FORMATO DE GUÍAS DE LAS PRÁCTICAS/LABORATORIO Y CENTROS DE SIMULACIÓN[1]

ASIGNATURA: Integración de sistemas NÚMERO DE LA PRÁCTICA: 5

NOMBRE ESTUDIANTE: Naobe Ovando

TEMA DE LA PRÁCTICA: REST - 2.

OBJETIVO DE LA PRÁCTICA: Implementar un servicio REST con .NET en C# que tenga acceso a base de datos, aplicando MVC, listar los datos en un formulario Web.

MARCO TEÓRICO:

- **MVC**.
- **REST**.
- **Métodos HTTP**.

RECURSOS, MATERIAL Y EQUIPO: Computador con el entorno integrado de desarrollo Visual Studio 2022, todas las dependencias necesarias para desarrollar aplicaciones ASP web utilizando el Framework .NET. SQL-Server, Navegador. SOAPUI o Postman.

ENUNCIADO, INSTRUCCIONES, ACTIVIDADES POR DESARROLLAR Y/O REGISTRO DE DATOS:

1. Los servicios y aplicaciones creadas deben incluir en la parte superior los siguientes comentarios:

// NOMBRE APELLIDOS:
// PARALELO:
// SI – INTEGRACIÓN DE SISTEMAS
// FECHA:
// PRÁCTICA No. #

2. Todos los servicios y aplicaciones deben estar debidamente comentados de manera que se pueda entender el código fuente.

[1] El Formato de Guías de las Prácticas corresponde al contenido de las guías o pautas que se seguirán durante el desarrollo de las prácticas de laboratorio.

EJERCICIO:

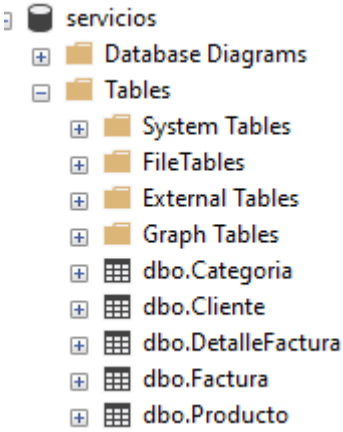
Utilizando el Visual Studio 2022 desarrollar una aplicación con el lenguaje C# para crear un servicio REST que acceda a tablas en una base de datos, crear el MVC, realizar las operaciones correspondientes y probar los resultados con una herramienta apropiada.

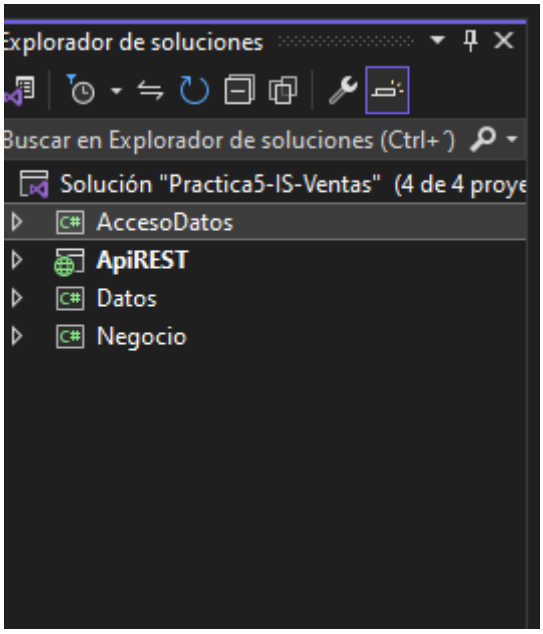
INSTRUCCIONES:

