**ESTRUCTURA DE LA APLICACIÓN**

ModulOTEC utiliza la siguiente distribución de directorios en etapa de desarrollo.

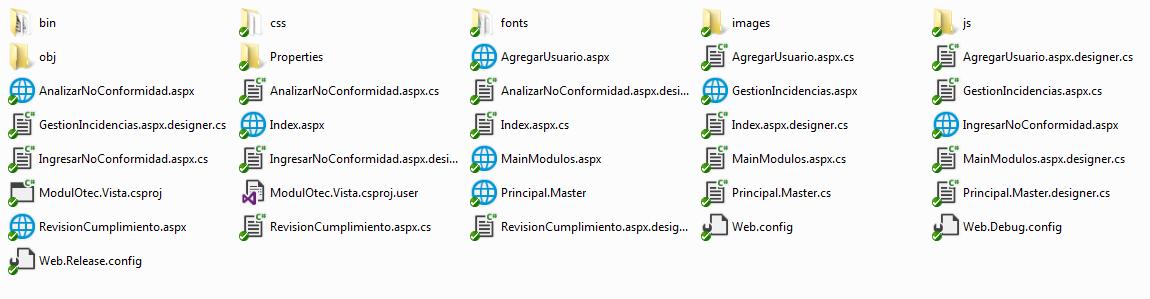


Ilustración 1 - Vista del Directorio de la Aplicación

**Bin:** ASP.NET reconoce la carpeta Bin de un sitio web como una carpeta especial para determinados tipos de contenido. Una carpeta Bin está diseñada para contener ensamblados compilados (archivos .dll) para controles y componentes ASP.NET personalizados, en este caso se encuentran los archivos de EntityFramework y los dll de los apartados de vistas, entidades, datos y negocio del sistema.

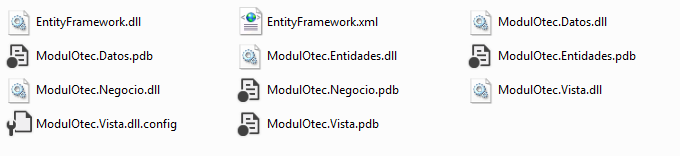


Ilustración 2 - Carpeta bin

**Css:** En esta carpeta se encuentran los archivos de hojas de estilos externos que utilizará el sistema, acá podemos hallar las hojas de estilo del framework css Bootstrap, y el archivo css con los estilos personalizados de esta web.

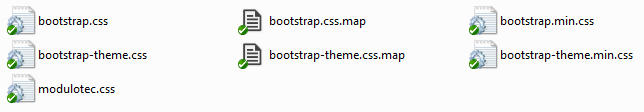


Ilustración 3 - Carpeta css

**Fonts:** Se almacenan las fuentes que utiliza actualmente sistema.

**Images:** Se almacenan las fuentes que utiliza actualmente sistema.

**Js:** En esta carpeta se incluyen las librerías JavaScript del proyecto, se pueden observar las librerías js de Bootstrap y Jquery.

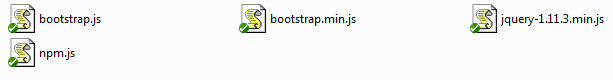


Ilustración 4 - Carpeta js

**Obj:** Se almacenan archivos generados por Visual Studio en el proceso de depuración del sistema.

**Properties:** Se guardan archivos que guardan la información general sobre un ensamblado.

**CAPA DE LÓGICA DE NEGOCIO**

La capa de negocio se comunica con la capa de presentación, para recibir las solicitudes de información y presentar los resultados, y con la capa de datos para almacenar o recuperar datos de la base de datos.

**Estructura de la capa**

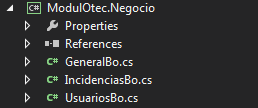


Ilustración 5 - Estructura Capa de Negocio

Se pueden observar tres clases dentro de la estructura de la capa, GeneralBo.cs, IncidenciasBo.cs y UsuariosBo.cs, además incorpora las propiedades de la capa (Properties) y las Referencias a diferentes componentes del proyecto.

