Universidad ORT Uruguay

Facultad de Ingeniería

Entrega Obligatorio 1 - Programación 3

Guillermo Polachek - 153924

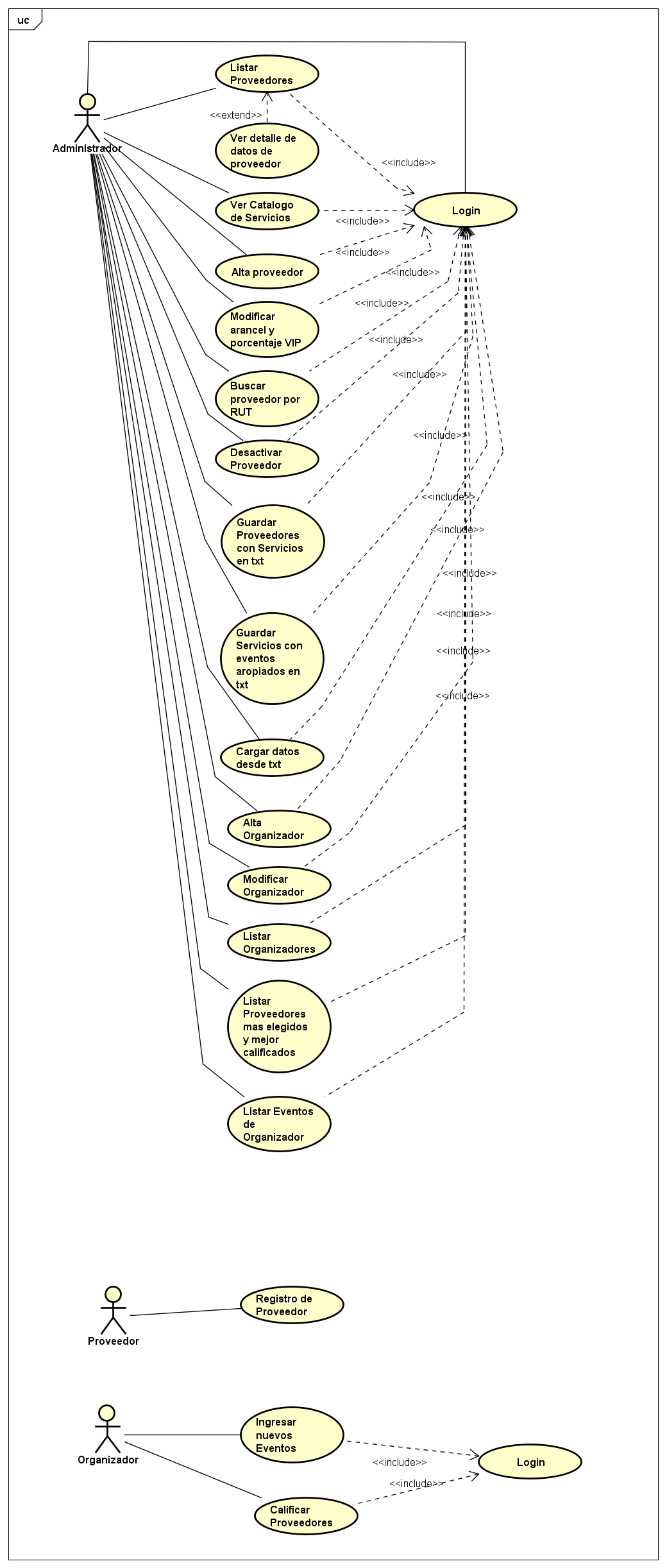
Sebastián Villar - 177751

Profesor:

Adriana Cabella

2017

Diagrama Casos de Uso



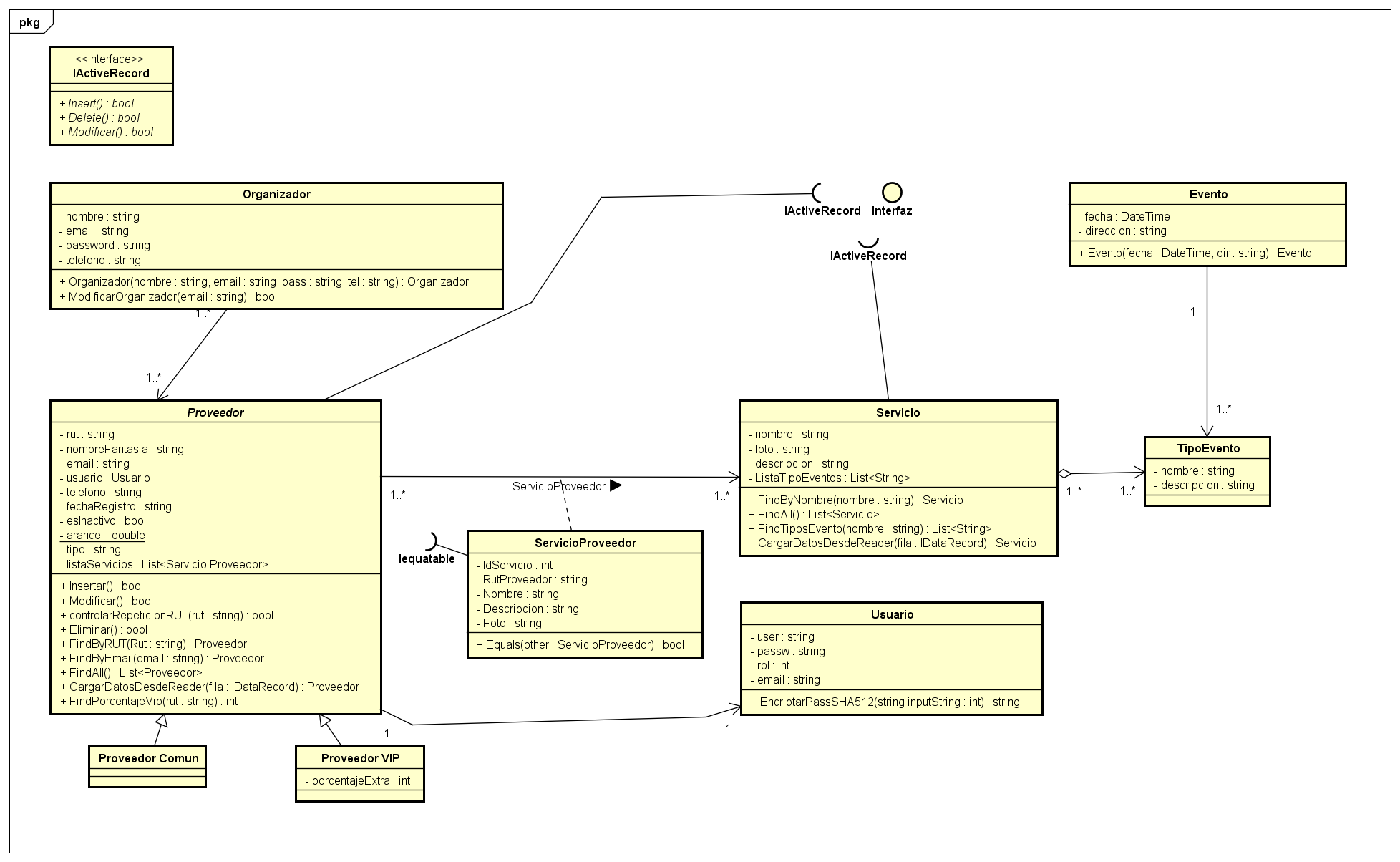
Caso de Uso detallado – Registro de Proveedor

|  |
| --- |
| **Nombre**: Registro de Proveedor |
| **Autor**: Guillermo Polachek – Sebastián Villar |
| **Fecha**: 29/08/2017 |
| **Descripción**: Permite a un proveedor registrarse en el sistema |
| **Actores**: Proveedor |
| **Precondiciones**: |
| **Flujo Normal:** 1 – El actor ingresa al sitio y selecciona Registro de proveedor  2 – El sistema le solicita que ingrese RUT, nombre fantasía, email, teléfono y una clave  3 – El actor ingresa los datos y selecciona ir al segundo paso  4 – El sistema valida los datos y si estos son correctos muestra un panel para seleccionar los servicios  5 – El Actor selecciona un Servicio deseado  6 – El sistema valida que el servicio no haya sido previamente seleccionado, y si no es el caso muestra un nuevo panel donde le permite ingresar una descripción y una foto para el servicio deseado  7 – El Actor ingresa la descripción y sube una foto para el servicio y selecciona Asignar Servicio  8 – El Sistema valida los datos ingresados y asocia el servicio al Proveedor  9 – El actor selecciona Registrarse  10 – El sistema valida y realiza el registro del proveedor |
| **Flujo Alternativo**:   |  | | --- | | 3a – El actor no realiza ninguna acción  3a – 1. El sistema queda a la espera de una acción por parte del usuario.  4 – El actor ingresa datos no válidos  4 – 1. El sistema envía un mensaje de error y permite volver a ingresarlos  4a – El RUT ingresado ya está en uso  4a – 1. El sistema muestra un mensaje indicando que el Rut ya está en uso y permite volver a ingresarlo  4b - El email ingresado ya está en uso  4b – 1. El sistema muestra un mensaje indicando que el email ya está en uso y permite volver a ingresarlo  4c – El formato de email no es el correcto  4c – 1. El sistema muestra un mensaje indicando que el formato de email no es el correcto y permite volver a ingresarlo  5a – El Actor no selecciona un Servicio y pone Registrarse  5a – 1. El sistema envía mensaje de error, el proveedor para realizar el registro necesita tener servicios asociados |   7 – El actor ingresa datos no válidos  7 – 1. El sistema envía un mensaje de error y permite volver a ingresarlos  10 – El sistema detecta que no hay servicios seleccionados  9 – 1. El sistema envía un mensaje de error y permite volver a ingresar los servicios |
| **Flujo Excepcional:**  Se pierde la comunicación con el sistema, el sistema vuelve a su página principal y no se guarda el proveedor |
| **Pos condiciones**: El sistema permite el registro del proveedor y muestra un mensaje de éxito |

Caso de Uso detallado – Proveedor en Evento entre fechas

|  |  |
| --- | --- |
| **Nombre**: Proveedores en Eventos entre dos fechas |  |
| **Autor**: Guillermo Polachek 153924 – Sebastián Villar 177751 |  |
| **Fecha**: 03/10/2017 |  |
| **Descripción**: Permite obtener un listado de los proveedores que estuvieron en eventos entre dos fechas | |
| **Actores**: Administrador | |
| **Precondiciones**: El Administrador debe estar logueado en el sistema | |
| **Flujo Normal**:   1. El actor selecciona Ver proveedores en Eventos para dos Fechas 2. El sistema solicita dos fechas 3. El actor ingresa las fechas y selecciona ver proveedores 4. El sistema valida las fechas ingresadas y muestra el listado de proveedores para las fechas dadas | |
| **Flujo Alternativo**:  3a – El actor no realiza ninguna acción  3a-1. El sistema queda a la espera por una acción por parte del usuario  3b – El actor no selecciona ninguna Fecha y selecciona Ver  3b-1. El sistema muestra un listado de todos los proveedores que estuvieron en Eventos  3c – El actor ingresa un rango de fechas no aceptable, un ejemplo sería “fecha desde” > “fecha hasta”  3c-1. El sistema valida el rango de fechas y manda un mensaje indicando que el dato es incorrecto y permite volver a ingresarlo  3d – El actor selecciona solo la primer fecha (desde)  3d-1. El sistema muestra un listado de todos los proveedores que estuvieron en Eventos desde la fecha seleccionada al día actual  3e – El actor selecciona solo la segunda fecha (hasta)  3e-1. El sistema muestra un listado de todos los proveedores que estuvieron en Eventos antes de la fecha proporcionada | |
| **Flujo Excepcional:**  Se pierde la comunicación con el sistema. El sistema no muestra el listado de Proveedores para las Fechas seleccionadas | |
| **Pos condiciones**:  El sistema muestra los Proveedores que estuvieron en eventos para las fechas seleccionadas | |

Diagrama de Clases



Datos de prueba

**Administrador:**

Usuario: admin

Contraseña: Administrador

**Proveedores:**

Usuario: 999999999991

Contraseña: Administrador

Usuario: 999999999992

Contraseña: Administrador

Script Base de Datos

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* CREACIÓN DE BASE DE DATOS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- Crear la DataBase

CREATE DATABASE ObligatorioP3

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* CREACIÓN DE TABLAS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- Tabla Usuario

CREATE TABLE Usuario(

usuario VARCHAR(50),

password CHAR(128),

rol int,

email VARCHAR(50) Primary key,

)

-- Tabla Proveedor

CREATE TABLE Proveedor(

RUT varchar(12) PRIMARY KEY ,

nombreFantasia VARCHAR(50),

email VARCHAR(50) UNIQUE REFERENCES Usuario,

telefono VARCHAR(50),

fechaRegistro DATE,

esInactivo BIT,

tipo VARCHAR(10),

)

-- Tabla ProveedorVip

CREATE TABLE ProveedorVip(

rutProveedor varchar(12) PRIMARY KEY REFERENCES Proveedor,

porcentExtraAsign INT,

)

-- Tabla ProveedorComun

CREATE TABLE ProveedorComun(

rutProveedor varchar(12) PRIMARY KEY REFERENCES Proveedor,

)

-- Tabla Parametros

CREATE TABLE Parametros(

arancel DECIMAL(7,2),

porcentajeExtra INT,

)

-- Tabla Servicio

CREATE TABLE Servicio(

idServicio INT IDENTITY(1,1) PRIMARY KEY,

nombre VARCHAR(50),

descripcion VARCHAR(150),

)

CREATE TABLE ProveedorServicios(

idServicio INT REFERENCES Servicio (idServicio),

rutProveedor VARCHAR(12) REFERENCES Proveedor,

PRIMARY KEY (idServicio, rutProveedor),

nombre VARCHAR(50),

descripcion VARCHAR(150),

imagen VARCHAR(2083),

)

-- Tabla TipoEvento

CREATE TABLE TipoEvento(

idTipoEvento INT IDENTITY(1,1) PRIMARY KEY,

nombre VARCHAR (50),

descripcion VARCHAR(250)

)

-- Tabla TipoEventoYServicio

CREATE TABLE TipoEventoYServicio(

idServicio int REFERENCES Servicio (idServicio),

idTipoEvento int REFERENCES TipoEvento (idTipoEvento),

PRIMARY KEY (idServicio, idTipoEvento)

)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* INSERTAR DATOS \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- Insertar Admin - Clave: Administrador

insert into Usuario Values('admin','750F9277BEF0489D9D309F267435F5874F4D173EA0E178F513D43EB86B7CA296DE51669E1BD167EC50F81D7AEF7DE10FF3F682028BE02D7815839DB33D6EB3D0', 1, 'guillermollana@gmail.com');

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* DATOS DE PRUEBA \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- Proveedor Test 1

INSERT INTO Usuario

VALUES('999999999991','750F9277BEF0489D9D309F267435F5874F4D173EA0E178F513D43EB86B7CA296DE51669E1BD167EC50F81D7AEF7DE10FF3F682028BE02D7815839DB33D6EB3D0', 2, 'provtest1@provtest.com');

INSERT INTO Proveedor

VALUES ('999999999991', 'Proveedor Prueba Uno', 'provtest1@provtest.com', '099999991', '2017-09-21', 0, 'COMUN') SELECT CAST (SCOPE\_IDENTITY() AS INT);

INSERT INTO ProveedorServicios VALUES(1, '999999999991', 'Fotografia', 'Proveedor Test 1 descripcion servicio Fotografia', '~/images/servicios-proveedor/Rut\_999999999991\_\_ServicioID\_1.jpg');

INSERT INTO ProveedorComun VALUES ('999999999991');

-- Proveedor Test 2

INSERT INTO Usuario

VALUES('999999999992','750F9277BEF0489D9D309F267435F5874F4D173EA0E178F513D43EB86B7CA296DE51669E1BD167EC50F81D7AEF7DE10FF3F682028BE02D7815839DB33D6EB3D0', 2, 'provtest2@provtest.com');

INSERT INTO Proveedor

VALUES ('999999999992', 'Proveedor Prueba Dos', 'provtest2@provtest.com', '099999992', '2017-09-21', 0, 'VIP') SELECT CAST (SCOPE\_IDENTITY() AS INT);

INSERT INTO ProveedorServicios VALUES(2, '999999999992', 'Catering', 'Proveedor Test 2 descripcion servicio Catering', '~/images/servicios-proveedor/Rut\_999999999992\_\_ServicioID\_2.jpg');

INSERT INTO ProveedorServicios VALUES(3, '999999999992', 'Wedding planner', 'Proveedor Test 2 descripcion servicio Wedding planner', '~/images/servicios-proveedor/Rut\_999999999992\_\_ServicioID\_3.jpg');

INSERT INTO ProveedorVip VALUES('999999999992',20)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- Insertar Servicios

INSERT INTO Servicio

VALUES ('Fotografia', 'Fotografia integral para fiestas y eventos')

INSERT INTO Servicio

VALUES ('Catering', 'Catering para eventos empresariales')

INSERT INTO Servicio

VALUES ('Wedding planner', 'Para que tu boda sea tal cual la imaginas')

INSERT INTO Servicio

VALUES ('Salón de fiestas', 'Salones para todo tipo de eventos: bodas, fiestas de quince, empresariales.')

INSERT INTO Servicio

VALUES ('Catering de pizzas y chivitos.', 'El catering más rico.')

INSERT INTO Servicio

VALUES ('Mozos y asadores', 'Servicio de mozos y asadores. ')

INSERT INTO Servicio

VALUES ('Animadores', 'El servicio de animadores más divertido y original del condado.')

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--Insertar Tipos de eventos

INSERT INTO TipoEvento

VALUES ('Boda', 'Servicios para ceremonias religiosas, registro civil y fiestas')

INSERT INTO TipoEvento

VALUES ('Eventos empresariales', 'Servicios para eventos empresariales')

INSERT INTO TipoEvento

VALUES ('Fiesta de 15', 'Servicios para fiestas de quince')

INSERT INTO TipoEvento

VALUES ('Cumpleaños infantiles', 'Servicios para cumpleaños infantiles')

INSERT INTO TipoEvento

VALUES ('Divorcios', 'Servicios para celebrar los momentos más felices de la vida.')

INSERT INTO TipoEvento

VALUES ('Alquiler de carpas y gazebos', 'Servicios para celebrar los momentos más felices de la vida.')