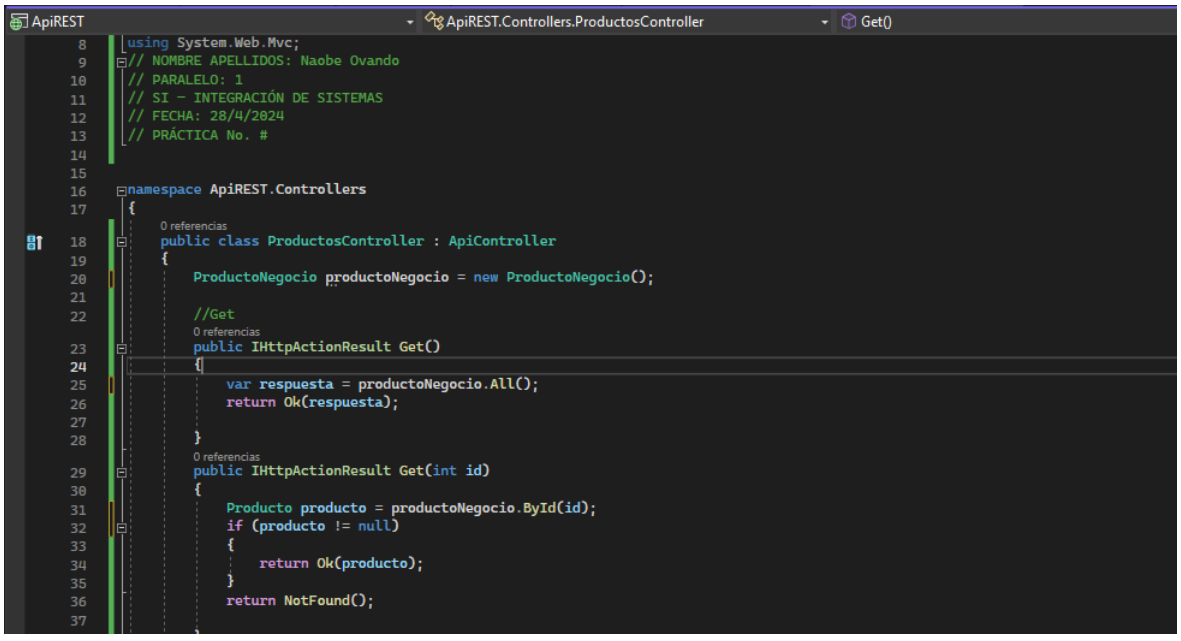
- Crear un proyecto (Aplicación Web ASP.NET (.NET Framework))
- Asignar como plantilla vacía.
- Crear un formulario Web para Listar los datos de una de las tablas.
- Implementar los métodos correspondientes para realizar la Serialización de los datos enviados por el servidor y mostrar los datos en un GridView.
- Crear un nuevo formulario Web para Insertar un registro de una tabla.
- Implementar los métodos correspondientes para realizar la deserialización de los datos enviados al servidor para ejecutar la inserción del registro.
- Verificar los datos utilizando el servidor SQL o con SOAPUI o Postman.

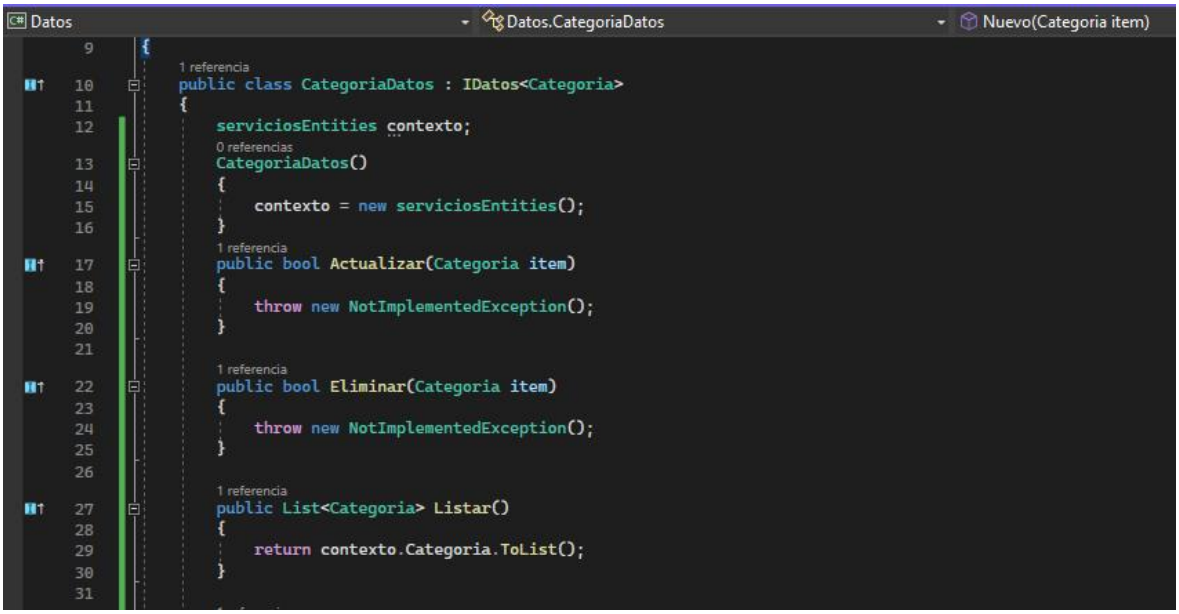
```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE PROCEDURE SP_Clientes

AS
BEGIN

    SELECT *from Cliente where Cliente.idCategoria=1
END
GO
```

```
<connectionStrings>
  <add name="serviciosEntities" connectionString="metadata=res://*/ModelVentas.csdl|res://*/ModelVentas.ssdl|res://*/ModelVentas.msl;provider=System.Data.SqlClient;provider
</connectionStrings>
```

