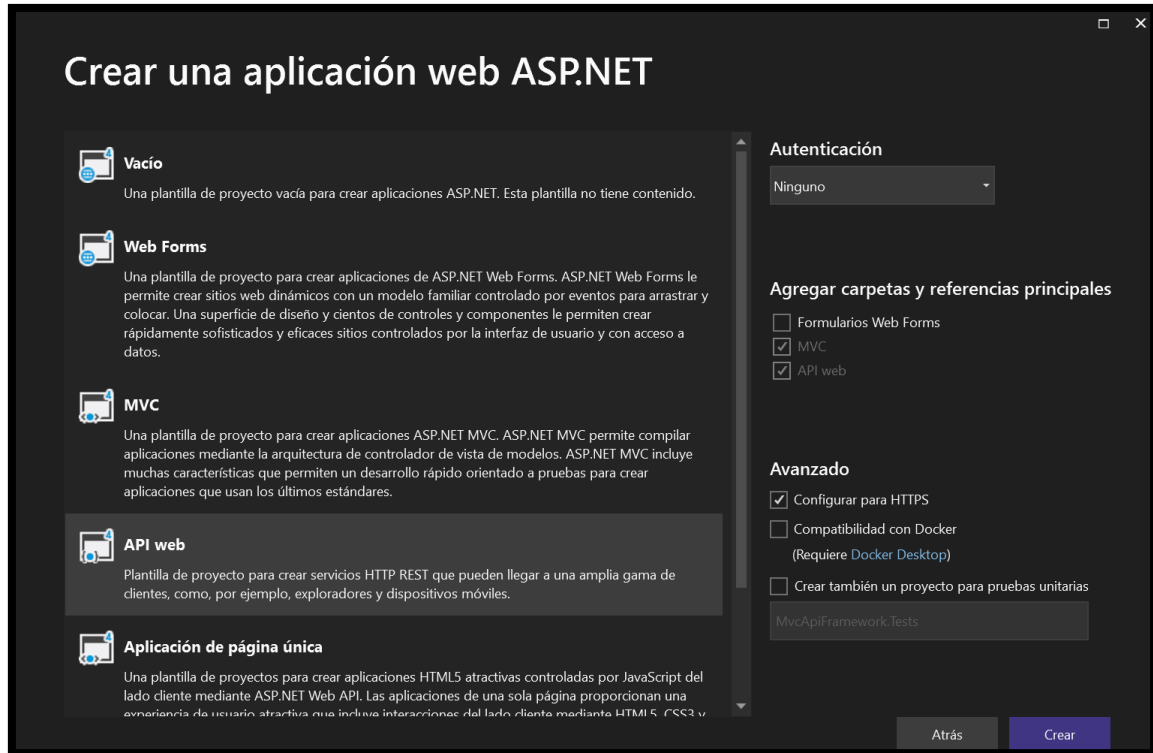




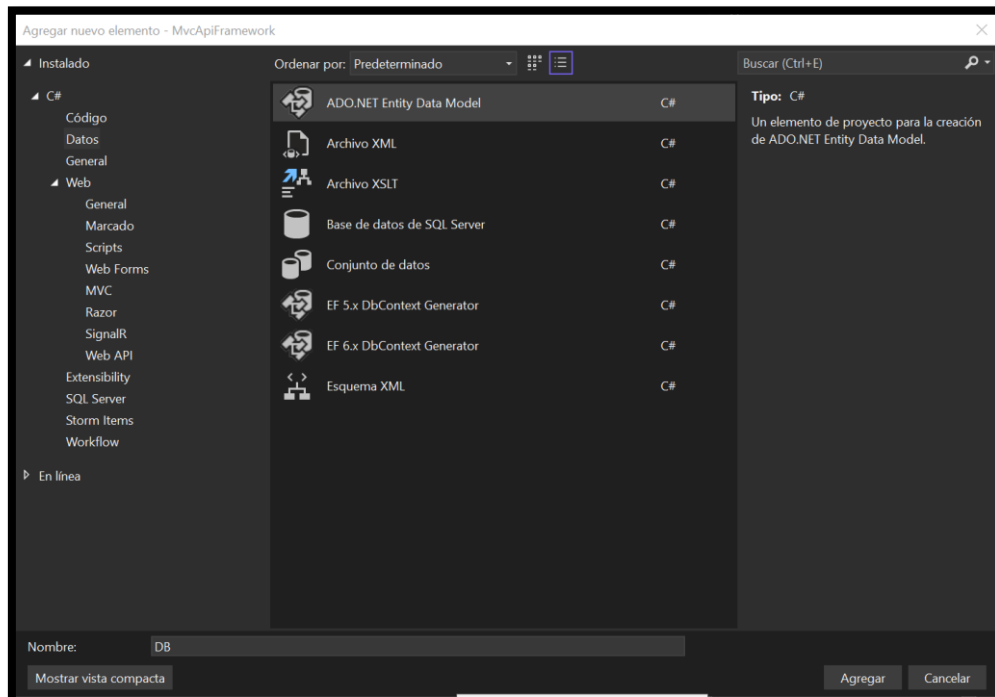
UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Seleccionar web api

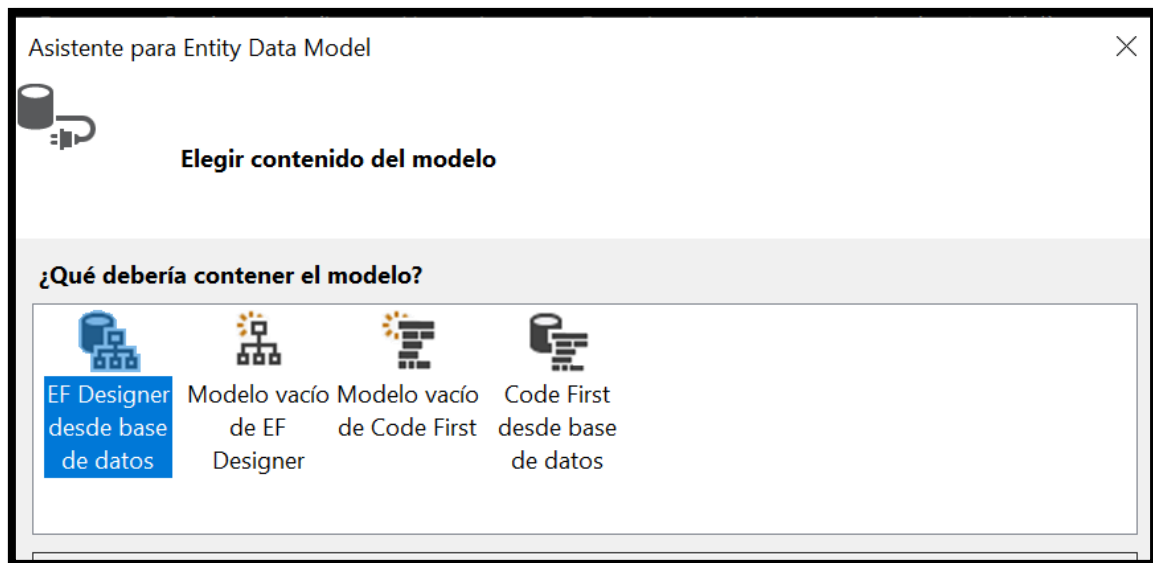


Agregar un nuevo elemento al proyecto





UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Agregar una nueva conexión

Realizar un test de conexión para verificar que está conectado



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS

Asistente para Entity Data Model

Elegir la conexión de datos

¿Qué conexión de datos debe usar la aplicación para conectarse a la base de datos?

desktop-405nfgt\analisisjohn.CrudMvcApi.dbo Nueva conexión...

Esta cadena de conexión parece contener datos confidenciales (por ejemplo, una contraseña) que son necesarios para conectarse con la base de datos. Almacenar datos confidenciales en la cadena de conexión puede suponer un riesgo para la seguridad. ¿Desea incluir estos datos en la cadena de conexión?

☐ No, excluir datos confidenciales de la cadena de conexión. Los estableceré en el código de mi aplicación.

☐ Sí, incluir datos confidenciales en la cadena de conexión.

Cadena de conexión:

```
metadata=res://*/DB.csdl|res://*/DB.ssdl|res://*/DB.msl;provider=System.Data.SqlClient;provider connection string="data source=DESKTOP-405NFGT\ANALISISJOHN;initial catalog=CrudMvcApi;integrated security=True;trustservercertificate=True;MultipleActiveResultSets=True;App=EntityFramework"
```

☒ Guardar configuración de conexión en Web.Config como:

CrudMvcApiEntities

< Anterior Siguiente > Finalizar Cancelar

Asistente para Entity Data Model

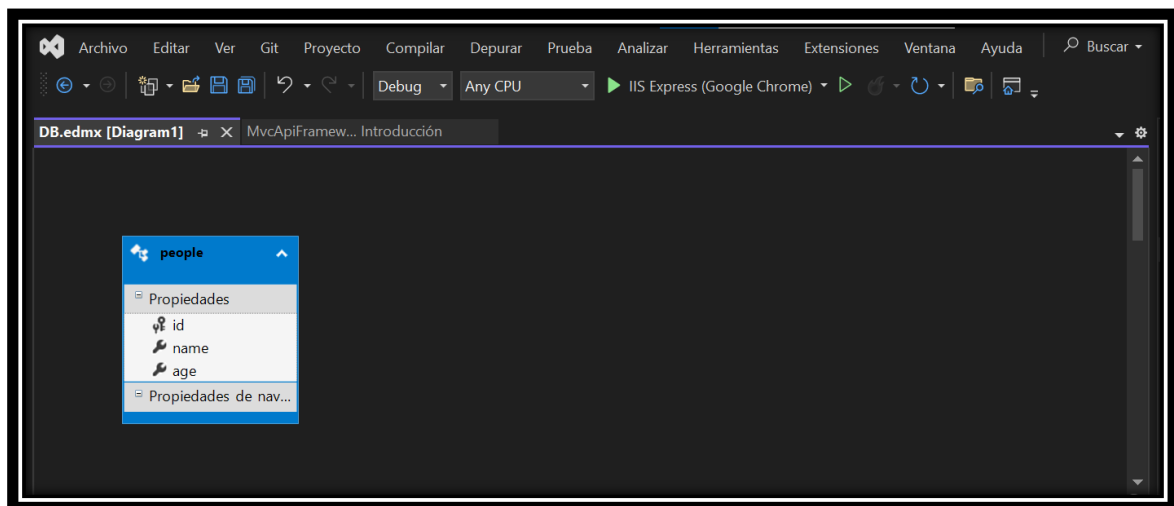
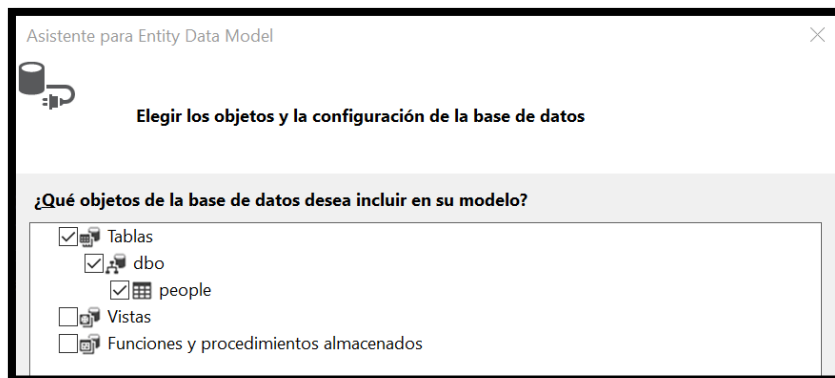
Elija su versión

¿Qué versión de Entity Framework desea usar?