**GeneralBo (Clase)**

Clase utilizada con la finalidad de cargar la información general de las vistas

**GeneralBo (using):** Bibliotecas de clases que incluye la Clase GeneralBo.

**System:** Contiene clases fundamentales y clases base que definen tipos de datos de referencia y de valor de uso frecuente, eventos y controladores de eventos, interfaces, atributos y excepciones de procesamiento.

**System.Collections.Generic:** Contiene interfaces y clases que definen colecciones genéricas, lo que permite que los usuarios creen colecciones fuertemente tipadas para proporcionar una mayor seguridad de tipos y un rendimiento mejor que los de las colecciones no genéricas fuertemente tipadas.

**System.Linq:** Proporciona clases e interfaces que admiten consultas que utilizan Language-integrated query (LINQ).

**Entidades:** Contiene clases para la utilización de la base de datos a través de objetos y entidades.

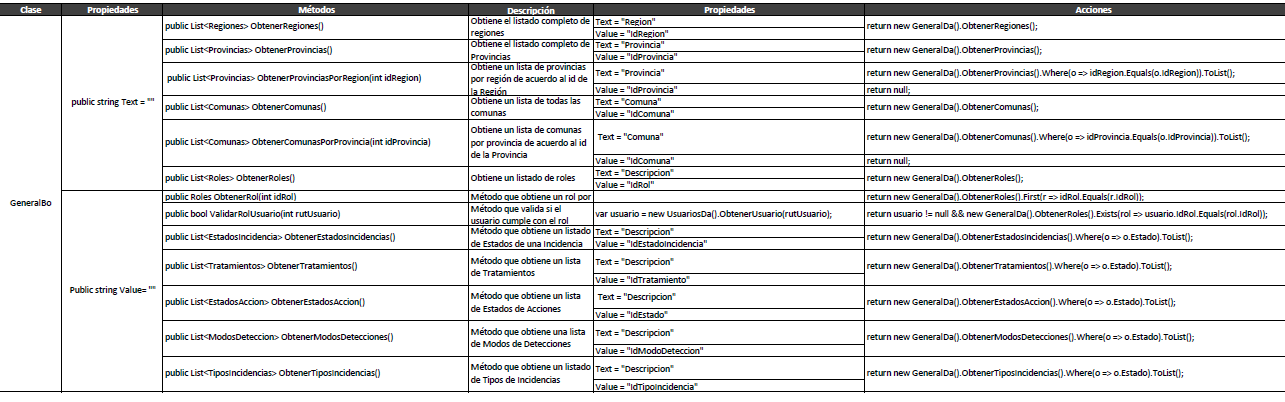
**Datos:** Contiene acceso a las clases y métodos de la capa de datos del sistema, para solicitar el almacenamiento o la recuperación de datos.

Ilustración 6 - Métodos Clase GeneralBo

**IncidenciasBo (Clase)**

**IncidenciasBo (using):** Bibliotecas de clases que incluye la Clase **IncidenciasBo**.

**System.Collections.Generic:** Contiene interfaces y clases que definen colecciones genéricas, lo que permite que los usuarios creen colecciones fuertemente tipadas para proporcionar una mayor seguridad de tipos y un rendimiento mejor que los de las colecciones no genéricas fuertemente tipadas.

**Datos:** Contiene acceso a las clases y métodos de la capa de datos del sistema, para solicitar el almacenamiento o la recuperación de datos.