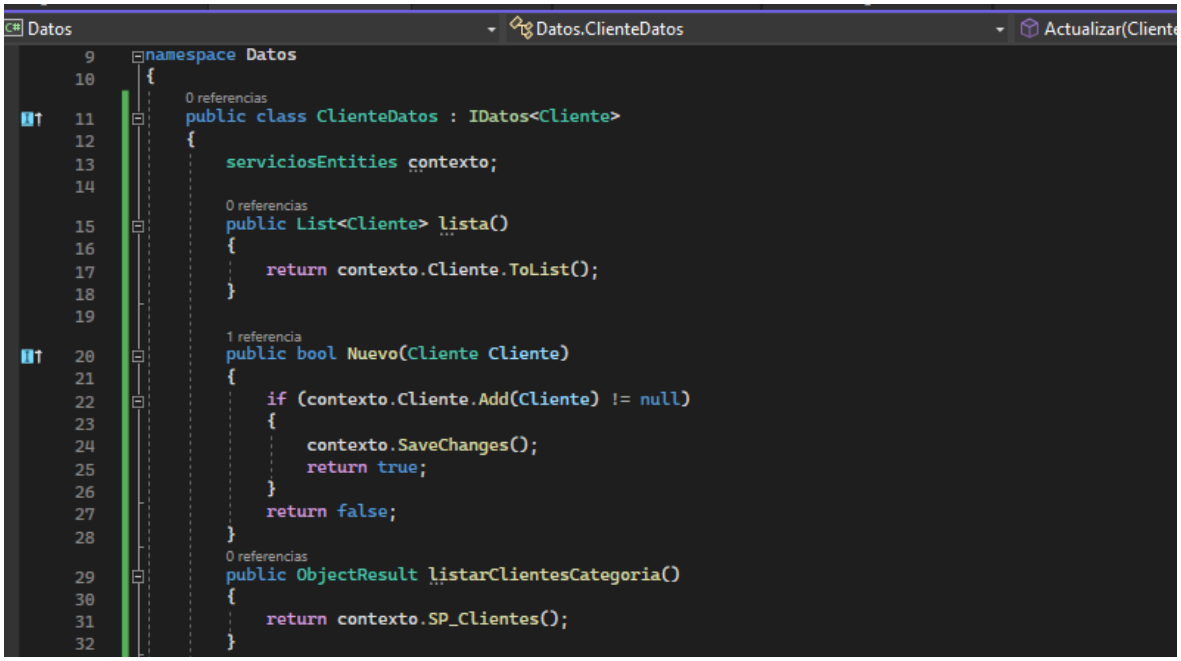



Explorador de soluciones
Buscar en Explorador de soluciones (Ctrl+;)
Solución "Practica5-IS-Ventas" (4 de 4 proye
AccesoDatos
ApiREST
Datos
Negocio


ApiREST
ApiREST.Controllers.ProductosController
Get()
using System.Web.Mvc;
// NOMBRE APELLIDOS: Naobe Ovando
// PARALELO: 1
// SI – INTEGRACIÓN DE SISTEMAS
// FECHA: 28/4/2024
// PRÁCTICA No. #
namespace ApiREST.Controllers
{
0 referencias
public class ProductosController : ApiController
{
ProductoNegocio productoNegocio = new ProductoNegocio();
//Get
0 referencias
public IHttpActionResult Get()
{
var respuesta = productoNegocio.All();
return Ok(respuesta);
}
0 referencias
public IHttpActionResult Get(int id)
{
Producto producto = productoNegocio.ById(id);
if (producto != null)
{
return Ok(producto);
}
return NotFound();