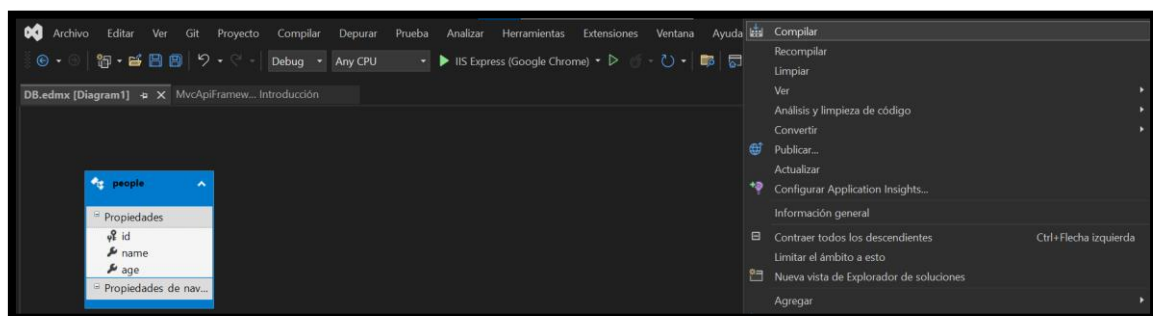
☒ Entity Framework 6.x

☐ Entity Framework 5.0

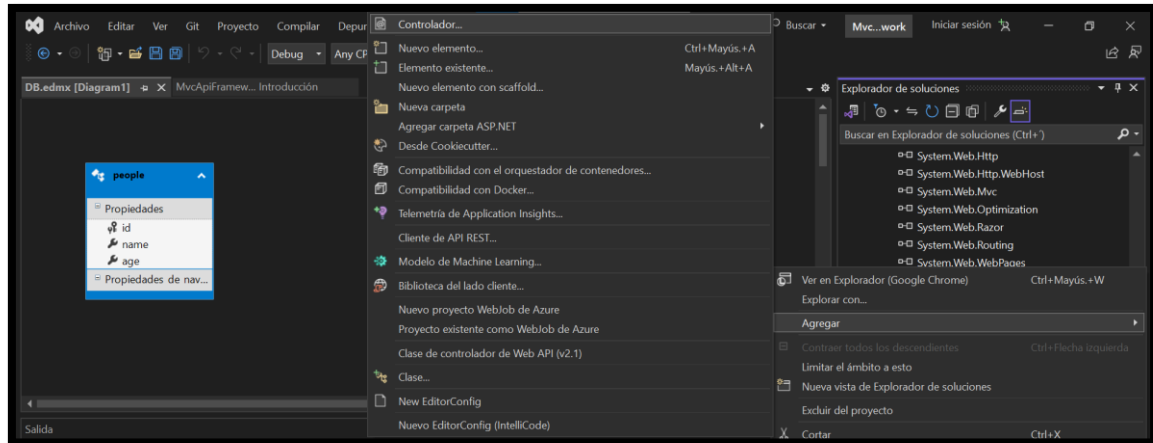
i También es posible instalar y usar otras versiones de Entity Framework.
[Obtener más información sobre esto](#)



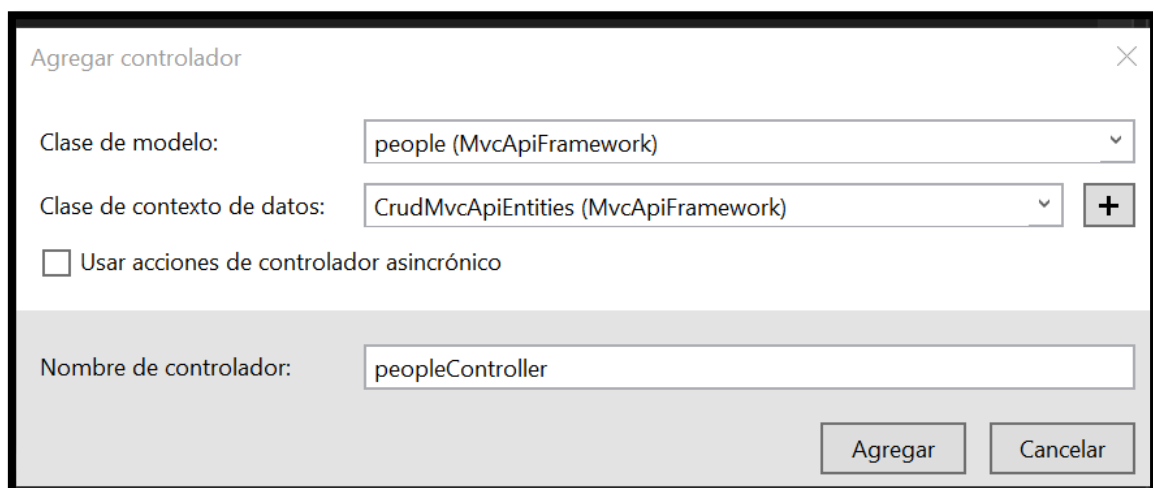
Crear un controlador a través de scaffolding, primero compilar el proyecto



Agregar un nuevo controlador

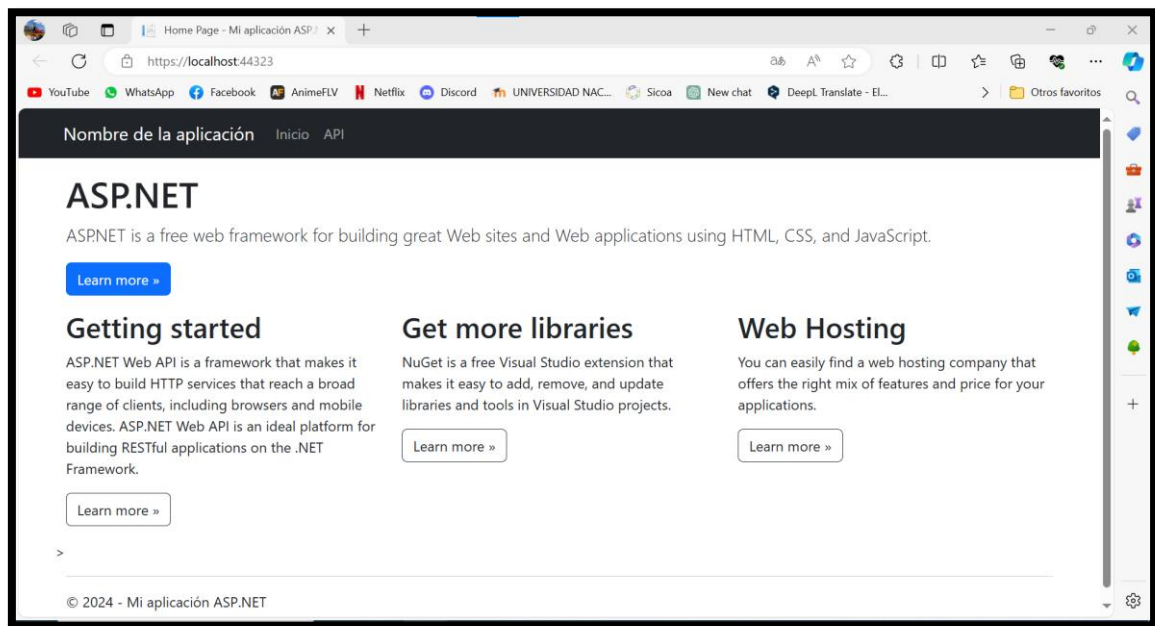


Escoger la clase people que corresponde a la tabla que vamos a crear



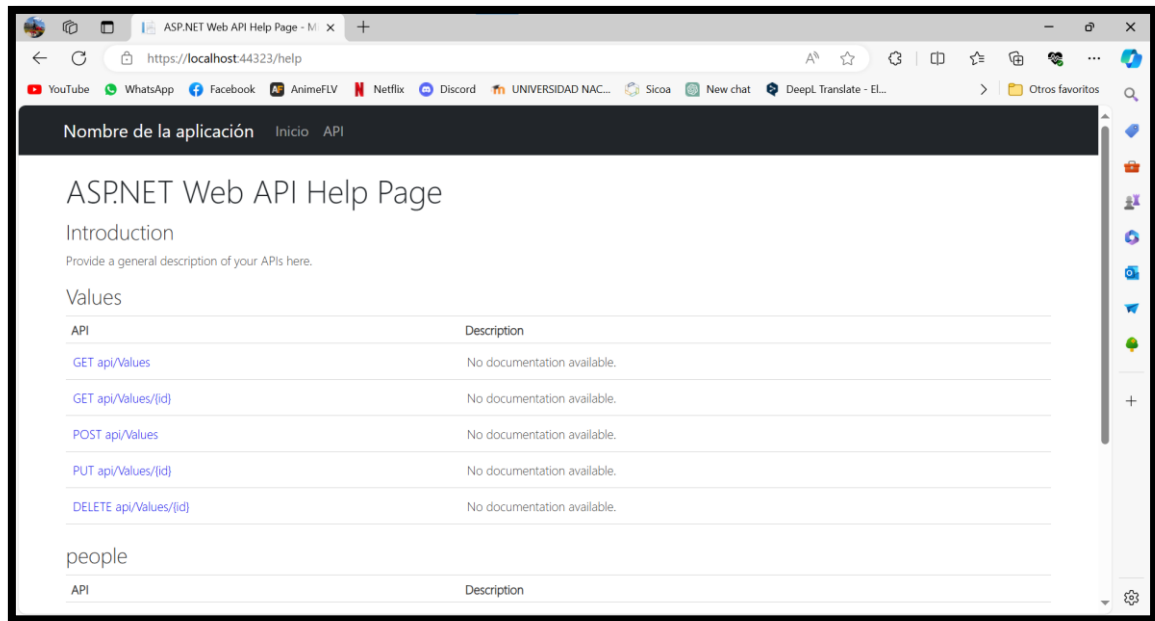


UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS

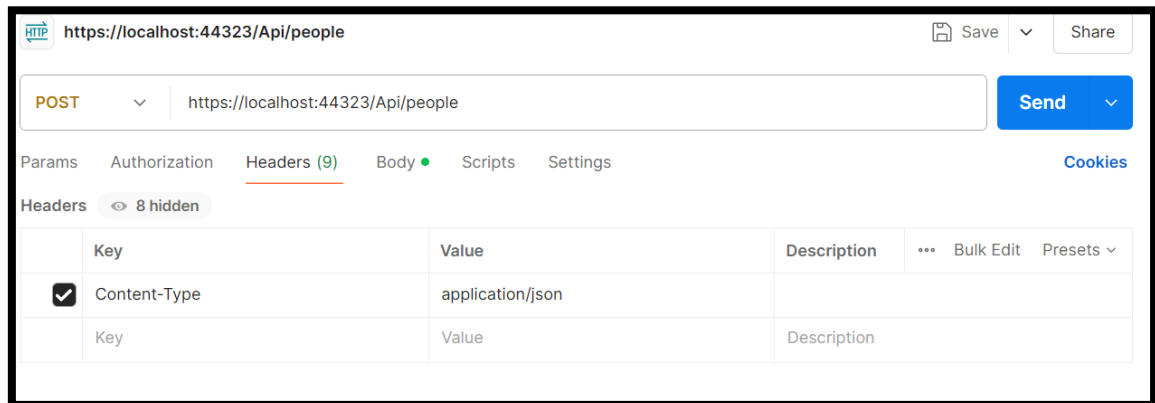




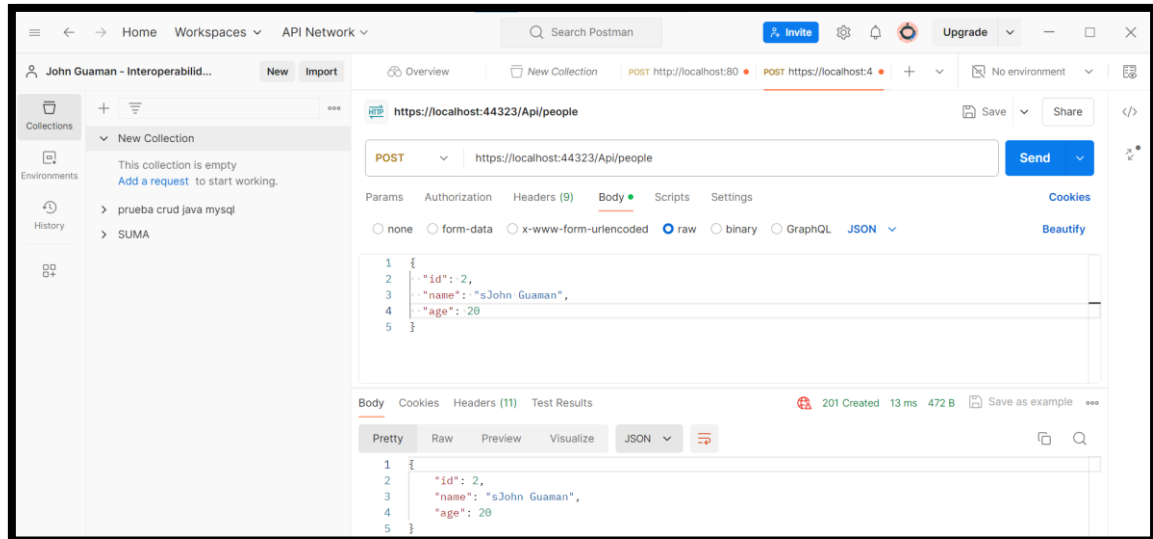
UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