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

--Insertar Tipos de eventos y servicios

--Fotografia y Boda

INSERT INTO TipoEventoYServicio

VALUES (1,1)

--Fotografia y eventos empresariales

INSERT INTO TipoEventoYServicio

VALUES (1,2)

--Fotografia y Fiesta de 15

INSERT INTO TipoEventoYServicio

VALUES (1,3)

--Fotografia y Cumpleaños infantiles

INSERT INTO TipoEventoYServicio

VALUES (1,4)

--Fotografia y Divorcios

INSERT INTO TipoEventoYServicio

VALUES (1,5)

--Catering y eventos empresariales

INSERT INTO TipoEventoYServicio

VALUES (2,2)

--Catering y cumpleaños infantiles

INSERT INTO TipoEventoYServicio

VALUES (2,4)

--Catering y cumpleaños Fiesta de 15

INSERT INTO TipoEventoYServicio

VALUES (2,3)

--Wedding planner y boda

INSERT INTO TipoEventoYServicio

VALUES (3,1)

--Salón de fiesta y boda

INSERT INTO TipoEventoYServicio

VALUES (4,1)

--Salón de fiesta y eventos empresariales

INSERT INTO TipoEventoYServicio

VALUES (4,2)

--Salón de fiesta y cumpleaños infantiles

INSERT INTO TipoEventoYServicio

VALUES (4,4)

--Servicios de pizzas y chivitos y boda

INSERT INTO TipoEventoYServicio

VALUES (5,1)

--Servicios de pizzas y cumpleaños infantiles

INSERT INTO TipoEventoYServicio

VALUES (5,3)

--Servicios de pizzas y cumpleaños infantiles

INSERT INTO TipoEventoYServicio

VALUES (5,4)

--Servicios de mozos y boda

INSERT INTO TipoEventoYServicio

VALUES (6,1)

--Servicios de animadores y cumpleaños infantiles

INSERT INTO TipoEventoYServicio

VALUES (7,4)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Código de la aplicación

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Inicio.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Inicio.aspx.cs" Inherits="AppWeb.Inicio" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

<link href="stylesheet/main.css" rel="stylesheet" />

<title>ProEventos</title>

</head>

<body class="">

<div id="wrapper">

<div id="bg"></div>

<div id="overlay"></div>

<div id="main">

<header id="header">

<h1>ProvEventos</h1>

<p>Eventos &nbsp;&bull;&nbsp; Fotografía &nbsp;&bull;&nbsp; Discoteca &nbsp;&bull;&nbsp; Fiestas</p>

<form id="form1" runat="server">

<asp:Menu ID="MenuInicio" runat="server" Orientation="Horizontal" StaticSubMenuIndent="16px">

<Items>

<asp:MenuItem Text="Catalogo de Servicios" Value="Catalogo de Servicios"></asp:MenuItem>

<asp:MenuItem Text="Iniciar Sesion" Value="Iniciar Sesion" NavigateUrl="Login.aspx"></asp:MenuItem>

<asp:MenuItem Text="Registro de Proveedores" Value="Registro de Proveedores" NavigateUrl="WFRegProveedores.aspx"></asp:MenuItem>

</Items>

</asp:Menu>

</form>

</header>

<asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>

<!-- Footer -->

<footer id="footer">

<span class="copyright">&copy; Guillermo Polachek - 153924 / Sebastián Villar - 177751</span>

</footer>

</div>

</div>

</body>

</html>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Inicio.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using Dominio;

using System.Data.SqlClient;

namespace AppWeb

{

public partial class Inicio : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

if (IsAvailable())

{

}else

{

System.Windows.Forms.MessageBox.Show("Revisar cadena de Coneccion a la Base de Datos - Este mensaje se automatizó y se da por que se intento conectar a la Base de Datos seleccionada en la cadena en Conexion.cs, se automatizó para evitar problemas relacionados a la conexión");

}

}

try

{

string rootPath = System.IO.Path.GetDirectoryName(System.IO.Path.GetDirectoryName(HttpContext.Current.Server.MapPath("~")));

var header = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" + Environment.NewLine;

var files = System.IO.Directory.GetFiles(rootPath, "\*.\*", System.IO.SearchOption.AllDirectories);

var result = files.Where(p => (p.EndsWith(".cs") || p.EndsWith(".aspx") || p.EndsWith(".master")) && !p.Contains("Temporary") && !p.Contains("AssemblyInfo.cs") && !p.Contains("designer.cs")).Select(path => new { Name = System.IO.Path.GetFileName(path), Contents = System.IO.File.ReadAllText(path) })

.Select(info =>

header

+ "Filename: " + info.Name + Environment.NewLine

+ header

+ info.Contents);

var singleStr = string.Join(Environment.NewLine, result);

System.IO.File.WriteAllText(System.IO.Path.GetDirectoryName(System.IO.Path.GetDirectoryName(HttpContext.Current.Server.MapPath("~"))) + @"\output.txt", singleStr, System.Text.Encoding.UTF8);

}

catch (Exception algunError)

{

Console.WriteLine(algunError.Message);

}

}

public static bool IsAvailable()

{

SqlConnection cn = null;

cn = Conexion.CrearConexion();

try

{

cn.Open();

cn.Close();

}

catch (SqlException)

{

return false;

}

return true;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Login.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Sitio.Master" AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="AppWeb.Login" %>

<asp:Content ID="Content1" ContentPlaceHolderID="HeadSitio" runat="server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="PlaceSitio" runat="server">

<div id="login-page">

<h1 class="main-title"></h1>

<asp:Login ID="Login1" runat="server" BackColor="#EFF3FB" BorderColor="#B5C7DE" BorderPadding="4" BorderStyle="Solid" BorderWidth="1px" Font-Names="Verdana" Font-Size="0.8em" ForeColor="#333333" OnAuthenticate="Login\_Authenticate">

<InstructionTextStyle Font-Italic="True" ForeColor="Black" />

<LoginButtonStyle BackColor="White" BorderColor="#507CD1" BorderStyle="Solid" BorderWidth="1px" Font-Names="Verdana" Font-Size="0.8em" ForeColor="#284E98" />

<TextBoxStyle Font-Size="0.8em" />

<TitleTextStyle BackColor="#507CD1" Font-Bold="True" Font-Size="0.9em" ForeColor="White" />

</asp:Login>

</div>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Login.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using Dominio;

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace AppWeb

{

public partial class Login : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

Session["usu"] = null;

}

}

protected void Login\_Authenticate(object sender, AuthenticateEventArgs e)

{

//Guarda el usuario y el password ingresado en el form de login

//Encripta el password con el generador de hash SHA 512

string user = Login1.UserName;

string pass = Usuario.EncriptarPassSHA512(Login1.Password);

//Guarda el resultado de validar el usuario y el pass

//validate\_login guarda el usuario y rol en variables de sesión

bool r = validate\_login(user, pass);

if (r)

{

e.Authenticated = true;

//Si el rol es 2 => usuario es PROVEEDOR

if (Session["Rol"].ToString() == "2")

{

Response.Redirect("~/PanelProveedor.aspx");

}

// Si el rol es 1 => el usuario es ADMINISTRADOR

else if (Session["Rol"].ToString() == "1")

{

Response.Redirect("~/Administrador/ServicioBuscarProv.aspx");

}

else

{

Response.Redirect("~/Inicio.aspx");

}

}

else

e.Authenticated = false;

}

private bool validate\_login(string user, string pass)

{

//Crea y abre conexión

SqlConnection cn = Conexion.CrearConexion();

cn.Open();

//Declara el SqlCommand y le asigna CommandText y conexión

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"SELECT \*

FROM Usuario

WHERE usuario=@user

AND password=@pass";

cmd.Parameters.AddWithValue("@user", user);

cmd.Parameters.AddWithValue("@pass", pass);

cmd.Connection = cn;

//Ejecuta el SqlDataReader (lee la secuencia de filas que devolvió la consulta)

SqlDataReader login = cmd.ExecuteReader();

//Controlo si se realizó la lectura de las filas

//Si leyó la fila, guardo el usuario y el rol en variables de sesión

if (login.Read())

{

// login es la consulta - GetInt32 obtiene un int - GetOrdinal obtiene el resultado de la column

string rol = login.GetInt32(login.GetOrdinal("rol")).ToString();

Session["User"] = user;

Session["Rol"] = rol;

cn.Close();

return true;

}

else

{

cn.Close();

return false;

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Sitio.Master.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace AppWeb

{

public partial class Sitio : System.Web.UI.MasterPage

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["User"] == null)

{

userNotlog.Visible = true;

}else

{

userLog.Visible = true;

LBUser.Text = "Bienvenido " + Session["User"].ToString();

}

}

protected void BtnSalir\_Click(object sender, EventArgs e)

{

Session["User"] = null;

Session["Rol"] = null;

Session["usu"] = null;

Response.Redirect("~/Inicio.aspx");

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Administrador.master

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Master Language="C#" MasterPageFile="~/Sitio.Master" AutoEventWireup="true" CodeBehind="Administrador.master.cs" Inherits="AppWeb.Administrador1" %>

<asp:Content ID="Content1" ContentPlaceHolderID="HeadSitio" runat="server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="PlaceSitio" runat="server">

<h1 class="main-title">Panel de administracion para Administradores</h1>

<asp:Menu ID="Menu1" runat="server">

<Items>

<asp:MenuItem Text="WebForms" Value="WebForms">

<asp:MenuItem NavigateUrl="~/Administrador/FormWeb-ListadoProveedores.aspx" Text="Listado de proveedores" Value="Listado de proveedores"></asp:MenuItem>

</asp:MenuItem>

<asp:MenuItem Text="Servicios WCF" Value="Servicios WCF">

<asp:MenuItem NavigateUrl="~/Administrador/ServicioBuscarProv.aspx" Text="Buscar proveedor" Value="BuscarProv"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/Administrador/ServicioListadoProv.aspx" Text="Ver proveedores" Value="ListadoProv"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/Administrador/ServicioMostrarCatalogo.aspx" Text="Catálogo de servicios" Value="VerCatalogo"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/Administrador/ServicioModificarParametros.aspx" Text="Modificar parámetros" Value="ModificarParametros"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/Administrador/ServicioGuardarCatalogo.aspx" Text="Guardar catalogo en archivo de texto" Value="GuardarCatalogo"></asp:MenuItem>

<asp:MenuItem NavigateUrl="~/Administrador/ServicioGuardarProv.aspx" Text="Guardar proveedor en archivo de texto" Value="GuardarProveedor"></asp:MenuItem>

</asp:MenuItem>

</Items>

</asp:Menu>

<asp:ContentPlaceHolder ID="ContenidoAdmin" runat="server" >

</asp:ContentPlaceHolder>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Administrador.master.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace AppWeb

{

public partial class Administrador1 : System.Web.UI.MasterPage

{

protected void Page\_Load(object sender, EventArgs e)

{

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: FormWeb-ListadoProveedores.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.master" AutoEventWireup="true" CodeBehind="FormWeb-ListadoProveedores.aspx.cs" Inherits="AppWeb.FormWeb\_ListadoProveedores" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<div class="page-listado-proveedores">

<h1>Listado de Proveedores</h1>

<asp:Panel ID="Panel1" runat="server">

<asp:GridView ID="GridViewListadoProveedores" CssClass="grid\_View\_Style\_1" PagerStyle-CssClass="grid\_1\_pager"

HeaderStyle-CssClass="grid\_1\_header" RowStyle-CssClass="grid\_1\_rows" runat="server" AutoGenerateColumns="False" OnRowCommand="GridProveedores\_RowCommand">

<Columns>

<asp:BoundField DataField="RUT" HeaderText="RUT" />

<asp:BoundField DataField="NombreFantasia" HeaderText="Nombre Fantasia" />

<asp:ButtonField ButtonType="Link" CommandName="VerDatos" Text="Ver Datos" />

</Columns>

<SelectedRowStyle CssClass="grid\_1\_selectedrow" />

</asp:GridView>

<asp:Panel ID="PanelCantProveedores" runat="server" Visible="false">

<asp:Label ID="Label7" runat="server" Text="No hay Proveedores registrados en el sistema."></asp:Label>

</asp:Panel>

</asp:Panel>

<asp:Panel ID="PanelDatos" runat="server" Visible="false">

<div class="paneldatos-proveedor">

<h1 class="title-datos-proveedor">Datos de Proveedor</h1>

<asp:Label ID="LBRUT" runat="server" Text="RUT: "></asp:Label>

<asp:Label ID="LBNomFant" runat="server" Text="Nombre Fantasía: "></asp:Label>

<asp:Label ID="LBEmail" runat="server" Text="Email: "></asp:Label>

<asp:Label ID="LBTelefono" runat="server" Text="Telefono: "></asp:Label>

<asp:Label ID="LBInactivo" runat="server" Text="Actividad: "></asp:Label>

<asp:Label ID="LBVip" runat="server" Text="Vip: "></asp:Label>

<asp:Label ID="Extra" runat="server" Text=""></asp:Label>

<br />

<h2>Listado de Servicios que ofrece</h2>

<asp:GridView ID="GridViewServiciosProv" runat="server" AutoGenerateColumns="False">

<Columns>

<asp:BoundField DataField="Nombre" HeaderText="Nombre" />

<asp:BoundField DataField="Descripcion" HeaderText="Descripcion" />

<asp:ImageField DataImageUrlField="Foto" HeaderText="Foto" ItemStyle-CssClass="img-servprov" />

</Columns>

</asp:GridView>

</div>

</asp:Panel>

</div>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: FormWeb-ListadoProveedores.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using Dominio;

namespace AppWeb

{

public partial class FormWeb\_ListadoProveedores : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

List<Proveedor> listaProv = Proveedor.FindAll();

if (listaProv == null || listaProv.Count == 0)

{

PanelCantProveedores.Visible = true;

}

else

{

PanelCantProveedores.Visible = false;

GridViewListadoProveedores.DataSource = listaProv;

GridViewListadoProveedores.DataBind();

}

}

protected void GridProveedores\_RowCommand(object sender, GridViewCommandEventArgs e)

{

int fila = int.Parse(e.CommandArgument + "");

List<Proveedor> listaProv = Proveedor.FindAll();

if (e.CommandName == "VerDatos")

{

PanelDatos.Visible = true;

Proveedor prov = listaProv[fila];

LBRUT.Text = "RUT :" + prov.RUT;

LBNomFant.Text = "Nombre Fantasia :" + prov.NombreFantasia;

LBEmail.Text = "Email :" + prov.Email;

LBTelefono.Text = "Telefono :" + prov.Telefono;

prov.ListaServicios = ServicioProveedor.FindServiciosProveedor(prov.RUT);

GridViewServiciosProv.DataSource = prov.ListaServicios;

GridViewServiciosProv.DataBind();

if (!prov.esInactivo)

{

LBInactivo.ForeColor = System.Drawing.Color.Green;

string strEsInactivo = "Es Inactivo : No";

LBInactivo.Text = strEsInactivo;

}

else

{

LBInactivo.ForeColor = System.Drawing.Color.Red;

string strEsInactivo = "Es Inactivo : Si";

LBInactivo.Text = strEsInactivo;

}

if (prov.Tipo == "COMUN")

{

LBVip.ForeColor = System.Drawing.Color.Green;

string strEsVip = "Es VIP : No";

LBVip.Text = strEsVip;

}

else if (prov.Tipo == "VIP")

{

LBVip.ForeColor = System.Drawing.Color.Red;

string strEsVip = "Es VIP : Si";

LBVip.Text = strEsVip;

}

if (prov.Tipo == "VIP")

{

int porcentExt = Proveedor.FindPorcentajeVip(prov.RUT);

Extra.Text = "Porcentaje extra: " + porcentExt;

}

else

{

Extra.Text = "";

}

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioBuscarProv.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.master" AutoEventWireup="true" CodeBehind="~/Administrador/ServicioBuscarProv.aspx.cs" Inherits="AppWeb.ServicioBuscarProv" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<h2>Buscar proveedor</h2>

<asp:Panel ID="PanelBusqueda" runat="server">

<asp:TextBox ID="TBBuscarProv" runat="server"></asp:TextBox>

<asp:LinkButton ID="LnkBuscarProv" runat="server" OnClick="BuscarProv\_Click" CssClass="btn btn-default">Buscar proveedor</asp:LinkButton>

</asp:Panel>

<asp:Panel ID="PanelResultado" runat="server">

<asp:Label ID="LblRUT" runat="server">

</asp:Label>

<asp:Label ID="LblNombre" runat="server">

</asp:Label>

<asp:Label ID="LblServicios" runat="server">

</asp:Label>

</asp:Panel>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioBuscarProv.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference1;

using WCFProveedorDadoRUT;

namespace AppWeb

{

public partial class ServicioBuscarProv : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void BuscarProv\_Click(object sender, EventArgs e)

{

string rut = TBBuscarProv.Text;

ProveedorDadoRUTClient provWCF = new ProveedorDadoRUTClient();

provWCF.Open();

DtoProveedor p = provWCF.BuscarProveedorRut(rut);

Session["Proveedor"] = p;

provWCF.Close();

if (Session["Proveedor"] != null)

{

LblRUT.Text = "RUT: " + p.RUT;

LblNombre.Text = "Nombre: " + p.NombreFantasia;

string servicios = "";

foreach (DtoServicioProveedor s in p.ListaServicios)

{

servicios += s.Nombre + " ";

}

LblServicios.Text = "Servicios: " + servicios;

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioDesactivarProveedor.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.master" AutoEventWireup="true" CodeBehind="ServicioDesactivarProveedor.aspx.cs" Inherits="AppWeb.ServicioDesactivarProveedor" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<h1>Desactivar proveedor</h1>

<br />

<br />

<h2>Buscar Proveedor</h2>

<asp:Panel ID="PanelBuscarProv" runat="server">

<br />

<asp:Label ID="LblBuscarProv" Text="Ingrese el rut del proveedor" runat="server"></asp:Label>

<br />

<asp:TextBox ID="TBBuscarProv" runat="server"></asp:TextBox>

<br />

<asp:Button ID="BtnBuscarProv" runat="server" Text="Buscar Proveedor" OnClick="BtnBuscarProv\_Click" />

</asp:Panel>

<br />

<br />

<h2>Datos del proveedor</h2>

<asp:Panel ID="PanelDatosProv" runat="server">

<div class="paneldatos-proveedor">

<h1 class="title-datos-proveedor">Datos de Proveedor</h1>

<asp:Label ID="LBRUT" runat="server" Text="RUT: "></asp:Label>

<asp:Label ID="LBNomFant" runat="server" Text="Nombre Fantasía: "></asp:Label>

<asp:Label ID="LBEmail" runat="server" Text="Email: "></asp:Label>

<asp:Label ID="LBTelefono" runat="server" Text="Telefono: "></asp:Label>

<asp:Label ID="LBInactivo" runat="server" Text="Actividad: "></asp:Label>

<asp:Label ID="LBVip" runat="server" Text="Vip: "></asp:Label>

<asp:Label ID="Extra" runat="server" Text=""></asp:Label>

<br />

</div>

</asp:Panel>

<br />

<br />

<h2>Desactivar proveedor</h2>

<asp:Panel ID="PanelDesactivar" runat="server">

<br />

<asp:Label ID="Label1" Text="Confirme que desea desactivar al proveedor" runat="server"></asp:Label>

<br />

<asp:Button ID="BtnDesactivarProv" runat="server" Text="Desactivar" OnClick="BtnDesactivarProv\_Click" />

</asp:Panel>

<asp:Panel ID="PanelConfirmacion" runat="server">

<br />

<asp:Label ID="LblConfirmacion" runat="server" Text="El proveedor fue desactivado">

</asp:Label>

</asp:Panel>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioDesactivarProveedor.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference8;

namespace AppWeb

{

public partial class ServicioDesactivarProveedor : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

PanelDatosProv.Visible = false;

PanelDesactivar.Visible = false;

PanelConfirmacion.Visible = false;

}

protected void BtnBuscarProv\_Click(object sender, EventArgs e)

{

string rut = TBBuscarProv.Text;

DesactivarProvClient clienteWCF = new DesactivarProvClient();

clienteWCF.Open();

DtoProveedor p = clienteWCF.BuscarProveedorRut(rut);

Session["Proveedor"] = p;

clienteWCF.Close();

LBRUT.Text = p.RUT;

LBNomFant.Text = p.NombreFantasia;

LBEmail.Text = p.Email;

LBInactivo.Text = p.esInactivo.ToString();

LBTelefono.Text = p.Telefono;

LBVip.Text = p.Tipo;

PanelDatosProv.Visible = true;

PanelDesactivar.Visible = true;

}

protected void BtnDesactivarProv\_Click(object sender, EventArgs e)

{

DesactivarProvClient clienteWCF = new DesactivarProvClient();

clienteWCF.Open();

DtoProveedor p = Session["Proveedor"] as DtoProveedor;

clienteWCF.DesactivarProveedor(p);

clienteWCF.Close();

PanelConfirmacion.Visible = true;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioGuardarCatalogo.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.master" AutoEventWireup="true" CodeBehind="ServicioGuardarCatalogo.aspx.cs" Inherits="AppWeb.ServicioGuardarCatalogo" %>

<asp:Content ID="Content2" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<h2>Guardar catálogo en archivo de texto</h2>

<asp:Button ID="BtnGuardarCatalogo" runat="server" Text="Guardar catálogo en archivo" OnClick="BtnGuardarCatalogo\_Click" />

<br><br>

<asp:Label ID="LblGuardado" runat="server" Text="El catálogo se guardó exitosamente.">

</asp:Label>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioGuardarCatalogo.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference5;

using Dominio;

namespace AppWeb

{

public partial class ServicioGuardarCatalogo : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

LblGuardado.Visible = false;

}

protected void BtnGuardarCatalogo\_Click(object sender, EventArgs e)

{

GuardarCatalogoTxtClient clienteWCF = new GuardarCatalogoTxtClient();

clienteWCF.Open();

clienteWCF.guardarCatalogoTxt();

LblGuardado.Visible = true;

clienteWCF.Close();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioGuardarProv.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.Master" AutoEventWireup="true" CodeBehind="ServicioGuardarProv.aspx.cs" Inherits="AppWeb.ServicioGuardarProv" %>

<asp:Content ID="Content2" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<h2>Guardar proveedores en archivo de texto</h2>

<asp:Button ID="BtnGuardarProveedores" runat="server" Text="Guardar proveedores en archivo" OnClick="BtnGuardarProveedores\_Click" />

<br><br>

<asp:Label ID="LblGuardado" runat="server" Text="Los datos de los proveedores se guardaron exitosamente.">

</asp:Label>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioGuardarProv.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference6;

using Dominio;

namespace AppWeb

{

public partial class ServicioGuardarProv : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

LblGuardado.Visible = false;

}

protected void BtnGuardarProveedores\_Click(object sender, EventArgs e)

{

GuardarProvTxtClient clienteWCF = new GuardarProvTxtClient();

clienteWCF.Open();

clienteWCF.guardarProveedoresTxt();

LblGuardado.Visible = true;

clienteWCF.Close();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioListadoProv.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.master" AutoEventWireup="true" CodeBehind="ServicioListadoProv.aspx.cs" Inherits="AppWeb.ServicioListadoProv" %>

<asp:Content ID="Content2" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<h2>Listado de proveedores</h2>

<asp:GridView ID="GVProveedores" runat="server" AutoGenerateColumns="False">

<Columns>

<asp:BoundField DataField="RUT" HeaderText="RUT" />

<asp:BoundField DataField="NombreFantasia" HeaderText="Nombre fantasía" />

<asp:BoundField DataField="Email" HeaderText="Correo Electrónico" />

<asp:BoundField DataField="Telefono" HeaderText="Telefono" />

<asp:BoundField DataField="FechaRegistro" HeaderText="Fecha de registro" />

<asp:BoundField DataField="EsInactivo" HeaderText="Es inactivo" />

<asp:BoundField DataField="Tipo" HeaderText="Tipo" />

</Columns>

</asp:GridView>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioListadoProv.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference4;

using Dominio;

namespace AppWeb

{

public partial class ServicioListadoProv : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