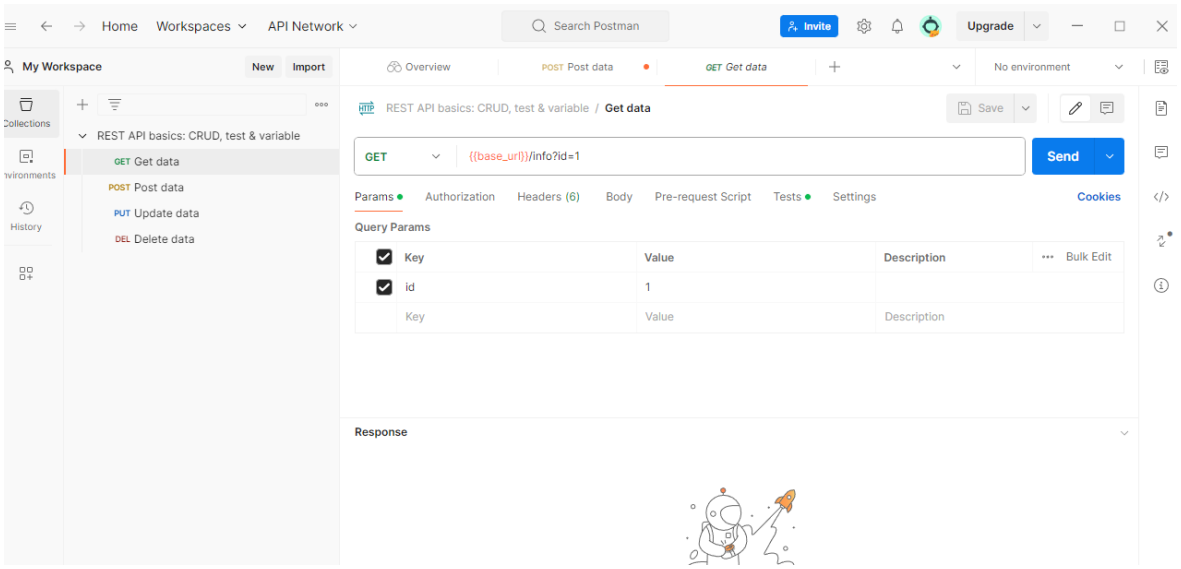

```
{
    1 referencia
    public class CategoriaDatos : IDatos<Categoria>
    {
        serviciosEntities contexto;
        0 referencias
        CategoriaDatos()
        {
            contexto = new serviciosEntities();
        }
        1 referencia
        public bool Actualizar(Categoria item)
        {
            throw new NotImplementedException();
        }

        1 referencia
        public bool Eliminar(Categoria item)
        {
            throw new NotImplementedException();
        }

        1 referencia
        public List<Categoria> Listar()
        {
            return contexto.Categoria.ToList();
        }
```




```
namespace Datos
{
    0 referencias
    public class ClienteDatos : IDatos<Cliente>
    {
        serviciosEntities contexto;

        0 referencias
        public List<Cliente> lista()
        {
            return contexto.Cliente.ToList();
        }

        1 referencia
        public bool Nuevo(Cliente Cliente)
        {
            if (contexto.Cliente.Add(Cliente) != null)
            {
                contexto.SaveChanges();
                return true;
            }
            return false;
        }
        0 referencias
        public ObjectResult listarClientesCategoria()
        {
            return contexto.SP_Clientes();
        }
```

```
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6
7   namespace Datos
8   {
        3 referencias
9       internal interface IDatos <T>
10      {
            3 referencias
11          List<T> Listar();
            3 referencias
12          bool Actualizar(T item);
            3 referencias
13          bool Nuevo(T item);
            3 referencias
14          bool Eliminar(T item);
15      }
16  }
17
```

```
Datos                    Datos.ProductoDatos          Eliminar(Producto item)
10  namespace Datos
11  {
        3 referencias
12      public class ProductoDatos :IDatos<Producto>
13      {
14          serviciosEntities contexto;
            1 referencia
15          public ProductoDatos()
16          {
17              contexto = new serviciosEntities();
18          }
19
            1 referencia
20          public List<Producto> lista()
21          {
22              return contexto.Producto.ToList();
23          }
24
            3 referencias
25          public bool Nuevo(Producto producto)
26          {
27              if (contexto.Producto.Add(producto)!=null)
28                  {
29                  contexto.SaveChanges();
30                  return true;
31              }
32              return false;
```



CUESTIONARIO:

1. Explique con sus palabras el proceso de serialización.
   La serialización es el proceso de convertir un objeto o estructura de datos en un formato que pueda ser almacenado o transmitido, generalmente en forma de bytes. Esto es útil cuando se necesita guardar el estado de un objeto o enviarlo a través de una red.
2. Explique con sus palabras el proceso de deserialización.
   La deserialización es el proceso inverso de la serialización. Consiste en tomar los datos serializados (por ejemplo, en forma de bytes) y convertirlos de nuevo en un objeto o estructura de datos en la memoria del programa. Esto es esencial para recuperar la información que ha sido previamente serializada.
3. ¿Qué características importantes tiene POSTMAN?
   POSTMAN es una herramienta de desarrollo de API muy útil y versátil que tiene varias características importantes:
   Interfaz de usuario amigable: POSTMAN proporciona una interfaz gráfica fácil de usar que permite a los desarrolladores crear, probar y depurar solicitudes HTTP y API de manera eficiente.
   Colecciones y entornos: Permite organizar solicitudes en colecciones, lo que facilita la gestión y ejecución de conjuntos de solicitudes relacionadas. Los entornos permiten la configuración

dinámica de variables para adaptarse a diferentes escenarios de desarrollo (por ejemplo, entornos de prueba, producción, etc.).