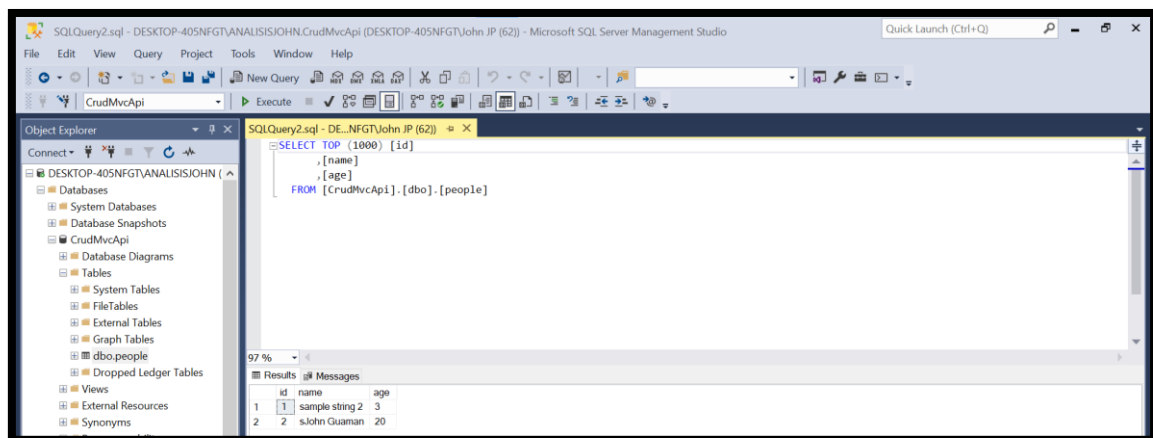
Agregar el content type en postman



Probar ingresar datos desde postman



Y como podemos ver se agrego los datos correspondientes en la base de datos que se creó inicialmente

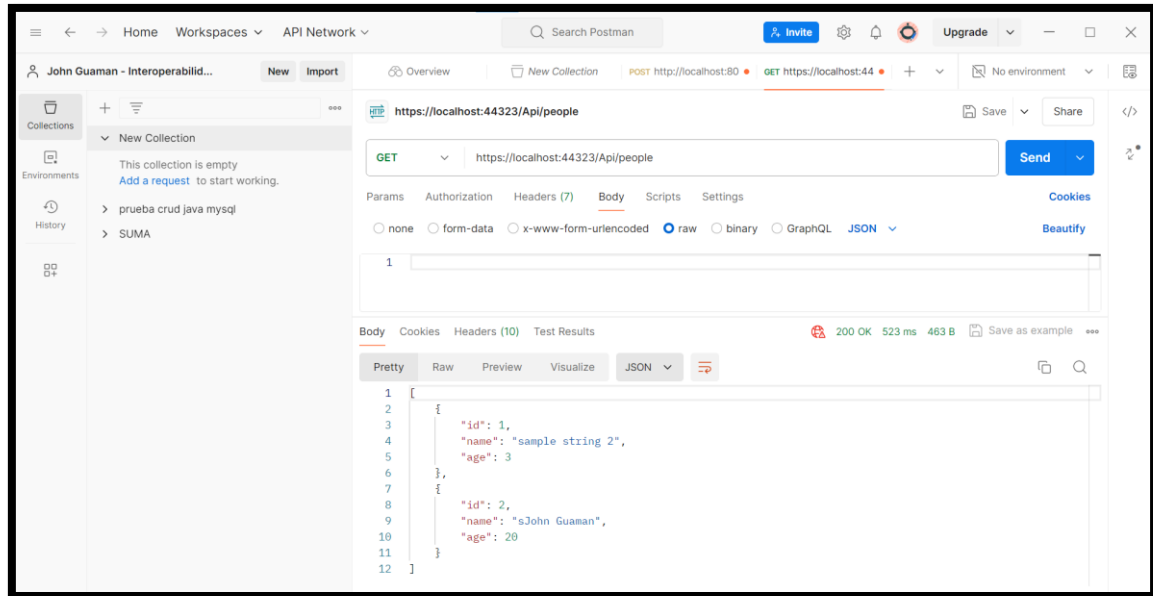




UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Se puede consultar desde postman también



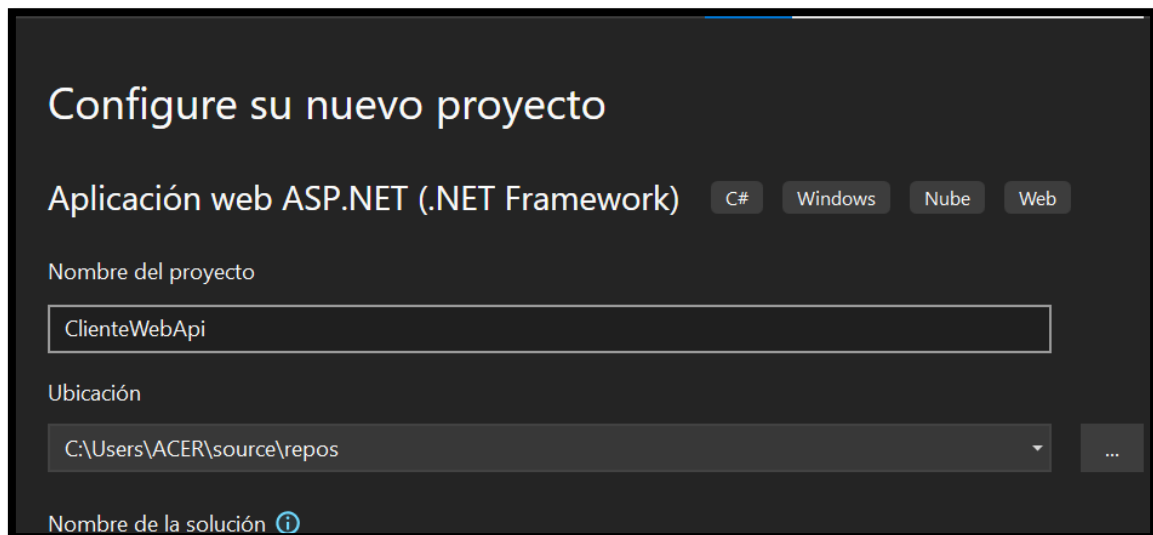
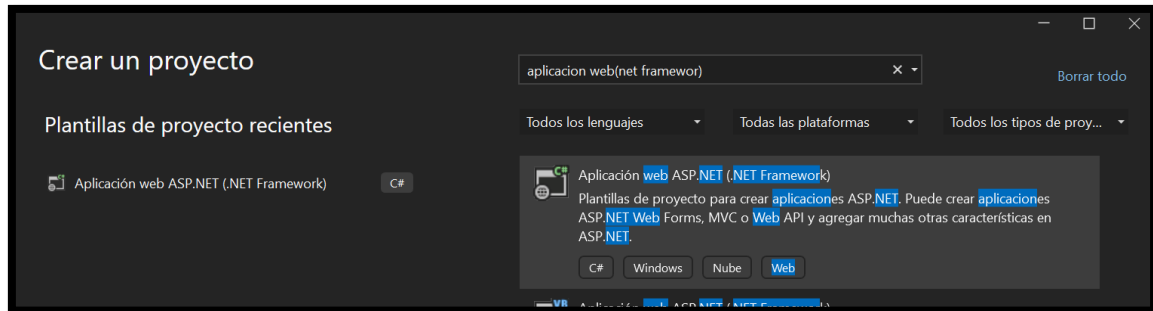


UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS

PRACTICA 2

CLIENTE PARA CONSUMIR UN CRUD

Crear un nuevo proyecto








UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS





Crear una aplicación web ASP.NET

**Vacío**
Una plantilla de proyecto vacía para crear aplicaciones ASP.NET. Esta plantilla no tiene contenido.

**Web Forms**
Una plantilla de proyecto para crear aplicaciones de ASP.NET Web Forms. ASP.NET Web Forms le permite crear sitios web dinámicos con un modelo familiar controlado por eventos para arrastrar y colocar. Una superficie de diseño y cientos de controles y componentes le permiten crear rápidamente sofisticados y eficaces sitios controlados por la interfaz de usuario y con acceso a datos.

**MVC**
Una plantilla de proyecto para crear aplicaciones ASP.NET MVC. ASP.NET MVC permite compilar aplicaciones mediante la arquitectura de controlador de vista de modelos. ASP.NET MVC incluye muchas características que permiten un desarrollo rápido orientado a pruebas para crear aplicaciones que usan los últimos estándares.

**API web**
Plantilla de proyecto para crear servicios HTTP REST que pueden llegar a una amplia gama de clientes, como, por ejemplo, exploradores y dispositivos móviles.

**Aplicación de página única**
Una plantilla de proyectos para crear aplicaciones HTML5 atractivas controladas por JavaScript del lado cliente mediante ASP.NET Web API. Las aplicaciones de una sola página proporcionan una experiencia de usuario atractiva que incluye interacciones del lado cliente mediante HTML5, CSS3 u

Autenticación
Ninguno

Agregar carpetas y referencias principales
☐ Formularios Web Forms
☒ MVC
☐ API web


Avanzado
☒ Configurar para HTTPS
☐ Compatibilidad con Docker
(Requiere [Docker Desktop](#))
☐ Crear también un proyecto para pruebas unitarias
ClienteWebApi.Tests