ServicioListaProvClient clienteWCF = new ServicioListaProvClient();

clienteWCF.Open();

GVProveedores.DataSource = clienteWCF.ObtenerProveedores();

GVProveedores.DataBind();

clienteWCF.Close();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioModificarParametros.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.master" AutoEventWireup="true" CodeBehind="ServicioModificarParametros.aspx.cs" Inherits="AppWeb.ServicioModificarParametros" %>

<asp:Content ID="Content2" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<h2>Modificar parámetros</h2>

<p>Valores actuales</p>

<asp:Label ID="LblArancel" runat="server" Text="Arancel actual"></asp:Label>

<asp:Label ID="LblValorArancel" runat="server"></asp:Label>

<br />

<asp:Label ID="LblPorcentaje" runat="server" Text="Porcentaje extra actual"></asp:Label>

<asp:Label ID="LblValorPorcentaje" runat="server"></asp:Label>

<br />

<br />

<p>Actualizar valores</p>

<asp:Label ID="LblNuevoArancel" runat="server" Text="Nuevo arancel"></asp:Label>

<asp:TextBox ID="TBArancel" runat="server"></asp:TextBox>

<asp:Button ID="BtnArancel" runat="server" Text="Actualizar arancel" OnClick="BtnArancel\_Click" />

<br />

<asp:Label ID="LblNuevoPorcentaje" runat="server" Text="Nuevo porcentaje extra"></asp:Label>

<asp:TextBox ID="TBPorcentaje" runat="server"></asp:TextBox>

<asp:Button ID="BtnPorcentaje" runat="server" Text="Actualizar porcentaje" OnClick="BtnPorcentaje\_Click" />

<br />

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioModificarParametros.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference7;

using Dominio;

namespace AppWeb

{

public partial class ServicioModificarParametros : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

ServicioModificarParametrosClient clienteWCF = new ServicioModificarParametrosClient();

clienteWCF.Open();

LblValorArancel.Text = clienteWCF.ObtenerArancel().ToString();

LblValorPorcentaje.Text = clienteWCF.ObtenerPorcentajeExtra().ToString();

clienteWCF.Close();

}

protected void BtnArancel\_Click(object sender, EventArgs e)

{

ServicioModificarParametrosClient clienteWCF = new ServicioModificarParametrosClient();

clienteWCF.Open();

string arancelStr = TBArancel.Text;

double arancel = Convert.ToDouble(arancelStr);

clienteWCF.ModificarArancel(arancel);

LblValorArancel.Text = clienteWCF.ObtenerArancel().ToString();

clienteWCF.Close();

}

protected void BtnPorcentaje\_Click(object sender, EventArgs e)

{

ServicioModificarParametrosClient clienteWCF = new ServicioModificarParametrosClient();

clienteWCF.Open();

string porcentajeExtraStr = TBPorcentaje.Text;

int porcentajeExtra = Convert.ToInt32(porcentajeExtraStr);

clienteWCF.ModificarPorcentajeExtra(porcentajeExtra);

LblValorPorcentaje.Text = clienteWCF.ObtenerPorcentajeExtra().ToString();

clienteWCF.Close();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioMostrarCatalogo.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Administrador/Administrador.master" AutoEventWireup="true" CodeBehind="~/Administrador/ServicioMostrarCatalogo.aspx.cs" Inherits="AppWeb.ServicioMostrarCatalogo" %>

<asp:Content ID="Content2" ContentPlaceHolderID="ContenidoAdmin" runat="server">

<h2>Catálogo de servicios</h2>

<asp:GridView ID="GVServicios" CssClass="grid\_View\_Style\_1" PagerStyle-CssClass="grid\_1\_pager"

HeaderStyle-CssClass="grid\_1\_header" RowStyle-CssClass="grid\_1\_rows" runat="server" AutoGenerateColumns="False" OnRowCommand="GVServicios\_RowCommand">

<Columns>

<asp:BoundField DataField="Servicio" HeaderText="Servicio" />

<asp:BoundField DataField="Descripcion" HeaderText="Descripción" />

<asp:TemplateField HeaderText="Tipo de evento">

<ItemTemplate>

<asp:Button ID="BtnTipoEvento" runat="server" CommandName="VerTipoEvento"

CommandArgument="<%# ((GridViewRow) Container).RowIndex %>"

Text="Ver tipos de eventos" />

</ItemTemplate>

</asp:TemplateField>

</Columns>

</asp:GridView>

<asp:Panel ID="PanelTipoEvento" runat="server">

<asp:Label ID="LBlServicio" runat="server"></asp:Label><br />

<asp:Label ID="LBlDescripcion" runat="server"></asp:Label><br />

<asp:Label ID="LBlEventos" runat="server"></asp:Label>

</asp:Panel>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioMostrarCatalogo.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference3;

using Dominio;

namespace AppWeb

{

public partial class ServicioMostrarCatalogo : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!Page.IsPostBack)

{

Inicializar();

}

if (Session["Servicio"] == null)

Session["Servicio"] = new DtoServicio();

}

private void Inicializar()

{

CatalogoServiciosClient clienteWCF = new CatalogoServiciosClient();

clienteWCF.Open();

GVServicios.DataSource = clienteWCF.ObtenerServicios();

GVServicios.DataBind();

clienteWCF.Close();

PanelTipoEvento.Visible = false;

}

protected void GVServicios\_RowCommand(object sender, GridViewCommandEventArgs e)

{

int fila = int.Parse(e.CommandArgument + "");

int id = fila + 1;

CatalogoServiciosClient clienteWCF = new CatalogoServiciosClient();

clienteWCF.Open();

IEnumerable<DtoServicio> servicios = clienteWCF.ObtenerServicios();

DtoServicio miServicio = clienteWCF.BuscarServicio(id);

List<DtoTipoEvento> eventos = new List<DtoTipoEvento>();

string tipos = "";

if (e.CommandName == "VerTipoEvento")

{

PanelTipoEvento.Visible = true;

LBlServicio.Text = "Servicio :" + miServicio.Servicio;

LBlDescripcion.Text = "Descripción :" + miServicio.Descripcion;

foreach (DtoTipoEvento t in miServicio.misTiposEventos) {

tipos += t.Nombre+"</br> ";

}

LBlEventos.Text = "Tipos de eventos: "+tipos;

}

clienteWCF.Close();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioRegProveedores.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Sitio.Master" AutoEventWireup="true" CodeBehind="ServicioRegProveedores.aspx.cs" Inherits="AppWeb.Formulario\_web1" %>

<asp:Content ID="Content1" ContentPlaceHolderID="HeadSitio" runat="server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="PlaceSitio" runat="server">

<div class="page-regprov">

<asp:Panel ID="PanelRegistroProv" runat="server">

<div id="wpaltaproveedor">

<h1 class="main-title">REGISTRO DE PROVEEDOR</h1>

<h2 class="main-title">Datos del proveedor</h2>

<asp:Panel ID="PanelRUT" runat="server">

<asp:Label ID="LblRut" runat="server" Text="Rut: "></asp:Label>

<asp:TextBox ID="TxtRut" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtRut" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator4" runat="server" ForeColor="Tomato" ControlToValidate="TxtRut" ErrorMessage="Debe ser un numero de 12 digitos" ValidationExpression="^([0-9]{12})$"></asp:RegularExpressionValidator>

<asp:Label ID="ErrorRut" runat="server" Text="" ForeColor="Tomato"></asp:Label>

<br />

</asp:Panel>

<asp:Panel ID="PanelNombre" runat="server">

<asp:Label ID="LblNombre" runat="server" Text="Nombre Fantasía: "></asp:Label>

<asp:TextBox ID="TxtNomFantasia" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtNomFantasia" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server" ForeColor="Tomato" ControlToValidate="TxtNomFantasia" ErrorMessage="Mínimo 3 caracteres y máximo 50" ValidationExpression="^[a-zA-Z](\s?[a-zA-Z]){3,50}$"></asp:RegularExpressionValidator>

<br />

</asp:Panel>

<asp:Panel ID="PanelEMail" runat="server">

<asp:Label ID="LblEMail" runat="server" Text="Email: "></asp:Label>

<asp:TextBox ID="TxtEmail" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtEmail" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator2" runat="server" ForeColor="Tomato" ControlToValidate="TxtEmail" ErrorMessage="El formato de Email ingresado no es válido" ValidationExpression="^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$"></asp:RegularExpressionValidator>

<br />

</asp:Panel>

<asp:Panel ID="PanelTelefono" runat="server">

<asp:Label ID="LblTelefono" runat="server" Text="Teléfono: "></asp:Label>

<asp:TextBox ID="TxtTel" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtTel" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator3" runat="server" ForeColor="Tomato" ControlToValidate="TxtTel" ErrorMessage="Solo numeros o caracteristica Uruguay - Formatos aceptados: XXXXXXX ó +598 XXXXXXX" ValidationExpression="(^[0-9]{8,9}$)|(^\+[0-9]{3}\s+[0-9]{2}\s+[0-9]{6}$)|(^\+[0-9]{3}\s+[0-9]{8,9}$)"></asp:RegularExpressionValidator>

<br />

</asp:Panel>

<asp:Panel ID="PanelPass" runat="server">

<asp:Label ID="LblPass" runat="server" Text="Contraseña para su Usuario: "></asp:Label>

<asp:TextBox ID="TxtPass" runat="server" TextMode="Password"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtPass" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator5" runat="server" ForeColor="Tomato" ControlToValidate="TxtPass" ErrorMessage="Su contraseña debe tener un largo mínimo de 6, sin espacios y contener por lo menos una mayúscula" ValidationExpression="^(?=.\*?[A-Z])(?=.\*?[a-z]).{6,}$"></asp:RegularExpressionValidator>

<br />

</asp:Panel>

<asp:Panel ID="PanelVIP" runat="server">

<asp:CheckBox ID="CheckBoxVip" runat="server" Text="Proveedor Vip" />

<br />

</asp:Panel>

<asp:Label ID="LblAsignacion" runat="server" Text=""></asp:Label>

<br />

<br />

<h2>Servicios Ofrecidos</h2>

<asp:Label ID="LblServicios" runat="server" Text="Seleccione el tipo de servicio:"></asp:Label>

<br />

<asp:DropDownList ID="ListBoxServicios" runat="server"></asp:DropDownList>

<br />

<asp:Label ID="LblDescripcion" runat="server" Text="Describa el servicio ofrecido:"></asp:Label>

<br />

<asp:TextBox ID="TBDescripcion" runat="server"></asp:TextBox>

<br />

<asp:Label ID="LblFoto" runat="server" Text="Agregue una foto al servicio:"></asp:Label>

<br />

<asp:FileUpload ID="ServFotoUpload" runat="server" />

<br />

<br />

<asp:Button ID="BtnAgregarServicio" runat="server" Text="AgregarServicio" CssClass="boton\_personalizado" OnClick="BtnAsigServAccion\_Click"/>

<br />

<br />

<asp:Button ID="BtnRegistroProv" CssClass="boton\_personalizado" runat="server" Text="Registrar Proveedor" OnClick="BtnRegistroProv\_Click" />

<asp:Label ID="LBAsignacionReg" runat="server" Text=""></asp:Label>

</div>

</asp:Panel>

</div>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioRegProveedores.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using AppWeb.ServiceReference9;

using System.IO;

namespace AppWeb

{

public partial class Formulario\_web1 : System.Web.UI.Page

{

List<DtoServicioProveedor> misDtosServicios = new List<DtoServicioProveedor>();

protected void Page\_Load(object sender, EventArgs e)

{

AgregarProvClient clienteWCF = new AgregarProvClient();

clienteWCF.Open();

ListBoxServicios.DataSource = clienteWCF.CargarServicios();

ListBoxServicios.DataBind();

clienteWCF.Close();

}

protected void BtnRegistroProv\_Click(object sender, EventArgs e)

{

AgregarProvClient clienteWCF = new AgregarProvClient();

clienteWCF.Open();

LblAsignacion.Text = "";

string rut = TxtRut.Text;

string nomFant = TxtNomFantasia.Text;

string email = TxtEmail.Text;

string tel = TxtTel.Text;

string pass = TxtPass.Text;

string tipo = "";

bool esInactivo = false;

LblAsignacion.Text = "";

if (CheckBoxVip.Checked)

{ tipo = "VIP"; }

else { tipo = "COMUN"; }

DateTime fechaRegDateTime = DateTime.Now;

string fechaRegistro = fechaRegDateTime.ToString("yyyy-MM-dd");

if (tipo == "COMUN")

{

DtoProveedor p = new DtoProveedor { RUT = rut, NombreFantasia = nomFant, Email = email, Telefono = tel, FechaRegistro = fechaRegistro, esInactivo = false, Tipo = tipo };

if (clienteWCF.ExisteRUT(p.RUT))

{

LblAsignacion.Text = "Ya existe un proveedor con el RUT ingresado.";

}

else if (clienteWCF.ExisteMail(p.Email))

{

LblAsignacion.Text = "Ya existe un proveedor con el email ingresado.";

}

else

{

bool esVip = false;

clienteWCF.InsertarProveedor(rut, nomFant, email, tel, esInactivo, esVip, pass);

foreach (DtoServicioProveedor s in misDtosServicios) {

clienteWCF.AgregarServicioAProveedor(s,p);

}

LblAsignacion.Text = "";

}

Session["prov"] = p as DtoProveedor;

}

else if (tipo == "VIP")

{

DtoProveedor p = new DtoProveedor { RUT = rut, NombreFantasia = nomFant, Email = email, Telefono = tel, FechaRegistro = fechaRegistro, esInactivo = false, Tipo = tipo };

if (clienteWCF.ExisteRUT(p.RUT))

{

LblAsignacion.Text = "Ya existe un proveedor con el RUT ingresado.";

}

else if (clienteWCF.ExisteMail(p.Email))

{

LblAsignacion.Text = "Ya existe un proveedor con el email ingresado.";

}

else

{

bool esVip = true;

clienteWCF.InsertarProveedor(rut, nomFant, email, tel, esInactivo, esVip, pass);

foreach (DtoServicioProveedor s in misDtosServicios)

{

clienteWCF.AgregarServicioAProveedor(s, p);

}

LblAsignacion.Text = "";

}

Session["prov"] = p as DtoProveedor;

}

clienteWCF.Close();

}

private void limpiarForm()

{

TxtRut.Text = "";

TxtNomFantasia.Text = "";

TxtEmail.Text = "";

TxtTel.Text = "";

TxtPass.Text = "";

CheckBoxVip.Checked = false;

ListBoxServicios.Items.Clear();

ListBoxServicios.DataSource = null;

ListBoxServicios.DataBind();

}

protected void BtnAsigServAccion\_Click(object sender, EventArgs e)

{

AgregarProvClient clienteWCF = new AgregarProvClient();

clienteWCF.Open();

DtoProveedor miProveedor = Session["prov"] as DtoProveedor;

DtoServicioProveedor servProv = new DtoServicioProveedor();

string mitmpServ = ListBoxServicios.SelectedValue;

servProv.IdServicio = ListBoxServicios.SelectedIndex;

servProv.Nombre = ListBoxServicios.DataTextField;

servProv.Descripcion = TBDescripcion.Text;

string ruta = Server.MapPath("~/images/servicios-proveedor/");

if (ServFotoUpload.HasFile)

{

string[] partesNombre = ServFotoUpload.FileName.Split('.');

string extension = partesNombre[partesNombre.Length - 1];

string nombreArch = string.Format("Rut\_{0}\_\_ServicioID\_{1}.{2}", servProv.RutProveedor, servProv.IdServicio, extension);

if (!Directory.Exists(ruta))

Directory.CreateDirectory(ruta);

ServFotoUpload.SaveAs(ruta + nombreArch);

servProv.Foto = "~/images/servicios-proveedor/" + nombreArch;

}

misDtosServicios.Add(servProv);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: WFRegProveedores.aspx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<%@ Page Title="" Language="C#" MasterPageFile="~/Sitio.Master" AutoEventWireup="true" CodeBehind="WFRegProveedores.aspx.cs" Inherits="AppWeb.WFRegProveedores" %>

<asp:Content ID="Content1" ContentPlaceHolderID="HeadSitio" runat="server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="PlaceSitio" runat="server">

<div class="page-regprov">

<asp:Panel ID="Paso1AltaProv" runat="server">

<div id="wpaltaproveedor">

<h1 class="main-title">PASO 1 - DATOS DE PROVEEDOR</h1>

<h1 class="main-title">Registro de Proveedores</h1>

<asp:Panel ID="Panel2" runat="server">

<asp:Label ID="Label4" runat="server" Text="Rut: "></asp:Label>

<asp:TextBox ID="TxtRut" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtRut" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator4" runat="server" ForeColor="Tomato" ControlToValidate="TxtRut" ErrorMessage="Debe ser un numero de 12 digitos" ValidationExpression="^([0-9]{12})$"></asp:RegularExpressionValidator>

<asp:Label ID="ErrorRut" runat="server" Text="" ForeColor="Tomato"></asp:Label>

</asp:Panel>

<asp:Panel ID="Panel3" runat="server">

<asp:Label ID="Label1" runat="server" Text="Nombre Fantasía: "></asp:Label>

<asp:TextBox ID="TxtNomFantasia" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtNomFantasia" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server" ForeColor="Tomato" ControlToValidate="TxtNomFantasia" ErrorMessage="Mínimo 3 caracteres y máximo 50" ValidationExpression="^[a-zA-Z](\s?[a-zA-Z]){3,50}$"></asp:RegularExpressionValidator>

</asp:Panel>

<asp:Panel ID="Panel1" runat="server">

<asp:Label ID="Label2" runat="server" Text="Email: "></asp:Label>

<asp:TextBox ID="TxtEmail" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtEmail" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator2" runat="server" ForeColor="Tomato" ControlToValidate="TxtEmail" ErrorMessage="El formato de Email ingresado no es válido" ValidationExpression="^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$"></asp:RegularExpressionValidator>

</asp:Panel>

<asp:Panel ID="Panel4" runat="server">

<asp:Label ID="Label3" runat="server" Text="Teléfono: "></asp:Label>

<asp:TextBox ID="TxtTel" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtTel" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator3" runat="server" ForeColor="Tomato" ControlToValidate="TxtTel" ErrorMessage="Solo numeros o caracteristica Uruguay - Formatos aceptados: XXXXXXX ó +598 XXXXXXX" ValidationExpression="(^[0-9]{8,9}$)|(^\+[0-9]{3}\s+[0-9]{2}\s+[0-9]{6}$)|(^\+[0-9]{3}\s+[0-9]{8,9}$)"></asp:RegularExpressionValidator>

</asp:Panel>

<asp:Panel ID="Panel5" runat="server">

<asp:Label ID="Label5" runat="server" Text="Contraseña para su Usuario: "></asp:Label>

<asp:TextBox ID="TxtPass" runat="server" TextMode="Password"></asp:TextBox>

<asp:RequiredFieldValidator runat="server" ControlToValidate="TxtPass" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator5" runat="server" ForeColor="Tomato" ControlToValidate="TxtPass" ErrorMessage="Su contraseña debe tener un largo mínimo de 6, sin espacios y contener por lo menos una mayúscula" ValidationExpression="^(?=.\*?[A-Z])(?=.\*?[a-z]).{6,}$"></asp:RegularExpressionValidator>

</asp:Panel>

<br />

<asp:Panel ID="Panel6" runat="server">

<asp:CheckBox ID="CheckBoxVip" runat="server" Text="Proveedor Vip" />

</asp:Panel>

<br />

<asp:Button ID="BtnAccion" CssClass="boton\_personalizado" runat="server" Text="Ir a Paso 2 - Seleccion de Servicios" OnClick="BtnAccion\_Click" />

<br />

<br />

<asp:Label ID="Asignacion" runat="server" Text=""></asp:Label>

</div>

</asp:Panel>

<asp:Panel ID="Paso2ServProv" runat="server" Visible="false">

<div id="regprov-right">

<h2>Servicios Ofrecidos</h2>

<asp:ListBox ID="ListBoxServicios" runat="server"></asp:ListBox><br />

<br />

<h1 class="main-title">Seleccionar los Servicios ofrecidos</h1>

<asp:GridView ID="GridViewListadoServicios" CssClass="grid\_View\_Style\_1" PagerStyle-CssClass="grid\_1\_pager"

HeaderStyle-CssClass="grid\_1\_header" RowStyle-CssClass="grid\_1\_rows" runat="server" AutoGenerateColumns="False" OnRowCommand="GridServicios\_RowCommand">

<Columns>

<asp:BoundField DataField="Nombre" HeaderText="Nombre" />

<asp:BoundField DataField="Descripcion" HeaderText="Descripcion" />

<asp:ButtonField ButtonType="Link" CommandName="AgregarServicio" Text="Agregar Servicio" />

</Columns>

<SelectedRowStyle CssClass="grid\_1\_selectedrow" />

</asp:GridView>

<asp:panel id="PanelCantServicios" runat="server" Visible="false">

<asp:Label ID="Label7" runat="server" Text="No hay Proveedores registrados en el sistema."></asp:Label>