**Entidades:** Contiene clases para la utilización de la base de datos a través de objetos y entidades.

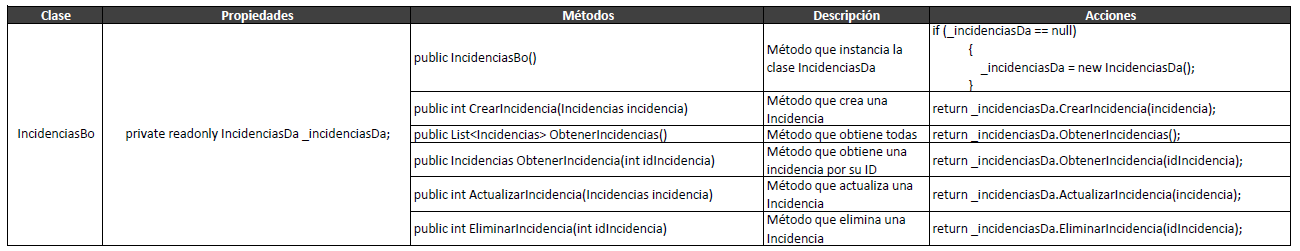


Ilustración 7 - Métodos Clase IncidenciasBo

**UsuariosBo (Clase)**

**UsuariosBo (using):** Bibliotecas de clases que incluye la Clase **UsuariosBo.**

**System.Collections.Generic:** Contiene interfaces y clases que definen colecciones genéricas, lo que permite que los usuarios creen colecciones fuertemente tipadas para proporcionar una mayor seguridad de tipos y un rendimiento mejor que los de las colecciones no genéricas fuertemente tipadas.

**Datos:** Contiene acceso a las clases y métodos de la capa de datos del sistema, para solicitar el almacenamiento o la recuperación de datos.

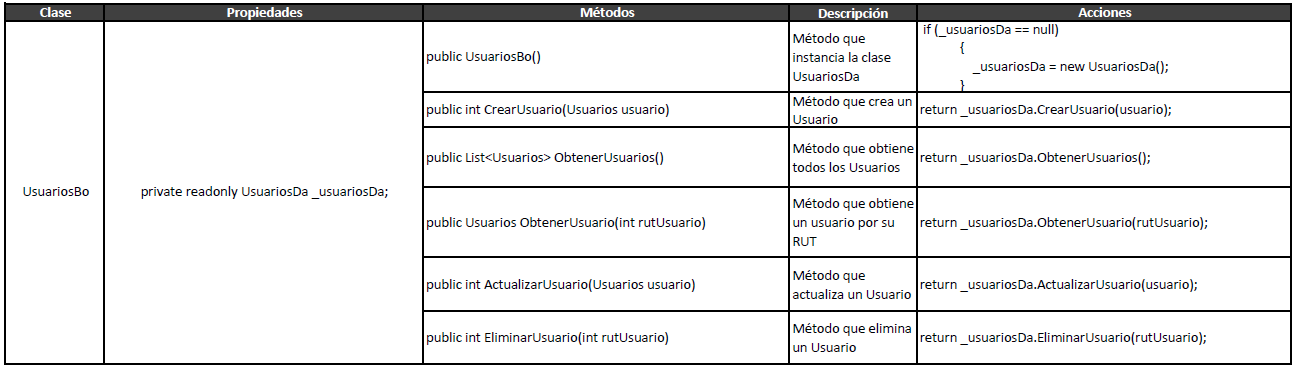
**Entidades:** Contiene clases para la utilización de la base de datos a través de objetos y entidades.

Ilustración 8 - Métodos Clase UsuariosBo

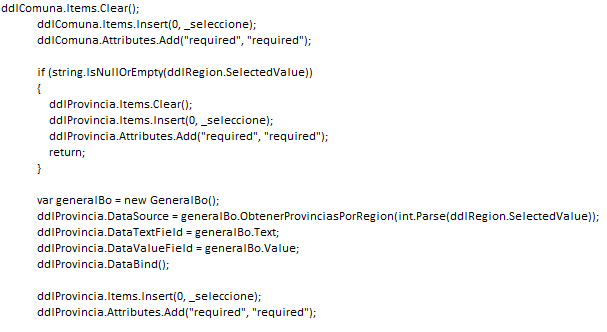
.

**Item:** AgregarUsuario.aspx

**Control de formulario:** DropDownList "ddlRegion"

**Evento:** OnSelectedIndexChanged="RegionSelectedIndexChanged"

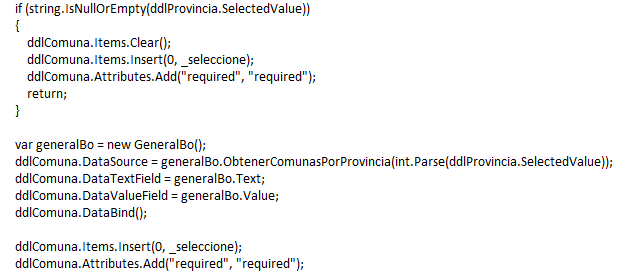
**Código Fuente:**



**Descripción:** Al cambiar la opción seleccionada deldropdownlist "ddlRegion", realiza un auto post back y carga las provincias de acuerdo al Identificador de la Región seleccionada.

**Item:** DropDownList "ddlProvincia"

**Control de formulario:** OnSelectedIndexChanged="ProvinciaSelectedIndexChanged"

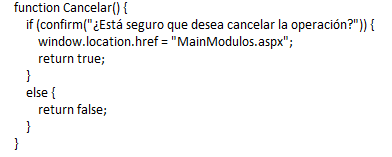
**Código Fuente:** 

**Descripción:** Al cambiar la opción seleccionada del dropdownlist "ddlProvincia", realiza un auto post back y carga las Comunas de acuerdo al Identificador de la Provincia seleccionada.

**Control de Formulario: Button "btnCancelar"**

**Evento:** **OnClientClick="return Cancelar();"**

**Código Fuente:**

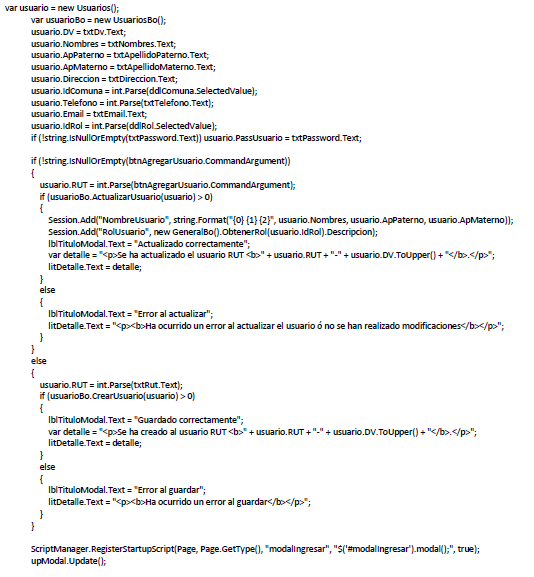


**Descripción:** Validación Javascript, al ser presionado el botón Cancelar, la validación es activada, el código genera una ventana de confirmación, preguntando al Usuario si está seguro que desea cancelar la operación.

**Control de Formulario:** Button "btnAgregarUsuario"

**Evento:** OnClick="AgregarUsuarioClick"

**Código Fuente:**



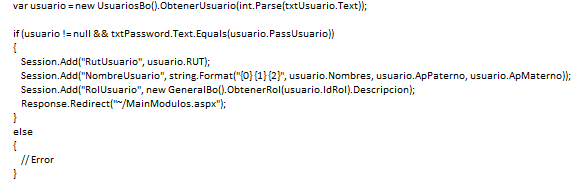
**Descripción:** Al darle click al botón Agregar Usuario, el sistema comprueba que los datos ingresados estén validados, asigna los campos de texto y seleccionables a las propiedades de la clase Usuarios(), en caso de que el proceso sea satisfactorio se genera un Modal al usuario que muestra en mensake "Se ha actualizado el usuario (RUT del usuario actualizado)", en caso de que la operación falle muestra el siguiente mensaje en el modal "Ha ocurrido un error al actualizar el usuario o no se han realizado modificaciones".

**Item:** Index.aspx

**Control de Formulario:** LinkButton "lnkIngresar"

**Evento:** OnClick="IngresarOnClick"

**Código Fuente:**



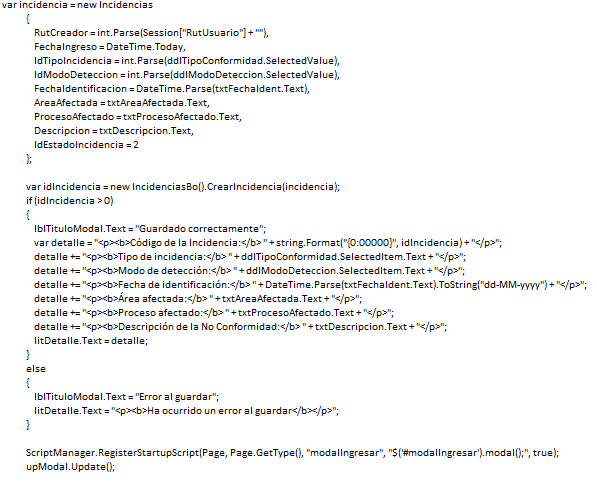
**Descripción:** Obtiene los datos ingresados en las cajas de texto del formulario y los compara a través de la capa de datos con la información del usuario, en caso de coicindir ambos campos (usuario y contraseña) redirecciona a la pagina MainModulos.aspx, en caso de error muestra un mensaje de error.

**ITEM:** IngresarNoConformidad.aspx

**Control de Formulario:** Button "btnIngresar"

**Evento:** OnClick="IngresarIncidencia"

**Código Fuente:**

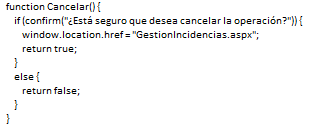


**Descripción:** Asigna los valores de los campos del formulario a las propiedades de las Incidencia, formateando los datos para que coincidan con los especificados en la base de datos, Ingresa la incidencia, en caso de que el proceso sea satisfactorio muestra una ventana con los detalles de la inserción, el caso de error, muestra el siguiente mensaje de error "Ha ocurrido un error al guardar".

**Control de Formulario:** Button "btnCancelar"

**Evento:** onclick="return Cancelar();"

**Código Fuente:**

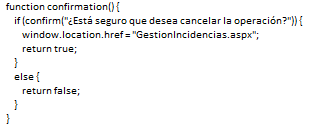


**Descripción:** Añade un archivo seleccionable desde el equipo, hacia el sistema, se asocia al número de análisis en caso de ser correctamente asociado, en caso de error muestra el siguiente mensaje "Ha ocurrido un error al guardar el documento".

**Control de Formulario:** Button "btnCancelar"

**Evento:** OnClientClick="return confirmation();"

**Código Fuente:**



**Descripción:** Validación Javascript, al ser presionado el botón Cancelar, la validación es activada, el código genera una ventana de confirmación, preguntando al Usuario si está seguro que desea cancelar la operación.

**CAPA DE ACCESO A DATOS**

Es donde se escribe el código que habla con la base de datos. Es en esta capa donde se definen las consultas a la base de datos (el SQL). No realiza validaciones entre entidades, sólo las validaciones que van hacia la base de datos. La intención de esta capa es que el resto de la aplicación no se preocupe de los detalles (en cierta medida) de la estructura de la base de datos y trabaje a un nivel de objetos. Es la encargada de la entrada y salida de datos hacia y desde la base de datos. Ya que no debe de preocuparse de las reglas de negocio el código de la capa debería de ser fácil de seguir y de mantener. Las capas construidas encima de esta (es decir que usen esta capa) deben de dejar de pensar en tablas y registros para pensar en objetos y colecciones.