Atrás

Crear

Ir al administrador de paquetes

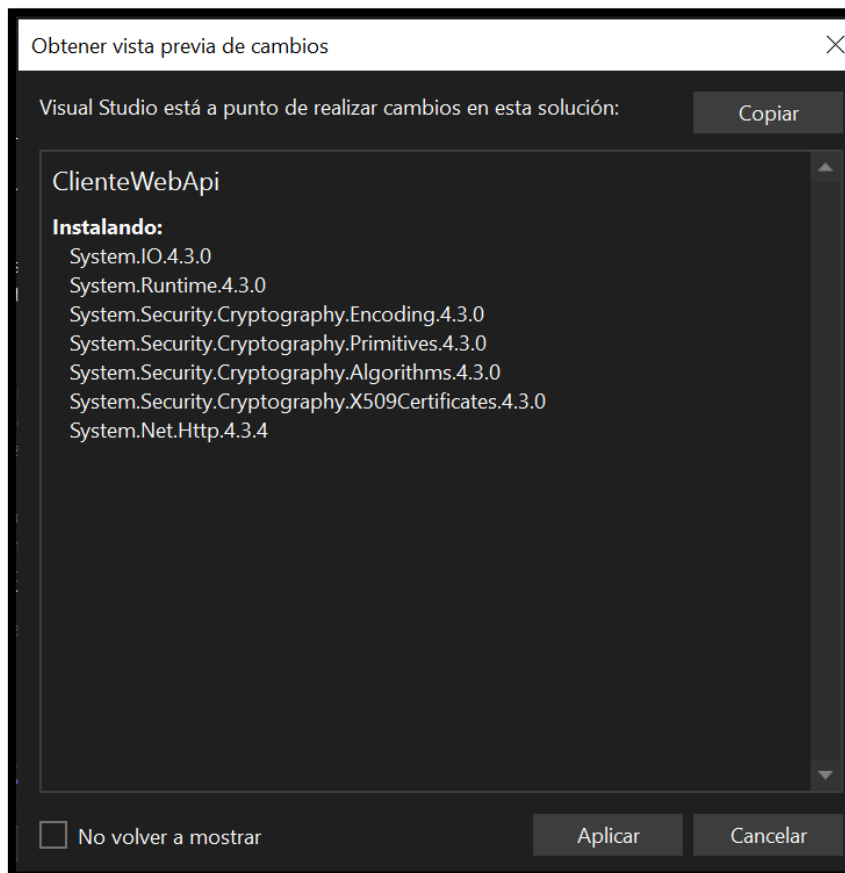
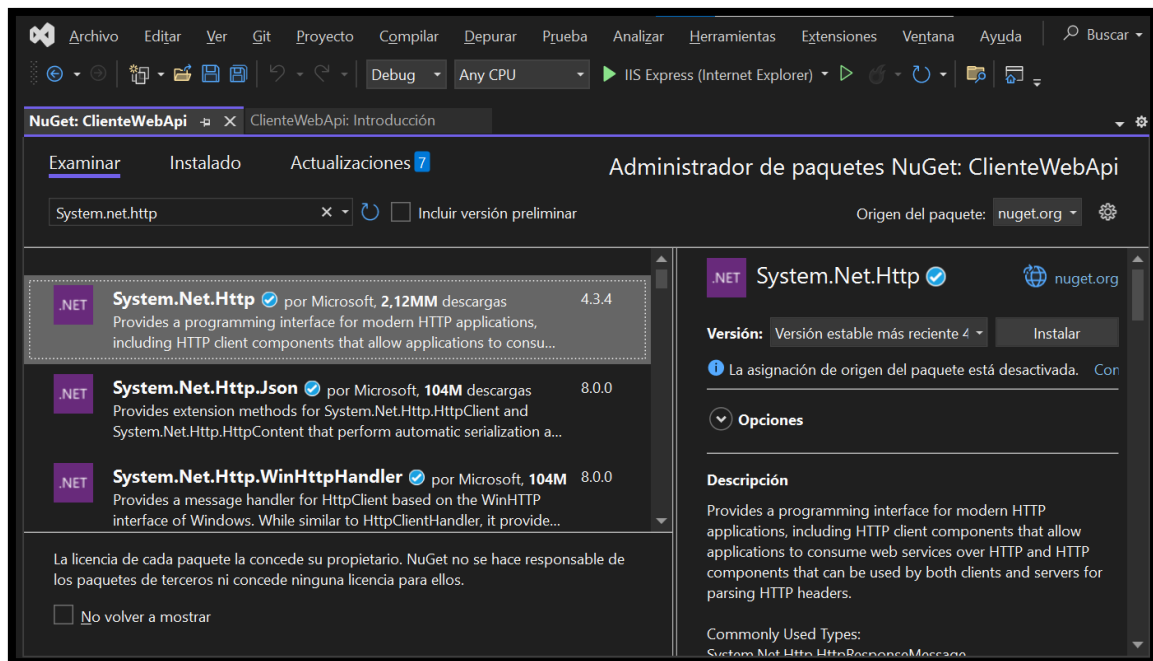
Agregar

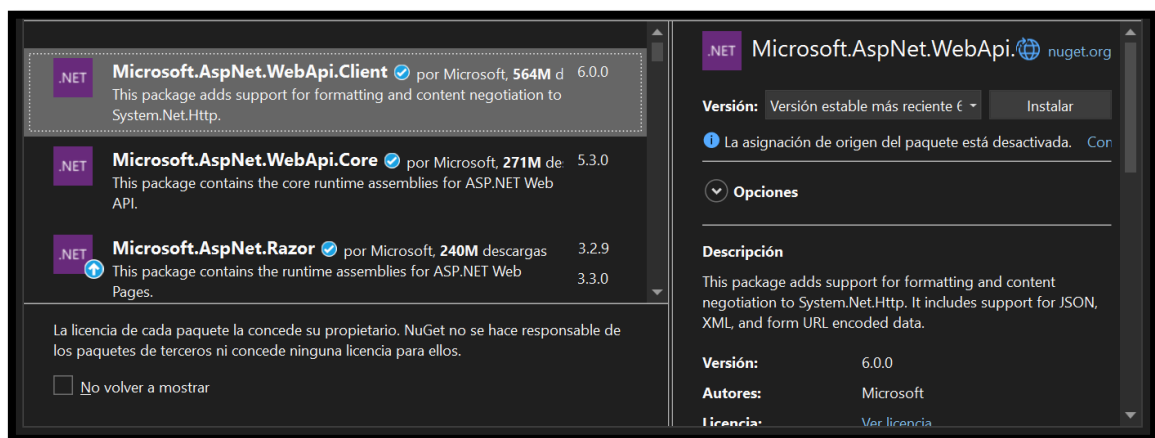
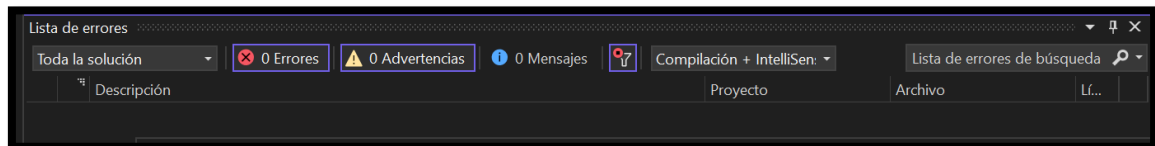
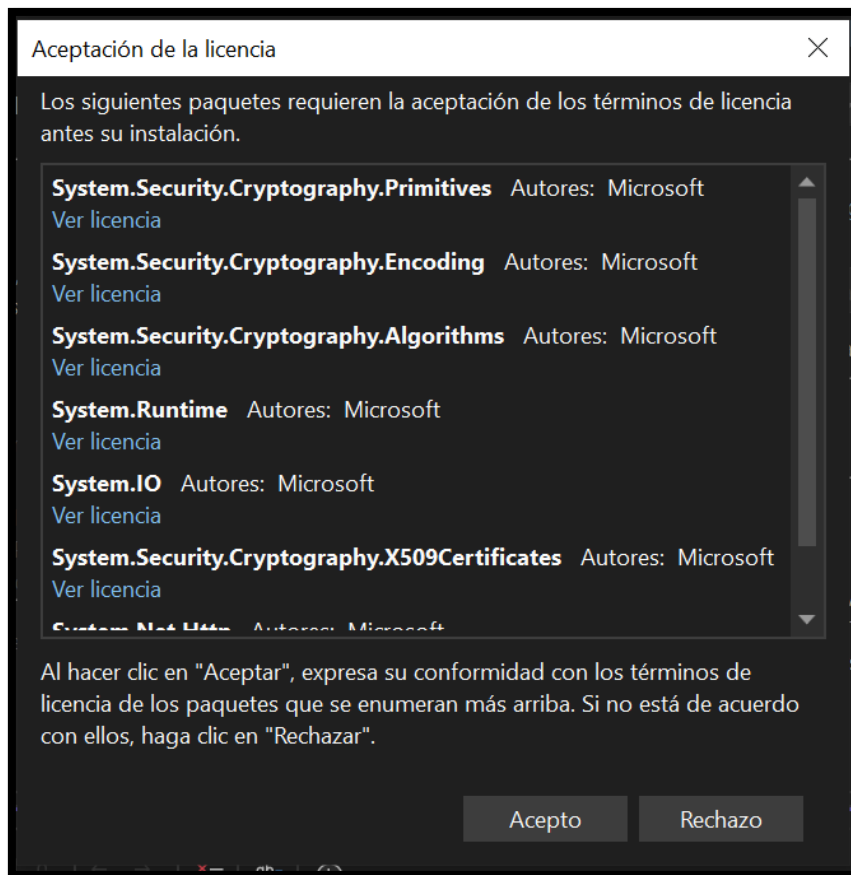
**Administrar paquetes NuGet...**

**Administrar bibliotecas del lado cliente...**



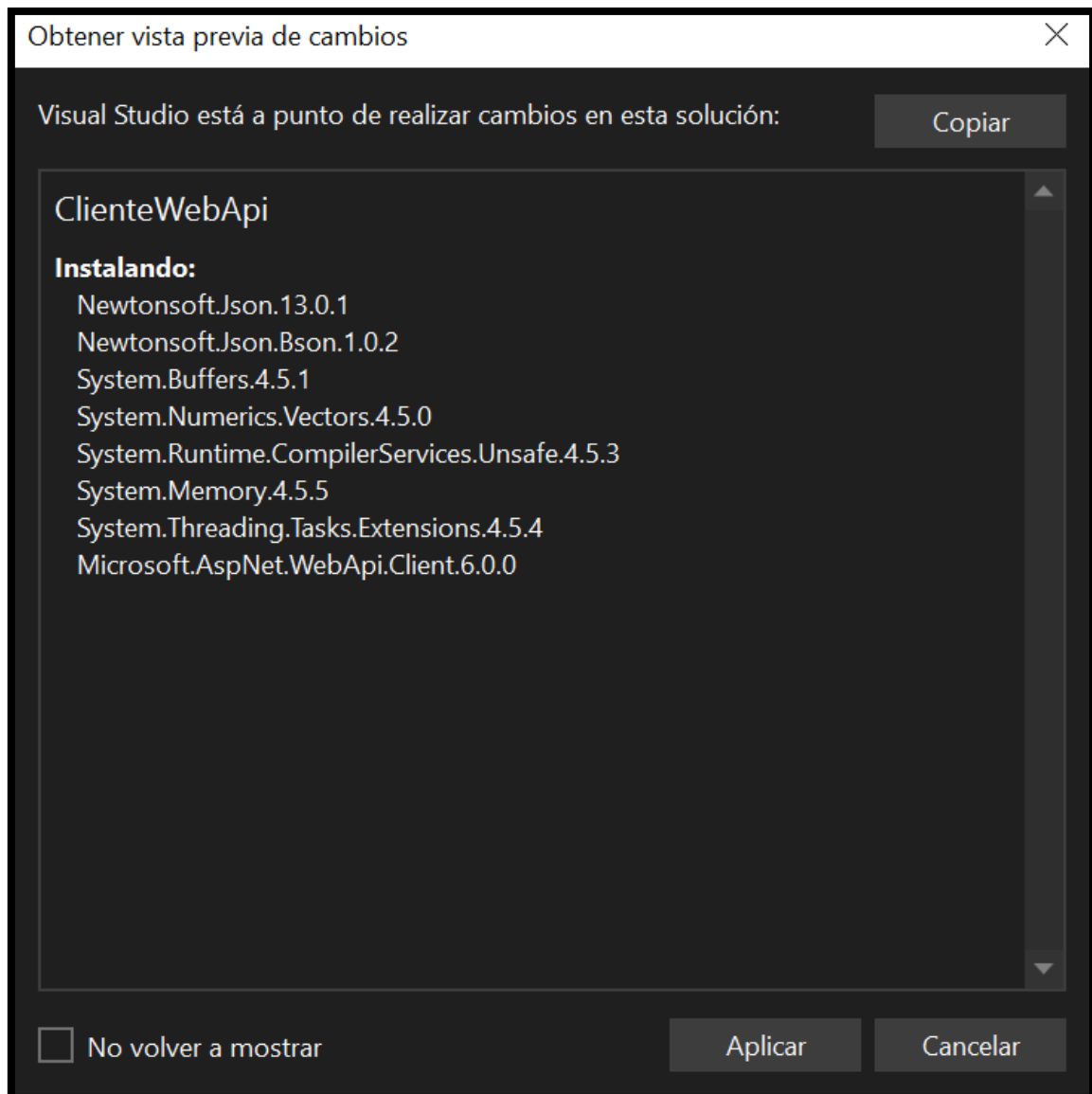
UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS





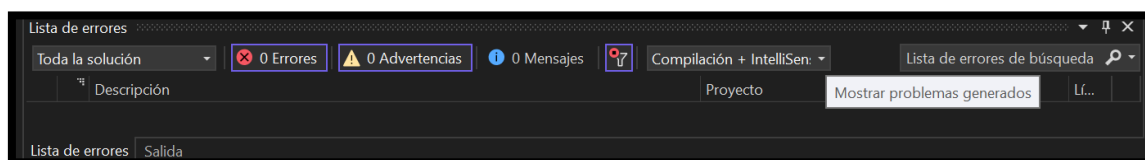
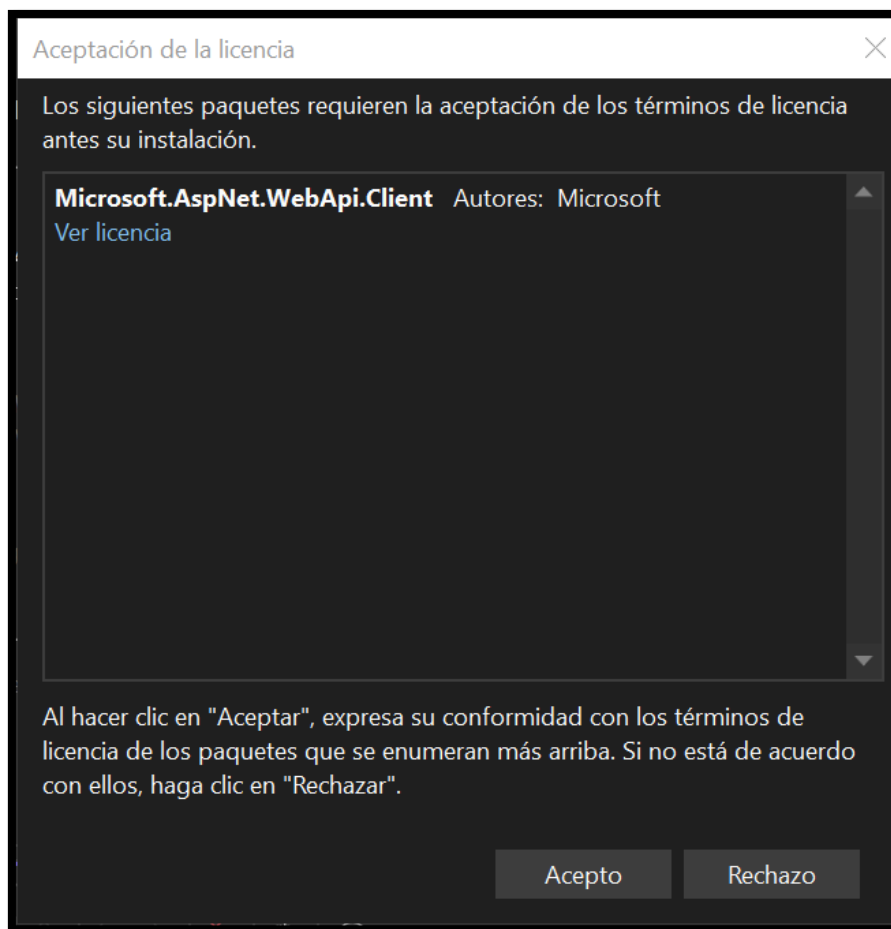


UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



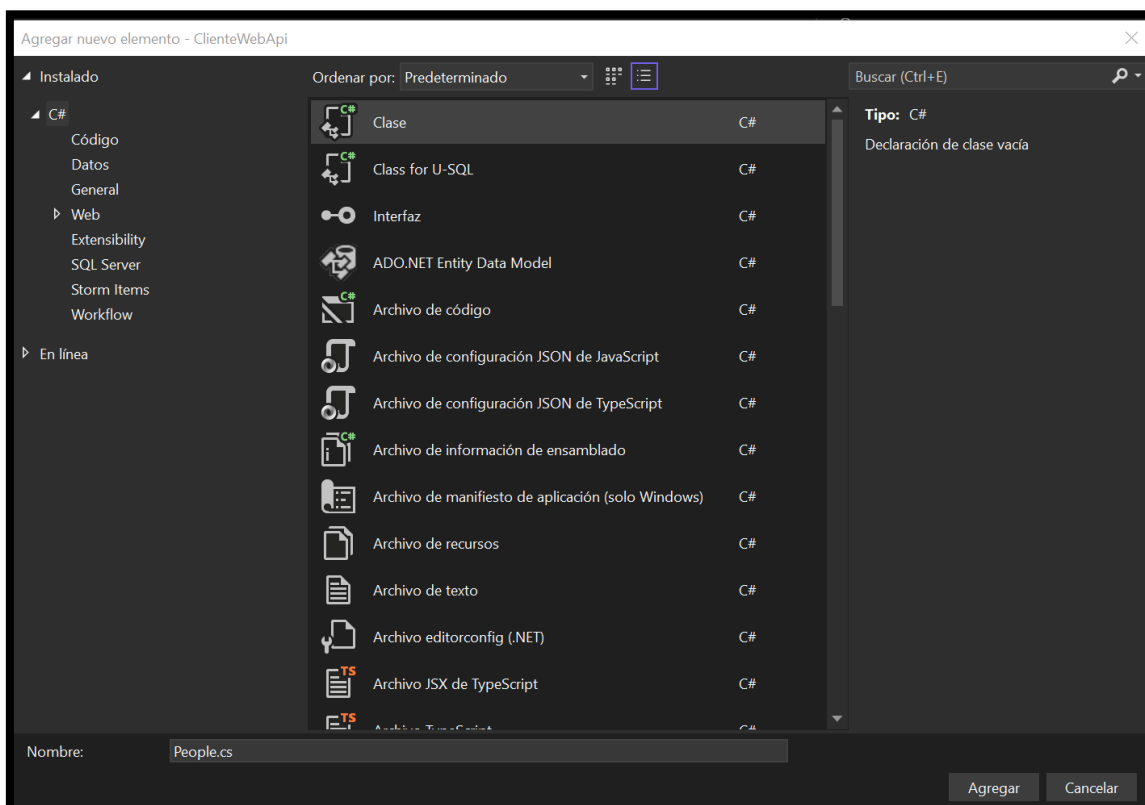
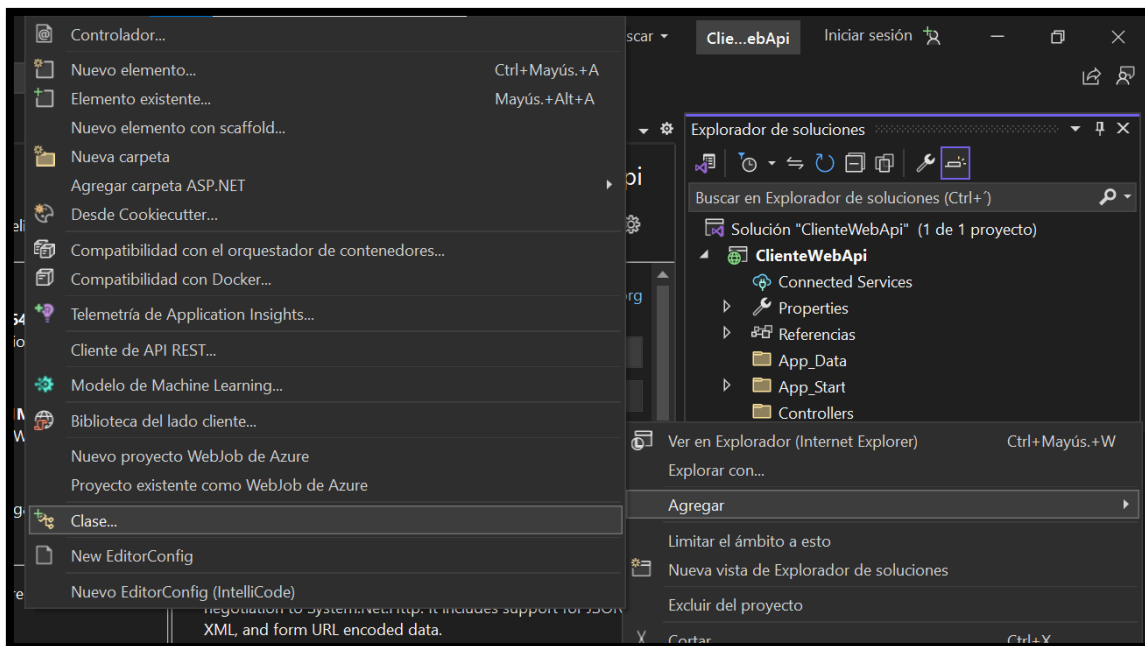


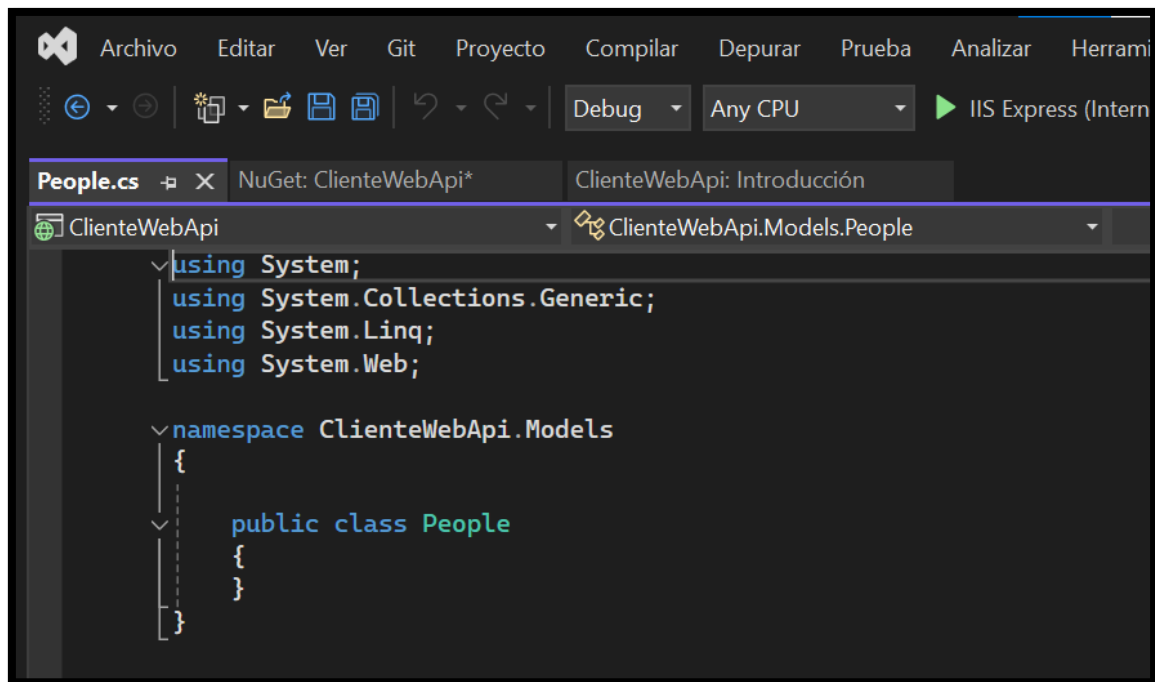
UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS





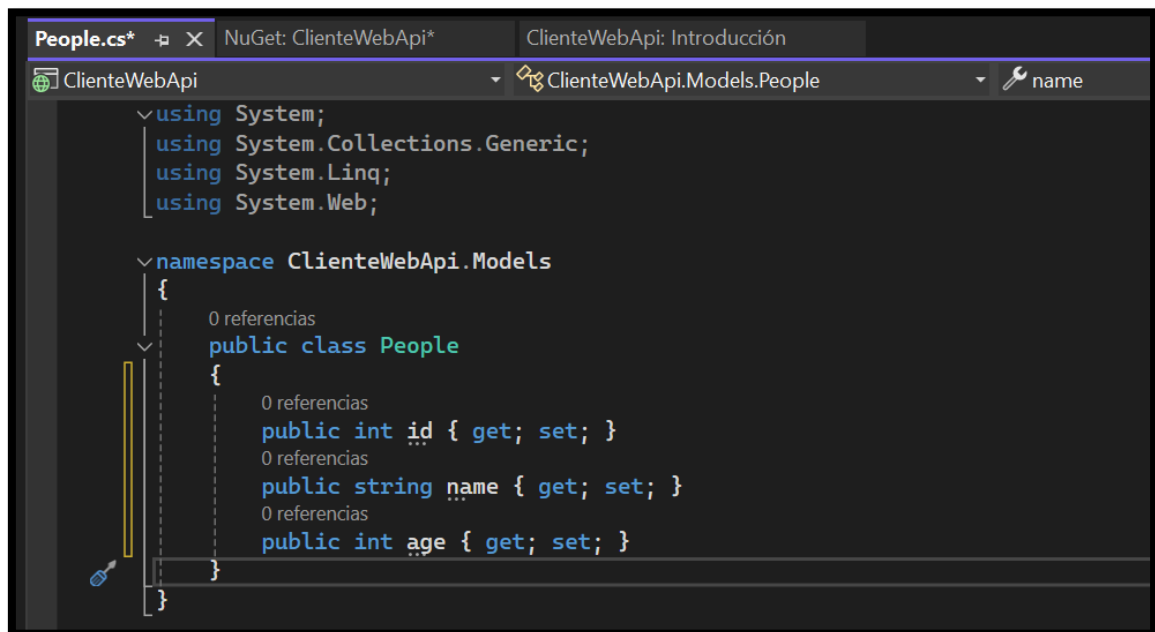
UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS





```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

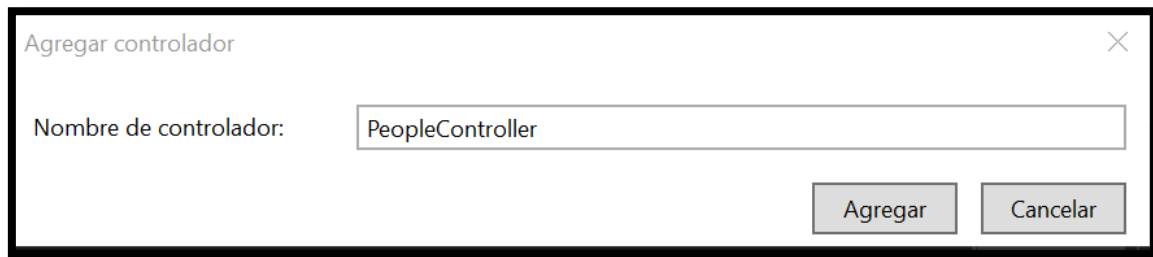
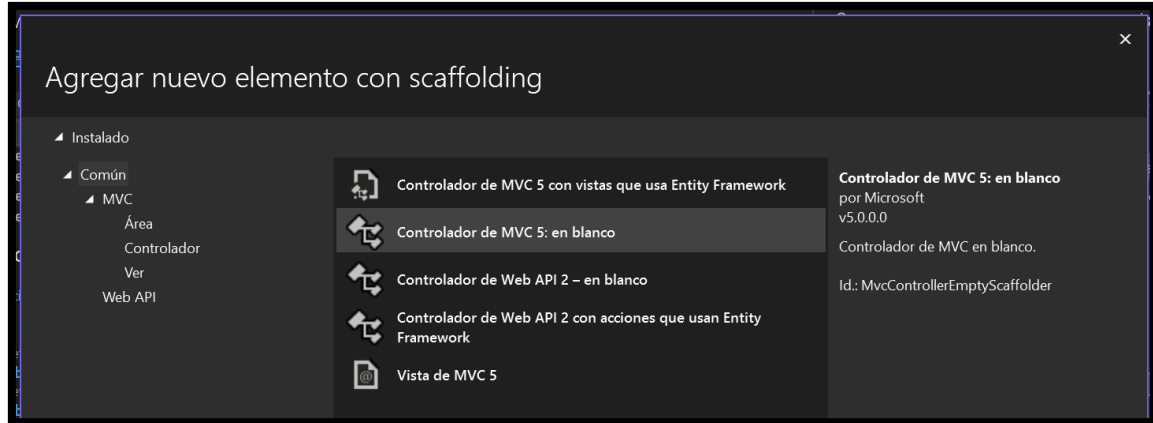
namespace ClienteWebApi.Models
{
    public class People
    {
    }
}
```



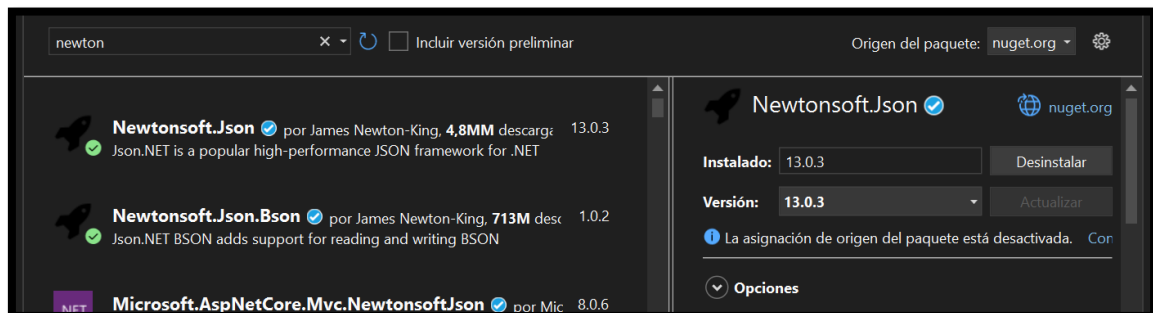
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace ClienteWebApi.Models
{
    0 referencias
    public class People
    {
        0 referencias
        public int id { get; set; }
        0 referencias
        public string name { get; set; }
        0 referencias
        public int age { get; set; }
    }
}
```

Agregar un controlador

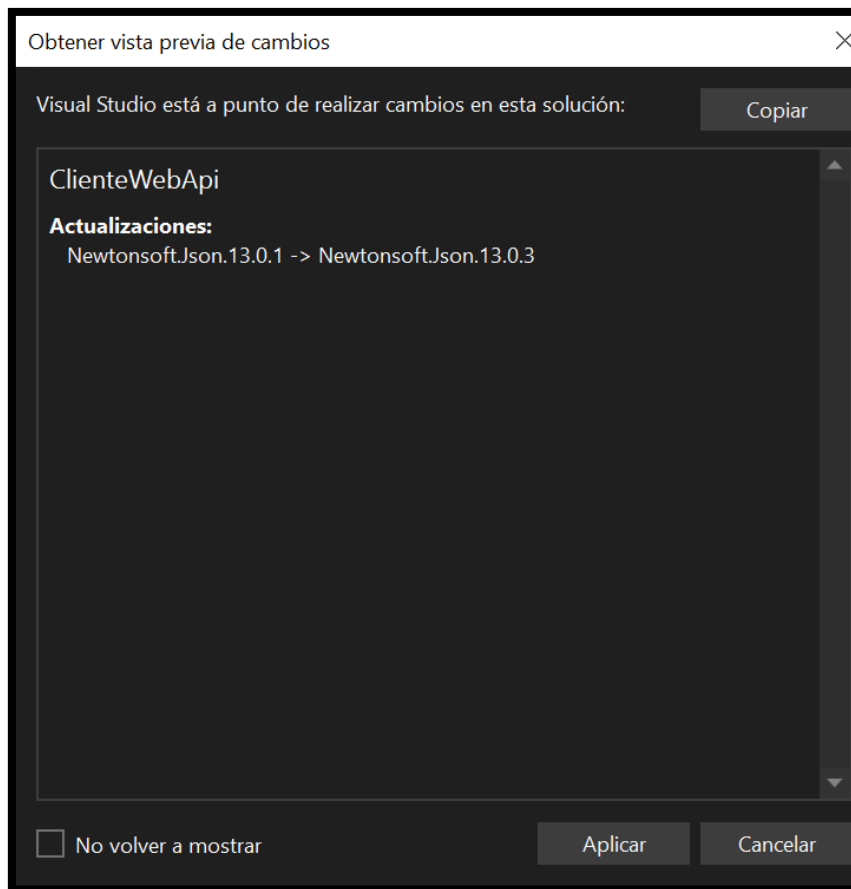


Agregar o actualizar paquete json

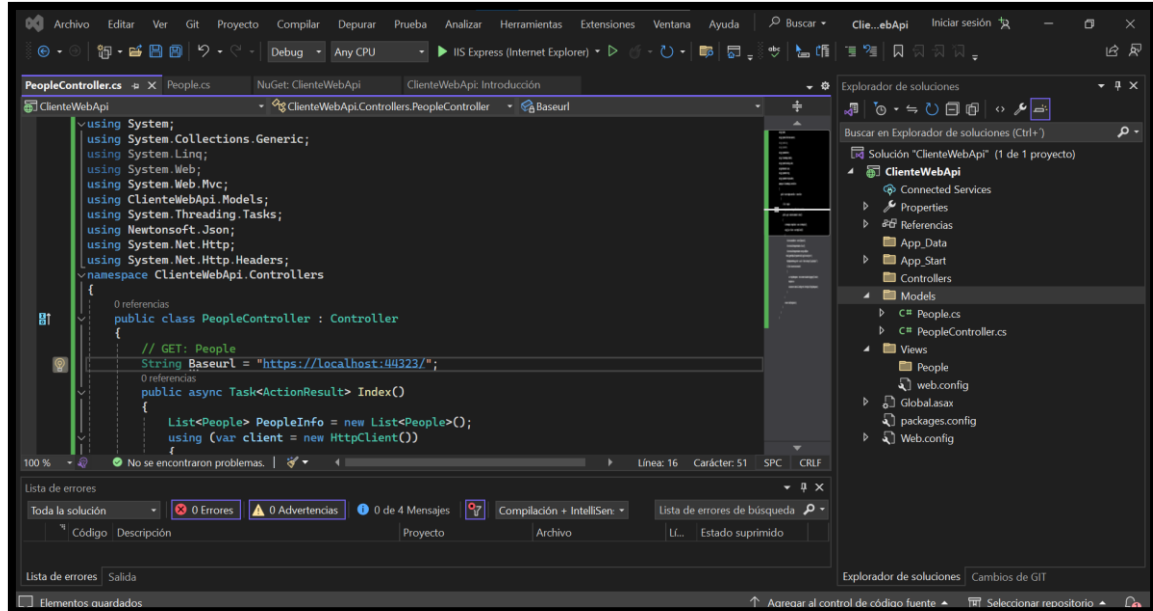




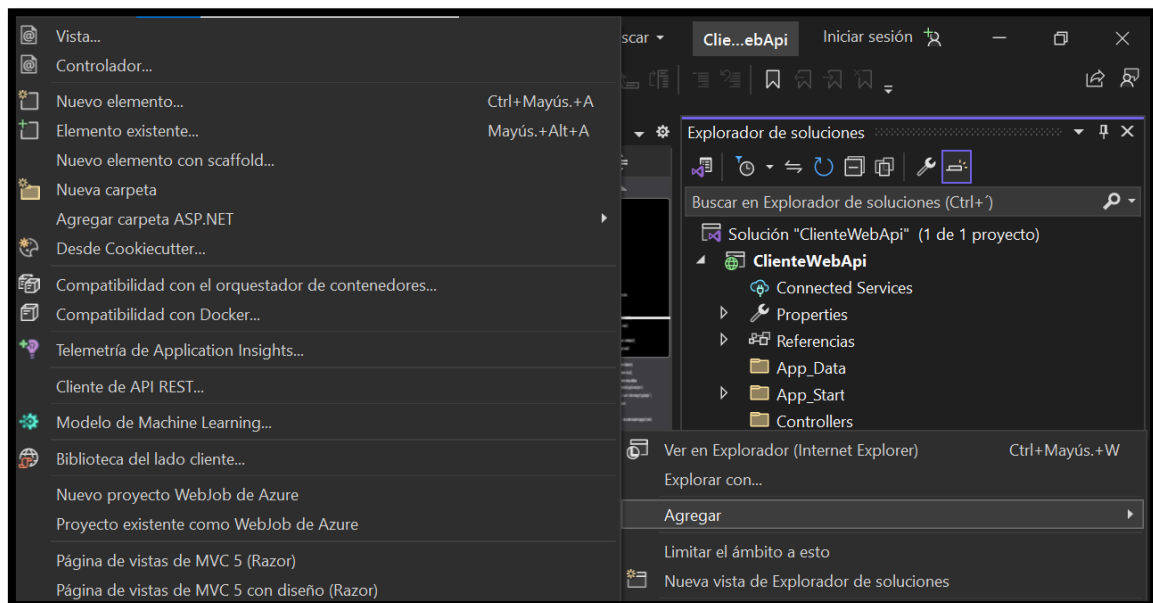
UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Dentro del controlador de la tabla añadimos las librerías adecuadas y el código siguiente.



Añadir una vista





UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS

Agregar vista

Nombre de vista:

Plantilla:

Clase de modelo:

Opciones:

☐ Crear como vista parcial

☒ Hacer referencia a bibliotecas de scripts

☒ Usar página de diseño:

...

(Dejar en blanco si se define en un archivo _viewstart de Razor)