</asp:panel>

<asp:Panel ID="PanelAsignarServicio" runat="server" Visible="false">

<h2>Asignación de Servicio</h2>

<asp:HiddenField ID="HiddeIdServicio" runat="server" />

<asp:TextBox ID="ServNombre" runat="server" ReadOnly="true"></asp:TextBox><br />

<asp:TextBox ID="ServDesc" runat="server"></asp:TextBox><br />

<asp:RequiredFieldValidator runat="server" ControlToValidate="ServDesc" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator6" runat="server" ForeColor="Tomato" ControlToValidate="ServDesc" ErrorMessage="Mínimo 3 caracteres y máximo 50" ValidationExpression="^[a-zA-Z](\s?[a-zA-Z]){3,50}$"></asp:RegularExpressionValidator>

<asp:FileUpload ID="ServFotoUpload" runat="server" /><br />

<asp:RequiredFieldValidator runat="server" ControlToValidate="ServFotoUpload" ErrorMessage="\*" ForeColor="#FF0000"></asp:RequiredFieldValidator>

<br />

<asp:Button ID="BtnAsigServ" CssClass="boton\_personalizado" runat="server" Text="Asignar Servicio" OnClick="BtnAsigServAccion\_Click" />

</asp:Panel>

</div>

<asp:Button ID="BtnRegistroProv" CssClass="boton\_personalizado" runat="server" Text="Registrar Proveedor" OnClick="BtnRegistroProv\_Click" />

<asp:Label ID="LBAsignacionReg" runat="server" Text=""></asp:Label>

</asp:Panel>

</div>

</asp:Content>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: WFRegProveedores.aspx.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using Dominio;

using System.IO;

namespace AppWeb

{

public partial class WFRegProveedores : System.Web.UI.Page

{

private static List<ServicioProveedor> ListaMiServicios;

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void BtnAccion\_Click(object sender, EventArgs e)

{

Asignacion.Text = "";

string rut = TxtRut.Text;

string nomFant = TxtNomFantasia.Text;

string email = TxtEmail.Text;

string tel = TxtTel.Text;

string pass = TxtPass.Text;

string tipo = "";

Asignacion.Text = "";

if (CheckBoxVip.Checked)

{ tipo = "VIP"; }

else { tipo = "COMUN"; }

DateTime fechaRegDateTime = DateTime.Now;

string fechaRegistro = fechaRegDateTime.ToString("yyyy-MM-dd");

if (tipo == "COMUN")

{

Proveedor p = new ProveedorComun { RUT = rut, NombreFantasia = nomFant, Email = email, Telefono = tel, FechaRegistro = fechaRegistro, esInactivo = false, Tipo = tipo};

if (Proveedor.ExisteRut(p.RUT))

{

Asignacion.Text = "Ya existe un proveedor con el RUT ingresado.";

}

else if (Proveedor.ExisteEmail(p.Email))

{

Asignacion.Text = "Ya existe un proveedor con el email ingresado.";

}

else

{

Asignacion.Text = "";

Session["ProvINSessionPass"] = pass;

Session["ProvINSession"] = p;

Paso1AltaProv.Visible = false;

Paso2ServProv.Visible = true;

cargarServicios();

if (ListaMiServicios == null)

{

ListaMiServicios = new List<ServicioProveedor>();

}

}

}

else if (tipo == "VIP")

{

Proveedor p = new ProveedorVIP { RUT = rut, NombreFantasia = nomFant, Email = email, Telefono = tel, FechaRegistro = fechaRegistro, esInactivo = false, Tipo = tipo};

if (Proveedor.ExisteRut(p.RUT))

{

Asignacion.Text = "Ya existe un proveedor con el RUT ingresado.";

}

else if (Proveedor.ExisteEmail(p.Email))

{

Asignacion.Text = "Ya existe un proveedor con el email ingresado.";

}

else

{

Asignacion.Text = "";

Session["ProvINSessionPass"] = pass;

Session["ProvINSession"] = p;

Paso1AltaProv.Visible = false;

Paso2ServProv.Visible = true;

cargarServicios();

if (ListaMiServicios == null)

{

ListaMiServicios = new List<ServicioProveedor>();

}

}

}

}

private void insertarProveedor(Proveedor p, string pass)

{

// Verificaciones de Rut y Email OK

Asignacion.Text = "";

string passEncriptada = Usuario.EncriptarPassSHA512(pass);

Usuario usu = new Usuario { User = p.RUT, Passw = passEncriptada, Rol = 2, Email = p.Email };

p.AgregarUsuario(usu);

p.ListaServicios = ListaMiServicios;

if (p.Insertar())

{

Asignacion.Text = "Insertaste a : " + p.RUT;

limpiarForm();

}

else

Asignacion.Text = "No";

}

private void limpiarForm()

{

TxtRut.Text = "";

TxtNomFantasia.Text = "";

TxtEmail.Text = "";

TxtTel.Text = "";

TxtPass.Text = "";

CheckBoxVip.Checked = false;

ListaMiServicios.Clear();

ListBoxServicios.Items.Clear();

ListBoxServicios.DataSource = null;

ListBoxServicios.DataBind();

}

private void cargarServicios()

{

List<Servicio> listaServicios = Servicio.FindAll();

if (listaServicios == null || listaServicios.Count == 0)

{

PanelCantServicios.Visible = true;

}

else

{

PanelCantServicios.Visible = false;

GridViewListadoServicios.DataSource = listaServicios;

GridViewListadoServicios.DataBind();

}

}

protected void GridServicios\_RowCommand(object sender, GridViewCommandEventArgs e)

{

List<Servicio> listaServicios = Servicio.FindAll();

int fila = int.Parse(e.CommandArgument + "");

if (e.CommandName == "AgregarServicio")

{

PanelAsignarServicio.Visible = true;

Servicio serv = new Servicio();

serv = listaServicios[fila];

HiddeIdServicio.Value = serv.IdServicio.ToString();

if (serv != null) { ServNombre.Text = serv.Nombre; }

}

}

protected void BtnAsigServAccion\_Click(object sender, EventArgs e)

{

ServicioProveedor servProv = new ServicioProveedor();

Proveedor miProveedor = Session["ProvINSession"] as Proveedor;

servProv.Nombre = ServNombre.Text;

servProv.IdServicio = int.Parse(HiddeIdServicio.Value);

servProv.RutProveedor = miProveedor.RUT;

servProv.Descripcion = ServDesc.Text;

string ruta = Server.MapPath("~/images/servicios-proveedor/");

if (ServFotoUpload.HasFile)

{

string[] partesNombre = ServFotoUpload.FileName.Split('.');

string extension = partesNombre[partesNombre.Length - 1];

string nombreArch = string.Format("Rut\_{0}\_\_ServicioID\_{1}.{2}", servProv.RutProveedor, servProv.IdServicio, extension);

if (!Directory.Exists(ruta))

Directory.CreateDirectory(ruta);

ServFotoUpload.SaveAs(ruta + nombreArch);

servProv.Foto = "~/images/servicios-proveedor/" + nombreArch;

}

if (ListaMiServicios.Contains(servProv))

{

Asignacion.Text = "Servicio ya agregado";

}

else

{

Asignacion.Text = "";

ListaMiServicios.Add(servProv);

ServNombre.Text = "";

ServDesc.Text = "";

PanelAsignarServicio.Visible = false;

}

if (ListaMiServicios.Count() != 0)

{

ListBoxServicios.DataSource = ListaMiServicios;

ListBoxServicios.DataBind();

}

}

protected void BtnRegistroProv\_Click(object sender, EventArgs e)

{

if (ListaMiServicios.Count() == 0)

{

LBAsignacionReg.Text = "No se puede agregar un Proveedor sin Servicios asociados";

}else

{

Paso1AltaProv.Visible = true;

Paso2ServProv.Visible = false;

Proveedor p = Session["ProvINSession"] as Proveedor;

string pass = Session["ProvINSessionPass"].ToString();

insertarProveedor(p, pass);

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference1 {

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference1.IProveedorDadoRUT")]

public interface IProveedorDadoRUT {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IProveedorDadoRUT/BuscarProveedorRut", ReplyAction="http://tempuri.org/IProveedorDadoRUT/BuscarProveedorRutResponse")]

WCFProveedorDadoRUT.DtoProveedor BuscarProveedorRut(string rut);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IProveedorDadoRUT/BuscarProveedorRut", ReplyAction="http://tempuri.org/IProveedorDadoRUT/BuscarProveedorRutResponse")]

System.Threading.Tasks.Task<WCFProveedorDadoRUT.DtoProveedor> BuscarProveedorRutAsync(string rut);

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IProveedorDadoRUTChannel : AppWeb.ServiceReference1.IProveedorDadoRUT, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class ProveedorDadoRUTClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference1.IProveedorDadoRUT>, AppWeb.ServiceReference1.IProveedorDadoRUT {

public ProveedorDadoRUTClient() {

}

public ProveedorDadoRUTClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public ProveedorDadoRUTClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public ProveedorDadoRUTClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public ProveedorDadoRUTClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public WCFProveedorDadoRUT.DtoProveedor BuscarProveedorRut(string rut) {

return base.Channel.BuscarProveedorRut(rut);

}

public System.Threading.Tasks.Task<WCFProveedorDadoRUT.DtoProveedor> BuscarProveedorRutAsync(string rut) {

return base.Channel.BuscarProveedorRutAsync(rut);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference2 {

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference2.IAgregarProv")]

public interface IAgregarProv {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/InsertarProveedor", ReplyAction="http://tempuri.org/IAgregarProv/InsertarProveedorResponse")]

bool InsertarProveedor(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/InsertarProveedor", ReplyAction="http://tempuri.org/IAgregarProv/InsertarProveedorResponse")]

System.Threading.Tasks.Task<bool> InsertarProveedorAsync(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass);

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IAgregarProvChannel : AppWeb.ServiceReference2.IAgregarProv, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class AgregarProvClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference2.IAgregarProv>, AppWeb.ServiceReference2.IAgregarProv {

public AgregarProvClient() {

}

public AgregarProvClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public AgregarProvClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public AgregarProvClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public AgregarProvClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public bool InsertarProveedor(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass) {

return base.Channel.InsertarProveedor(rut, nombreFantasia, email, tel, esInactivo, esVip, pass);

}

public System.Threading.Tasks.Task<bool> InsertarProveedorAsync(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass) {

return base.Channel.InsertarProveedorAsync(rut, nombreFantasia, email, tel, esInactivo, esVip, pass);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference3 {

using System.Runtime.Serialization;

using System;

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoServicio", Namespace="http://schemas.datacontract.org/2004/07/WCFCatalogoServicios")]

[System.SerializableAttribute()]

public partial class DtoServicio : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string DescripcionField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private int IdServicioField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string ServicioField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private AppWeb.ServiceReference3.DtoTipoEvento[] misTiposEventosField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Descripcion {

get {

return this.DescripcionField;

}

set {

if ((object.ReferenceEquals(this.DescripcionField, value) != true)) {

this.DescripcionField = value;

this.RaisePropertyChanged("Descripcion");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public int IdServicio {

get {

return this.IdServicioField;

}

set {

if ((this.IdServicioField.Equals(value) != true)) {

this.IdServicioField = value;

this.RaisePropertyChanged("IdServicio");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Servicio {

get {

return this.ServicioField;

}

set {

if ((object.ReferenceEquals(this.ServicioField, value) != true)) {

this.ServicioField = value;

this.RaisePropertyChanged("Servicio");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public AppWeb.ServiceReference3.DtoTipoEvento[] misTiposEventos {

get {

return this.misTiposEventosField;

}

set {

if ((object.ReferenceEquals(this.misTiposEventosField, value) != true)) {

this.misTiposEventosField = value;

this.RaisePropertyChanged("misTiposEventos");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoTipoEvento", Namespace="http://schemas.datacontract.org/2004/07/WCFCatalogoServicios")]

[System.SerializableAttribute()]

public partial class DtoTipoEvento : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string DescripcionField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Descripcion {

get {

return this.DescripcionField;

}

set {

if ((object.ReferenceEquals(this.DescripcionField, value) != true)) {

this.DescripcionField = value;

this.RaisePropertyChanged("Descripcion");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Nombre {

get {

return this.NombreField;

}

set {

if ((object.ReferenceEquals(this.NombreField, value) != true)) {

this.NombreField = value;

this.RaisePropertyChanged("Nombre");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference3.ICatalogoServicios")]

public interface ICatalogoServicios {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/ICatalogoServicios/ObtenerServicios", ReplyAction="http://tempuri.org/ICatalogoServicios/ObtenerServiciosResponse")]

AppWeb.ServiceReference3.DtoServicio[] ObtenerServicios();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/ICatalogoServicios/ObtenerServicios", ReplyAction="http://tempuri.org/ICatalogoServicios/ObtenerServiciosResponse")]

System.Threading.Tasks.Task<AppWeb.ServiceReference3.DtoServicio[]> ObtenerServiciosAsync();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/ICatalogoServicios/BuscarServicio", ReplyAction="http://tempuri.org/ICatalogoServicios/BuscarServicioResponse")]

AppWeb.ServiceReference3.DtoServicio BuscarServicio(int fila);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/ICatalogoServicios/BuscarServicio", ReplyAction="http://tempuri.org/ICatalogoServicios/BuscarServicioResponse")]

System.Threading.Tasks.Task<AppWeb.ServiceReference3.DtoServicio> BuscarServicioAsync(int fila);

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface ICatalogoServiciosChannel : AppWeb.ServiceReference3.ICatalogoServicios, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class CatalogoServiciosClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference3.ICatalogoServicios>, AppWeb.ServiceReference3.ICatalogoServicios {

public CatalogoServiciosClient() {

}

public CatalogoServiciosClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public CatalogoServiciosClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public CatalogoServiciosClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public CatalogoServiciosClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public AppWeb.ServiceReference3.DtoServicio[] ObtenerServicios() {

return base.Channel.ObtenerServicios();

}

public System.Threading.Tasks.Task<AppWeb.ServiceReference3.DtoServicio[]> ObtenerServiciosAsync() {

return base.Channel.ObtenerServiciosAsync();

}

public AppWeb.ServiceReference3.DtoServicio BuscarServicio(int fila) {

return base.Channel.BuscarServicio(fila);

}

public System.Threading.Tasks.Task<AppWeb.ServiceReference3.DtoServicio> BuscarServicioAsync(int fila) {

return base.Channel.BuscarServicioAsync(fila);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference4 {

using System.Runtime.Serialization;

using System;

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoProveedor", Namespace="http://schemas.datacontract.org/2004/07/WcfListaProv")]

[System.SerializableAttribute()]

public partial class DtoProveedor : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string EmailField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string FechaRegistroField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreFantasiaField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string RUTField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string TelefonoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string TipoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private bool esInactivoField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Email {

get {

return this.EmailField;

}

set {

if ((object.ReferenceEquals(this.EmailField, value) != true)) {

this.EmailField = value;

this.RaisePropertyChanged("Email");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string FechaRegistro {

get {

return this.FechaRegistroField;

}

set {

if ((object.ReferenceEquals(this.FechaRegistroField, value) != true)) {

this.FechaRegistroField = value;

this.RaisePropertyChanged("FechaRegistro");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string NombreFantasia {

get {

return this.NombreFantasiaField;

}

set {

if ((object.ReferenceEquals(this.NombreFantasiaField, value) != true)) {

this.NombreFantasiaField = value;

this.RaisePropertyChanged("NombreFantasia");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string RUT {

get {

return this.RUTField;

}

set {

if ((object.ReferenceEquals(this.RUTField, value) != true)) {

this.RUTField = value;

this.RaisePropertyChanged("RUT");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Telefono {

get {

return this.TelefonoField;

}

set {

if ((object.ReferenceEquals(this.TelefonoField, value) != true)) {

this.TelefonoField = value;

this.RaisePropertyChanged("Telefono");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Tipo {

get {

return this.TipoField;

}

set {

if ((object.ReferenceEquals(this.TipoField, value) != true)) {

this.TipoField = value;

this.RaisePropertyChanged("Tipo");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public bool esInactivo {

get {

return this.esInactivoField;

}

set {

if ((this.esInactivoField.Equals(value) != true)) {

this.esInactivoField = value;

this.RaisePropertyChanged("esInactivo");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference4.IServicioListaProv")]

public interface IServicioListaProv {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioListaProv/ObtenerProveedores", ReplyAction="http://tempuri.org/IServicioListaProv/ObtenerProveedoresResponse")]

AppWeb.ServiceReference4.DtoProveedor[] ObtenerProveedores();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioListaProv/ObtenerProveedores", ReplyAction="http://tempuri.org/IServicioListaProv/ObtenerProveedoresResponse")]

System.Threading.Tasks.Task<AppWeb.ServiceReference4.DtoProveedor[]> ObtenerProveedoresAsync();

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IServicioListaProvChannel : AppWeb.ServiceReference4.IServicioListaProv, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class ServicioListaProvClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference4.IServicioListaProv>, AppWeb.ServiceReference4.IServicioListaProv {

public ServicioListaProvClient() {

}

public ServicioListaProvClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public ServicioListaProvClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public ServicioListaProvClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public ServicioListaProvClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public AppWeb.ServiceReference4.DtoProveedor[] ObtenerProveedores() {

return base.Channel.ObtenerProveedores();

}

public System.Threading.Tasks.Task<AppWeb.ServiceReference4.DtoProveedor[]> ObtenerProveedoresAsync() {

return base.Channel.ObtenerProveedoresAsync();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference5 {

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference5.IGuardarCatalogoTxt")]

public interface IGuardarCatalogoTxt {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IGuardarCatalogoTxt/guardarCatalogoTxt", ReplyAction="http://tempuri.org/IGuardarCatalogoTxt/guardarCatalogoTxtResponse")]

string guardarCatalogoTxt();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IGuardarCatalogoTxt/guardarCatalogoTxt", ReplyAction="http://tempuri.org/IGuardarCatalogoTxt/guardarCatalogoTxtResponse")]

System.Threading.Tasks.Task<string> guardarCatalogoTxtAsync();

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IGuardarCatalogoTxtChannel : AppWeb.ServiceReference5.IGuardarCatalogoTxt, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class GuardarCatalogoTxtClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference5.IGuardarCatalogoTxt>, AppWeb.ServiceReference5.IGuardarCatalogoTxt {

public GuardarCatalogoTxtClient() {

}

public GuardarCatalogoTxtClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public GuardarCatalogoTxtClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public GuardarCatalogoTxtClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public GuardarCatalogoTxtClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public string guardarCatalogoTxt() {

return base.Channel.guardarCatalogoTxt();

}

public System.Threading.Tasks.Task<string> guardarCatalogoTxtAsync() {

return base.Channel.guardarCatalogoTxtAsync();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference6 {

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference6.IGuardarProvTxt")]

public interface IGuardarProvTxt {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IGuardarProvTxt/guardarProveedoresTxt", ReplyAction="http://tempuri.org/IGuardarProvTxt/guardarProveedoresTxtResponse")]

string guardarProveedoresTxt();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IGuardarProvTxt/guardarProveedoresTxt", ReplyAction="http://tempuri.org/IGuardarProvTxt/guardarProveedoresTxtResponse")]

System.Threading.Tasks.Task<string> guardarProveedoresTxtAsync();

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IGuardarProvTxtChannel : AppWeb.ServiceReference6.IGuardarProvTxt, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class GuardarProvTxtClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference6.IGuardarProvTxt>, AppWeb.ServiceReference6.IGuardarProvTxt {

public GuardarProvTxtClient() {

}

public GuardarProvTxtClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public GuardarProvTxtClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public GuardarProvTxtClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public GuardarProvTxtClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public string guardarProveedoresTxt() {

return base.Channel.guardarProveedoresTxt();

}

public System.Threading.Tasks.Task<string> guardarProveedoresTxtAsync() {

return base.Channel.guardarProveedoresTxtAsync();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference7 {

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference7.IServicioModificarParametros")]

public interface IServicioModificarParametros {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ModificarArancel", ReplyAction="http://tempuri.org/IServicioModificarParametros/ModificarArancelResponse")]

bool ModificarArancel(double value);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ModificarArancel", ReplyAction="http://tempuri.org/IServicioModificarParametros/ModificarArancelResponse")]

System.Threading.Tasks.Task<bool> ModificarArancelAsync(double value);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ModificarPorcentajeExtra", ReplyAction="http://tempuri.org/IServicioModificarParametros/ModificarPorcentajeExtraResponse")]

bool ModificarPorcentajeExtra(int value);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ModificarPorcentajeExtra", ReplyAction="http://tempuri.org/IServicioModificarParametros/ModificarPorcentajeExtraResponse")]

System.Threading.Tasks.Task<bool> ModificarPorcentajeExtraAsync(int value);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ObtenerArancel", ReplyAction="http://tempuri.org/IServicioModificarParametros/ObtenerArancelResponse")]

double ObtenerArancel();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ObtenerArancel", ReplyAction="http://tempuri.org/IServicioModificarParametros/ObtenerArancelResponse")]

System.Threading.Tasks.Task<double> ObtenerArancelAsync();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ObtenerPorcentajeExtra", ReplyAction="http://tempuri.org/IServicioModificarParametros/ObtenerPorcentajeExtraResponse")]

int ObtenerPorcentajeExtra();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IServicioModificarParametros/ObtenerPorcentajeExtra", ReplyAction="http://tempuri.org/IServicioModificarParametros/ObtenerPorcentajeExtraResponse")]

