	UNIVERSIDAD DON BOSCO **FACULTAD DE ESTUDIOS TECNOLÓGICOS** **ESCUELA DE COMPUTACIÓN**
CICLO: 01/2019	*GUIA DE LABORATORIO #3* **Nombre de la Práctica:** **Controles Web y Validación de Formularios Web en ASP.NET** **Lugar de Ejecución:** **Centro de cómputo** **Tiempo Estimado:** **2 horas con 30 minutos** **MATERIA:** **Lenguaje de Programación II**

I. OBJETIVOS

En esta guía de práctica se pretende:
1. Introducir a los estudiantes en el uso de controles web para registrar la información en una página web
2. Desarrollar las habilidades necesarias para manipular mediante programación la interfaz creada para una página web
3. Hacer uso de fuentes de datos no tradicionales para la gestión de información en formularios
4. Implementar controles de validación combinados con controles para asegurar la integridad de la información recibida

II. INTRODUCCIÓN TEÓRICA

Controles Web en ASP.NET
Los controles de servidor web ASP.NET son objetos de páginas web ASP.NET que se ejecutan cuando se solicita la página y que representan marcado en un explorador. Muchos controles de servidor web son similares a elementos HTML conocidos, como botones y cuadros de texto. Otros controles abarcan comportamiento complejo, por ejemplo controles de calendario y controles que administran las conexiones de datos.

Los controles web tienen propiedades similares a las de los controles de formulario de Windows incluyendo Texto, Habilitado, Visible, Font, ReadOnly, y así sucesivamente. Hay algunas diferencias importantes:

- La propiedad de control de Windows (Name) tiene el mismo nombre que la propiedad ID de controles Web
- Los controles web tienen una propiedad AutoPostBack
- Los controles web pierden las propiedades de ejecución cuando el usuario se aleja de esa página
- Debe guardar el estado de conservar las propiedades de ejecución

Tipos de controles soportados por ASP.NET
Cuando crea páginas Web ASP.NET, puede utilizar estos tipos de controles:

- Controles de servidor HTML: Elementos HTML expuestos al servidor para que se puedan programar. Los controles de servidor HTML exponen un modelo de objeto que se relacionan muy estrechamente con los elementos HTML que representan.
- Controles de servidor Web: Controles con más funciones incorporadas que los controles de servidor HTML. Los controles de servidor Web incluyen no sólo controles de formulario como botones y cuadros de texto, sino también controles con fines especiales como un calendario, menús y un control de vista de árbol. Los controles de servidor Web son más abstractos que los controles de servidor HTML pues su modelo de objetos no refleja necesariamente la sintaxis HTML.
- Controles de validación: Controles que incorporan lógica para permitirle comprobar los controles de entrada de los usuarios como el control TextBox. Los controles de validación le permiten comprobar

un campo necesario, su adecuación a un valor o un modelo de caracteres concreto, comprobar que un valor se encuentra en un intervalo predefinido, etc.

- Controles de usuario: Controles que crea como páginas Web ASP.NET. Se pueden incrustar controles de usuario de ASP.NET en otras páginas Web ASP.NET; esta es una forma sencilla de crear barras de herramientas y otros elementos reutilizables.

Controles de servidor HTML

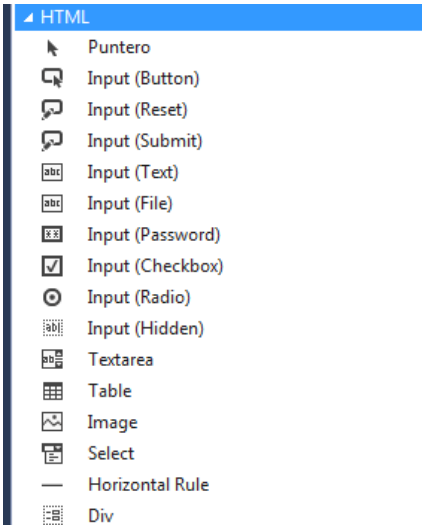
Los controles de servidor HTML son elementos HTML (o elementos en otro marcado compatible, como XHTML) que contienen atributos que los convierten en programables en código del servidor. De forma predeterminada, los elementos HTML en una página Web ASP.NET no están disponibles para el servidor. En su lugar, se tratan como texto opaco y se pasan al explorador. Sin embargo, cuando se convierten en controles de servidor HTML, los elementos HTML quedan expuestos como elementos programables en el servidor.
El modelo de objetos de los controles de servidor HTML se relaciona estrechamente con el de los elementos correspondientes. Por ejemplo, los atributos HTML se exponen en controles de servidor HTML como propiedades.

