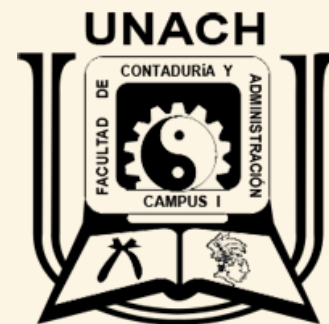




Universidad Autónoma de Chiapas  
Fac. de Cont. y Admón. C-I  
Lic. Sistemas Computacionales



**DOCENTE:** Mtro. Rigoberto Pérez Ovando

**ASIGNATURA:** Servicios Web

**INTEGRANTES:** Acosta Cambrano José Felipe

Arias Domínguez Eliel.

Gómez Domínguez Baldomero

**SEMESTRE Y GRUPO:** 9no. Semestre

**FECHA:** 22 – Octubre - 2021

**Tuxtla Gutiérrez, Chiapas, México.**



## **Caso Práctico 4 - Creando una aplicación Cliente-Servidor**

### **Manual para crear un Service Web**

---

#### **Web Service**

Es un sistema software diseñado para soportar la interoperabilidad máquina – máquina a través de una red. Este tiene una interfaz descrita en un formato que puede ser procesado por una máquina (específicamente WSDL).

Un Web Service es una comunicación por medio de mensajes SOAP entre diferentes equipos a través de una red.

#### **SOAP – Simple Object Access Protocol**

Es un protocolo de comunicación, el cual permite la comunicación entre aplicaciones a través de mensajes por medio de Internet. Es independiente de la plataforma, y del lenguaje. Esta basado en XML y es la base principal de los Web Services. Los mensajes SOAP son documento XML propiamente dicho, pero esto lo veremos más adelante cuando veamos un ejemplo de un mensaje SOAP.

#### **NuSOAP**

Es un kit de herramientas (ToolKit) para desarrollar Web Services bajo el lenguaje PHP. Está compuesto por una serie de clases que nos harán mucho más fácil el desarrollo de Web Services. Provee soporte para el desarrollo de clientes (aquellos que consumen los Web Services) y de servidores (aquellos que los proveen). NuSOAP está basado en SOAP 1.1, WSDL 1.1 y HTTP 1.0/1.1

Link de repositorio PARTE SERVIDOR:

<https://github.com/ElielAr/ServiceWeb-Practica4>

Link de Web Service en Hosting

<http://app-9c76bc9f-d6b8-4221-83fd-9e8cd702204f.cleverapps.io/>

## Como crear un Web Service

### Parte Servidor

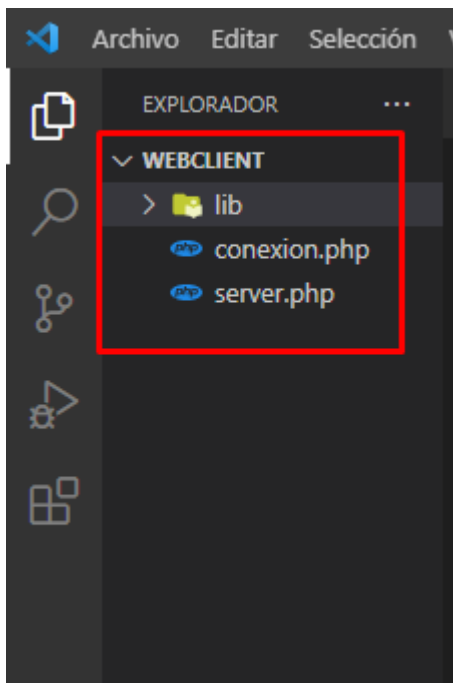
#### 1.- Instalación de NuSOAP

La instalación es bastante sencilla, solo basta ir a la página en sourceforge de NuSOAP y descargar el archivo comprimido:

<https://sourceforge.net/projects/nusoap/>



2.- En nuestro Editor de código favorito creamos la siguiente estructura para nuestro proyecto

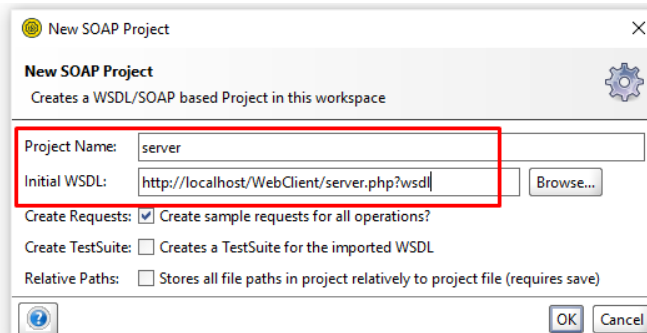


3.- Una vez que tenemos la librería de NUSOAP en nuestro directorio ya podemos empezar a programar nuestro Web Service con PHP. Para ello en el archivo **server.php** con el siguiente código:

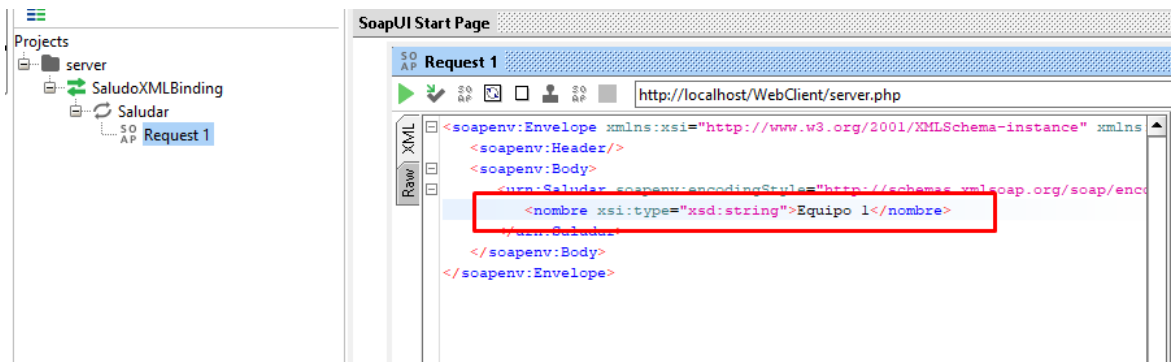
```
server.php > ...
1  <?php
2
3  // incluyendo la librería de nusopa
4  require_once('lib/nussoap.php');
5
6  // Configurando el web service
7  $server = new soap_server();
8  $server->configureWSDL("SaludoXML", "urn:SaludoXMLwsdl");
9  $server->wsdl->schemaTargetNamespace = "urn:SaludoXMLwsdl";
10
11 // Nuestra función de ejemplo
12 function Saludar($nombre) {
13     return 'Hola Mundo' . trim($nombre);
14 }
15
16 // Registramos nuestra función Saludar con su parámetro nombre
17 $server->register(
18     'Saludar', // Nombre del método
19     array('nombre' => 'xsd:string'), // Parámetros de entrada
20     array('return' => 'xsd:string'), // Parámetros de salida
21     'urn:SaludoXMLwsdl', // Nombre del workspace
22     'urn:SaludoXMLwsdl#Saludar', // Acción soap
23     'rpc', // Estilo
24     'encoded', // Uso
25     'Saluda a la persona' // Documentación
26 );
27
28 $HTTP_RAW_POST_DATA = isset($GLOBALS['HTTP_RAW_POST_DATA']) ? $GLOBALS['HTTP_RAW_POST_DATA'] : '';
29
30 $server->service($HTTP_RAW_POST_DATA);
31
```

4.- Para comprobar lo anterior usamos SoapUI (es una herramienta, desarrollada en java, para la realización de pruebas a aplicaciones con arquitectura orientada a servicio y transferencia de estado representacional.)