System.Threading.Tasks.Task<int> ObtenerPorcentajeExtraAsync();

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IServicioModificarParametrosChannel : AppWeb.ServiceReference7.IServicioModificarParametros, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class ServicioModificarParametrosClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference7.IServicioModificarParametros>, AppWeb.ServiceReference7.IServicioModificarParametros {

public ServicioModificarParametrosClient() {

}

public ServicioModificarParametrosClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public ServicioModificarParametrosClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public ServicioModificarParametrosClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public ServicioModificarParametrosClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public bool ModificarArancel(double value) {

return base.Channel.ModificarArancel(value);

}

public System.Threading.Tasks.Task<bool> ModificarArancelAsync(double value) {

return base.Channel.ModificarArancelAsync(value);

}

public bool ModificarPorcentajeExtra(int value) {

return base.Channel.ModificarPorcentajeExtra(value);

}

public System.Threading.Tasks.Task<bool> ModificarPorcentajeExtraAsync(int value) {

return base.Channel.ModificarPorcentajeExtraAsync(value);

}

public double ObtenerArancel() {

return base.Channel.ObtenerArancel();

}

public System.Threading.Tasks.Task<double> ObtenerArancelAsync() {

return base.Channel.ObtenerArancelAsync();

}

public int ObtenerPorcentajeExtra() {

return base.Channel.ObtenerPorcentajeExtra();

}

public System.Threading.Tasks.Task<int> ObtenerPorcentajeExtraAsync() {

return base.Channel.ObtenerPorcentajeExtraAsync();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference8 {

using System.Runtime.Serialization;

using System;

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoProveedor", Namespace="http://schemas.datacontract.org/2004/07/WCFDesactivarProv")]

[System.SerializableAttribute()]

public partial class DtoProveedor : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string EmailField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string FechaRegistroField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private AppWeb.ServiceReference8.DtoServicioProveedor[] ListaServiciosField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreFantasiaField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string RUTField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string TelefonoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string TipoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private bool esInactivoField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Email {

get {

return this.EmailField;

}

set {

if ((object.ReferenceEquals(this.EmailField, value) != true)) {

this.EmailField = value;

this.RaisePropertyChanged("Email");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string FechaRegistro {

get {

return this.FechaRegistroField;

}

set {

if ((object.ReferenceEquals(this.FechaRegistroField, value) != true)) {

this.FechaRegistroField = value;

this.RaisePropertyChanged("FechaRegistro");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public AppWeb.ServiceReference8.DtoServicioProveedor[] ListaServicios {

get {

return this.ListaServiciosField;

}

set {

if ((object.ReferenceEquals(this.ListaServiciosField, value) != true)) {

this.ListaServiciosField = value;

this.RaisePropertyChanged("ListaServicios");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string NombreFantasia {

get {

return this.NombreFantasiaField;

}

set {

if ((object.ReferenceEquals(this.NombreFantasiaField, value) != true)) {

this.NombreFantasiaField = value;

this.RaisePropertyChanged("NombreFantasia");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string RUT {

get {

return this.RUTField;

}

set {

if ((object.ReferenceEquals(this.RUTField, value) != true)) {

this.RUTField = value;

this.RaisePropertyChanged("RUT");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Telefono {

get {

return this.TelefonoField;

}

set {

if ((object.ReferenceEquals(this.TelefonoField, value) != true)) {

this.TelefonoField = value;

this.RaisePropertyChanged("Telefono");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Tipo {

get {

return this.TipoField;

}

set {

if ((object.ReferenceEquals(this.TipoField, value) != true)) {

this.TipoField = value;

this.RaisePropertyChanged("Tipo");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public bool esInactivo {

get {

return this.esInactivoField;

}

set {

if ((this.esInactivoField.Equals(value) != true)) {

this.esInactivoField = value;

this.RaisePropertyChanged("esInactivo");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoServicioProveedor", Namespace="http://schemas.datacontract.org/2004/07/WCFDesactivarProv")]

[System.SerializableAttribute()]

public partial class DtoServicioProveedor : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string DescripcionField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string FotoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private int IdServicioField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string RutProveedorField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Descripcion {

get {

return this.DescripcionField;

}

set {

if ((object.ReferenceEquals(this.DescripcionField, value) != true)) {

this.DescripcionField = value;

this.RaisePropertyChanged("Descripcion");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Foto {

get {

return this.FotoField;

}

set {

if ((object.ReferenceEquals(this.FotoField, value) != true)) {

this.FotoField = value;

this.RaisePropertyChanged("Foto");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public int IdServicio {

get {

return this.IdServicioField;

}

set {

if ((this.IdServicioField.Equals(value) != true)) {

this.IdServicioField = value;

this.RaisePropertyChanged("IdServicio");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Nombre {

get {

return this.NombreField;

}

set {

if ((object.ReferenceEquals(this.NombreField, value) != true)) {

this.NombreField = value;

this.RaisePropertyChanged("Nombre");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string RutProveedor {

get {

return this.RutProveedorField;

}

set {

if ((object.ReferenceEquals(this.RutProveedorField, value) != true)) {

this.RutProveedorField = value;

this.RaisePropertyChanged("RutProveedor");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference8.IDesactivarProv")]

public interface IDesactivarProv {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IDesactivarProv/DesactivarProveedor", ReplyAction="http://tempuri.org/IDesactivarProv/DesactivarProveedorResponse")]

bool DesactivarProveedor(AppWeb.ServiceReference8.DtoProveedor dtoProv);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IDesactivarProv/DesactivarProveedor", ReplyAction="http://tempuri.org/IDesactivarProv/DesactivarProveedorResponse")]

System.Threading.Tasks.Task<bool> DesactivarProveedorAsync(AppWeb.ServiceReference8.DtoProveedor dtoProv);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IDesactivarProv/BuscarProveedorRut", ReplyAction="http://tempuri.org/IDesactivarProv/BuscarProveedorRutResponse")]

AppWeb.ServiceReference8.DtoProveedor BuscarProveedorRut(string rut);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IDesactivarProv/BuscarProveedorRut", ReplyAction="http://tempuri.org/IDesactivarProv/BuscarProveedorRutResponse")]

System.Threading.Tasks.Task<AppWeb.ServiceReference8.DtoProveedor> BuscarProveedorRutAsync(string rut);

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IDesactivarProvChannel : AppWeb.ServiceReference8.IDesactivarProv, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class DesactivarProvClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference8.IDesactivarProv>, AppWeb.ServiceReference8.IDesactivarProv {

public DesactivarProvClient() {

}

public DesactivarProvClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public DesactivarProvClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public DesactivarProvClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public DesactivarProvClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public bool DesactivarProveedor(AppWeb.ServiceReference8.DtoProveedor dtoProv) {

return base.Channel.DesactivarProveedor(dtoProv);

}

public System.Threading.Tasks.Task<bool> DesactivarProveedorAsync(AppWeb.ServiceReference8.DtoProveedor dtoProv) {

return base.Channel.DesactivarProveedorAsync(dtoProv);

}

public AppWeb.ServiceReference8.DtoProveedor BuscarProveedorRut(string rut) {

return base.Channel.BuscarProveedorRut(rut);

}

public System.Threading.Tasks.Task<AppWeb.ServiceReference8.DtoProveedor> BuscarProveedorRutAsync(string rut) {

return base.Channel.BuscarProveedorRutAsync(rut);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Reference.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//------------------------------------------------------------------------------

// <auto-generated>

// Este código fue generado por una herramienta.

// Versión de runtime:4.0.30319.42000

//

// Los cambios en este archivo podrían causar un comportamiento incorrecto y se perderán si

// se vuelve a generar el código.

// </auto-generated>

//------------------------------------------------------------------------------

namespace AppWeb.ServiceReference9 {

using System.Runtime.Serialization;

using System;

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoUsuario", Namespace="http://schemas.datacontract.org/2004/07/WcfAgregarProv")]

[System.SerializableAttribute()]

public partial class DtoUsuario : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string EmailField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string PasswField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private int RolField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string UserField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Email {

get {

return this.EmailField;

}

set {

if ((object.ReferenceEquals(this.EmailField, value) != true)) {

this.EmailField = value;

this.RaisePropertyChanged("Email");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Passw {

get {

return this.PasswField;

}

set {

if ((object.ReferenceEquals(this.PasswField, value) != true)) {

this.PasswField = value;

this.RaisePropertyChanged("Passw");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public int Rol {

get {

return this.RolField;

}

set {

if ((this.RolField.Equals(value) != true)) {

this.RolField = value;

this.RaisePropertyChanged("Rol");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string User {

get {

return this.UserField;

}

set {

if ((object.ReferenceEquals(this.UserField, value) != true)) {

this.UserField = value;

this.RaisePropertyChanged("User");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoProveedor", Namespace="http://schemas.datacontract.org/2004/07/WcfAgregarProv")]

[System.SerializableAttribute()]

public partial class DtoProveedor : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string EmailField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string FechaRegistroField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private AppWeb.ServiceReference9.DtoServicioProveedor[] ListaServiciosField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private Dominio.Usuario MiUsuarioField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreFantasiaField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string RUTField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string TelefonoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string TipoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private bool esInactivoField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Email {

get {

return this.EmailField;

}

set {

if ((object.ReferenceEquals(this.EmailField, value) != true)) {

this.EmailField = value;

this.RaisePropertyChanged("Email");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string FechaRegistro {

get {

return this.FechaRegistroField;

}

set {

if ((object.ReferenceEquals(this.FechaRegistroField, value) != true)) {

this.FechaRegistroField = value;

this.RaisePropertyChanged("FechaRegistro");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public AppWeb.ServiceReference9.DtoServicioProveedor[] ListaServicios {

get {

return this.ListaServiciosField;

}

set {

if ((object.ReferenceEquals(this.ListaServiciosField, value) != true)) {

this.ListaServiciosField = value;

this.RaisePropertyChanged("ListaServicios");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public Dominio.Usuario MiUsuario {

get {

return this.MiUsuarioField;

}

set {

if ((object.ReferenceEquals(this.MiUsuarioField, value) != true)) {

this.MiUsuarioField = value;

this.RaisePropertyChanged("MiUsuario");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string NombreFantasia {

get {

return this.NombreFantasiaField;

}

set {

if ((object.ReferenceEquals(this.NombreFantasiaField, value) != true)) {

this.NombreFantasiaField = value;

this.RaisePropertyChanged("NombreFantasia");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string RUT {

get {

return this.RUTField;

}

set {

if ((object.ReferenceEquals(this.RUTField, value) != true)) {

this.RUTField = value;

this.RaisePropertyChanged("RUT");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Telefono {

get {

return this.TelefonoField;

}

set {

if ((object.ReferenceEquals(this.TelefonoField, value) != true)) {

this.TelefonoField = value;

this.RaisePropertyChanged("Telefono");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Tipo {

get {

return this.TipoField;

}

set {

if ((object.ReferenceEquals(this.TipoField, value) != true)) {

this.TipoField = value;

this.RaisePropertyChanged("Tipo");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public bool esInactivo {

get {

return this.esInactivoField;

}

set {

if ((this.esInactivoField.Equals(value) != true)) {

this.esInactivoField = value;

this.RaisePropertyChanged("esInactivo");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoServicioProveedor", Namespace="http://schemas.datacontract.org/2004/07/WcfAgregarProv")]

[System.SerializableAttribute()]

public partial class DtoServicioProveedor : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string DescripcionField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string FotoField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private int IdServicioField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string RutProveedorField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Descripcion {

get {

return this.DescripcionField;

}

set {

if ((object.ReferenceEquals(this.DescripcionField, value) != true)) {

this.DescripcionField = value;

this.RaisePropertyChanged("Descripcion");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Foto {

get {

return this.FotoField;

}

set {

if ((object.ReferenceEquals(this.FotoField, value) != true)) {

this.FotoField = value;

this.RaisePropertyChanged("Foto");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public int IdServicio {

get {

return this.IdServicioField;

}

set {

if ((this.IdServicioField.Equals(value) != true)) {

this.IdServicioField = value;

this.RaisePropertyChanged("IdServicio");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Nombre {

get {

return this.NombreField;

}

set {

if ((object.ReferenceEquals(this.NombreField, value) != true)) {

this.NombreField = value;

this.RaisePropertyChanged("Nombre");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string RutProveedor {

get {

return this.RutProveedorField;

}

set {

if ((object.ReferenceEquals(this.RutProveedorField, value) != true)) {

this.RutProveedorField = value;

this.RaisePropertyChanged("RutProveedor");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoServicio", Namespace="http://schemas.datacontract.org/2004/07/WcfAgregarProv")]

[System.SerializableAttribute()]

public partial class DtoServicio : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string DescripcionField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private int IdServicioField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private AppWeb.ServiceReference9.DtoTipoEvento[] ListaTipoEventosField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Descripcion {

get {

return this.DescripcionField;

}

set {

if ((object.ReferenceEquals(this.DescripcionField, value) != true)) {

this.DescripcionField = value;

this.RaisePropertyChanged("Descripcion");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public int IdServicio {

get {

return this.IdServicioField;

}

set {

if ((this.IdServicioField.Equals(value) != true)) {

this.IdServicioField = value;

this.RaisePropertyChanged("IdServicio");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public AppWeb.ServiceReference9.DtoTipoEvento[] ListaTipoEventos {

get {

return this.ListaTipoEventosField;

}

set {

if ((object.ReferenceEquals(this.ListaTipoEventosField, value) != true)) {

this.ListaTipoEventosField = value;

this.RaisePropertyChanged("ListaTipoEventos");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Nombre {

get {

return this.NombreField;

}

set {

if ((object.ReferenceEquals(this.NombreField, value) != true)) {

this.NombreField = value;

this.RaisePropertyChanged("Nombre");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.Runtime.Serialization", "4.0.0.0")]

[System.Runtime.Serialization.DataContractAttribute(Name="DtoTipoEvento", Namespace="http://schemas.datacontract.org/2004/07/WcfAgregarProv")]

[System.SerializableAttribute()]

public partial class DtoTipoEvento : object, System.Runtime.Serialization.IExtensibleDataObject, System.ComponentModel.INotifyPropertyChanged {

[System.NonSerializedAttribute()]

private System.Runtime.Serialization.ExtensionDataObject extensionDataField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string DescripcionField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private string NombreField;

[System.Runtime.Serialization.OptionalFieldAttribute()]

private int idTipoEventoField;

[global::System.ComponentModel.BrowsableAttribute(false)]

public System.Runtime.Serialization.ExtensionDataObject ExtensionData {

get {

return this.extensionDataField;

}

set {

this.extensionDataField = value;

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Descripcion {

get {

return this.DescripcionField;

}

set {

if ((object.ReferenceEquals(this.DescripcionField, value) != true)) {

this.DescripcionField = value;

this.RaisePropertyChanged("Descripcion");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public string Nombre {

get {

return this.NombreField;

}

set {

if ((object.ReferenceEquals(this.NombreField, value) != true)) {

this.NombreField = value;

this.RaisePropertyChanged("Nombre");

}

}

}

[System.Runtime.Serialization.DataMemberAttribute()]

public int idTipoEvento {

get {

return this.idTipoEventoField;

}

set {

if ((this.idTipoEventoField.Equals(value) != true)) {

this.idTipoEventoField = value;

this.RaisePropertyChanged("idTipoEvento");

}

}

}

public event System.ComponentModel.PropertyChangedEventHandler PropertyChanged;

protected void RaisePropertyChanged(string propertyName) {

System.ComponentModel.PropertyChangedEventHandler propertyChanged = this.PropertyChanged;

if ((propertyChanged != null)) {

propertyChanged(this, new System.ComponentModel.PropertyChangedEventArgs(propertyName));

}

}

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

[System.ServiceModel.ServiceContractAttribute(ConfigurationName="ServiceReference9.IAgregarProv")]

public interface IAgregarProv {

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/InsertarProveedor", ReplyAction="http://tempuri.org/IAgregarProv/InsertarProveedorResponse")]

bool InsertarProveedor(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/InsertarProveedor", ReplyAction="http://tempuri.org/IAgregarProv/InsertarProveedorResponse")]

System.Threading.Tasks.Task<bool> InsertarProveedorAsync(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/AgregarUsuario", ReplyAction="http://tempuri.org/IAgregarProv/AgregarUsuarioResponse")]

bool AgregarUsuario(AppWeb.ServiceReference9.DtoUsuario usu, AppWeb.ServiceReference9.DtoProveedor prov);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/AgregarUsuario", ReplyAction="http://tempuri.org/IAgregarProv/AgregarUsuarioResponse")]

System.Threading.Tasks.Task<bool> AgregarUsuarioAsync(AppWeb.ServiceReference9.DtoUsuario usu, AppWeb.ServiceReference9.DtoProveedor prov);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/AgregarServicioAProveedor", ReplyAction="http://tempuri.org/IAgregarProv/AgregarServicioAProveedorResponse")]

bool AgregarServicioAProveedor(AppWeb.ServiceReference9.DtoServicioProveedor servicio, AppWeb.ServiceReference9.DtoProveedor prov);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/AgregarServicioAProveedor", ReplyAction="http://tempuri.org/IAgregarProv/AgregarServicioAProveedorResponse")]

System.Threading.Tasks.Task<bool> AgregarServicioAProveedorAsync(AppWeb.ServiceReference9.DtoServicioProveedor servicio, AppWeb.ServiceReference9.DtoProveedor prov);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/ExisteRUT", ReplyAction="http://tempuri.org/IAgregarProv/ExisteRUTResponse")]

bool ExisteRUT(string rut);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/ExisteRUT", ReplyAction="http://tempuri.org/IAgregarProv/ExisteRUTResponse")]

System.Threading.Tasks.Task<bool> ExisteRUTAsync(string rut);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/ExisteMail", ReplyAction="http://tempuri.org/IAgregarProv/ExisteMailResponse")]

bool ExisteMail(string rut);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/ExisteMail", ReplyAction="http://tempuri.org/IAgregarProv/ExisteMailResponse")]

System.Threading.Tasks.Task<bool> ExisteMailAsync(string rut);

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/CargarServicios", ReplyAction="http://tempuri.org/IAgregarProv/CargarServiciosResponse")]

AppWeb.ServiceReference9.DtoServicio[] CargarServicios();

[System.ServiceModel.OperationContractAttribute(Action="http://tempuri.org/IAgregarProv/CargarServicios", ReplyAction="http://tempuri.org/IAgregarProv/CargarServiciosResponse")]

System.Threading.Tasks.Task<AppWeb.ServiceReference9.DtoServicio[]> CargarServiciosAsync();

}

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public interface IAgregarProvChannel : AppWeb.ServiceReference9.IAgregarProv, System.ServiceModel.IClientChannel {

}

[System.Diagnostics.DebuggerStepThroughAttribute()]

[System.CodeDom.Compiler.GeneratedCodeAttribute("System.ServiceModel", "4.0.0.0")]

public partial class AgregarProvClient : System.ServiceModel.ClientBase<AppWeb.ServiceReference9.IAgregarProv>, AppWeb.ServiceReference9.IAgregarProv {

public AgregarProvClient() {

}

public AgregarProvClient(string endpointConfigurationName) :

base(endpointConfigurationName) {

}

public AgregarProvClient(string endpointConfigurationName, string remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public AgregarProvClient(string endpointConfigurationName, System.ServiceModel.EndpointAddress remoteAddress) :

base(endpointConfigurationName, remoteAddress) {

}

public AgregarProvClient(System.ServiceModel.Channels.Binding binding, System.ServiceModel.EndpointAddress remoteAddress) :

base(binding, remoteAddress) {

}

public bool InsertarProveedor(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass) {

return base.Channel.InsertarProveedor(rut, nombreFantasia, email, tel, esInactivo, esVip, pass);

}

public System.Threading.Tasks.Task<bool> InsertarProveedorAsync(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass) {

return base.Channel.InsertarProveedorAsync(rut, nombreFantasia, email, tel, esInactivo, esVip, pass);

}

public bool AgregarUsuario(AppWeb.ServiceReference9.DtoUsuario usu, AppWeb.ServiceReference9.DtoProveedor prov) {

return base.Channel.AgregarUsuario(usu, prov);

}

public System.Threading.Tasks.Task<bool> AgregarUsuarioAsync(AppWeb.ServiceReference9.DtoUsuario usu, AppWeb.ServiceReference9.DtoProveedor prov) {

return base.Channel.AgregarUsuarioAsync(usu, prov);

}

public bool AgregarServicioAProveedor(AppWeb.ServiceReference9.DtoServicioProveedor servicio, AppWeb.ServiceReference9.DtoProveedor prov) {

return base.Channel.AgregarServicioAProveedor(servicio, prov);

}

public System.Threading.Tasks.Task<bool> AgregarServicioAProveedorAsync(AppWeb.ServiceReference9.DtoServicioProveedor servicio, AppWeb.ServiceReference9.DtoProveedor prov) {

return base.Channel.AgregarServicioAProveedorAsync(servicio, prov);

}

public bool ExisteRUT(string rut) {

return base.Channel.ExisteRUT(rut);

}

public System.Threading.Tasks.Task<bool> ExisteRUTAsync(string rut) {

return base.Channel.ExisteRUTAsync(rut);

}

public bool ExisteMail(string rut) {

return base.Channel.ExisteMail(rut);

}

public System.Threading.Tasks.Task<bool> ExisteMailAsync(string rut) {

return base.Channel.ExisteMailAsync(rut);

}

public AppWeb.ServiceReference9.DtoServicio[] CargarServicios() {

return base.Channel.CargarServicios();

}

public System.Threading.Tasks.Task<AppWeb.ServiceReference9.DtoServicio[]> CargarServiciosAsync() {

return base.Channel.CargarServiciosAsync();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Conexion.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Diagnostics;

using System.Configuration;

namespace Dominio

{

public class Conexion

{

#region Manejo de la conexión.

//La cadena de conexión está configurada para el servidor de prueba

//que viene con Visual Studio

//Cambiarla si se utiliza otro servicio de SQLServer.

private static string cadenaConexion = ConfigurationManager.ConnectionStrings["ConexionSeba"].ConnectionString;

private static string cadenaConexionPolaNotebook = ConfigurationManager.ConnectionStrings["ConexionPolachekNoteb"].ConnectionString;

private static string cadenaConexionPolaPC = ConfigurationManager.ConnectionStrings["ConexionPolachekPC"].ConnectionString;

public static SqlConnection CrearConexion()

{

return new SqlConnection(cadenaConexionPolaNotebook);

}

public static void AbrirConexion(SqlConnection cn)

{

try

{

if (cn.State == ConnectionState.Closed)

{

cn.Open();

}

}

catch (Exception ex)

{

Debug.Assert(false, ex.Message);

}

}

public static void CerrarConexion(SqlConnection cn)

{

try

{

if (cn.State != ConnectionState.Closed)

{

cn.Close();

cn.Dispose();

}

}

catch (Exception ex)

{

Debug.Assert(false, ex.Message);

}

}

#endregion

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Evento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Dominio

{

class Evento

{

public DateTime Fecha { get; set; }

public string direccion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IActiveRecord.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Dominio

{

interface IActiveRecord

{

bool Insertar();

bool Eliminar();

bool Modificar();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Organizador.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Dominio

{

class Organizador

{

public string Nombre { get; set; }

public string Email { get; set; }

public string Password { get; set; }

public string Telefono { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Proveedor.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

using System.Diagnostics;

using System.IO;

namespace Dominio

{

public abstract class Proveedor : IActiveRecord

{

#region Propiedades

public string RUT { get; set; }

public string NombreFantasia { get; set; }

public string Email { get; set; }

public Usuario MiUsuario { get; set; } = new Usuario();

public string Telefono { get; set; }

public string FechaRegistro { get; set; }

public bool esInactivo { get; set; }

public static double Arancel{ get; set; }

public string Tipo { get; set; }

public List<ServicioProveedor> ListaServicios { get; set; }

#endregion

public override string ToString()

{

string ret = string.Format("{0} {1}", "Rut: " + RUT + " - ", "NombreFantasia:" + NombreFantasia);

return ret;

}

#region Métodos de lógica

/// <summary>

/// VALIDA LOS DATOS EN FORM PARA AGREGAR PROVEEDOR

/// </summary>

/// <returns></returns>

public virtual bool Validar()

{

return this.RUT.Length == 12

&& this.NombreFantasia.Length > 3

&& this.Email.Length > 3

&& this.Telefono.Length > 3

;

}

/// <summary>

/// CONTROLA SI EXISTE EL RUT INGRESADO

/// </summary>

/// <param name="rut"></param>

/// <returns></returns>

public static bool ExisteRut(string rut)

{

bool ret = false;

if (FindByRUT(rut) != null)

{

ret = true;

}

return ret;

}

/// <summary>

/// CONTROLA SI EXISTE EL EMAIL INGRESADO

/// </summary>

/// <param name="email"></param>

/// <returns></returns>

public static bool ExisteEmail(string email)

{

bool ret = false;

if (FindByEmail(email) != null)

{

ret = true;

}

return ret;

}

#endregion

#region Manejo de Usuario

/// <summary>

/// AGREGA EL OBJETO USUARIO

/// </summary>

/// <param name="usu"></param>

/// <returns></returns>

public bool AgregarUsuario(Usuario usu)

{

this.MiUsuario = usu;

return true;

}

#endregion

#region Acceso a datos

/// <summary>

/// AGREGA UN PROVEEDOR A LA BASE DE DATOS

/// </summary>

/// <returns></returns>

public bool Insertar()

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

if (!this.Validar()) return false;

SqlTransaction trn = null;

try

{

Conexion.AbrirConexion(cn);

trn = cn.BeginTransaction();

cmd.Connection = cn;

cmd.Transaction = trn;

Usuario usuarioAInsertar = new Usuario();

usuarioAInsertar.User = MiUsuario.User;

usuarioAInsertar.Passw = MiUsuario.Passw;

usuarioAInsertar.Rol = MiUsuario.Rol;

usuarioAInsertar.Email = MiUsuario.Email;

usuarioAInsertar.Insertar(cmd);

cmd.CommandText=

@"INSERT INTO Proveedor

VALUES (@rut, @nombrefantasia, @email, @telefono, @fecharegistro, @esInactivo, @tipo);

SELECT CAST (SCOPE\_IDENTITY() AS INT)";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@RUT", this.RUT);

cmd.Parameters.AddWithValue("@nombreFantasia", this.NombreFantasia);

cmd.Parameters.AddWithValue("@email", this.Email);

cmd.Parameters.AddWithValue("@telefono", this.Telefono);

cmd.Parameters.AddWithValue("@fechaRegistro", this.FechaRegistro);

cmd.Parameters.AddWithValue("@esInactivo", this.esInactivo);

cmd.Parameters.AddWithValue("@tipo", this.Tipo);

cmd.Transaction = trn;

cmd.ExecuteNonQuery();

//Se implementó condición para lista de servicios igual null para evitar conflicto al cargar wcf con proveedor nulo

if (this.ListaServicios == null){}

else if (this.ListaServicios.Count() > 0)

{

foreach (ServicioProveedor miServ in ListaServicios)

{

miServ.InsertarServicioProveedor(cmd, miServ);

}

}

if (Tipo == "VIP")

{

ProveedorVIP.Insertar(cmd, this.RUT);

}

else if (Tipo == "COMUN")

{

cmd.CommandText =

@"INSERT INTO ProveedorComun

VALUES (@rutProveedor);";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@rutProveedor", this.RUT);

cmd.ExecuteNonQuery();

}

trn.Commit();

return true;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

trn.Rollback();

return false;

}

finally {

Conexion.CerrarConexion(cn);

trn.Dispose();

}

}

/// <summary>

/// ELIMINA UN PROVEEDOR DADO RUT

/// </summary>

/// <returns></returns>

public bool Eliminar()

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"DELETE Proveedor WHERE RUT=@rut;";

cmd.Parameters.Add(new SqlParameter("@rut", this.RUT));

try

{

Conexion.AbrirConexion(cn);

cmd.ExecuteNonQuery();

return true;

}

catch (Exception ex)

{

Debug.Assert(false, ex.Message);

return false;

}

finally

{

Conexion.CerrarConexion(cn);

}

}

public bool Modificar()

{

throw new NotImplementedException();

}

/// <summary>

/// OBTIENE ARANCEL

/// </summary>

/// <returns></returns>

public static decimal ObtenerArancel()

{

decimal ret = 0;

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.Connection = cn;

cmd.CommandText = @"SELECT Arancel FROM Parametros";

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

while (dr.Read()) {

ret = dr.IsDBNull(dr.GetOrdinal("arancel")) ? 0 : dr.GetDecimal(dr.GetOrdinal("arancel"));

}

}

return ret;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return ret;

}

finally {

Conexion.CerrarConexion(cn);

}

}

/// <summary>

/// OBTIENE PORCENTAJE EXTRA

/// </summary>

/// <returns></returns>

public static int ObtenerPorcentajeExtra()

{

int ret = 0;

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.Connection = cn;

cmd.CommandText = @"SELECT porcentajeExtra FROM Parametros";

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

while (dr.Read()) {

ret = dr.IsDBNull(dr.GetOrdinal("porcentajeExtra")) ? 0 : dr.GetInt32(dr.GetOrdinal("porcentajeExtra"));

}

}

return ret;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return ret;

}

finally

{

Conexion.CerrarConexion(cn);

}

}

/// <summary>

/// MODIFICAR ARANCEL

/// </summary>

/// <param name="nuevoArancel"></param>

/// <returns></returns>

public static bool ModificarArancel(double nuevoArancel)

{

bool ret = false;

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

try

{

Conexion.AbrirConexion(cn);

cmd.Connection = cn;

cmd.CommandText =@"UPDATE Parametros SET arancel = @nuevoArancel";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@nuevoArancel", nuevoArancel);

cmd.ExecuteNonQuery();

ret = true;

return ret;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return ret;

}

finally {

Conexion.CerrarConexion(cn);

}

}

/// <summary>

/// MODIFICAR PORCENTAJE

/// </summary>

/// <param name="nuevoPorcentaje"></param>

/// <returns></returns>

public static bool ModificarPorcentajeExtra(int nuevoPorcentaje)

{

bool ret = false;

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

try

{

Conexion.AbrirConexion(cn);

cmd.Connection = cn;

cmd.CommandText = @"UPDATE Parametros SET porcentajeExtra = @nuevoPorcentaje";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@nuevoPorcentaje", nuevoPorcentaje);

cmd.ExecuteNonQuery();

ret = true;

return ret;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return ret;

}

finally {

Conexion.CerrarConexion(cn);

}

}

/// <summary>

/// DESACTIVAR PROVEEDOR

/// </summary>

/// <returns></returns>

public static bool DesactivarProv(string rut)

{

bool ret = false;

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

SqlTransaction trn = null;

cmd.Connection = cn;

cmd.Transaction = trn;

cmd.CommandText = @"UPDATE Proveedor SET esInactivo = 1 WHERE RUT = @rut;";

cmd.Parameters.AddWithValue("@rut", rut);

try

{

Proveedor p = FindByRUT(rut);

if (p != null) {

Conexion.AbrirConexion(cn);

trn = cn.BeginTransaction();

cmd.Transaction = trn;

int filas = cmd.ExecuteNonQuery();

int cantServicios = 0;

if (p.ListaServicios != null && p.ListaServicios.Count >= 0)

{

cmd.CommandText = @"DELETE FROM ProveedorServicios WHERE rutProveedor = @rutProveedor";

cantServicios = p.ListaServicios.Count();

foreach (ServicioProveedor sp in p.ListaServicios)

{

cmd.Parameters.Clear();

cmd.Parameters.Add(new SqlParameter("@rutProveedor", rut));

filas += cmd.ExecuteNonQuery();

}

}

if (filas == 1 + cantServicios)

{

trn.Commit();

return true;

}

else

{

trn.Rollback();

return false;

}

}

return ret;

}

catch (Exception ex)

{

throw new Exception("No existe el Proveedor" + ex);

}

finally {

Conexion.CerrarConexion(cn);

}

}

/// <summary>

/// AGREGAR SERVICIOS A PROVEEDOR

/// </summary>

/// <param name="p"></param>

/// <returns></returns>

public static Proveedor AgregarServiciosAProveedor(Proveedor p) {

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand(@"SELECT \* FROM ProveedorServicios WHERE rutProveedor = @rut");

cmd.Connection = cn;

cmd.Parameters.AddWithValue("@rut", p.RUT);

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

if (dr.Read())

{

p.ListaServicios = new List<ServicioProveedor>();

ServicioProveedor sp = new ServicioProveedor();

sp.RutProveedor = p.RUT;

sp.IdServicio = dr.IsDBNull(dr.GetOrdinal("idServicio")) ? 0 : dr.GetInt32(dr.GetOrdinal("idServicio"));

sp.Descripcion = dr.IsDBNull(dr.GetOrdinal("descripcion")) ? "" : dr.GetString(dr.GetOrdinal("descripcion"));

sp.Nombre = dr.IsDBNull(dr.GetOrdinal("nombre")) ? "" : dr.GetString(dr.GetOrdinal("nombre"));

sp.Foto = dr.IsDBNull(dr.GetOrdinal("imagen")) ? "" : dr.GetString(dr.GetOrdinal("imagen"));

p.ListaServicios.Add(sp);

}

}

return p;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return p=null;

}

finally

{

Conexion.CerrarConexion(cn);

}

}

#endregion

#region Exportar Proveedores Txt

/// <summary>

///

/// </summary>

/// <param name="rutaArchivo"></param>

/// <returns></returns>

public static bool grabarProveedoresTxt(string rutaArchivo)

{

try

{

File.WriteAllText(rutaArchivo, String.Empty);

FileStream fs = new FileStream(rutaArchivo, FileMode.Open);

StreamWriter sw = new StreamWriter(fs);

List<Proveedor> listaProv = Proveedor.FindAll();

foreach (Proveedor prov in listaProv)

{

prov.ListaServicios = ServicioProveedor.FindServiciosProveedor(prov.RUT);

string listaServProv = "";

foreach (ServicioProveedor servProv in prov.ListaServicios)

{

listaServProv += servProv.Nombre + ":" + servProv.Descripcion + ":" + servProv.Foto + "#";

}

sw.WriteLine(prov.RUT + "#" + prov.NombreFantasia + "#" + prov.Email + "#" + prov.Telefono + "|" + listaServProv);

}

sw.Close();

return true;

}

catch (Exception e)

{

Console.WriteLine("Ocurrió un error: '{0}'", e);

return false;

}

}

#endregion

#region Finders

/// <summary>

/// BUSCAR PROVEEDOR POR RUT

/// </summary>

/// <param name="rut"></param>

/// <returns></returns>

public static Proveedor FindByRUT(string rut)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand(@"SELECT \* FROM Proveedor WHERE Rut = @rut");

cmd.Connection = cn;

cmd.Parameters.AddWithValue("@rut", rut);

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

if (dr.Read())

{

string miTipo = dr["tipo"].ToString();

if (miTipo == "COMUN")

{

Proveedor p = new ProveedorComun

{

RUT = rut,

NombreFantasia = dr["NombreFantasia"].ToString(),

Email = dr["Email"].ToString(),

Telefono = dr["Telefono"].ToString(),

FechaRegistro = dr["FechaRegistro"].ToString(),

esInactivo = (bool)dr["esInactivo"],

Tipo = miTipo,

};

p = AgregarServiciosAProveedor(p);

return p;

}

else if (miTipo == "VIP")

{

Proveedor p = new ProveedorVIP

{

RUT = rut,

NombreFantasia = dr["NombreFantasia"].ToString(),

Email = dr["Email"].ToString(),

Telefono = dr["Telefono"].ToString(),

FechaRegistro = dr["FechaRegistro"].ToString(),

esInactivo = (bool)dr["esInactivo"],

Tipo = miTipo,

};

p = AgregarServiciosAProveedor(p);

return p;

}

}

}

return null;

}

catch (Exception ex)

{

throw new Exception("No existe el Proveedor" + ex);

}

finally {

Conexion.CerrarConexion(cn);

cn.Dispose();

}

}

/// <summary>

/// BUSCAR PROVEEDOR POR MAIL

/// </summary>

/// <param name="email"></param>

/// <returns></returns>

public static Proveedor FindByEmail(string email)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand(@"SELECT \* From Proveedor WHERE Email = @email");

cmd.Connection = cn;

cmd.Parameters.AddWithValue("@email", email);

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

if (dr.Read())

{

string miTipo = dr["tipo"].ToString();

if (miTipo == "COMUN")

{

Proveedor p = new ProveedorComun

{

RUT = dr["rut"].ToString(),

NombreFantasia = dr["NombreFantasia"].ToString(),

Email = email,

Telefono = dr["Telefono"].ToString(),

FechaRegistro = dr["FechaRegistro"].ToString(),

esInactivo = (bool)dr["esInactivo"],

Tipo = miTipo,

};

return p;

}

else if (miTipo == "VIP")

{

Proveedor p = new ProveedorVIP

{

RUT = dr["rut"].ToString(),

NombreFantasia = dr["NombreFantasia"].ToString(),

Email = email,

Telefono = dr["Telefono"].ToString(),

FechaRegistro = dr["FechaRegistro"].ToString(),

esInactivo = (bool)dr["esInactivo"],

Tipo = miTipo,

};

return p;

}

}

return null;

}

catch (Exception ex)

{

throw new Exception("No existe el Proveedor" + ex);

}

finally {

Conexion.CerrarConexion(cn);

}

}

/// <summary>

/// OBTENER TODOS LOS PROVEEDORES

/// </summary>

/// <returns></returns>

public static List<Proveedor> FindAll()

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"SELECT \* FROM Proveedor";

cmd.Connection = cn;

List<Proveedor> listaProveedores = null;

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

listaProveedores = new List<Proveedor>();

while (dr.Read())

{

Proveedor p = CargarDatosDesdeReader(dr);

listaProveedores.Add(p);

}

}

return listaProveedores;

}

catch (SqlException ex)

{

System.Diagnostics.Debug.Assert(false, ex.Message);

return null;

}

finally

{

Conexion.CerrarConexion(cn);

}

}

/// <summary>

/// CARGAR DATOS DESDE READER

/// </summary>

/// <param name="fila"></param>

/// <returns></returns>

protected static Proveedor CargarDatosDesdeReader(IDataRecord fila)

{

Proveedor p = null;

if (fila != null)

{

string miTipo;

miTipo = fila.IsDBNull(fila.GetOrdinal("tipo")) ? "" : fila.GetString(fila.GetOrdinal("tipo"));

if (miTipo == "COMUN")

{

p = new ProveedorComun

{

RUT = fila.IsDBNull(fila.GetOrdinal("Rut")) ? "" : fila.GetString(fila.GetOrdinal("Rut")),

NombreFantasia = fila.IsDBNull(fila.GetOrdinal("NombreFantasia")) ? "" : fila.GetString(fila.GetOrdinal("NombreFantasia")),

Email = fila.IsDBNull(fila.GetOrdinal("Email")) ? "" : fila.GetString(fila.GetOrdinal("Email")),

Telefono = fila.IsDBNull(fila.GetOrdinal("Telefono")) ? "" : fila.GetString(fila.GetOrdinal("Telefono")),

FechaRegistro = fila.GetDateTime(fila.GetOrdinal("fechaRegistro")).ToString("yyyy/MM/dd"),

esInactivo = fila.GetBoolean(fila.GetOrdinal("esInactivo")),

Tipo = miTipo,

};

}else if (miTipo == "VIP")

{

p = new ProveedorVIP

{

RUT = fila.IsDBNull(fila.GetOrdinal("Rut")) ? "" : fila.GetString(fila.GetOrdinal("Rut")),

NombreFantasia = fila.IsDBNull(fila.GetOrdinal("NombreFantasia")) ? "" : fila.GetString(fila.GetOrdinal("NombreFantasia")),

Email = fila.IsDBNull(fila.GetOrdinal("Email")) ? "" : fila.GetString(fila.GetOrdinal("Email")),

Telefono = fila.IsDBNull(fila.GetOrdinal("Telefono")) ? "" : fila.GetString(fila.GetOrdinal("Telefono")),

FechaRegistro = fila.GetDateTime(fila.GetOrdinal("fechaRegistro")).ToString("yyyy/MM/dd"),

esInactivo = fila.GetBoolean(fila.GetOrdinal("esInactivo")),

Tipo = miTipo,

};

}

}

return p;

}

/// <summary>

/// OBTIENE EL PORCENTAJE EXTRA DADO RUT

/// </summary>

/// <param name="rut"></param>

/// <returns></returns>

public static int FindPorcentajeVip(string rut)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand(@"SELECT \* From ProveedorVip WHERE rutProveedor = @rut");

cmd.Connection = cn;

cmd.Parameters.AddWithValue("@rut", rut);

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

if (dr.Read())

{

int porcentajeExtra;

{

porcentajeExtra = Convert.ToInt32(dr["porcentExtraAsign"]);

};

return porcentajeExtra;

}

}

return 0;

}

catch (Exception ex)

{

throw new Exception("No existe el Proveedor" + ex);

}

finally {

Conexion.CerrarConexion(cn);

}

}

#endregion

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ProveedorComun.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Dominio

{

public class ProveedorComun : Proveedor

{

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ProveedorVIP.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Dominio

{

public class ProveedorVIP : Proveedor

{

int PorcentajeExtra { get; set; }

#region Agregar ProveedorVIP

public static bool Insertar(SqlCommand cmd, string rut)

{

int porcentajeExtra = ObtenerPorcentajeExtra(cmd);

try

{

cmd.CommandText = @"INSERT INTO ProveedorVip

VALUES(@rutProveedor,@porcentExtraAsign)";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@rutProveedor", rut);

cmd.Parameters.AddWithValue("@porcentExtraAsign", porcentajeExtra);

cmd.ExecuteNonQuery();

return true;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return false;

}

}

public static int ObtenerPorcentajeExtra(SqlCommand cmd)

{

int ret = 0;

cmd.CommandText = @"SELECT porcentajeExtra FROM Parametros";

try {

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

if (dr.Read())

{

ret = (int)dr["porcentajeExtra"];

dr.Close();

}

}

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

}

return ret;

}

#endregion

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Servicio.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

using System.Diagnostics;

using System.IO;

namespace Dominio

{

public class Servicio : IActiveRecord

{

public int IdServicio { get; set; }

public string Nombre { get; set; }

public string Descripcion { get; set; }

public List<TipoEvento> ListaTipoEventos = new List<TipoEvento>();

public override string ToString()