Cualquier elemento HTML de una página se puede convertir en control de servidor HTML agregando el atributo runat="server". Durante el análisis, el marco de trabajo de la página ASP.NET crea instancias de todos los elementos que contienen el atributo runat="server". Si desea hacer referencia al control como un miembro dentro del código, también deberá asignarle un atributo id al control.

El marco de trabajo de la página proporciona controles de servidor HTML predefinidos para los elementos HTML que se utilizan con más frecuencia dinámicamente en una página: el elemento form, los elementos input (cuadro de texto, casilla, botón Enviar), el elemento select, etc. Estos controles de servidor HTML predefinidos comparten las propiedades básicas del control genérico y, además, cada control normalmente proporciona su propio conjunto de propiedades y su propio evento.

Los controles de servidor HTML ofrecen las funciones siguientes:

- Un modelo de objetos que pueda volver a programar en el servidor con las técnicas habituales orientadas a objetos. Los controles de servidor exponen propiedades que permiten manipular los atributos de marcado del control mediante programación en el código del servidor.
- Un conjunto de eventos para los que pueda escribir controles de eventos de la misma forma que lo haría en un formulario basado en cliente, con la excepción de que un evento se controla en código del servidor.
- La capacidad de controlar eventos en un script de cliente.
- Mantenimiento automático del estado del control. Cuando la página realiza una acción de ida y vuelta al servidor, los valores que el usuario escriba en los controles de servidor HTML se mantendrán automáticamente y la página se devuelve al explorador.
- Interacción con los controles de validación ASP.NET para poder comprobar que un usuario ha escrito la información adecuada en un control.
- Enlace de datos a una o varias de las propiedades del control.
- Compatibilidad con estilos si la página Web ASP.NET se muestra en un explorador que admite hojas de estilos en cascada.
- Paso a través de atributos personalizados. Pueden agregarse los atributos que se necesiten a un control de servidor HTML: el marco de trabajo de páginas los representará sin ningún cambio en la funcionalidad. Esto permite agregar atributos específicos del explorador a los controles.



Controles de servidor web

Los controles de servidor Web son un segundo conjunto de controles diseñado con otro enfoque. No se asignan necesariamente uno a uno a controles de servidor HTML. En lugar de ello, se definen como controles abstractos, en los que el marcado real representado por el control puede ser muy diferente al modelo con respecto al que se han programado. Por ejemplo, un control RadioButtonList de servidor Web podría representarse en una tabla o como un texto en línea con otro marcado.

Los controles de servidor Web incluyen controles de formulario tradicionales como botones y cuadros de texto, además de controles complejos, como, por ejemplo, las tablas. También incluyen controles que proporcionan funcionalidad de formulario de uso frecuente, como la presentación de datos en cuadrícula, la elección de fechas, la visualización de menús, etc.

Los controles de servidor Web ofrecen todas las funciones descritas anteriormente para los controles de servidor HTML (excepto la asignación uno a uno a elementos) y estas funciones adicionales:

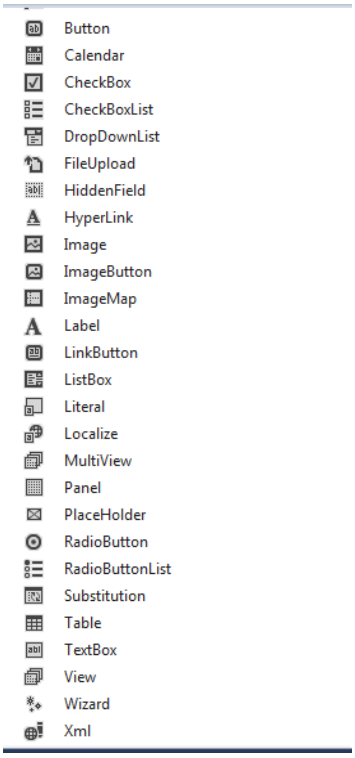
- Un modelo de objetos enriquecido que proporciona capacidades de programación de tipo seguro.
- Detección automática del explorador. Los controles pueden detectar las funciones del explorador y representar el marcado adecuado.
- Para algunos controles, la capacidad para definir su propio diseño para el control utilizando Templates.
- Para algunos controles, la capacidad de especificar si un evento del control provoca un envío inmediato al servidor o, en su lugar, se almacena en caché y se activa cuando se envía la página.
- Compatibilidad para temas, lo que le permite definir un aspecto uniforme para los controles en todo el sitio.
- Capacidad para pasar eventos de un control anidado (como un botón en una tabla) al control contenedor.