```
RouteConfig.cs | index.cshtml | PeopleController.cs | People.cs | NuGet: ClienteWebApi | ClienteWebApi: Introducción
ClienteWebApi | ClienteWebApi.RouteConfig | RegisterRoutes(RouteCollection routes)
using System.Web;
using System.Web.Mvc;
using System.Web.Routing;

namespace ClienteWebApi
{
    1 referencia
    public class RouteConfig
    {
        1 referencia
        public static void RegisterRoutes(RouteCollection routes)
        {
            routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

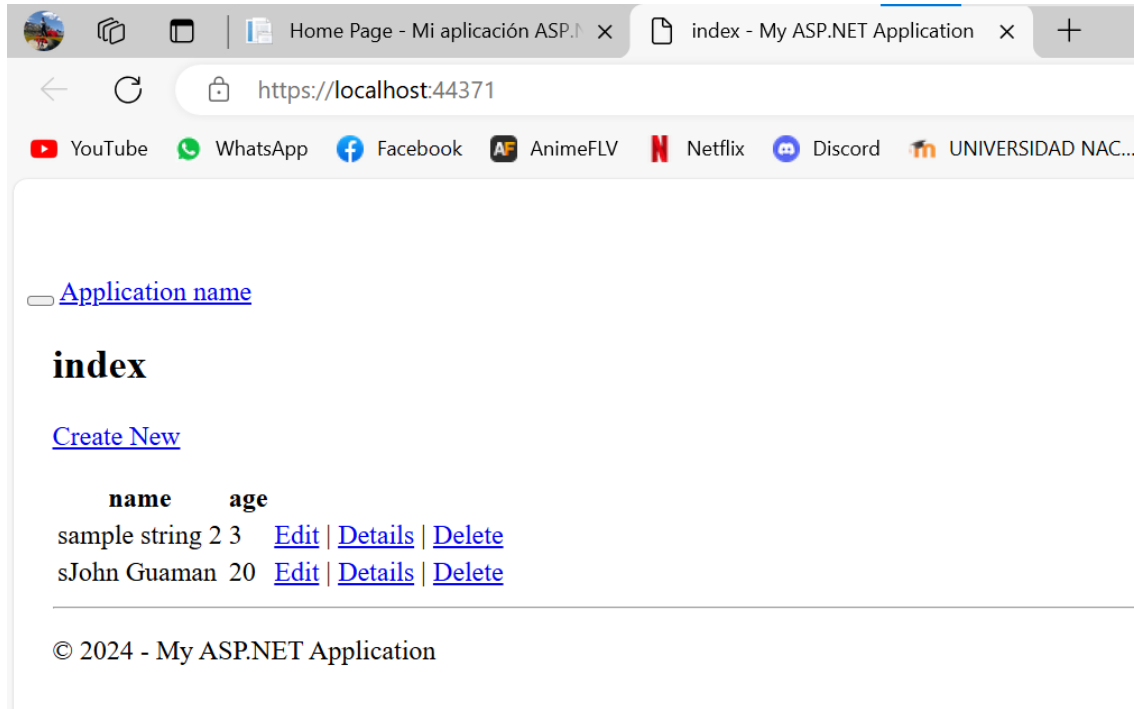
            routes.MapRoute(
                name: "Default",
                url: "{controller}/{action}/{id}",
                defaults: new { controller = "People", action = "Index", id = UrlParameter.Optional
            );
        }
    }
}
```



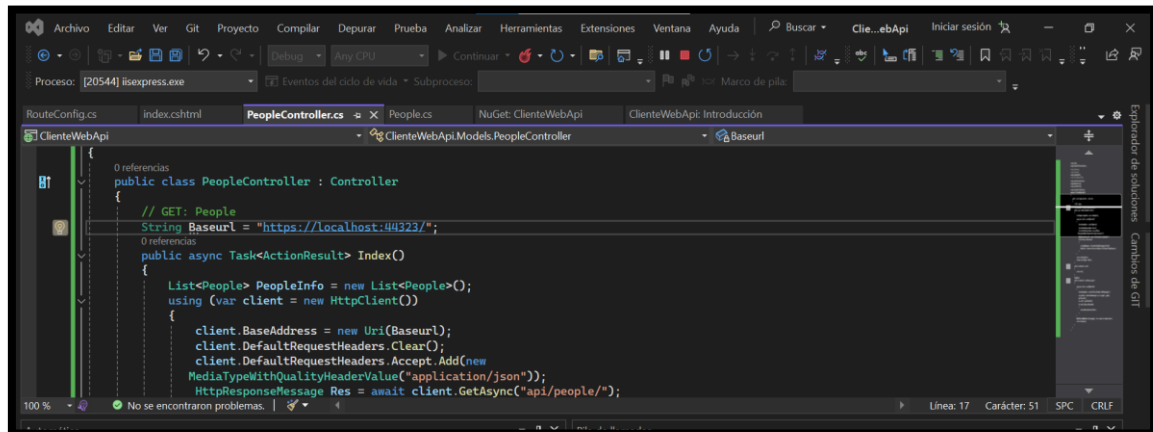
UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Probar la aplicación

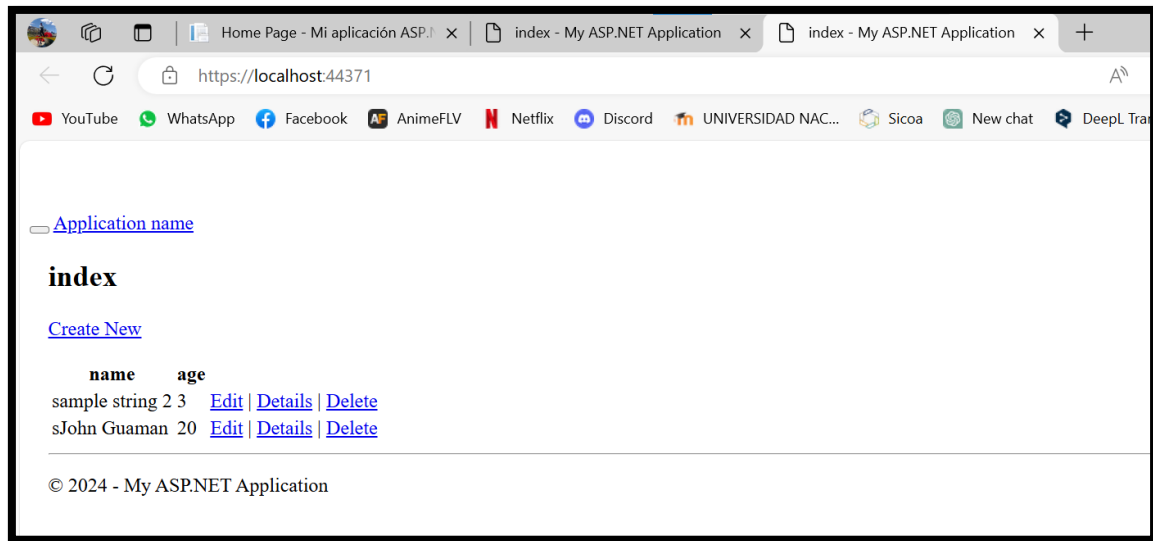


Modificar el controlador para añadir el créate.





UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Añadir la vista.

Agregar vista

Nombre de vista:

Create

Plantilla:

Create

Clase de modelo:

People (ClienteWebApi.Models)

Opciones:

☐ Crear como vista parcial

☒ Hacer referencia a bibliotecas de scripts

☒ Usar página de diseño:

...

(Dejar en blanco si se define en un archivo _viewstart de Razor)

Agregar

Cancelar



<localhost:44323/api/people>

[Application name](#)

Create

People

id

name

age

Create

[Back to List](#)

© 2024 - My ASP.NET Application

[Application name](#)

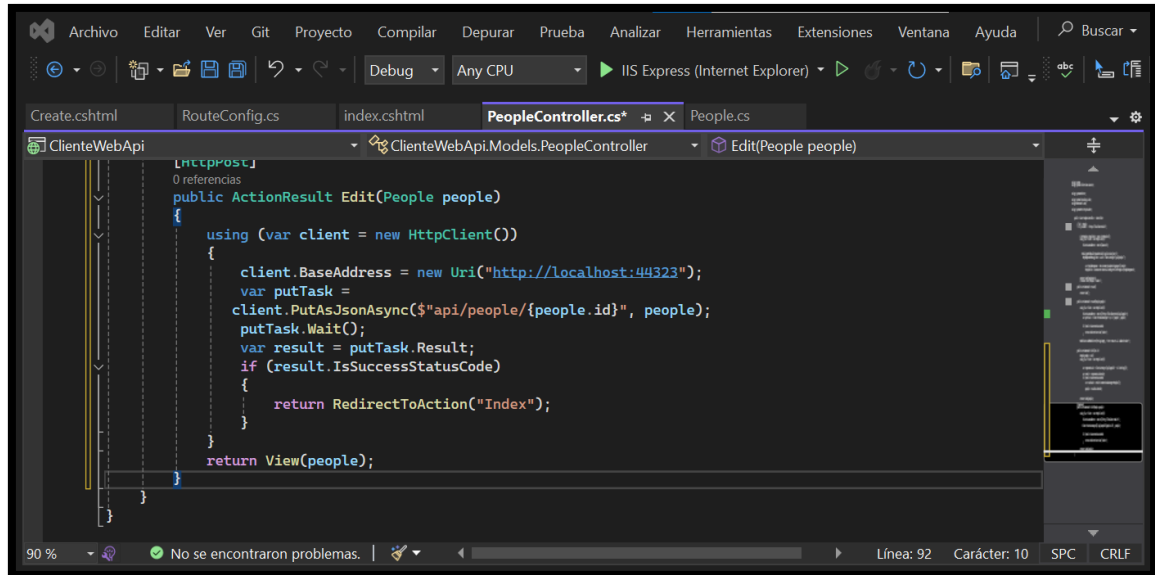
index

[Create New](#)

name	age	
sample string 2	3	Edit Details Delete
sJohn Guaman	20	Edit Details Delete
Guaman	20	Edit Details Delete

© 2024 - My ASP.NET Application

Agregar el controlador



```
[HttpPost]
public ActionResult Edit(People people)
{
    using (var client = new HttpClient())
    {
        client.BaseAddress = new Uri("http://localhost:44323");
        var putTask =
            client.PutAsJsonAsync($"api/people/{people.id}", people);
        putTask.Wait();
        var result = putTask.Result;
        if (result.IsSuccessStatusCode)
        {
            return RedirectToAction("Index");
        }
    }
    return View(people);
}
```

Añadir la vista

Agregar vista

Nombre de vista:

Edit

Plantilla:

Edit

Clase de modelo:

People (ClienteWebApi.Models)

Opciones:

☐ Crear como vista parcial

☒ Hacer referencia a bibliotecas de scripts

☒ Usar página de diseño:

...

(Dejar en blanco si se define en un archivo _viewstart de Razor)

Agregar

Cancelar



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