Automatización y scripting: POSTMAN permite automatizar tareas mediante la creación de scripts en JavaScript. Esto es útil para realizar pruebas automatizadas, gestionar flujos de trabajo complejos y realizar tareas repetitivas.

Monitoreo de API: Ofrece capacidades de monitoreo para supervisar el rendimiento de las API, detectar problemas y obtener métricas útiles para el análisis y la optimización.

Colaboración y compartición: Facilita la colaboración entre equipos al permitir compartir colecciones, entornos y resultados de pruebas de manera sencilla.

Soporte multiplataforma: Está disponible en varias plataformas, incluyendo Windows, macOS y Linux, lo que lo hace accesible para una amplia gama de desarrolladores.

CONCLUSIONES Y RECOMENDACIONES:

1. La serialización y deserialización son procesos fundamentales en el desarrollo de software para el manejo eficiente de datos, almacenamiento y comunicación entre sistemas.

2. POSTMAN es una herramienta esencial para los desarrolladores de API, que ofrece una amplia gama de funcionalidades para facilitar el desarrollo, pruebas y monitorización de servicios web.

3. El uso adecuado de POSTMAN puede mejorar la productividad, la calidad y la colaboración en equipos de desarrollo de software.

Recomendaciones

1. Familiarízate con los conceptos de serialización y deserialización para comprender mejor el manejo de datos en tus aplicaciones.

2. Utiliza POSTMAN para probar y depurar tus API de forma eficiente, aprovechando sus capacidades de automatización y scripting.

3. Comparte colecciones y entornos de POSTMAN con tu equipo para mejorar la colaboración y la eficacia en el desarrollo de software.

MATRIZ DE EVALUACIÓN DE CONOCIMIENTOS Y DESTREZAS DEL ESTUDIANTE EN LA PRÁCTICA (RÚBRICA)

	CRITERIOS DE EVALUACIÓN DE LOS PARÁMETROS			
Criterios	**Excelente**	**Muy bueno**	**Bueno**	**Bajo**
Desempeño en el laboratorio (toma de datos, realización de cálculos, realización de programa, obtención de resultados, obtención de un producto, aplicación de una herramienta, realización de un procedimiento para experimento, etc.)	El estudiante desarrolla el 100% los programas de acuerdo a lo que se solicita, el código permite que el programa se entienda, el programa crea los resultados requeridos (4 puntos)	El estudiante desarrolla el 60 % de los programas de acuerdo a lo solicitado, no obtiene todos los resultados requeridos (2 puntos)	El estudiante desarrolla menos del 60% de los programas de acuerdo a lo solicitado, no obtiene todos los resultados requeridos. (1 puntos)	El estudiante no desarrolla lo que la práctica solicita. O no entrega la práctica (0 puntos)
Cuestionario	El estudiante contesta el cuestionario completo de forma correcta (3 puntos)	El estudiante contesta la mitad del cuestionario de forma correcta (2 puntos)	El estudiante contesta una pregunta del cuestionario de forma correcta (1 puntos)	El estudiante no contesta el cuestionario (0 punto)
Conclusiones y Recomendaciones	Todas las conclusiones son adecuadas, objetivas y aplicables. Las recomendaciones tienen coherencia y se	Las conclusiones son parcialmente adecuadas, objetivas y aplicables Las recomendaciones se relacionan con el tema de la	Las conclusiones son simples, no son objetivas y no se pueden aplicar. Las recomendaciones no tienen coherencia o no se	No existen conclusiones ni recomendaciones

	relacionan con el tema de la práctica (3 puntos)	práctica, pero no son coherentes. (2 puntos)	relacionan con el tema de la práctica. (1 puntos)	 (0 punto)

Fecha de elaboración del documento:		2020-08-20			
Elaborado por:	Patricio Coba M.	Revisado por:	Jorge Alarcón	Aprobado por:	Henry Roa
Cargo:	Docente	Cargo:	Coordinador Área	Cargo:	Coordinador Carrera
Firma:		Firma:		Firma:	

Nota: El archivo de las prácticas deberá guardarse con el siguiente formato:

Práctica_Nro01_PE_paralelo_NombreApellido