Los controles utilizan una sintaxis como la que se muestra a continuación:

```
<asp:button attributes runat="server" id="Button1" />
```

En este caso los atributos no son los de los elementos HTML. En lugar de ello, son propiedades del control Web. Cuando se ejecuta la página Web ASP.NET, el control de servidor Web se representa en la página utilizando el marcado apropiado, que con frecuencia no sólo depende del tipo de explorador sino también de

la configuración que haya realizado para el control. Por ejemplo, un control TextBox podría representarse como una etiqueta input o una etiqueta textarea dependiendo de sus propiedades.



✪ Label Control

- Etiqueta de control que muestra los datos del programa
- Utilice únicamente si el texto de la etiqueta cambiará en tiempo de ejecución
- Si el texto no cambia, es establecido como texto estático

✪ Text Control

- El control tiene entrada de texto por el usuario
- Propiedad TextMode puede ser:
    - SingleLine: permite que una sola línea de entrada
    - MultiLine: permite múltiples líneas de entrada
    - Password: caracteres escritos aparecen como asteriscos

✪ CheckBox Control

- Funciona de manera casi idéntica a los CheckBox en formularios de Windows
- Propiedad Text establece texto visible para el usuario
- Evaluar propiedad Checked en tiempo de ejecución para determinar si el control es verificado por el usuario
- TextAlign permite texto de la posición

✪ HyperLink Control

- Proporciona un enlace para ir a otra página
- Propiedad Text especifica el texto mostrado para el enlace
- Propiedad NavigateURL sostiene URL de destino

- Propiedad Target determina si se abre una nueva ventana del navegador para mostrar la nueva página
- Igualarán a _blank para abrir una ventana independiente

✪ ImageButton, LinkButton, y el RadioButtonList Control

- ImageButton ofrece una imagen que se puede hacer clic
  - Genera un evento click
  - Propiedad ImageURL especifica la ruta a la imagen
- LinkButton se comporta como un hipervínculo, pero genera un evento de clic RadioButtonList en un grupo de botones de radio
  - Funciones similares a un ListBox
  - Tiene SelectedIndex y SelectedValue propiedades

✪ ListBox Control

- Muy similar al control ListBox de Windows
  - Tiene una colección de artículos
  - Tiene las propiedades ListBox SelectedIndex, selectedItem y SelectedValue
  - Propiedad SelectionMode especifica si varios elementos de lista se pueden seleccionar
- Manejo de eventos SelectedIndexChanged
  - Debe establecer AutoPostBack true si este evento debe disparar inmediatamente después de una selección de usuario
  - Si no es así, desencadena el evento sólo después de otros causas de control forman para ser publicado de nuevo al servidor

✪ CheckBoxList y el DropDownList Control

- Control CheckBoxList parece grupo de casillas de verificación, pero funciona como un ListBox
  - Tiene una colección de artículos
  - Tiene las propiedades ListBox SelectedIndex, selectedItem y SelectedValue
  - Cada elemento tiene una propiedad seleccionada de Bool
- DropDownList similar a ComboBox excepto:
  - Valor inicial de SelectedIndex siempre cero por lo que el primer elemento siempre se muestra
  - Debe seleccionar el elemento de la lista, no puede de entrada clave

Controles de validación

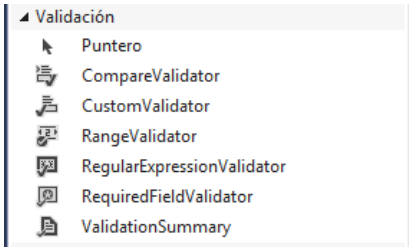
Un aspecto importante de la creación de páginas Web ASP.NET para la entrada de datos por el usuario consiste en poder comprobar que la información que introducen los usuarios es válida. ASP.NET ofrece un conjunto de controles de validación que proporciona una forma eficaz y fácil de usar para comprobar errores y, si es necesario, mostrar mensajes al usuario. En los temas de esta sección se describen los controles de validación y cómo utilizarlos. En la tabla siguiente se muestran los controles de validación de ASP.NET y cómo se utilizan.