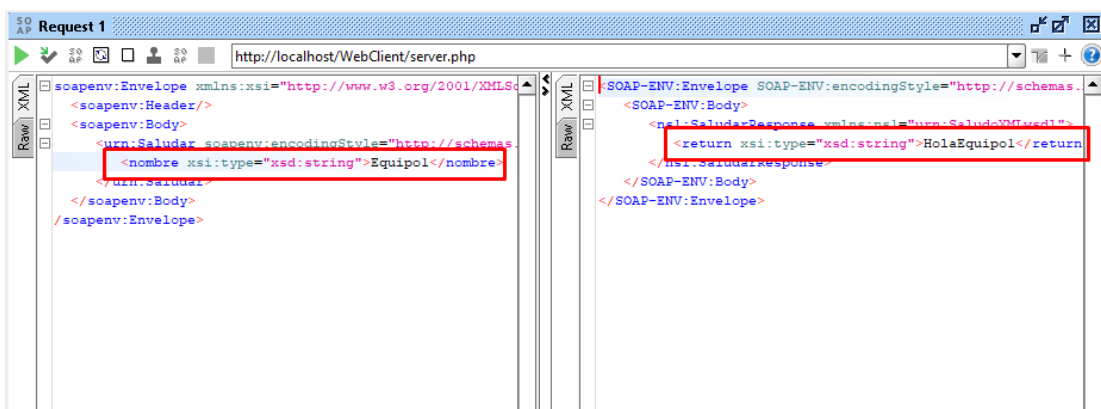
Abrimos nuestro SoapUI y ponemos la dirección de nuestro servidor local



una vez ingresado la dirección, nos dirigimos para realizar la prueba:



de entrada, como prueba no solicitará un nombre de regreso nos devolverá un saludo con el nombre que ingresamos.



**5.- Crear la base de datos, dentro de PhpMyAdmin dentro de la sentencia mysql ponemos el siguiente código para crear nuestra base de datos**

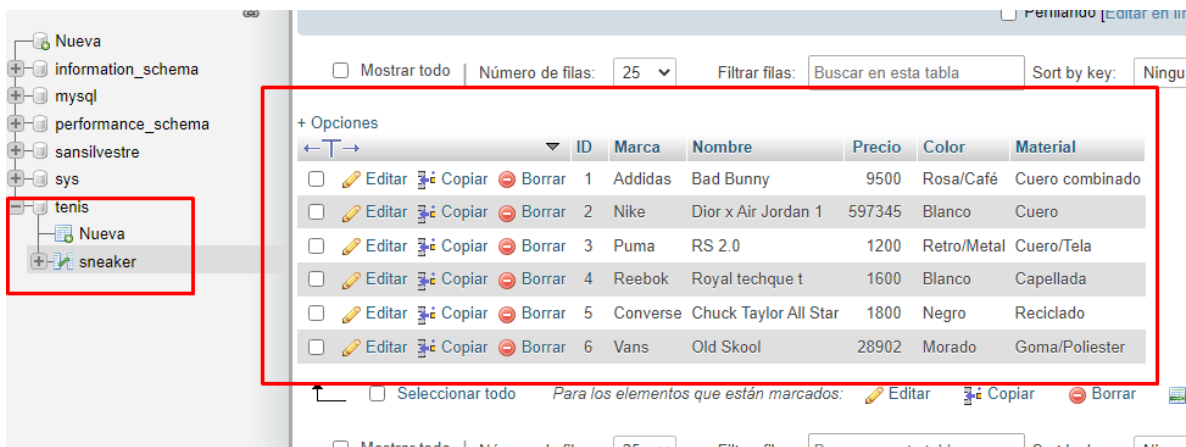
Código para crear nuestra base de datos y estructura de nuestra tabla con los valores siguientes

```
1 CREATE DATABASE IF NOT EXISTS `tenis` DEFAULT CHARACTER SET utf8 COLLATE utf8_spanish2_ci;
2 USE `tenis`;
3
4 DROP TABLE IF EXISTS `sneaker`;
5 CREATE TABLE IF NOT EXISTS `sneaker` (
6   `ID` int NOT NULL AUTO_INCREMENT,
7   `Marca` varchar(20) COLLATE utf8_spanish2_ci NOT NULL,
8   `Nombre` varchar(50) COLLATE utf8_spanish2_ci NOT NULL,
9   `Precio` int NOT NULL,
10  `Color` varchar(20) COLLATE utf8_spanish2_ci NOT NULL,
11  `Material` varchar(25) COLLATE utf8_spanish2_ci NOT NULL,
12  PRIMARY KEY (`ID`)
```

Código para el volcado de nuestra tabla creada

```
15
16 INSERT INTO `sneaker` (`ID`, `Marca`, `Nombre`, `Precio`, `Color`, `Material`) VALUES
17 (1, 'Addidas', 'Bad Bunny', 9500, 'Rosa/Café', 'Cuero combinado'),
18 (2, 'Nike', 'Dior x Air Jordan 1', 597345, 'Blanco', 'Cuero'),
19 (3, 'Puma', 'RS 2.0', 1200, 'Retro/Metal', 'Cuero/Tela'),
20 (4, 'Reebok', 'Royal techque t', 1600, 'Blanco', 'Capellada'),
21 (5, 'Converse', 'Chuck Taylor All Star', 1800, 'Negro', 'Reciclado'),
22 (6, 'Vans', 'Old Skool', 28902, 'Morado', 'Goma/Poliester');
```

Comprobamos que se halla creado la base de datos.



The screenshot shows a database management interface. On the left, a tree view shows the database structure with 'tenis' selected. The main area displays the 'sneaker' table with the following data:

ID	Marca	Nombre	Precio	Color	Material
1	Addidas	Bad Bunny	9500	Rosa/Café	Cuero combinado
2	Nike	Dior x Air Jordan 1	597345	Blanco	Cuero
3	Puma	RS 2.0	1200	Retro/Metal	Cuero/Tela
4	Reebok	Royal techque t	1600	Blanco	Capellada
5	Converse	Chuck Taylor All Star	1800	Negro	Reciclado
6	Vans	Old Skool	28902	Morado	Goma/Poliester

## 5.- Dentro de nuestra estructura de nuestro proyecto en el archivo conexión.php

```
conexion.php > ...
1  <?php
2
3  //Creamos la clase conexion para nuestro service web
4  class Conexion extends PDO{
5      private $tipo = 'mysql';
6      private $host = 'localhost'; //nombre del servidor
7      private $nombredb = 'tenis'; // nombre de nuestra base de datos
8      private $usuario = 'root'; //nombre de nuestro usuario
9      private $contra = ''; //contrase;a
10
11     public function __construct(){
12         try {
13             parent::__construct("{$this->tipo}:dbname={$this->nombredb};host={$this->host};charset=utf8", $this->usuario, $this->contra);
14         } catch (PDOException $e) {
15             echo 'Existe un error: ' . $e->getMessage();
16         }
17     }
18 }
19
```

**6.- para nuestro web service nos pide que realice lo siguiente: permita Insertar, modificar, eliminar y consultar los datos de la tabla que hemos creado.**

Ahora empezaremos con la construcción del servicio, para ello hemos subido al servidor el fichero con la librería nuSOAP y lo hemos descomprimido en un directorio llamado lib y realizamos la instancion de la conexión.

```
require_once 'conexion.php';  
require_once 'lib/nusoap.php';
```

Definimos el método, el cual consistirá en **inserciones** a la tabla creada, usando los parámetros de entrada Marca, Nombre, Precio, Color, Material y devolver un Id se indica la inserción del registro a la tabla.

```
//Metodo insertar  
function insertSneaker($marca,$nombre,$precio,$color,$material){  
    try{  
        $conexion= new Conexion();  
        $consulta=$conexion->prepare("INSERT INTO sneaker (Marca,Nombre,Precio,Color,Material)  
VALUES(:marca, :nombre, :precio, :color, :material)");  
        $consulta -> bindParam(":marca", $marca, PDO::PARAM_STR);  
        $consulta -> bindParam(":nombre", $nombre, PDO::PARAM_STR);  
        $consulta -> bindParam(":precio", $precio, PDO::PARAM_INT);  
        $consulta -> bindParam(":color", $color, PDO::PARAM_STR);  
        $consulta -> bindParam(":material", $material, PDO::PARAM_STR);  
        $consulta -> execute();  
        $ultimoID = $conexion->lastInsertId();  
        return join ("",array($ultimoID));  
    }catch (Exception $e){  
        return join ("",array(false));  
    }  
}
```

El siguiente paso consiste en la creación del servidor, para ello comenzamos incluyendo la librería, creando una instancia de la clase soap\_server.

```
$server = new nusoap_server();  
$conexion= new Conexion();
```

Activamos el método, indicándole el nombre, los parámetros de entrada y salida, la acción SOAP, el estilo, el uso y una descripción.

```
78 //Registro de INSERTAR del servicio SOAP
79 $server = new nusoap_server();
80 $server->configureWSDL("server", "urn:server");
81 $server->register ("insertSneakear",
82 array("marca" => "xsd:string", "nombre" => "xsd:string", "precio" => "xsd:int", "color" => "xsd:string", "material" => "xsd:string"),
83 array("return" => "xsd:string"),
84 "urn:server",
85 "urn:server#insertSneakear",
86 "rpc",
87 "encoded",
88 "Metodo que inserta Sneakear");
89
```

Finalmente invocamos el servicio web.

```
119
120 //Flujo de solo lectura
121 $post = file_get_contents('php://input');
122 $server->service($post);
123 ?>
```

Para la acción de **lectura** de un registro o obtención definimos el siguiente método con los parámetros de entrada y salida.

```
28 //Metodo de Obtener
29 function GetSneakear($id){
30     try{
31         global $conexion;
32         $sql = "SELECT * FROM sneaker WHERE ID = :id";
33         $stmt = $conexion->prepare($sql);
34         $stmt->bindParam(':id', $id);
35         $stmt->execute();
36         $data = $stmt->fetch(PDO::FETCH_ASSOC);
37         return json_encode($data);
38         $conexion = null;
39     }catch (Exception $e){
40         return join (" ", array(false));
41     }
42 }
```

```
// Registro de Leer
$server->register("GetSneakear",
array("id" => "xsd:string"),
array("data" => "xsd:string"),
"urn:server",
"urn:server#GetSneakear",
"rpc",
"encoded",
"Metodo que inserta Sneakear");
```



Para la acción de **eliminar** definimos la siguiente función

```
//Metodo de Eliminar
function DelSneakear($id){
    try{
        global $conexion;
        $sql = "DELETE FROM sneaker WHERE ID = :id";
        $stmt = $conexion->prepare($sql);
        $stmt->bindParam(':id', $id);
        $stmt->execute();
        $data = $stmt->fetch(PDO::FETCH_ASSOC);
        return json_encode(true);
        $conexion = null;
    }catch (Exception $e){
        return join(",",array(false));
    }
}
```

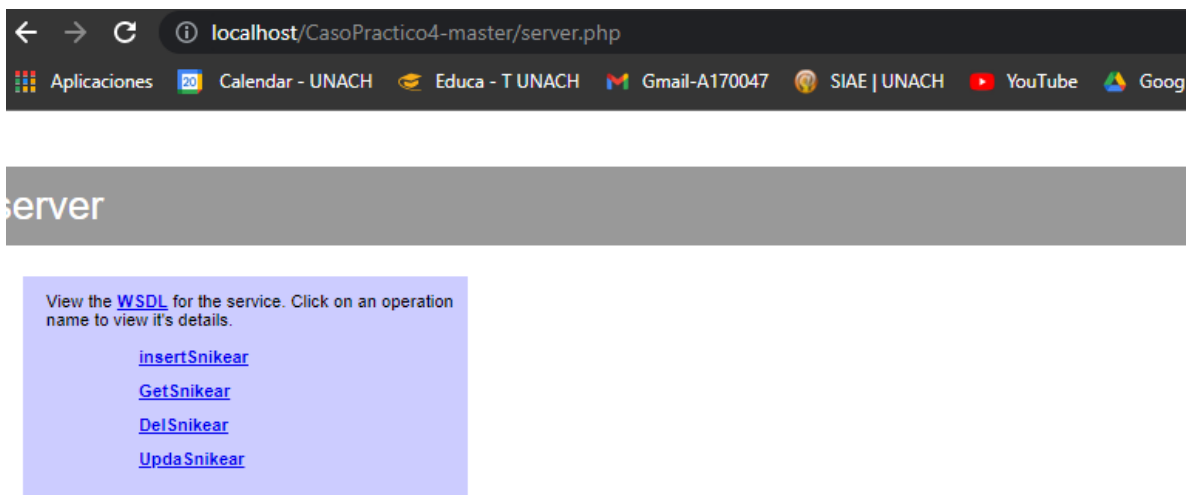
```
// Registro de Eliminar
$server->register("DelSneakear",
array("id" => "xsd:string"),
array("data" => "xsd:string"),
"urn:server",
"urn:server#DelSneakear",
"rpc",
"encoded",
"Metodo que elimina Tenis");
```