```
Archivo  Editar  Ver  Git  Proyecto  Compilar  Depurar  Prueba  Analizar  Herramientas  Extensiones  Ventana  Ayuda  Busca
Debug  Any CPU  Continuar  Eventos del ciclo de vida  Subproceso:  Marco de pila:
Proceso: [23796] iisexpress.exe
Edit.cshtml  Create.cshtml  RouteConfig.cs  index.cshtml  PeopleController.cs*  People.cs
@Html.EditorFor(model => model.name, new
{
    htmlAttributes = new
    {
        @class =
        "form-control"
    }
})
@Html.ValidationMessageFor(model => model.name, "", new { @class = "textdanger" })
</div>
</div>
<div class="form-group">
    @Html.LabelFor(model => model.age, htmlAttributes: new { @class = "control-label colmd-2" })
    <div class="col-md-10">
        @Html.EditorFor(model => model.age, new { htmlAttributes = new { @class = "formcontrol" } })
        @Html.ValidationMessageFor(model => model.age, "", new
        {
            @class = "text-danger"
        })
    </div>
</div>
```

☐ [Application name](#)

Edit

People

id

1

name

Prueba2

age

3

Save

[Back to List](#)

© 2024 - My ASP.NET Application

☐ [Application name](#)

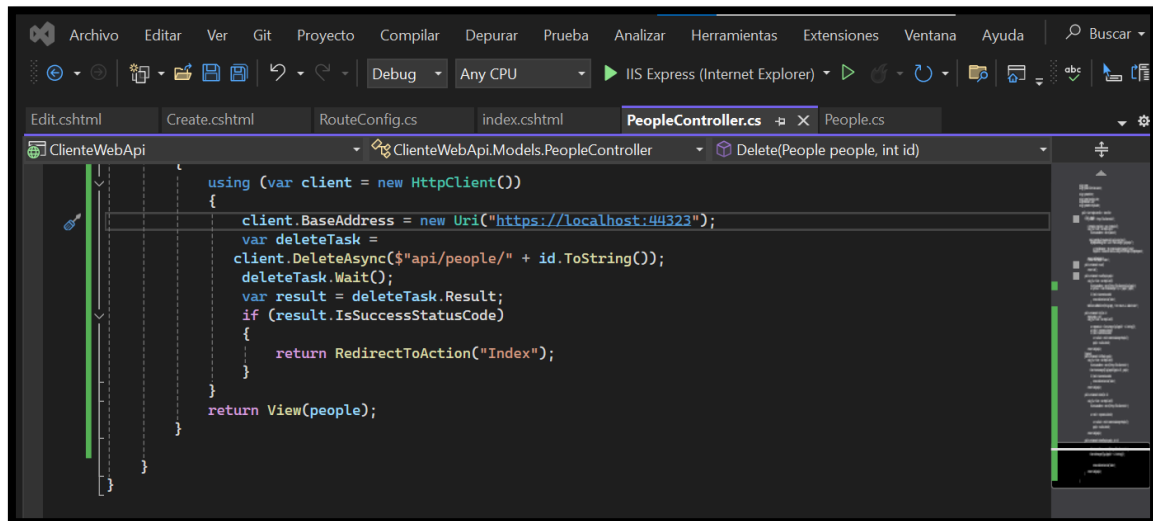
index

[Create New](#)

name	age	
Prueba2	3	Edit Details Delete
sJohn Guaman	20	Edit Details Delete
Guaman	20	Edit Details Delete
Prueba1	1	Edit Details Delete

© 2024 - My ASP.NET Application

Agregar un controlador para la eliminación



```
using (var client = new HttpClient())
{
    client.BaseAddress = new Uri("https://localhost:44323");
    var deleteTask =
    client.DeleteAsync($"api/people/" + id.ToString());
    deleteTask.Wait();
    var result = deleteTask.Result;
    if (result.IsSuccessStatusCode)
    {
        return RedirectToAction("Index");
    }
}

return View(people);
```



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Añadir la vista

Agregar vista

Nombre de vista:

Delete

Plantilla:

Delete

Clase de modelo:

People (ClienteWebApi.Models)

Opciones:

☐ Crear como vista parcial

☒ Hacer referencia a bibliotecas de scripts

☒ Usar página de diseño:

...

(Dejar en blanco si se define en un archivo _viewstart de Razor)

Agregar

Cancelar

Archivo

Editar

Ver

Git

Proyecto

Compilar

Depurar

Prueba

Analizar

Herramientas

Extensiones

Ventana

Ayuda

Buscar

Debug

Any CPU

IIS Express (Internet Explorer)

Delete.cshtml

Edit.cshtml

Create.cshtml

RouteConfig.cs

index.cshtml

PeopleController.cs

People.cs

</dt>

<@Html.DisplayNameFor(model => model.id)

</dt>

<dd>

<@Html.DisplayFor(model => model.id)

</dd>

<dt>

<@Html.DisplayNameFor(model => model.name)

</dt>

<dd>

<@Html.DisplayFor(model => model.name)

</dd>

<dt>

<@Html.DisplayNameFor(model => model.age)

</dt>

<dd>

<@Html.DisplayFor(model => model.age)

</dd>

</dl>

<@using (Html.BeginForm())

90 %

No se encontraron problemas.

Línea: 19

Carácter: 9

SPC

CRLF



☐ [Application name](#)

Delete

Are you sure you want to delete this?

People

id	1
name	Prueba2
age	3

| [Back to List](#)

© 2024 - My ASP.NET Application



[Application name](#)

index

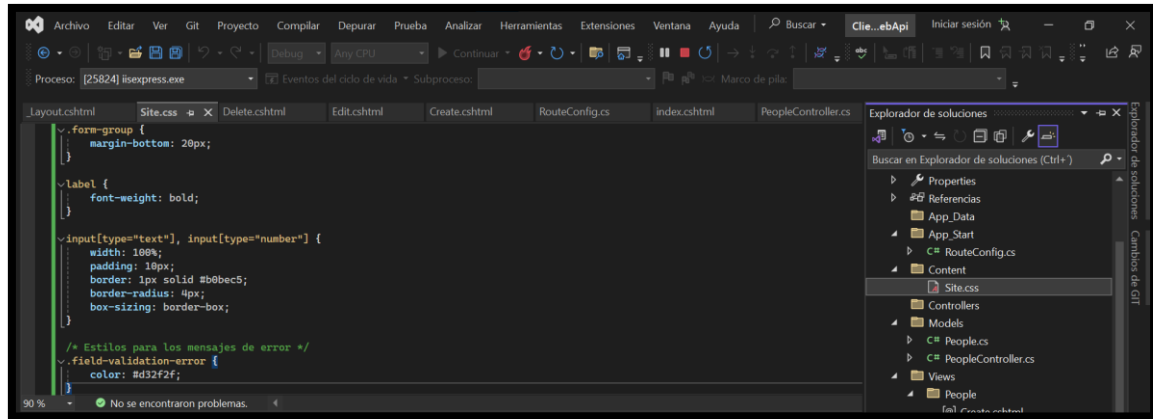
[Create New](#)

name	age	
sJohn Guaman	20	Edit Details Delete
Guaman	20	Edit Details Delete
Prueba1	1	Edit Details Delete

© 2024 - My ASP.NET Application

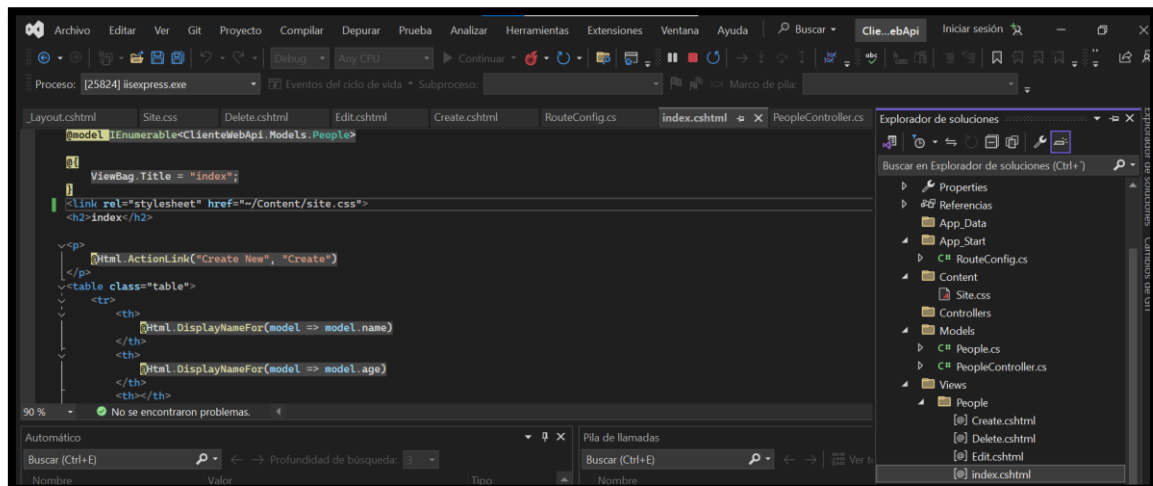
Por último, agregar estilos

Se implementa el css en site.css



```
form-group {  
  margin-bottom: 20px;  
}  
  
label {  
  font-weight: bold;  
}  
  
input[type="text"], input[type="number"] {  
  width: 100%;  
  padding: 10px;  
  border: 1px solid #b0bec5;  
  border-radius: 4px;  
  box-sizing: border-box;  
}  
  
/* Estilos para los mensajes de error */  
.field-validation-error {  
  color: #d32f2f;  
}
```

Y se llama al css en el index



```
@model IEnumerable<ClienteWebApi.Models.People>  
  
<@>ViewBag.Title = "index";  
<link rel="stylesheet" href="~/Content/site.css">  
<h2>index</h2>  
  
<p>@Html.ActionLink("Create New", "Create")</p>  
<table class="table">  
  <tr>  
    <th>  
      @Html.DisplayNameFor(model => model.name)  
    </th>  
    <th>  
      @Html.DisplayNameFor(model => model.age)  
    </th>  
  </tr>  
</table>
```




UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Resultados

Autor: John Guaman

index

[Create New](#)

name	age	
sJohn Guaman	20	Edit Details Delete
Prueba1	1	Edit Details Delete
Prueba4	19	Edit Details Delete

© 2024 - My ASP.NET Application



Conclusiones

En este informe, hemos explorado la implementación de un sistema web basado en la arquitectura MVC (Modelo-Vista-Controlador) que realiza operaciones CRUD (Crear, Leer, Actualizar y Eliminar). A través del análisis detallado de cada componente de MVC, se ha demostrado cómo este patrón de diseño promueve la separación de responsabilidades, lo que resulta en un código más organizado y mantenible. La creación de un prototipo funcional ha permitido ilustrar la interacción entre el cliente y el servidor, utilizando tecnologías web modernas como Node.js, Express.js y frameworks del lado del cliente como React. Este enfoque no solo facilita la gestión eficiente de datos, sino que también mejora la experiencia del usuario al proporcionar una interfaz interactiva y reactiva. En resumen, la implementación de MVC junto con operaciones CRUD se ha mostrado como una solución robusta y escalable para el desarrollo de aplicaciones web modernas.

Recomendación

Para futuros desarrollos y mejoras en sistemas web basados en la arquitectura MVC y operaciones CRUD, se recomienda adoptar metodologías ágiles de desarrollo. Estas metodologías, como Scrum o Kanban, permiten una mayor flexibilidad y capacidad de respuesta a los cambios, asegurando que el desarrollo se mantenga alineado con las necesidades del usuario final y los requisitos del negocio. Además, se sugiere invertir en pruebas automatizadas y en la integración continua para detectar y corregir errores de manera temprana, lo que incrementa la confiabilidad y la calidad del software. La capacitación continua en nuevas tecnologías y prácticas de desarrollo también es esencial para mantenerse actualizado y aprovechar las últimas innovaciones en el campo del desarrollo web, garantizando así la competitividad y la relevancia de las aplicaciones desarrolladas.



Referencias

Hernandez, U. (2015, 22 febrero). *MVC (Model, View, Controller) explicado*.

Codigofacilito. Recuperado 11 de junio de 2024, de

<https://codigofacilito.com/articulos/mvc-model-view-controller-explicado>

De TechTarget, C. (2021, 23 abril). *MySQL*. ComputerWeekly.es.

<https://www.computerweekly.com/es/definicion/MySQL>

Sydle. (2024, 6 febrero). *¿Qué es API? Ejemplos, ventajas y tipos*. Blog SYDLE.

<https://www.sydle.com/es/blog/api-6214f68876950e47761c40e7>

Anandmeg. (2023, 31 octubre). *¿Qué es el IDE de Visual Studio?* Microsoft Learn.

<https://learn.microsoft.com/es-es/visualstudio/get-started/visual-studio-ide?view=vs-2022>



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN TECNOLOGIAS DE LA INFORMACION
CUARTO SEMESTRE – INTEROPERABILIDAD DE PLATAFORMAS



Anexos: