MANUAL TÉCNICO

IMPLEMENTACIÓN DE UN PROTOTIPO DE SOFTWARE DE E-LEARNING PARA LA ENSEÑANZA DE SQLSERVER UTILIZANDO MICROSOFT SILVERLIGHT

1. Introducción

El sitio web es una Aplicación RIA (Rich Internet Applications) que utiliza las siguientes herramientas:

- IDE.- Microsoft Visual Studio 2010
- Tecnología.- Microsoft Silverlight versión 4
- Lenguaje de desarrollo.- C# csharp
- Motor de base de datos.- Microsoft SQL Server 2008.

El detalle está en la sección "5.3. Análisis y diseño" del informe de tesis

2. Objetivo general

Implementar un prototipo de software que ejemplifique la investigación realizada.

3. Objetivos específicos

- Utilizar los controles de usuario investigados en el marco teórico sobre el sitio web desarrollado.
- Implementar una arquitectura estable en n-capas.
- Utilizar un ORM(Entity Framework) en la capa de Acceso a Datos
- Usar una metodología SOA (Windows Comucation Fundation (WCF)) en la capa De Comunicación.

4. Alcance

Esta sección esta detallada en el informe de tesis en el ítem: "5.1. ALCANCE"

5. Diagramas UML

5.1. Diagramas de actividades

Esta sección esta detallada en el informe de tesis en el ítem: "5.3.4. DIAGRAMAS DE ACTIVIDADES"

5.2. Diagramas de Casos de Uso

Esta sección esta detallada en el informe de tesis en el ítem: "5.3.3. DIAGRAMAS DE CASOS DE USO"

5.3. Diagramas se secuencias

Esta sección esta detallada en el informe de tesis en el ítem: “5.3.5. DIAGRAMAS DE SECUENCIA”

5.4. Diagrama Entidad-Relación

Esta sección esta detallada en el informe de tesis en el ítem: “5.3.2. DIAGRAMA DE LA BASES DE DATOS”

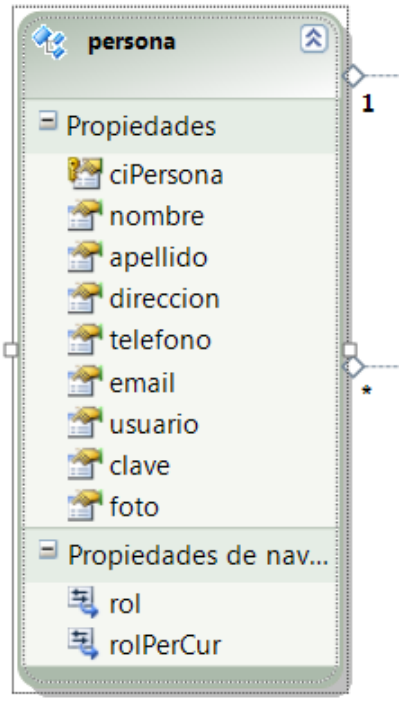
6. Procesos

Se toma como referencia a la clase persona para el análisis en cada capa del aplicativo:

Capa de Acceso a Datos

Explicación: Usó ORM (Entity Framework).

Archivo: ModelPMV.edmx



Código: ModelPMV.tt\ persona.cs

Código	Explicación
`using System;` `using System.Collections.Generic;` `using System.Collections.ObjectModel;` `using System.Collections.Specialized;` `using System.ComponentModel;` `using System.Globalization;` `using System.Runtime.Serialization;`	Librerías
`namespace AccesoDatos.PMV` `{` `[DataContract(IsReference = true)]` `[KnownType(typeof(rol))]` `[KnownType(typeof(rolPerCur))]` `public partial class persona: IObjectWithChangeTracker, INotifyPropertyChanged` `{` `#region Propiedades primitivas` `[DataMember]` `public string ciPersona` `{` `get { return _ciPersona; }` `set` `{` `if (_ciPersona != value)` `{` `if (ChangeTracker.ChangeTrackingEnabled && ChangeTracker.State != ObjectState.Added)` `{` `throw new InvalidOperationException("La propiedad 'ciPersona' forma parte de la clave del objeto y no se puede modificar. Solo se pueden realizar cambios en las propiedades de clave cuando no se realiza un seguimiento del objeto o su estado es Agregado.");` `}` `_ciPersona = value;` `OnPropertyChanged("ciPersona");` `}` `}` `}` `private string _ciPersona;`	Campos o variables
`#endregion` `#region ChangeTracking` `protected virtual void OnPropertyChanged(String propertyName)` `{` `if (ChangeTracker.State != ObjectState.Added && ChangeTracker.State != ObjectState.Deleted)` `{` `ChangeTracker.State = ObjectState.Modified;` `}` `if (_propertyChanged != null)`	Métodos que muestran el cambio de estado de los campos

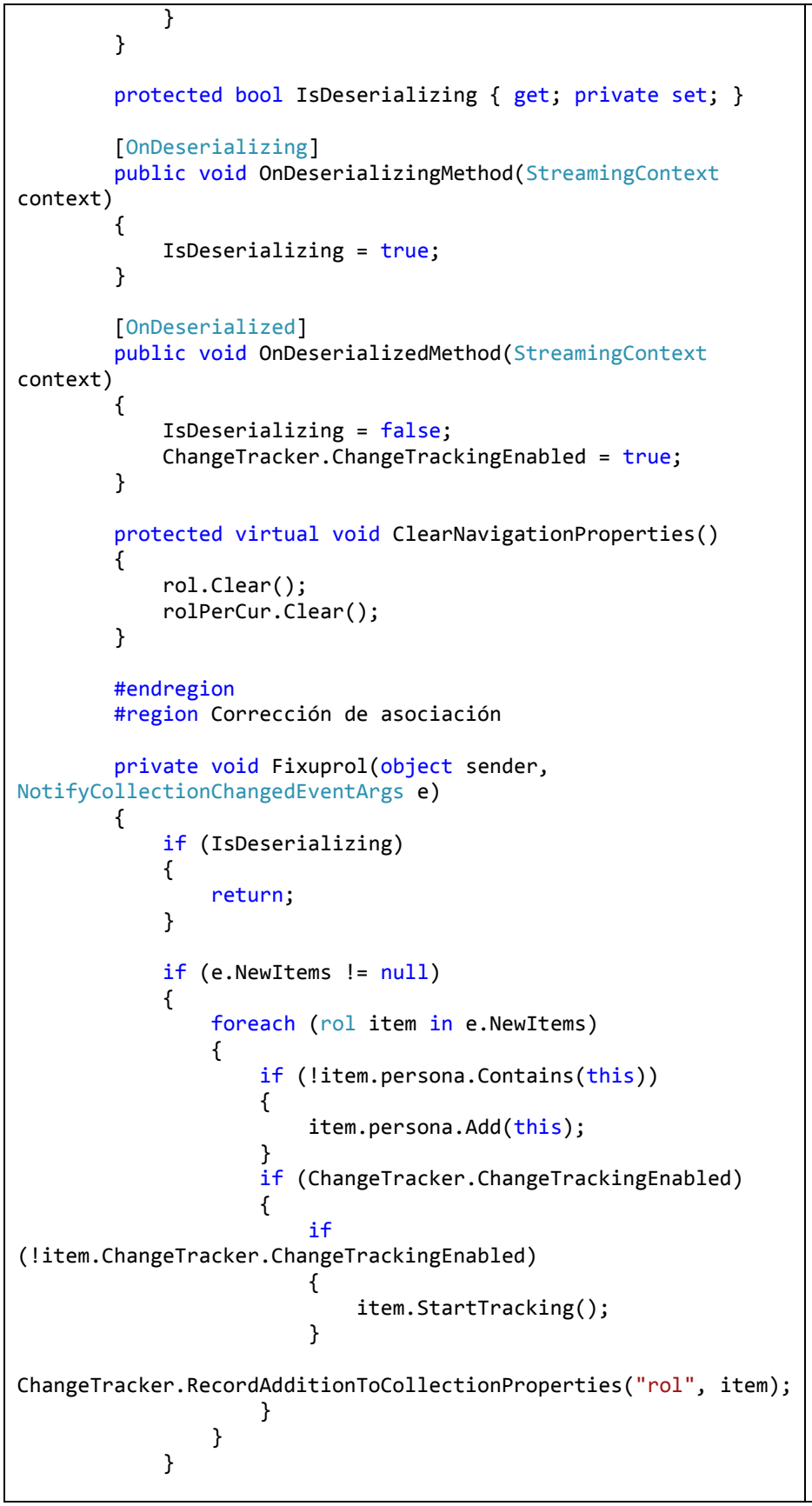
```
            {
                _propertyChanged(this, new
PropertyChangedEventArgs(propertyName));
            }
        }

        protected virtual void
OnNavigationPropertyChanged(String propertyName)
        {
            if (_propertyChanged != null)
            {
                _propertyChanged(this, new
PropertyChangedEventArgs(propertyName));
            }
        }

        event PropertyChangedEventHandler
INotifyPropertyChanged.PropertyChanged{ add { _propertyChanged
+= value; } remove { _propertyChanged -= value; } }
        private event PropertyChangedEventHandler
_propertyChanged;
        private ObjectChangeTracker _changeTracker;

        [DataMember]
        public ObjectChangeTracker ChangeTracker
        {
            get
            {
                if (_changeTracker == null)
                {
                    _changeTracker = new ObjectChangeTracker();
                    _changeTracker.ObjectStateChanging +=
HandleObjectStateChanging;
                }
                return _changeTracker;
            }
            set
            {
                if(_changeTracker != null)
                {
                    _changeTracker.ObjectStateChanging -=
HandleObjectStateChanging;
                }
                _changeTracker = value;
                if(_changeTracker != null)
                {
                    _changeTracker.ObjectStateChanging +=
HandleObjectStateChanging;
                }
            }
        }

        private void HandleObjectStateChanging(object sender,
ObjectStateChangingEventArgs e)
        {
            if (e.NewState == ObjectState.Deleted)
            {
                ClearNavigationProperties();
```

 ``` } } protected bool IsDeserializing { get; private set; } [OnDeserializing] public void OnDeserializingMethod(StreamingContext context) { IsDeserializing = true; } [OnDeserialized] public void OnDeserializedMethod(StreamingContext context) { IsDeserializing = false; ChangeTracker.ChangeTrackingEnabled = true; } protected virtual void ClearNavigationProperties() { rol.Clear(); rolPerCur.Clear(); } #endregion #region Corrección de asociación private void Fixuprol(object sender, NotifyCollectionChangedEventArgs e) { if (IsDeserializing) { return; } if (e.NewItems != null) { foreach (rol item in e.NewItems) { if (!item.persona.Contains(this)) { item.persona.Add(this); } if (ChangeTracker.ChangeTrackingEnabled) { if (!item.ChangeTracker.ChangeTrackingEnabled) { item.StartTracking(); } ChangeTracker.RecordAdditionToCollectionProperties("rol", item); } } } ```	Método que se usa para las asociaciones de tablas (UNION)

```
            if (e.OldItems != null)
            {
                foreach (rol item in e.OldItems)
                {
                    if (item.persona.Contains(this))
                    {
                        item.persona.Remove(this);
                    }
                    if (ChangeTracker.ChangeTrackingEnabled)
                    {

ChangeTracker.RecordRemovalFromCollectionProperties("rol",
item);
                    }
                }
            }
        }

        private void FixuprolPerCur(object sender,
NotifyCollectionChangedEventArgs e)
        {
            if (IsDeserializing)
            {
                return;
            }

            if (e.NewItems != null)
            {
                foreach (rolPerCur item in e.NewItems)
                {
                    item.persona = this;
                    if (ChangeTracker.ChangeTrackingEnabled)
                    {
                        if
(!item.ChangeTracker.ChangeTrackingEnabled)
                        {
                            item.StartTracking();
                        }

ChangeTracker.RecordAdditionToCollectionProperties("rolPerCur",
item);
                    }
                    // Este es el extremo principal en una
asociación que realiza eliminaciones en cascada.
                    // Actualizar la escucha de eventos para que
se refiera al nuevo extremo dependiente.
                    ChangeTracker.ObjectStateChanging +=
item.HandleCascadeDelete;
                }
            }

            if (e.OldItems != null)
            {
                foreach (rolPerCur item in e.OldItems)
                {
                    if (ReferenceEquals(item.persona, this))
                    {
                        item.persona = null;
```

`}` `if (ChangeTracker.ChangeTrackingEnabled)` `{` `ChangeTracker.RecordRemovalFromCollectionProperties("rolPerCur", item);` `// Eliminar el extremo dependiente de esta asociación de identificación. Si el estado actual es agregado,` `// permite que la relación se modifique sin eliminar el elemento dependiente.` `if (item.ChangeTracker.State != ObjectState.Added)` `{` `item.MarkAsDeleted();` `}` `}` `// Este es el extremo principal en una asociación que realiza eliminaciones en cascada.` `// Quitar el extremo dependiente anterior de la escucha de eventos.` `ChangeTracker.ObjectStateChanging -= item.HandleCascadeDelete;` `}` `}` `}` `#endregion` `}` `}`	

Capa de Negocio

Explicación: Capa que interactúa con su capa inferior Acceso a Datos para mostrar la lógica del negocio juntamente con la Store Procedures de la Base de datos.

Archivo: LogicaNegocio\ ReglasNegocio.cs

Código

Código	Explicación
`public List<persona> GetPersonas()` `{` `return new PersonaRepository().GetPersonas();` `}` `public List<persona> GetPersonasFiltro(string _ciPersona)` `{` `return new PersonaRepository().GetPersonasFiltro(_ciPersona);` `}`	Llamadas a la capa e Acceso a Datos

```
        public persona ValidarUsuario(string
_usuario, string _clave)
        {
                List<persona> _lstPersonas = new
List<persona>();
                _lstPersonas =
GetPersonas().Where(x => x.usuario == _usuario &&
x.clave == _clave).ToList();

                if (_lstPersonas.Count == 1)
                    return _lstPersonas.First();
                else
                    return new persona();
        }

        public List<rol> ObtenerRolesLogIN(string
cedula)
        {
                List<rolesUsuario> lstRoles = new
List<rolesUsuario>();
                lstRoles =
GetRolesPersona(cedula).Where(x => x.estado ==
true).ToList();

                if (lstRoles.Count > 0)
                {
                    List<rol> temp = new
List<rol>();

lstRoles.ForEach(delegate(rolesUsuario reg)
                    {

temp.Add(GetRol(reg.idRol));
                    });

                    return temp;
                }
                else
                    return new List<rol>();
        }


        public OperationStatus SavePersona(persona
_persona, List<rolesUsuario> roles)
        {
            return new
PersonaRepository().SavePersona(_persona, roles);
        }

        public OperationStatus SavePersona(persona
_persona)
        {
            return new
PersonaRepository().SavePersona(_persona);
        }
```

Capa de Comunicación

Explicación: Capa que interactúa con su capa inferior Negocio o (BBL) y expone en una interface las funciones del aplicativo. Se toma como un proyecto WCF que puede ser montado en un servidor web "IIS" local o remoto.

Código

Código	Explicación
Archivo: Service1.cs	
`public persona GetPersona(string _ciPersona)` `{` `try` `{` `LogicaNegocio.ReglasNegocio obj = new LogicaNegocio.ReglasNegocio();` `return obj.GetPersona(_ciPersona);` `}` `catch { throw; }` `}` `public List<persona> GetPersonas()` `{` `try` `{` `LogicaNegocio.ReglasNegocio obj = new LogicaNegocio.ReglasNegocio();` `return obj.GetPersonas();` `}` `catch { throw; }` `}` `public _Excepcion_lstpersona GetPersonasFiltro(string _ciPersona)` `{` `try` `{` `LogicaNegocio.ReglasNegocio obj = new LogicaNegocio.ReglasNegocio();` `return new _Excepcion_lstpersona()` `{` `lstpersona = obj.GetPersonasFiltro(_ciPersona),` `_Error = null`	Comunicación con la capa de Negocio

```
                };
            }
            catch (Exception error)
            {
                return new
_Excepcion_lstpersona()
                {
                    lstpersona = new
List<persona>(),
                    _Error = new
Error(error.Message, error.StackTrace)
                };
            }

        }

        public persona
ValidarUsuario(string _usuario, string
_clave)
        {
            try
            {
                List<persona>
_lstPersonas=new List<persona>();

_lstPersonas=GetPersonas().Where(x=>
x.usuario==_usuario &&
x.clave==_clave).ToList();


if(_lstPersonas.Count==1)
                    return
_lstPersonas.First();
                else
                    return new
persona();
            }
            catch { throw; }
        }
```

Archivo: IService1.cs.- Métodos expuestos por el servicio

```
[OperationContract]
        List<persona> GetPersonas();

        [OperationContract]
        _Excepcion_lstpersona GetPersonasFiltro(string _ciPersona);

        [OperationContract]
        _Excepcion_OperationStatus SavePersona(persona _persona, List<rolesUsuario>
roles);

        [OperationContract]
        OperationStatus SavePersonaSingle(persona _persona);
```

Capa de Presentación

Explicación: Se usa tecnología Microsoft Silverlihgt 4.0

Proyecto: PMV, PMV.Web

Código

Código	**Explicación**
Archivo: PMV\ Views\ DatosPersona.xaml	
Página:	

Validación de campos

```
<dataInput:Label Height="28" Width="100"  Margin="4" Target="{Binding
ElementName=txtCedula}" HorizontalAlignment="Right" />
                <TextBox Height="23" Width="171" x:Name="txtCedula" MaxLength="10"
Text="{Binding Mode=TwoWay, ValidatesOnExceptions=true,
NotifyOnValidationError=true, Path=CurrentPersona.ciPersona}"  Margin="4"
KeyUp="txtCedula_KeyUp" />
                <dataInput:DescriptionViewer Margin="4" Target="{Binding
ElementName=txtCedula}" />
```

Objeto: Se usa DataAnnotations

```
[DataMember]
        [StringLength(10,MinimumLength = 10 ,ErrorMessage = "Debe ingresar 10
caracteres")]
        [Required(ErrorMessage = "Campo requerido")]
        [Display (Name="Cédula",Description="Cédula de ciudadanía")]
        [RegularExpression(@"^\d\d\d\d\d\d\d\d\d\d$", ErrorMessage = "Ingresar 10
caracteres enteros.")]
        public string ciPersona
        {
            get { return _ciPersona; }
            set
            {
                if (_ciPersona != value)
                {
                    if (ChangeTracker.ChangeTrackingEnabled && ChangeTracker.State
!= ObjectState.Added)
                    {
                        throw new InvalidOperationException("La propiedad
```

```
'ciPersona' forma parte de la clave del objeto y no se puede modificar. Solo se
pueden realizar cambios en las propiedades de clave cuando no se realiza un
seguimiento del objeto o su estado es Agregado.");
                    }


                    Validator.ValidateProperty(value, new ValidationContext(this,
null, null) { MemberName = "ciPersona" });

                    _ciPersona = value;
                    OnPropertyChanged("ciPersona");
                }
            }
        }
        private string _ciPersona;
```

Archivo:
UserControls\ ucDatosPersona.xaml,
ViewModels\ IngresoDatosViewModel.cs

```
PMV._proxy.Service1Client proxy = new PMV._proxy.Service1Client();
            proxy.SavePersonaCompleted += (a, b) => {
                if (b.Result._Error != null)
                {
                    new ErrorWindow(b.Result._Error.Message,
b.Result._Error.StackTrace).Show();
                }
                else
                {
                    StatusMessage = (b.Result._OperationStatus.Status) ?
"Success!" : "Operación realizada con exito";
                }

                GetPersonas();
                _RolesStatic = new ObservableCollection<rolesUsuario>();
            };
            proxy.SavePersonaAsync(CurrentPersona,_RolesStatic);
```

Llamadas lambda a las funciones expuestas en el servicio web