para la acción de **actualizar** definimos la siguiente función

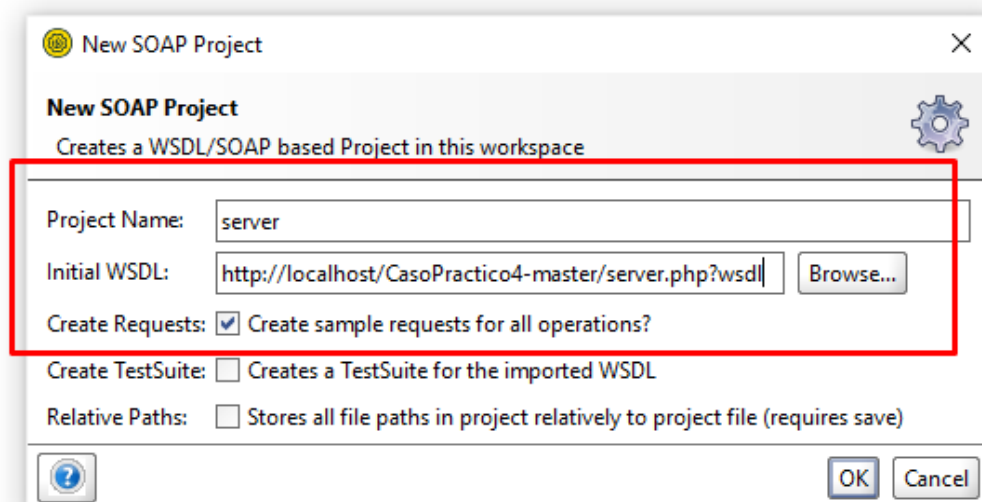
```
//Metodo de Actualizar
function UpdaSneakear($id,$marca,$nombre,$precio,$color,$material){
    try{
        $conexion= new Conexion();
        $consulta=$conexion->prepare("UPDATE sneaker SET Marca='$marca',Nombre='$nombre',Precio='$precio',Color='$color'");
        $consulta -> bindParam(":id", $id, PDO::PARAM_INT);
        $consulta -> bindParam(":marca", $marca, PDO::PARAM_STR);
        $consulta -> bindParam(":nombre", $nombre, PDO::PARAM_STR);
        $consulta -> bindParam(":precio", $precio, PDO::PARAM_INT);
        $consulta -> bindParam(":color", $color, PDO::PARAM_STR);
        $consulta -> bindParam(":material", $material, PDO::PARAM_STR);
        $consulta -> execute();
        $mensaje='Actualizado';
        return join(",",array($mensaje));
    }catch (Exception $e){
        return join(",",array(false));
    }
}
```

```
// Registo de Actualizar
$server->register("UpdaSnikear",
array("id" => "xsd:int", "marca"=>"xsd:string", "nombre"=>"xsd:string", "precio"=>"xsd:int", "color"=>"xsd:string", "ma
array("data" => "xsd:string"),
"urn:server",
"urn:server#UpdaSnikear",
"rpc",
"encoded",
"Metodo que Actualiza Tenis");
```

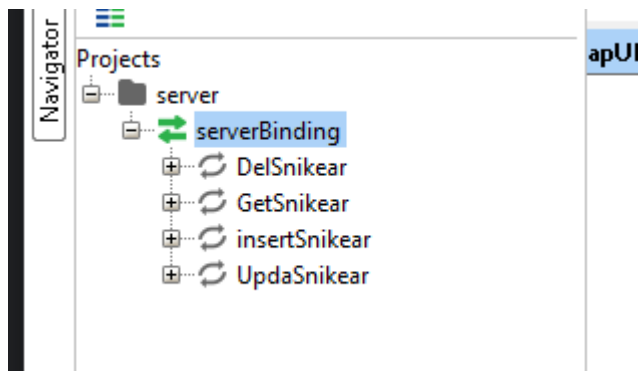
7.- para comprobar nos dirigimos a nuestro servidor local que se encuentre los métodos realizados.



8.- para comprobar nuestro servicio web este en funcionamiento usaremos SoapUI

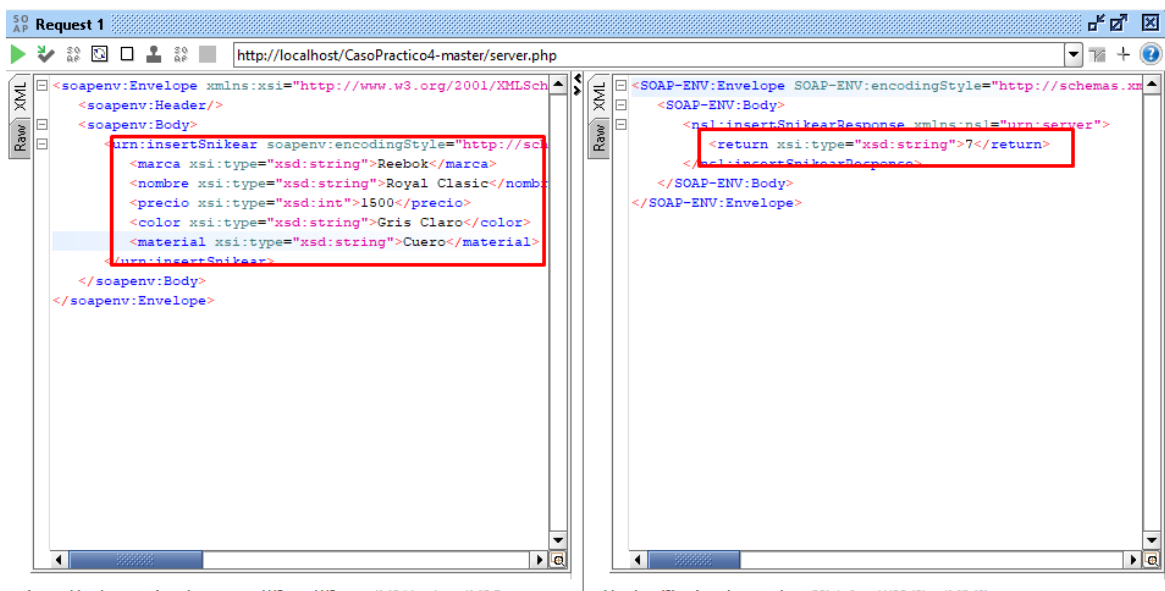


comprobamos que los métodos estén y vamos comprobando uno por uno



## INSERTAR

Insertamos los valores de entrada y de salida nos devuelve el id del producto como



## LECTURA

Insertamos el parámetro de entrada en este caso será el ID que se ingresó anteriormente