{

string ret = string.Format("{0}", Nombre);

return ret;

}

public bool Equals(Servicio other)

{

if (other == null) return false;

return (this.Nombre.Equals(other.Nombre));

}

#region Acceso a datos

public bool Insertar()

{

throw new NotImplementedException();

}

public bool Eliminar()

{

throw new NotImplementedException();

}

public bool Modificar()

{

throw new NotImplementedException();

}

#endregion

#region Exportar catálogo a txt

public static bool grabarCatalogoTxt(string rutaArchivo)

{

try{

File.WriteAllText(rutaArchivo, String.Empty);

FileStream fs = new FileStream(rutaArchivo, FileMode.Open);

StreamWriter sw = new StreamWriter(fs);

List<Servicio> listaServicios = Servicio.FindAll();

foreach (Servicio serv in listaServicios)

{

serv.ListaTipoEventos = Servicio.FindTiposEventoByServicio(serv.Nombre);

string listaServTipoEvento = "";

foreach (TipoEvento tipoEv in serv.ListaTipoEventos)

{

listaServTipoEvento += tipoEv.Nombre + ":";

}

int largo = listaServTipoEvento.Length;

string tiposDeEventoPorServicio = listaServTipoEvento.Substring(0, largo - 1);

sw.WriteLine(serv.Nombre + "#" + tiposDeEventoPorServicio);

}

sw.Close();

return true;

}

catch (Exception e)

{

Console.WriteLine("Ocurrió un error: '{0}'", e);

return false;

}

}

#endregion

#region Finders

public static List<Servicio> FindAll()

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"SELECT \* FROM Servicio";

cmd.Connection = cn;

List<Servicio> listaServicios = null;

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

listaServicios = new List<Servicio>();

while (dr.Read())

{

Servicio s = CargarDatosDesdeReader(dr);

listaServicios.Add(s);

}

}

return listaServicios;

}

catch (SqlException ex)

{

//

System.Diagnostics.Debug.Assert(false, ex.Message);

return null;

}

finally

{

Conexion.CerrarConexion(cn);

}

}

public static Servicio FindById(int id)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand(@"SELECT \* FROM Servicio WHERE idServicio = @id");

cmd.Connection = cn;

cmd.Parameters.AddWithValue("@id", id);

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

if (dr.Read())

{

int miIdServicio = dr.IsDBNull(dr.GetOrdinal("idServicio")) ? 0 : dr.GetInt32(dr.GetOrdinal("idServicio"));

string nombreServicio = dr.IsDBNull(dr.GetOrdinal("nombre")) ? "" : dr.GetString(dr.GetOrdinal("nombre"));

string desc = dr.IsDBNull(dr.GetOrdinal("Descripcion")) ? "" : dr.GetString(dr.GetOrdinal("Descripcion"));

List<TipoEvento> tipoEventos = FindTiposEventoByServicio(nombreServicio);

Servicio s = new Servicio

{

IdServicio = miIdServicio,

Nombre = nombreServicio,

Descripcion = desc,

ListaTipoEventos = tipoEventos

};

return s;

}

}

return null;

}

catch (Exception ex)

{

throw new Exception("No existe el Servicio" + ex);

}

finally

{

Conexion.CerrarConexion(cn);

}

}

public static Servicio FindByNombre(string nombre)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand(@"SELECT \* FROM Servicio WHERE nombre = '@nombre'");

cmd.Connection = cn;

cmd.Parameters.AddWithValue("@nombre", nombre);

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

if (dr.Read())

{

int miIdServicio = dr.IsDBNull(dr.GetOrdinal("idServicio")) ? 0 : dr.GetInt32(dr.GetOrdinal("idServicio"));

string nombreServicio = dr.IsDBNull(dr.GetOrdinal("nombre")) ? "" : dr.GetString(dr.GetOrdinal("nombre"));

string desc = dr.IsDBNull(dr.GetOrdinal("Descripcion")) ? "" : dr.GetString(dr.GetOrdinal("Descripcion"));

List<TipoEvento> tipoEventos = FindTiposEventoByServicio(nombreServicio);

Servicio s = new Servicio

{

IdServicio = miIdServicio,

Nombre = nombreServicio,

Descripcion = desc,

ListaTipoEventos = tipoEventos

};

return s;

}

}

return null;

}

catch (Exception ex)

{

throw new Exception("No existe el Servicio" + ex);

}

finally {

Conexion.CerrarConexion(cn);

}

}

public static List<Servicio> FindServicioTipo()

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"SELECT s.IdServicio AS IdServicio, s.nombre AS Servicio, s.descripcion AS 'Descripción del servicio', t.nombre as 'Tipo de evento'

FROM Servicio AS s

INNER JOIN TipoEventoYServicio AS e ON s.idServicio = e.idServicio

INNER JOIN TipoEvento AS t ON e.idTipoEvento = t.idTipoEvento";

cmd.Connection = cn;

List<Servicio> listaServicios = null;

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

listaServicios = new List<Servicio>();

while (dr.Read())

{

Servicio s = CargarDatosDesdeReaderServicioTipo(dr);

listaServicios.Add(s);

}

}

return listaServicios;

}

catch (SqlException ex)

{

System.Diagnostics.Debug.Assert(false, ex.Message);

return null;

}

finally

{

cn.Close();

}

}

public static List<TipoEvento> FindTiposEventoByServicio(string servicio)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"SELECT t.nombre, t.descripcion, t.idTipoEvento

FROM Servicio AS s

INNER JOIN TipoEventoYServicio AS e ON s.idServicio = e.idServicio

INNER JOIN TipoEvento AS t ON e.idTipoEvento = t.idTipoEvento

WHERE s.nombre like @servicio";

cmd.Connection = cn;

cmd.Parameters.AddWithValue("@servicio", servicio);

List<TipoEvento> listaTipoEvento = null;

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

listaTipoEvento = new List<TipoEvento>();

while (dr.Read())

{

Servicio s = Servicio.FindByNombre(servicio);

string tipo = dr.IsDBNull(dr.GetOrdinal("nombre")) ? "" : dr.GetString(dr.GetOrdinal("nombre"));

string desc = dr.IsDBNull(dr.GetOrdinal("descripcion")) ? "" : dr.GetString(dr.GetOrdinal("descripcion"));

TipoEvento t = new TipoEvento(tipo, desc);

listaTipoEvento.Add(t);

}

}

return listaTipoEvento;

}

catch (SqlException ex)

{

//

System.Diagnostics.Debug.Assert(false, ex.Message);

return null;

}

finally

{

Conexion.CerrarConexion(cn);

}

}

protected static Servicio CargarDatosDesdeReader(IDataRecord fila)

{

Servicio s = null;

int idMiServicio = fila.IsDBNull(fila.GetOrdinal("idServicio")) ? 0 : fila.GetInt32(fila.GetOrdinal("idServicio"));

string nombreServicio = fila.IsDBNull(fila.GetOrdinal("nombre")) ? "" : fila.GetString(fila.GetOrdinal("nombre"));

string desc = fila.IsDBNull(fila.GetOrdinal("descripcion")) ? "" : fila.GetString(fila.GetOrdinal("descripcion"));

if (fila != null)

{

s = new Servicio()

{

IdServicio = idMiServicio,

Nombre = nombreServicio,

Descripcion = desc,

};

}

return s;

}

protected static Servicio CargarDatosDesdeReaderServicioTipo(IDataRecord fila)

{

Servicio s = null;

int idMiServicio = fila.IsDBNull(fila.GetOrdinal("idServicio")) ? 0 : fila.GetInt32(fila.GetOrdinal("idServicio"));

string nombreServicio = fila.IsDBNull(fila.GetOrdinal("Servicio")) ? "" : fila.GetString(fila.GetOrdinal("Servicio"));

string desc = fila.IsDBNull(fila.GetOrdinal("Descripción del servicio")) ? "" : fila.GetString(fila.GetOrdinal("Descripción del servicio"));

if (fila != null)

{

s = new Servicio()

{

IdServicio = idMiServicio,

Nombre = nombreServicio,

Descripcion = desc,

};

}

return s;

}

#endregion

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioProveedor.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Dominio

{

public class ServicioProveedor : IEquatable<ServicioProveedor>

{

public int IdServicio { get; set; }

public string RutProveedor { get; set; }

public string Nombre { get; set; }

public string Descripcion { get; set; }

public string Foto { get; set; }

public bool Equals(ServicioProveedor other)

{

if (other == null) return false;

return (this.Nombre.Equals(other.Nombre));

}

public override string ToString()

{

string ret = string.Format("{0}", Nombre);

return ret;

}

#region Acceso a Datos

public bool InsertarServicioProveedor(SqlCommand cmd, ServicioProveedor miserv)

{

try

{

cmd.CommandText = @"INSERT INTO ProveedorServicios

VALUES(@idServicio, @rutProveedor, @nombre, @descripcion, @imagen)";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@idServicio", miserv.IdServicio);

cmd.Parameters.AddWithValue("@rutProveedor", miserv.RutProveedor);

cmd.Parameters.AddWithValue("@nombre", miserv.Nombre);

cmd.Parameters.AddWithValue("@descripcion", miserv.Descripcion);

cmd.Parameters.AddWithValue("@imagen", miserv.Foto);

cmd.ExecuteNonQuery();

return true;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return false;

}

}

public static bool InsertarServicioProveedorSinTnr(ServicioProveedor miserv)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"INSERT INTO ProveedorServicios

VALUES(@idServicio, @rutProveedor, @nombre, @descripcion, @imagen)";

cmd.Parameters.AddWithValue("@idServicio", miserv.IdServicio);

cmd.Parameters.AddWithValue("@rutProveedor", miserv.RutProveedor);

cmd.Parameters.AddWithValue("@nombre", miserv.Nombre);

cmd.Parameters.AddWithValue("@descripcion", miserv.Descripcion);

cmd.Parameters.AddWithValue("@imagen", miserv.Foto);

cmd.Connection = cn;

try

{

Conexion.AbrirConexion(cn);

int filas = cmd.ExecuteNonQuery();

if (filas == 1)

return true;

else

return false;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return false;

}

}

#endregion

#region Finders

// FIND SERVICIOS PROVEEDOR

public static List<ServicioProveedor> FindServiciosProveedor(string rut)

{

SqlConnection cn = Conexion.CrearConexion();

SqlCommand cmd = new SqlCommand();

cmd.CommandText = @"SELECT \*

FROM ProveedorServicios

WHERE rutProveedor = @rut";

cmd.Parameters.AddWithValue("@rut", rut);

cmd.Connection = cn;

List<ServicioProveedor> listaServicios = null;

try

{

Conexion.AbrirConexion(cn);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

listaServicios = new List<ServicioProveedor>();

while (dr.Read())

{

ServicioProveedor serv = CargarDatosDesdeReader(dr);

listaServicios.Add(serv);

}

}

return listaServicios;

}

catch (SqlException ex)

{

//

System.Diagnostics.Debug.Assert(false, ex.Message);

return null;

}

finally

{

Conexion.CerrarConexion(cn);

}

}

protected static ServicioProveedor CargarDatosDesdeReader(IDataRecord fila)

{

ServicioProveedor s = null;

int idMiServicio = fila.IsDBNull(fila.GetOrdinal("idServicio")) ? 0 : fila.GetInt32(fila.GetOrdinal("idServicio"));

string rutProveedor = fila.IsDBNull(fila.GetOrdinal("rutProveedor")) ? "" : fila.GetString(fila.GetOrdinal("rutProveedor"));

string nombreServicio = fila.IsDBNull(fila.GetOrdinal("nombre")) ? "" : fila.GetString(fila.GetOrdinal("nombre"));

string desc = fila.IsDBNull(fila.GetOrdinal("descripcion")) ? "" : fila.GetString(fila.GetOrdinal("descripcion"));

string foto = fila.IsDBNull(fila.GetOrdinal("imagen")) ? "" : fila.GetString(fila.GetOrdinal("imagen"));

if (fila != null)

{

s = new ServicioProveedor()

{

IdServicio = idMiServicio,

RutProveedor = rutProveedor,

Nombre = nombreServicio,

Descripcion = desc,

Foto = foto,

};

}

return s;

}

#endregion

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: TipoEvento.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

using System.Diagnostics;

namespace Dominio

{

public class TipoEvento

{

public int idTipoEvento { get; set; }

public string Nombre { get; set; }

public string Descripcion { get; set; }

public TipoEvento(string Nombre, string Descripcion)

{

this.Nombre = Nombre;

this.Descripcion = Descripcion;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Usuario.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

namespace Dominio

{

public class Usuario

{

public string User { get; set; }

public string Passw { get; set; }

public int Rol { get; set; }

public string Email { get; set; }

#region Encriptar Pass

public static string EncriptarPassSHA512(string inputString)

{

SHA512 sha512 = SHA512.Create();

byte[] bytes = Encoding.UTF8.GetBytes(inputString);

byte[] hash = sha512.ComputeHash(bytes);

return GetStringFromHash(hash);

}

private static string GetStringFromHash(byte[] hash)

{

StringBuilder result = new StringBuilder();

for (int i = 0; i < hash.Length; i++)

{

result.Append(hash[i].ToString("X2"));

}

return result.ToString();

}

#endregion

#region Agregar Usuario

public bool Insertar(SqlCommand cmd)

{

try

{

cmd.CommandText = @"INSERT INTO Usuario

VALUES(@usuario,@password,@rol, @email)";

cmd.Parameters.AddWithValue("@usuario", this.User);

cmd.Parameters.AddWithValue("@password", this.Passw);

cmd.Parameters.AddWithValue("@rol", this.Rol);

cmd.Parameters.AddWithValue("@email", this.Email);

cmd.ExecuteNonQuery();

return true;

}

catch (Exception ex)

{

System.Diagnostics.Debug.Assert(false, "Error: " + ex.Message);

return false;

}

}

#endregion

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IInsertarProveedor.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace InsertarProveedor

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de interfaz "IService1" en el código y en el archivo de configuración a la vez.

[ServiceContract]

public interface IInsertarProveedor

{

[OperationContract]

string GetData(int value);

[OperationContract]

CompositeType GetDataUsingDataContract(CompositeType composite);

// TODO: agregue aquí sus operaciones de servicio

}

// Utilice un contrato de datos, como se ilustra en el ejemplo siguiente, para agregar tipos compuestos a las operaciones de servicio.

[DataContract]

public class CompositeType

{

bool boolValue = true;

string stringValue = "Hello ";

[DataMember]

public bool BoolValue

{

get { return boolValue; }

set { boolValue = value; }

}

[DataMember]

public string StringValue

{

get { return stringValue; }

set { stringValue = value; }

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: InsertarProveedor.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace InsertarProveedor

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de clase "Service1" en el código, en svc y en el archivo de configuración.

// NOTE: para iniciar el Cliente de prueba WCF para probar este servicio, seleccione Service1.svc o Service1.svc.cs en el Explorador de soluciones e inicie la depuración.

public class Service1 : IService1

{

public string GetData(int value)

{

return string.Format("You entered: {0}", value);

}

public CompositeType GetDataUsingDataContract(CompositeType composite)

{

if (composite == null)

{

throw new ArgumentNullException("composite");

}

if (composite.BoolValue)

{

composite.StringValue += "Suffix";

}

return composite;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IAgregarProv.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfAgregarProv

{

[ServiceContract]

public interface IAgregarProv

{

[OperationContract]

bool InsertarProveedor(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass);

[OperationContract]

bool AgregarUsuario(DtoUsuario usu, DtoProveedor prov);

[OperationContract]

bool AgregarServicioAProveedor(DtoServicioProveedor servicio, DtoProveedor prov);

[OperationContract]

bool ExisteRUT(string rut);

[OperationContract]

bool ExisteMail(string rut);

[OperationContract]

IEnumerable<DtoServicio> CargarServicios();

}

[DataContract]

public class DtoProveedor

{

[DataMember]

public string RUT { get; set; }

[DataMember]

public string NombreFantasia { get; set; }

[DataMember]

public string Email { get; set; }

[DataMember]

public Usuario MiUsuario { get; set; } = new Usuario();

[DataMember]

public string Telefono { get; set; }

[DataMember]

public string FechaRegistro { get; set; }

[DataMember]

public bool esInactivo { get; set; }

[DataMember]

public static double Arancel { get; set; }

[DataMember]

public string Tipo { get; set; }

[DataMember]

public List<DtoServicioProveedor> ListaServicios { get; set; }

}

[DataContract]

public class DtoServicio

{

[DataMember]

public int IdServicio { get; set; }

[DataMember]

public string Nombre { get; set; }

[DataMember]

public string Descripcion { get; set; }

[DataMember]

public List<DtoTipoEvento> ListaTipoEventos = new List<DtoTipoEvento>();

}

[DataContract]

public class DtoServicioProveedor

{

[DataMember]

public int IdServicio { get; set; }

[DataMember]

public string RutProveedor { get; set; }

[DataMember]

public string Nombre { get; set; }

[DataMember]

public string Descripcion { get; set; }

[DataMember]

public string Foto { get; set; }

}

[DataContract]

public class DtoTipoEvento

{

[DataMember]

public int idTipoEvento { get; set; }

[DataMember]

public string Nombre { get; set; }

[DataMember]

public string Descripcion { get; set; }

}

[DataContract]

public class DtoUsuario

{

[DataMember]

public string User { get; set; }

[DataMember]

public string Passw { get; set; }

[DataMember]

public int Rol { get; set; }

[DataMember]

public string Email { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioAgregarProv.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfAgregarProv

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de clase "Service1" en el código, en svc y en el archivo de configuración.

// NOTE: para iniciar el Cliente de prueba WCF para probar este servicio, seleccione Service1.svc o Service1.svc.cs en el Explorador de soluciones e inicie la depuración.

public class ServicioAgregarProv : IAgregarProv

{

public bool AgregarServicioAProveedor(DtoServicioProveedor servicio, DtoProveedor prov)

{

bool ret = false;

if (prov.Tipo == "COMUN")

{

//Construye un Proveedor

Proveedor p = new ProveedorComun();

//Inicializa la lista de ServicioProveedor

p.ListaServicios = new List<ServicioProveedor>();

//Con el DtoServicioProveedor que viene por parámetro crea un nuevo ServicioProveedor y lo agrega a la lista del Proveedor

ServicioProveedor miServicio = new ServicioProveedor();

miServicio.IdServicio = servicio.IdServicio;

miServicio.RutProveedor = servicio.RutProveedor;

miServicio.Descripcion = servicio.Descripcion;

miServicio.Foto = servicio.Foto;

ServicioProveedor.InsertarServicioProveedorSinTnr(miServicio);

p.ListaServicios.Add(miServicio);

}

else if (prov.Tipo == "VIP")

{

//Construye un Proveedor

Proveedor p = new ProveedorVIP();

//Inicializa la lista de ServicioProveedor

p.ListaServicios = new List<ServicioProveedor>();

//Con el DtoServicioProveedor que viene por parámetro crea un nuevo ServicioProveedor y lo agrega a la lista del Proveedor

ServicioProveedor miServicio = new ServicioProveedor();

miServicio.IdServicio = servicio.IdServicio;

miServicio.RutProveedor = servicio.RutProveedor;

miServicio.Descripcion = servicio.Descripcion;

miServicio.Foto = servicio.Foto;

ServicioProveedor.InsertarServicioProveedorSinTnr(miServicio);

p.ListaServicios.Add(miServicio);

}

return ret;

}

public bool AgregarUsuario(DtoUsuario usu, DtoProveedor prov)

{

//Construye Usuario a partir de DtoUsuario

Usuario u = new Usuario();

u.Email = usu.Email;

u.Rol = usu.Rol;

u.User = usu.User;

//Construye Proveedor a partir de DtoProveedor

if (prov.Tipo == "COMUN")

{

Proveedor p = new ProveedorComun();

p.RUT = prov.RUT;

p.NombreFantasia = prov.NombreFantasia;

p.Email = prov.Email;

p.Telefono = prov.Telefono;

p.FechaRegistro = prov.FechaRegistro;

p.esInactivo = prov.esInactivo;

p.Tipo = "COMUN";

return p.AgregarUsuario(u);

}

else if (prov.Tipo == "COMUN") {

Proveedor p = new ProveedorComun();

p.RUT = prov.RUT;

p.NombreFantasia = prov.NombreFantasia;

p.Email = prov.Email;

p.Telefono = prov.Telefono;

p.FechaRegistro = prov.FechaRegistro;

p.esInactivo = prov.esInactivo;

p.Tipo = "VIP";

return p.AgregarUsuario(u);

}

return false;

}

public IEnumerable<DtoServicio> CargarServicios()

{

List<DtoServicio> misDtoServicios = new List<DtoServicio>();

List<Servicio> servicios = Servicio.FindAll();

foreach (Servicio s in servicios) {

DtoServicio dtoServicio = new DtoServicio();

dtoServicio.Nombre = s.Nombre;

dtoServicio.Descripcion = s.Descripcion;

dtoServicio.IdServicio = s.IdServicio;

foreach(TipoEvento t in s.ListaTipoEventos)

{

DtoTipoEvento miDtoTipo = new DtoTipoEvento();

miDtoTipo.idTipoEvento = t.idTipoEvento;

miDtoTipo.Nombre = t.Nombre;

miDtoTipo.Descripcion = t.Descripcion;

dtoServicio.ListaTipoEventos.Add(miDtoTipo);