**Estructura de la Capa**

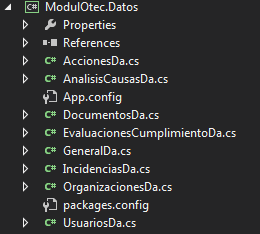
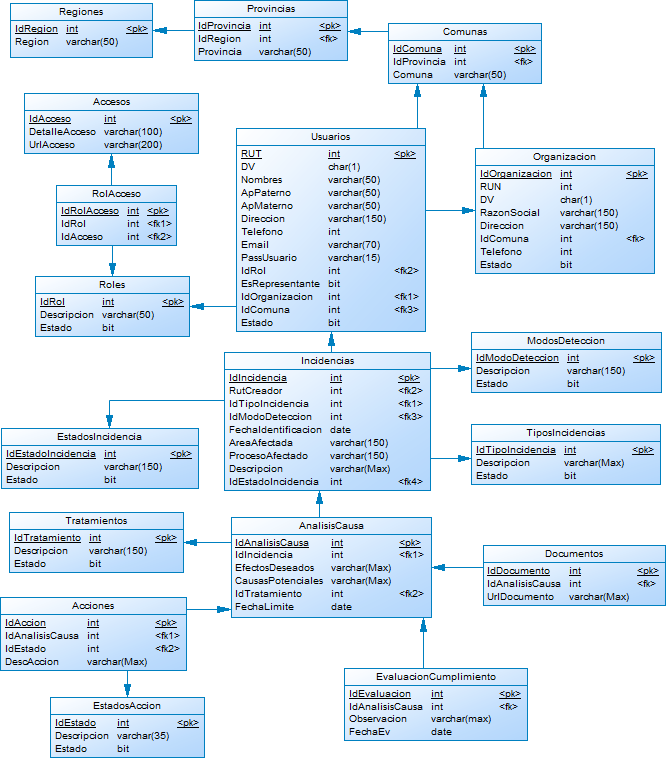


Ilustración 9 - Estructura capa de datos

**MODELO DE DATOS**

****

**LENGUAJE DE MANIPULACIÓN Y DEFINICIÓN DE DATOS**

Una vez creado el modelo de datos es necesaria la utilización de un lenguaje que nos permita la manipulación y el trabajo con la base de datos que almacenará la información.

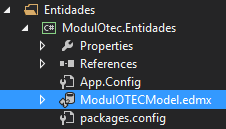
Este proyecto hace uso de SQL Server 2008 R2 para el almacenamiento de los datos, aplicando Lenguaje de Manipulación de Datos no procedural, justificado en la utilización del Framework Entity de Microsoft®.

**ENTITY FRAMEWORK**

“Entity Framework es la tecnología de acceso a datos de Microsoft recomendada para las nuevas aplicaciones.”

Se incorpora a este proyecto ModulOTEC Entity Framework dada su integración nativa con las tecnologías .NET, sus características permiten trabajar con datos en forma de objetos y propiedades específicos del dominio, como clientes y direcciones de cliente, sin tener que preocuparse por las tablas y columnas de la base de datos subyacente donde se almacenan estos datos. Con Entity Framework, se puede trabajar en un nivel mayor de abstracción al tratar con datos, y pueden crear y mantener aplicaciones orientadas a datos con menos código que en las aplicaciones tradicionales.

Se utiliza una importación al proyecto directamente desde VisualStudio el Modelo de Datos a utilizar a la capa de Entidades, evitando el trabajo con consultas sql desde la aplicación, esto gracias a Entity Framework.



**CADENA DE CONEXIÓN SQL SERVER**

<connectionStrings>

<add name="ModulOTECEntities" connectionString="metadata=res://\*/ModulOTECModel.csdl|res://\*/ModulOTECModel.ssdl|res://\*/ModulOTECModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;**data source=localhost**;**initial catalog=ModulOTEC**;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient" />

</connectionStrings>

**Data source:** Referencia al servidor donde se aloja el sistema, en el caso específico de la etapa de desarrollo del sistema este se encuentra corriendo desde un servidor local, por ello se incorpora la cadena “localhost”.

**Initial catalog:** En este apartado se identifica la base de datos creada en el servidor con la que trabajará el sistema (ModulOTEC), debe ser consistente con el modelo de datos incorporado al momento del desarrollo.

**DEFINICIONES**

**LINQ:** Language-Integrated Query (LINQ) es un conjunto de características presentado en Visual Studio 2008 que agrega capacidades de consulta eficaces a la sintaxis de los lenguajes C# y Visual Basic.LINQ incorpora patrones fáciles y estándar para consultar y actualizar datos, y la tecnología se puede ampliar para proporcionar compatibilidad prácticamente con cualquier tipo de almacén de datos.Visual Studio incluye ensamblados de proveedor LINQ que habilitan el uso de LINQ con colecciones de .NET Framework, bases de datos de SQL Server, conjuntos de datos ADO.NET y documentos XML.