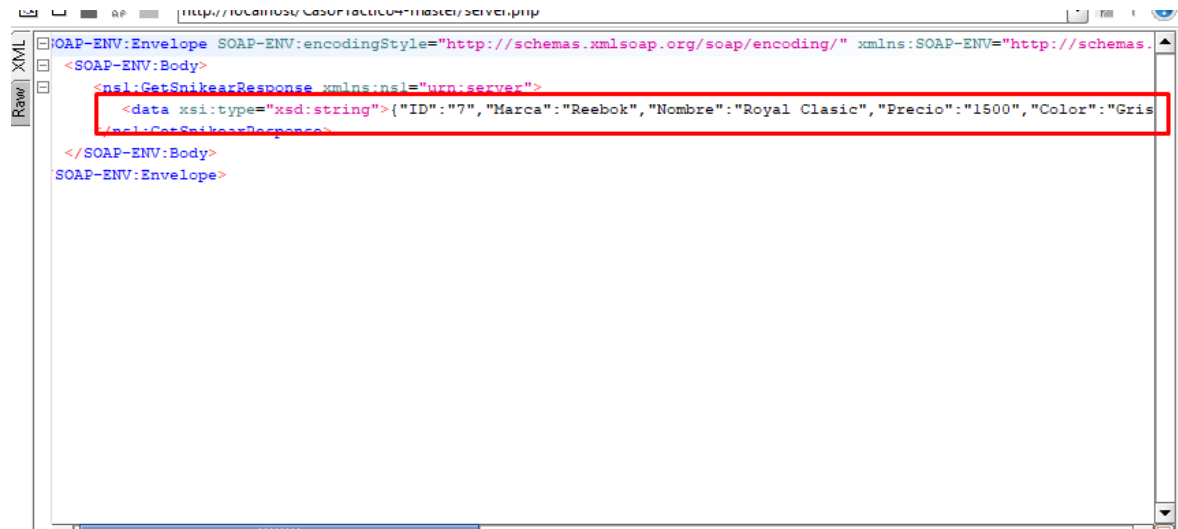


The screenshot shows a SOAP request XML document in a text editor. The XML is structured as follows:

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:GetSneaker soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <id xsi:type="xsd:string">7</id>
    </urn:GetSneaker>
  </soapenv:Body>
</soapenv:Envelope>
```

The `<id xsi:type="xsd:string">7</id>` element is highlighted with a red rectangle.

de salida nos devolverá los datos del registro que hemos buscado a través del ID



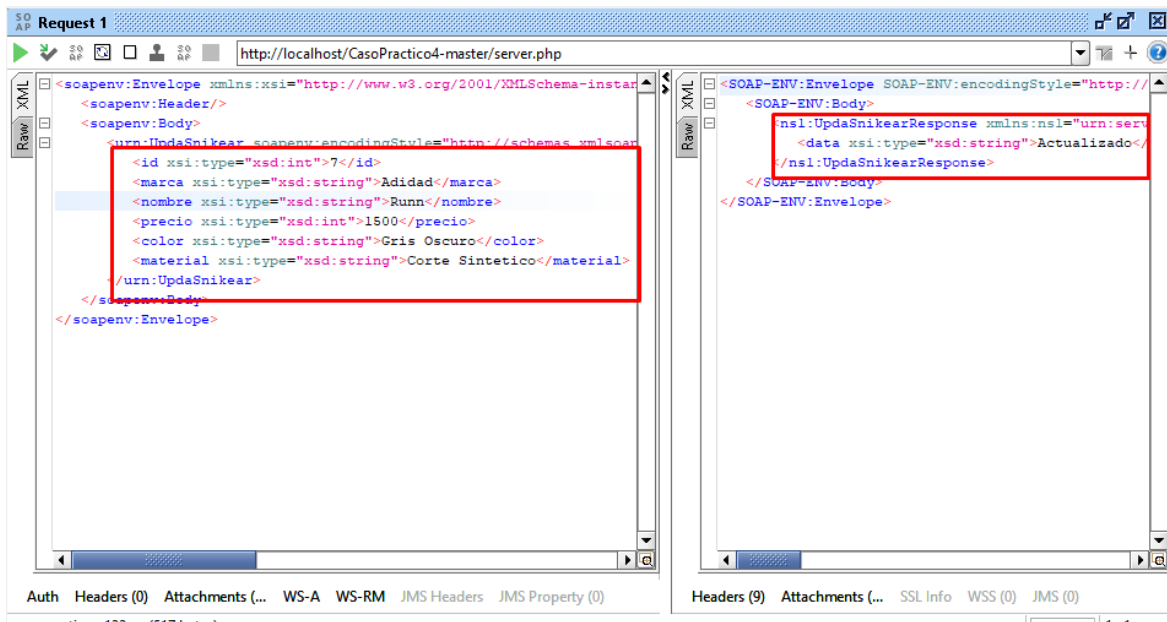
The screenshot shows a SOAP response XML document in a text editor. The XML is structured as follows:

```
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <ns1:GetSneakerResponse xmlns:ns1="urn:server">
      <data xsi:type="xsd:string">{"ID":"7","Marca":"Reebok","Nombre":"Royal Classic","Precio":"1500","Color":"Gris"}</data>
    </ns1:GetSneakerResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The `<data xsi:type="xsd:string">{"ID":"7","Marca":"Reebok","Nombre":"Royal Classic","Precio":"1500","Color":"Gris"}</data>` element is highlighted with a red rectangle.

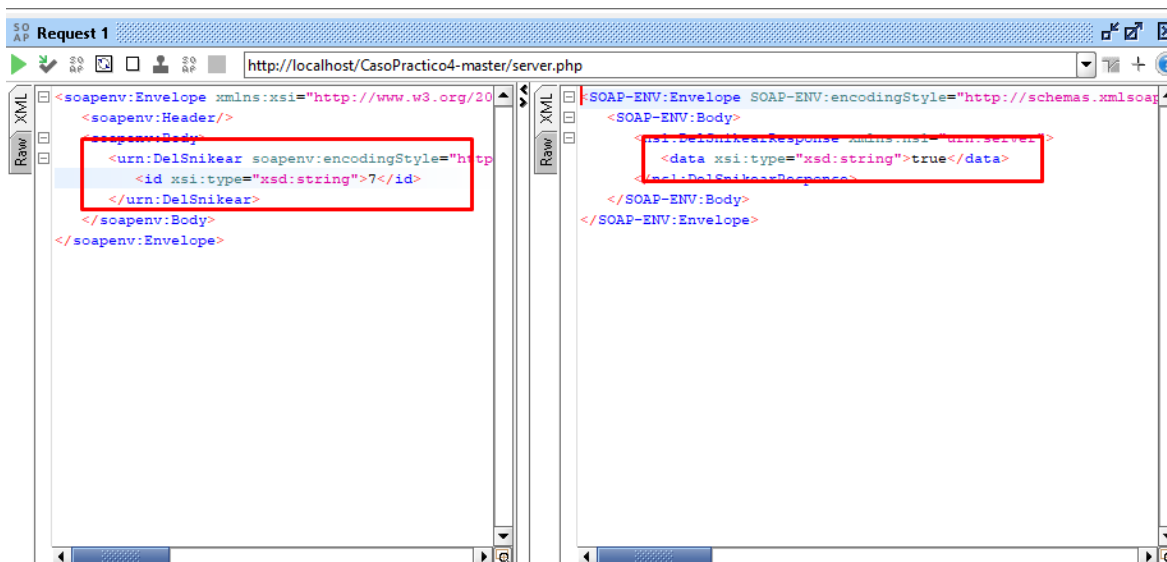
## MODIFICAR

Para ellos ponemos los valores del registro que hemos hecho de salida nos devolverá un mensaje **Actualizado** como seña que se ha modificado el registro



## ELIMINAR

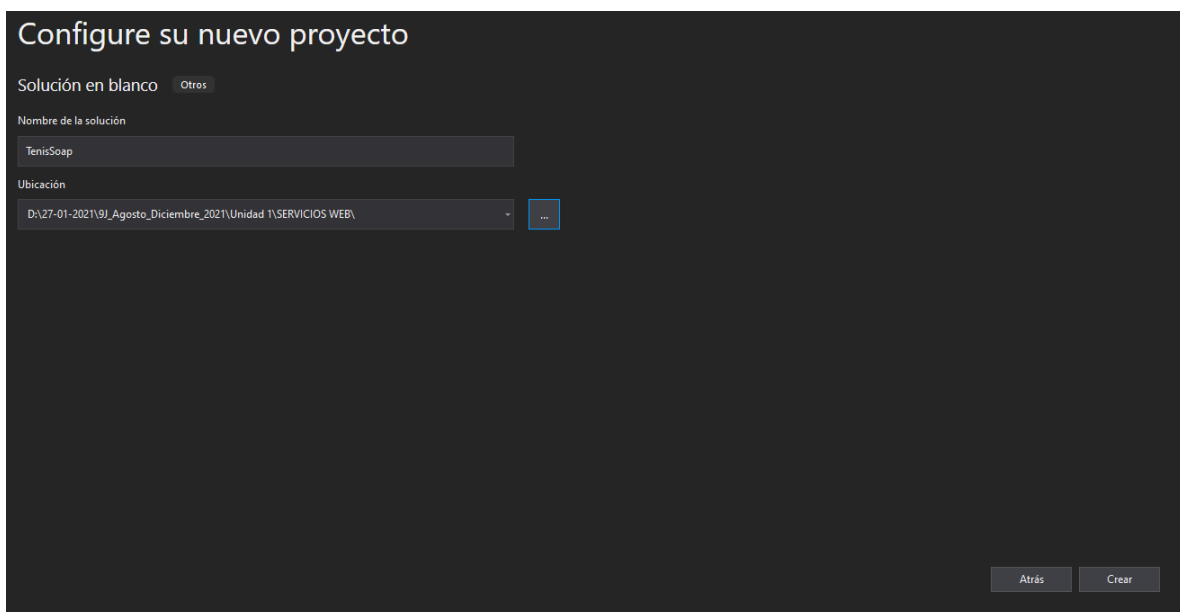
Para esto pondremos el id del registro y nos devolverá **true** como respuesta que se ha eliminado el registro



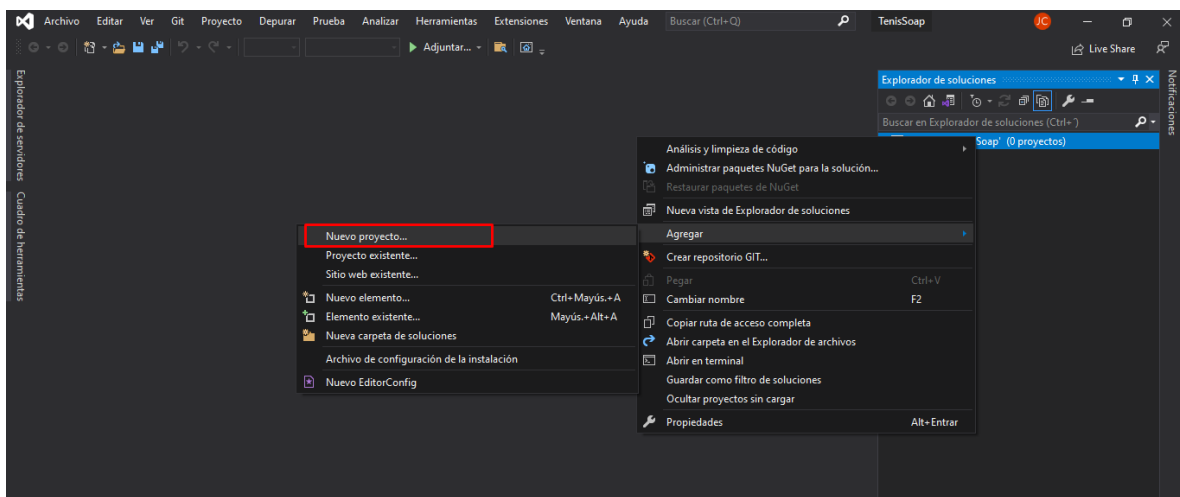
## PARTE CLIENTE

Procederemos a crear una aplicación web ASP.NET (.NET Framework) C# para el consumo del servicio. **ASP.NET** es una tecnología de **.NET Framework** que permite crear **aplicaciones web**. Para obtener más información sobre **ASP.NET**, vea: Documentación de **ASP.NET**. y procedemos a realizar los pasos a continuación:

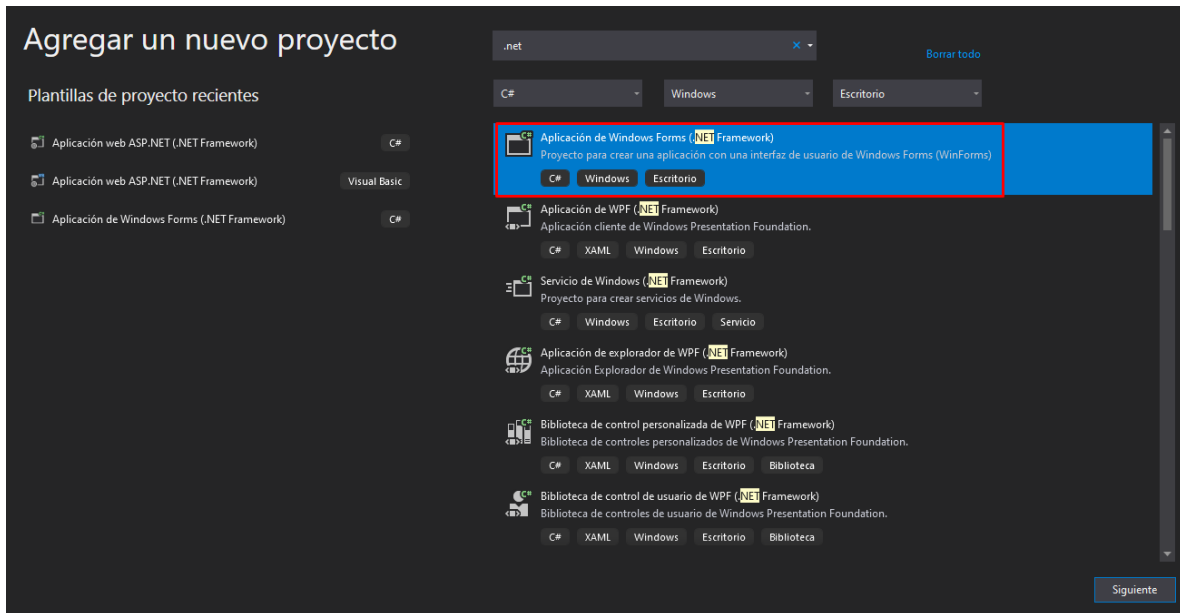
1. Abrir visual Studio y creamos una solución en blanco colocando el nombre de nuestro proyecto, para este ejemplo se denominará “TenisSoap” y lo guardamos en la ruta correspondiente.



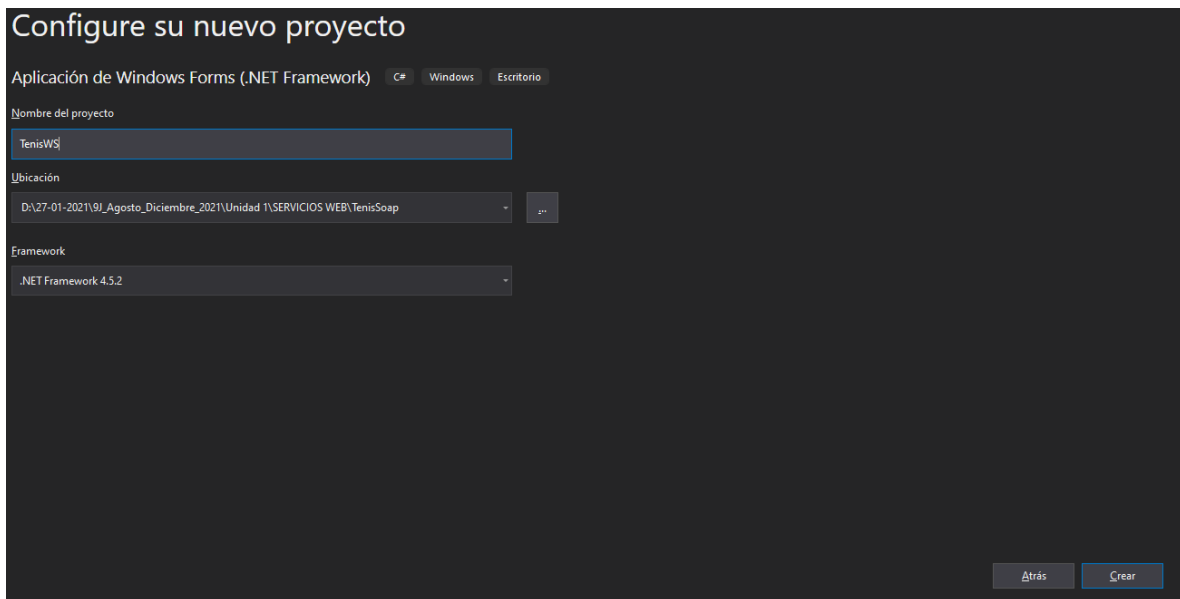
2. Hacemos clic derecho sobre la solución y le damos en agregar nuevo proyecto:



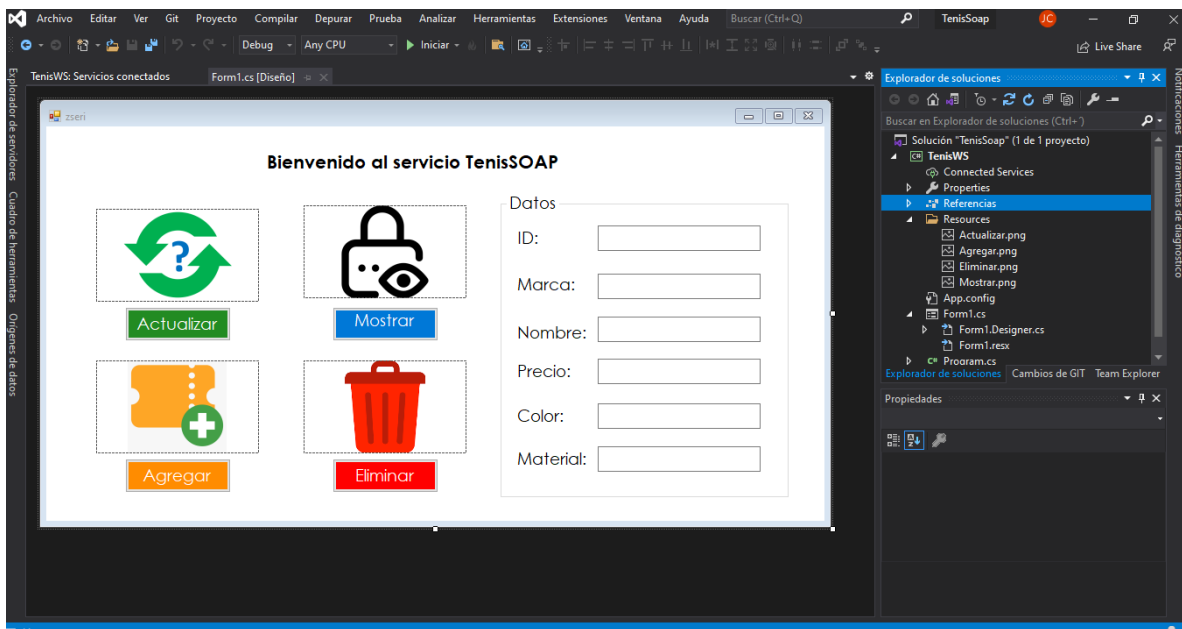
### 3. Buscamos la creación de una aplicación de .NET C#



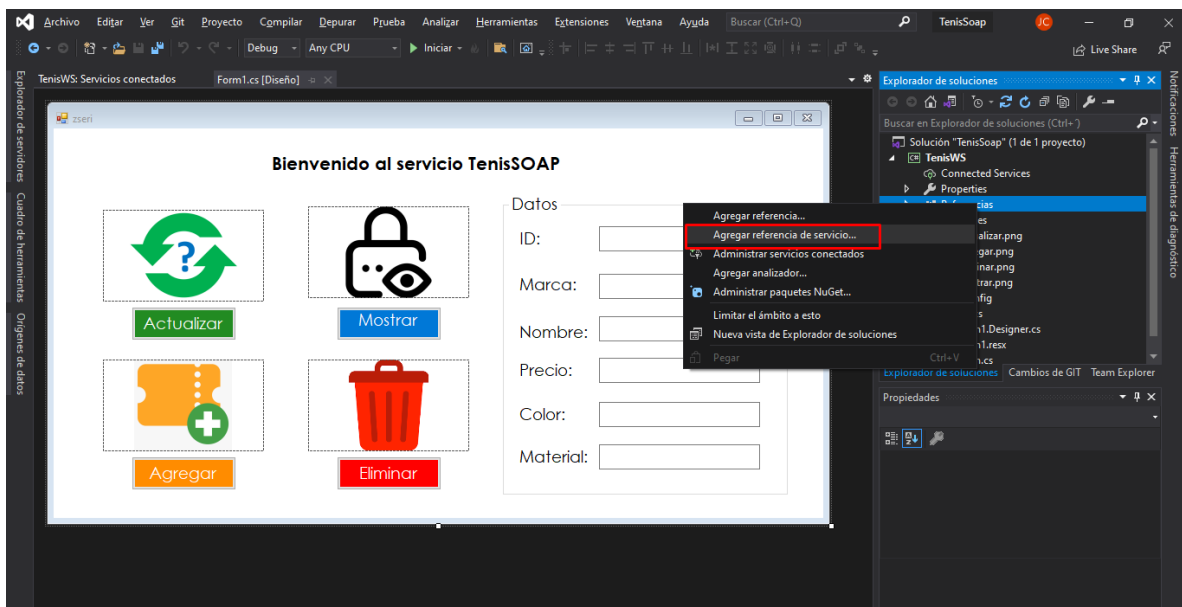
### 4. Definimos un nombre para el proyecto, en este caso “TenisWS” y la creamos



5. Agregamos button, textbox, label, etc., según consideremos necesario para la interfaz de nuestra aplicación.

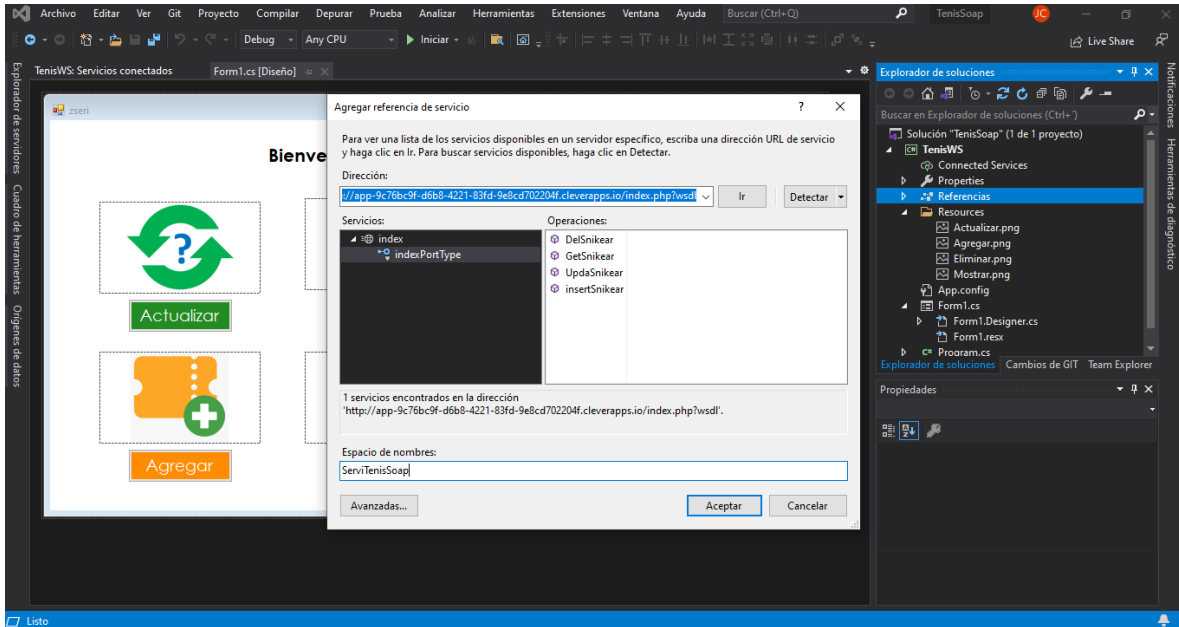


6. Procedemos a agregar la referencia del servicio SOAP con la dirección asignada, de la siguiente manera:





7. Agregamos un nombre identificador del servicio y lo agregamos:



8. Procedemos a programar cada una de las acciones y hacemos las pruebas correspondientes

```
}
private ServiTennisSoap.indexPortTypeClient con = new ServiTennisSoap.indexPortTypeClient();
//string resultado;
1 referencia
private void btnMostrar_Click(object sender, EventArgs e)
{
    txtResultado.Text = con.GetSnikear(txtID.Text.ToString()); MessageBox.Show("Listo");
}

1 referencia
private void btnEliminar_Click(object sender, EventArgs e)
{
    txtResultado.Text = con.DelSnikear(txtID.Text).ToString();
}

1 referencia
private void btnAgregar_Click(object sender, EventArgs e)
{
    txtResultado.Text = con.insertSnikear(txtMarca.Text, txtNombre.Text, Convert.ToInt32(txtID.Text));
}

1 referencia
private void btnActualizar_Click(object sender, EventArgs e)
{
    txtResultado.Text = con.UpdaSnikear(Convert.ToInt32(txtID.Text), txtMarca.Text, txtNombre.Text);
}
```

9. En este caso agregaremos un producto:

TenisSoap

### Bienvenido al servicio TennisSOAP

Material:

  
**Actualizar**

  
**Mostrar**

  
**Agregar**

  
**Eliminar**

**Datos**  
ID:   
Marca:   
Nombre:   
Precio:   
Color:   
Material:

	ID	Marca	Nombre	Precio	Color	Material
 Copy  Delete	1	Addidas	Bad Bunny	9500	Rosa/Café	Cuero combinado
 Copy  Delete	2	Nike	Dior x Air Jordan 1	597345	Blanco	Cuero
 Copy  Delete	3	Puma	RS 2.0	1200	Retro/Metal	Cuero/Tela
 Copy  Delete	4	Reebok	Royal techque t	1600	Blanco	Capellada
 Copy  Delete	5	Converse	Chuck Taylor All Star	1800	Negro	Reciclado
 Copy  Delete	6	dkuguwe	dugud	250	iewid	diheid
 Copy  Delete	9	fefe	fef	280	fefe	fee
 Copy  Delete	10	Nike	Taco	950	Rojo	Fino

check all    With selected:  Edit    Copy    Delete    Export