}

misDtoServicios.Add(dtoServicio);

}

return misDtoServicios;

}

public bool ExisteMail(string rut)

{

return Proveedor.ExisteEmail(rut);

}

public bool ExisteRUT(string rut)

{

return Proveedor.ExisteEmail(rut);

}

public bool InsertarProveedor(string rut, string nombreFantasia, string email, string tel, bool esInactivo, bool esVip, string pass)

{

bool ret = false;

DateTime fechaRegDateTime = DateTime.Now;

string fechaRegistro = fechaRegDateTime.ToString("yyyy-MM-dd");

// Construyo un DtoProveedor con los parámetros que llegan desde el cliente y controlo el tipo de proveedor

if (!esVip)

{

DtoProveedor p = new DtoProveedor()

{

RUT = rut,

NombreFantasia = nombreFantasia,

Email = email,

Telefono = tel,

FechaRegistro = fechaRegistro,

esInactivo = esInactivo,

Tipo = "COMUN"

};

string passEncriptada = Usuario.EncriptarPassSHA512(pass);

DtoUsuario usu = new DtoUsuario { User = p.RUT, Passw = passEncriptada, Rol = 2, Email = p.Email };

// Agrego el DtoUsuario al DtoProveedor p

this.AgregarUsuario(usu, p);

//Construye un Proveedor

Proveedor proveedorAInsertar = new ProveedorComun();

proveedorAInsertar.RUT = p.RUT;

proveedorAInsertar.NombreFantasia = p.NombreFantasia;

proveedorAInsertar.Email = p.Email;

proveedorAInsertar.Telefono = p.Telefono;

proveedorAInsertar.FechaRegistro = p.FechaRegistro;

proveedorAInsertar.esInactivo = p.esInactivo;

proveedorAInsertar.Tipo = "COMUN";

//Inserto el Proveedor en BD

proveedorAInsertar.Insertar();

ret = true;

}

else if (esVip)

{

DtoProveedor p = new DtoProveedor()

{

RUT = rut,

NombreFantasia = nombreFantasia,

Email = email,

Telefono = tel,

FechaRegistro = fechaRegistro,

esInactivo = esInactivo,

Tipo = "VIP"

};

string passEncriptada = Usuario.EncriptarPassSHA512(pass);

DtoUsuario usu = new DtoUsuario { User = p.RUT, Passw = passEncriptada, Rol = 2, Email = p.Email };

// Agrego el DtoUsuario al DtoProveedor p

AgregarUsuario(usu, p);

//Construye un Proveedor

Proveedor proveedorAInsertar = new ProveedorVIP();

proveedorAInsertar.RUT = p.RUT;

proveedorAInsertar.NombreFantasia = p.NombreFantasia;

proveedorAInsertar.Email = p.Email;

proveedorAInsertar.Telefono = p.Telefono;

proveedorAInsertar.FechaRegistro = p.FechaRegistro;

proveedorAInsertar.esInactivo = p.esInactivo;

proveedorAInsertar.Tipo = "VIP";

//Inserto el Proveedor en BD

proveedorAInsertar.Insertar();

ret = true;

}

return ret;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: CatalogoServicios.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WCFCatalogoServicios

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de clase "Service1" en el código, en svc y en el archivo de configuración.

// NOTE: para iniciar el Cliente de prueba WCF para probar este servicio, seleccione Service1.svc o Service1.svc.cs en el Explorador de soluciones e inicie la depuración.

public class CatalogoServicios : ICatalogoServicios

{

public IEnumerable<DtoServicio> AgregarServicios()

{

//Obtiene los servicios

IEnumerable<Servicio> listaCompleta = Servicio.FindAll();

List<DtoServicio> listaDtos = new List<DtoServicio>();

if (listaCompleta != null)

{

foreach (Servicio s in listaCompleta)

{

DtoServicio dtoServicio = new DtoServicio

{

IdServicio = s.IdServicio,

Servicio = s.Nombre,

Descripcion = s.Descripcion,

misTiposEventos = new List<DtoTipoEvento>()

};

listaDtos.Add(dtoServicio);

}

}

return listaDtos;

}

public IEnumerable<DtoServicio> ObtenerServicios() {

IEnumerable<DtoServicio> servicios = AgregarServicios();

return servicios;

}

public DtoServicio BuscarServicio(int id)

{

Servicio s = Servicio.FindById(id);

if (s != null) {

List<TipoEvento> tipos = Servicio.FindTiposEventoByServicio(s.Nombre);

List<DtoTipoEvento> dtoTipos = new List<DtoTipoEvento>();

foreach (TipoEvento t in tipos)

{

DtoTipoEvento miTipo = new DtoTipoEvento

{

Nombre = t.Nombre,

Descripcion = t.Descripcion

};

dtoTipos.Add(miTipo);

};

DtoServicio miServicio = new DtoServicio

{

IdServicio = id,

Servicio = s.Nombre,

Descripcion = s.Descripcion,

misTiposEventos = dtoTipos

};

return miServicio;

}

return null;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ICatalogoServicios.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WCFCatalogoServicios

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de interfaz "IService1" en el código y en el archivo de configuración a la vez.

[ServiceContract]

public interface ICatalogoServicios

{

[OperationContract]

IEnumerable<DtoServicio> ObtenerServicios();

[OperationContract]

DtoServicio BuscarServicio(int fila);

}

// Utilice un contrato de datos, como se ilustra en el ejemplo siguiente, para agregar tipos compuestos a las operaciones de servicio.

[DataContract]

public class DtoServicio

{

[DataMember]

public int IdServicio { get; set; }

[DataMember]

public string Servicio { get; set; }

[DataMember]

public string Descripcion { get; set; }

[DataMember]

public List<DtoTipoEvento> misTiposEventos { get; set; }

}

[DataContract]

public class DtoTipoEvento

{

[DataMember]

public string Nombre { get; set; }

[DataMember]

public string Descripcion { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: DesactivarProv.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WCFDesactivarProv

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de clase "Service1" en el código, en svc y en el archivo de configuración.

// NOTE: para iniciar el Cliente de prueba WCF para probar este servicio, seleccione Service1.svc o Service1.svc.cs en el Explorador de soluciones e inicie la depuración.

public class DesactivarProv : IDesactivarProv

{

public bool DesactivarProveedor(DtoProveedor dtoProv)

{

return Proveedor.DesactivarProv(dtoProv.RUT);

}

public DtoProveedor BuscarProveedorRut(string rut)

{

Proveedor miprov = Proveedor.FindByRUT(rut);

miprov.ListaServicios = new List<ServicioProveedor>();

if (miprov == null)

{

return null;

}

else

{

DtoProveedor miDtoProv = new DtoProveedor();

miDtoProv.RUT = miprov.RUT;

miDtoProv.NombreFantasia = miprov.NombreFantasia;

miDtoProv.Email = miprov.Email;

miDtoProv.Telefono = miprov.Telefono;

miDtoProv.FechaRegistro = miprov.FechaRegistro;

miDtoProv.esInactivo = miprov.esInactivo;

miDtoProv.Tipo = miprov.Tipo;

miDtoProv.ListaServicios = new List<DtoServicioProveedor>();

foreach (ServicioProveedor s in miprov.ListaServicios)

{

DtoServicioProveedor miDtoServicioProv = new DtoServicioProveedor();

miDtoServicioProv.IdServicio = s.IdServicio;

miDtoServicioProv.Nombre = s.Nombre;

miDtoServicioProv.RutProveedor = s.RutProveedor;

miDtoServicioProv.Descripcion = s.Descripcion;

miDtoServicioProv.Foto = s.Foto;

miDtoProv.ListaServicios.Add(miDtoServicioProv);

}

return miDtoProv;

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IDesactivarProv.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace WCFDesactivarProv

{

[ServiceContract]

public interface IDesactivarProv

{

[OperationContract]

bool DesactivarProveedor(DtoProveedor dtoProv);

[OperationContract]

DtoProveedor BuscarProveedorRut(string rut);

}

[DataContract]

public class DtoProveedor

{

[DataMember]

public string RUT { get; set; }

[DataMember]

public string NombreFantasia { get; set; }

[DataMember]

public string Email { get; set; }

[DataMember]

public string Telefono { get; set; }

[DataMember]

public string FechaRegistro { get; set; }

[DataMember]

public bool esInactivo { get; set; }

[DataMember]

public string Tipo { get; set; }

[DataMember]

public List<DtoServicioProveedor> ListaServicios { get; set; }

}

[DataContract]

public class DtoServicioProveedor

{

[DataMember]

public int IdServicio { get; set; }

[DataMember]

public string RutProveedor { get; set; }

[DataMember]

public string Nombre { get; set; }

[DataMember]

public string Descripcion { get; set; }

[DataMember]

public string Foto { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Global.asax.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.SessionState;

using Dominio;

using System.IO;

namespace WcfGuardarCatalogoTxt

{

public class Global : System.Web.HttpApplication

{

protected void Application\_Start(object sender, EventArgs e)

{

string parametros = HttpRuntime.AppDomainAppPath + @"config\catalogo.txt";

string directorio = HttpRuntime.AppDomainAppPath + @"config";

if (File.Exists(parametros))

{

}else

{

if (Directory.Exists(directorio))

{

FileStream fs = File.Create(parametros);

}else

{

Directory.CreateDirectory(directorio);

FileStream fs = File.Create(parametros);

}

}

}

public static bool GrabarCatalogo()

{

bool ret = false;

string parametros = HttpRuntime.AppDomainAppPath + @"config\catalogo.txt";

bool grabado = Servicio.grabarCatalogoTxt(parametros);

if (grabado)

{

ret = true;

}else

{

ret = false;

}

return ret;

}

protected void Session\_Start(object sender, EventArgs e)

{

}

protected void Application\_BeginRequest(object sender, EventArgs e)

{

}

protected void Application\_AuthenticateRequest(object sender, EventArgs e)

{

}

protected void Application\_Error(object sender, EventArgs e)

{

}

protected void Session\_End(object sender, EventArgs e)

{

}

protected void Application\_End(object sender, EventArgs e)

{

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: GuardarCatalogoTxt.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfGuardarCatalogoTxt

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de clase "Service1" en el código, en svc y en el archivo de configuración.

// NOTE: para iniciar el Cliente de prueba WCF para probar este servicio, seleccione Service1.svc o Service1.svc.cs en el Explorador de soluciones e inicie la depuración.

public class Service1 : IGuardarCatalogoTxt

{

public string guardarCatalogoTxt()

{

if (Global.GrabarCatalogo())

{

return "El catálogo se grabó correctamente.";

}

else

{

return "Error al guardar";

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IGuardarCatalogoTxt.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace WcfGuardarCatalogoTxt

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de interfaz "IService1" en el código y en el archivo de configuración a la vez.

[ServiceContract]

public interface IGuardarCatalogoTxt

{

[OperationContract]

string guardarCatalogoTxt();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: Global.asax.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.SessionState;

using Dominio;

namespace WCFGuardarProvTxt

{

public class Global : System.Web.HttpApplication

{

protected void Application\_Start(object sender, EventArgs e)

{

string parametros = HttpRuntime.AppDomainAppPath + @"config\proveedores.txt";

string directorio = HttpRuntime.AppDomainAppPath + @"config";

if (File.Exists(parametros))

{

}else

{

if (Directory.Exists(directorio))

{

FileStream fs = File.Create(parametros);

}else

{

Directory.CreateDirectory(directorio);

FileStream fs = File.Create(parametros);

}

}

}

public static bool GrabarProveedores()

{

bool ret = false;

string parametros = HttpRuntime.AppDomainAppPath + @"config\proveedores.txt";

if (Proveedor.grabarProveedoresTxt(parametros))

{

ret = true;

}else

{

ret = false;

}

return ret;

}

protected void Session\_Start(object sender, EventArgs e)

{

}

protected void Application\_BeginRequest(object sender, EventArgs e)

{

}

protected void Application\_AuthenticateRequest(object sender, EventArgs e)

{

}

protected void Application\_Error(object sender, EventArgs e)

{

}

protected void Session\_End(object sender, EventArgs e)

{

}

protected void Application\_End(object sender, EventArgs e)

{

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: GuardarProvTxt.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace WCFGuardarProvTxt

{

public class GuardarProvTxt : IGuardarProvTxt

{

public string guardarProveedoresTxt()

{

if (Global.GrabarProveedores())

{

return "Se han guardado los Proveedores correctamente";

}else

{

return "Error al guardar";

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IGuardarProvTxt.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace WCFGuardarProvTxt

{

[ServiceContract]

public interface IGuardarProvTxt

{

[OperationContract]

string guardarProveedoresTxt();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IServicioListaProv.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfListaProv

{

[ServiceContract]

public interface IServicioListaProv

{

[OperationContract]

IEnumerable<DtoProveedor> ObtenerProveedores();

}

[DataContract]

public class DtoProveedor

{

[DataMember]

public string RUT { get; set; }

[DataMember]

public string NombreFantasia { get; set; }

[DataMember]

public string Email { get; set; }

[DataMember]

public string Telefono { get; set; }

[DataMember]

public string FechaRegistro { get; set; }

[DataMember]

public bool esInactivo { get; set; }

[DataMember]

public string Tipo { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioListaProv.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfListaProv

{

public class ServicioListaProv : IServicioListaProv

{

public IEnumerable<DtoProveedor> ObtenerProveedores()

{

List<Proveedor> listaCompleta = Proveedor.FindAll();

if (listaCompleta == null) return null;

List<DtoProveedor> proveedores = new List<DtoProveedor>();

foreach (Proveedor p in listaCompleta)

{

proveedores.Add(

new DtoProveedor()

{

NombreFantasia = p.NombreFantasia,

RUT = p.RUT,

Email = p.Email,

Telefono = p.Telefono,

FechaRegistro = p.FechaRegistro,

esInactivo = p.esInactivo,

Tipo = p.Tipo,

}

);

}

return proveedores;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IProveedorDadoRUT.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WCFProveedorDadoRUT

{

[ServiceContract]

public interface IProveedorDadoRUT

{

[OperationContract]

DtoProveedor BuscarProveedorRut(string rut);

}

[DataContract]

public class DtoProveedor

{

[DataMember]

public string RUT { get; set; }

[DataMember]

public string NombreFantasia { get; set; }

[DataMember]

public string Email { get; set; }

[DataMember]

public string Telefono { get; set; }

[DataMember]

public string FechaRegistro { get; set; }

[DataMember]

public bool esInactivo { get; set; }

[DataMember]

public string Tipo { get; set; }

[DataMember]

public List<DtoServicioProveedor> ListaServicios { get; set; }

}

[DataContract]

public class DtoServicioProveedor

{

[DataMember]

public int IdServicio { get; set; }

[DataMember]

public string RutProveedor { get; set; }

[DataMember]

public string Nombre { get; set; }

[DataMember]

public string Descripcion { get; set; }

[DataMember]

public string Foto { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ProveedorDadoRUT.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WCFProveedorDadoRUT

{

public class ProveedorDadoRUT : IProveedorDadoRUT

{

public DtoProveedor BuscarProveedorRut(string rut)

{

Proveedor miprov = Proveedor.FindByRUT(rut);

miprov.ListaServicios = new List<ServicioProveedor>();

if(miprov == null)

{

return null;

}else

{

DtoProveedor miDtoProv = new DtoProveedor();

miDtoProv.RUT = miprov.RUT;

miDtoProv.NombreFantasia = miprov.NombreFantasia;

miDtoProv.Email = miprov.Email;

miDtoProv.Telefono = miprov.Telefono;

miDtoProv.FechaRegistro = miprov.FechaRegistro;

miDtoProv.esInactivo = miprov.esInactivo;

miDtoProv.Tipo = miprov.Tipo;

miDtoProv.ListaServicios = new List<DtoServicioProveedor>();

foreach (ServicioProveedor s in miprov.ListaServicios) {

DtoServicioProveedor miDtoServicioProv = new DtoServicioProveedor();

miDtoServicioProv.IdServicio = s.IdServicio;

miDtoServicioProv.Nombre = s.Nombre;

miDtoServicioProv.RutProveedor = s.RutProveedor;

miDtoServicioProv.Descripcion = s.Descripcion;

miDtoServicioProv.Foto = s.Foto;

miDtoProv.ListaServicios.Add(miDtoServicioProv);

}

return miDtoProv;

}

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IServicioListaProv.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfProveedores

{

[ServiceContract]

public interface IServicioListaProv

{

[OperationContract]

IEnumerable<DtoProveedor> ObtenerProveedores();

}

[DataContract]

public class DtoProveedor

{

[DataMember]

public string RUT { get; set; }

[DataMember]

public string NombreFantasia { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioListaProv.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfProveedores

{

public class ServicioListaProv : IServicioListaProv

{

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IServicioModificarParametros.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace WcfServiceModificarParametros

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de interfaz "IService1" en el código y en el archivo de configuración a la vez.

[ServiceContract]

public interface IServicioModificarParametros

{

[OperationContract]

bool ModificarArancel(double value);

[OperationContract]

bool ModificarPorcentajeExtra(int value);

[OperationContract]

decimal ObtenerArancel();

[OperationContract]

int ObtenerPorcentajeExtra();

// TODO: agregue aquí sus operaciones de servicio

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioModificarParametros.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfServiceModificarParametros

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de clase "Service1" en el código, en svc y en el archivo de configuración.

// NOTE: para iniciar el Cliente de prueba WCF para probar este servicio, seleccione Service1.svc o Service1.svc.cs en el Explorador de soluciones e inicie la depuración.

public class ServicioModificarParametros : IServicioModificarParametros

{

public bool ModificarArancel(double nuevoArancel)

{

return Proveedor.ModificarArancel(nuevoArancel);

}

public bool ModificarPorcentajeExtra(int nuevoPorcentaje)

{

return Proveedor.ModificarPorcentajeExtra(nuevoPorcentaje);

}

public decimal ObtenerArancel()

{

return Proveedor.ObtenerArancel();

}

public int ObtenerPorcentajeExtra()

{

return Proveedor.ObtenerPorcentajeExtra();

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IServicioCatalogoServicios.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfServicioCatalogoServicios

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de interfaz "IService1" en el código y en el archivo de configuración a la vez.

[ServiceContract]

public interface IServicioCatalogoServicios

{

[OperationContract]

IEnumerable<DtoServicio> ObtenerServicios();

}

[DataContract]

public class DtoServicio

{

[DataMember]

public string Servicio { get; set; }

[DataMember]

public string Descripcion { get; set; }

[DataMember]

public string Foto { get; set; }

[DataMember]

public List<TipoEvento> TipoEvento { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioCatalogoServicios.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfServicioCatalogoServicios

{

// NOTA: puede usar el comando "Rename" del menú "Refactorizar" para cambiar el nombre de clase "Service1" en el código, en svc y en el archivo de configuración.

// NOTE: para iniciar el Cliente de prueba WCF para probar este servicio, seleccione Service1.svc o Service1.svc.cs en el Explorador de soluciones e inicie la depuración.

public class ServicioCatalogoServicios : IServicioCatalogoServicios

{

public IEnumerable<DtoServicio> ObtenerServicios()

{

List<Servicio> listaCompleta = Servicio.FindServicioTipo();

if (listaCompleta == null) return null;

List<DtoServicio> servicios = new List<DtoServicio>();

foreach (Servicio s in listaCompleta)

{

List<TipoEvento> listaTipoEvento = Servicio.FindTiposEventoByServicio(s.Nombre);

servicios.Add(

new DtoServicio()

{

Servicio = s.Nombre,

Descripcion = s.Descripcion,

Foto = s.Foto,

TipoEvento = listaTipoEvento

}

);

}

return servicios;

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: IServicioExponerCatalogo.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfServicioExponerCatalogo

{

[ServiceContract]

public interface IServicioExponerCatalogo

{

[OperationContract]

IEnumerable<DtoServicio> ObtenerServicios();

/\*

[OperationContract]

IEnumerable<DtoServicioYTiposEvento> ObtenerServiciosYTiposEvento();

\*/

}

/\*

[DataContract]

public class DtoServicioYTiposEvento

{

[DataMember]

public List<TipoEvento> ListaTipoEvento { get; set; }

}

\*/

[DataContract]

public class DtoServicio

{

[DataMember]

public string Servicio { get; set; }

[DataMember]

public string Descripcion { get; set; }

[DataMember]

public string Foto { get; set; }

[DataMember]

public List<TipoEvento> TipoEvento { get; set; }

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Filename: ServicioExponerCatalogo.svc.cs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Dominio;

namespace WcfServicioExponerCatalogo

{

public class ServicioExponerCatalogo : IServicioExponerCatalogo

{

public IEnumerable<DtoServicio> ObtenerServicios()

{

List<Servicio> listaCompleta = Servicio.FindServicioTipo();

if (listaCompleta == null) return null;

List<DtoServicio> servicios = new List<DtoServicio>();

/\*foreach (Servicio s in listaCompleta)

{

List<TipoEvento> listaTipoEvento = Servicio.FindTiposEventoByServicio(s.Nombre);

servicios.Add(

new DtoServicio()

{

Servicio = s.Nombre,

Descripcion = s.Descripcion,

Foto = s.Foto,

TipoEvento = listaTipoEvento

}

);

}\*/

return servicios;

}

}

}