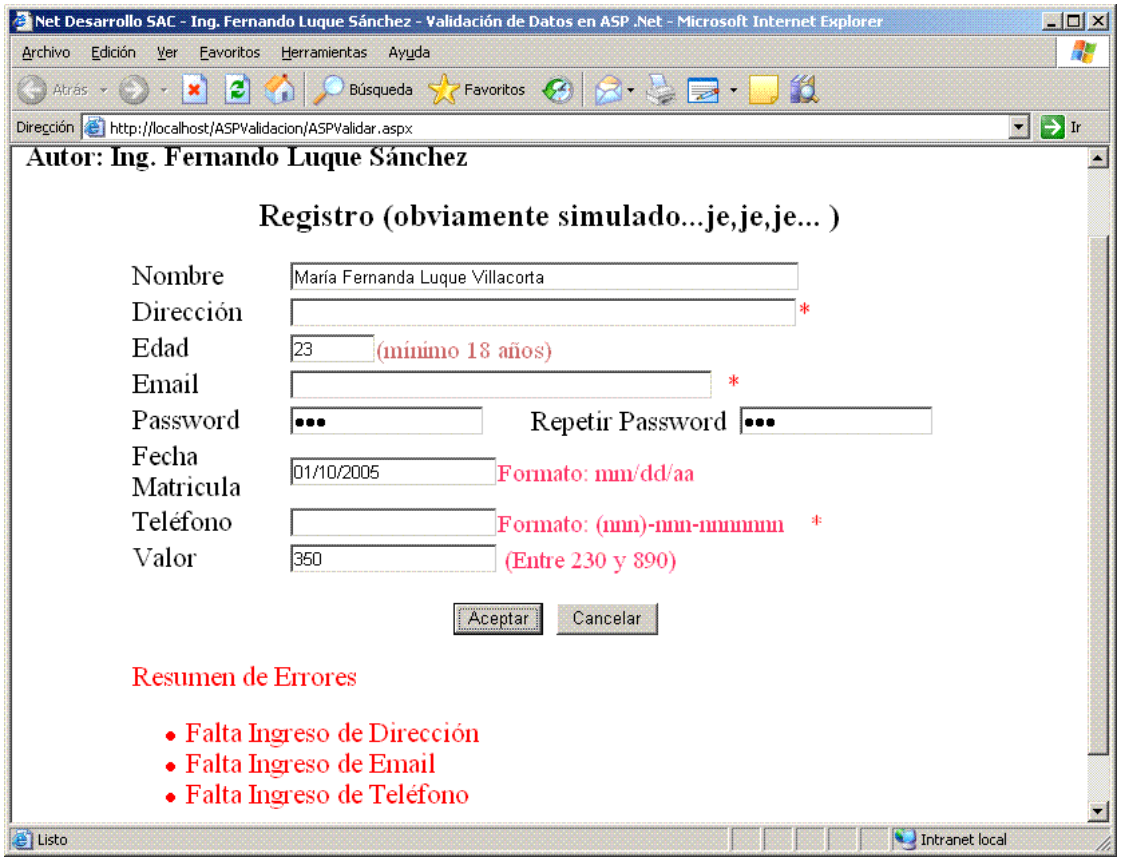
Tipo de validación	Utilizar control	Descripción
Entrada requerida.	**RequiredFieldValidator**	Garantiza que el usuario no omite una entrada.
Comparación con un valor	**CompareValidator**	Compara los datos proporcionados por el usuario con un valor constante, con el valor de otro control (mediante un operador de comparación como menor que, igual que o mayor que) o para un tipo de datos específico.
Comprobación del intervalo	**RangeValidator**	Comprueba que una entrada de usuario está entre los límites superior e inferior especificados. Se pueden

		comprobar los intervalos entre pares de números, caracteres alfabéticos y fechas.
Coincidencia de modelos	**RegularExpressionValidator**	Comprueba que la entrada del usuario coincide con un modelo definido por una expresión regular. Este tipo de validación permite comprobar secuencias de caracteres predecibles, como los que aparecen en las direcciones de correo electrónico, números de teléfono, códigos postales, etc.
Definida por el usuario	**CustomValidator**	Comprueba la entrada de usuario utilizando la validación lógica que ha escrito. Este tipo de validación permite comprobar valores derivados en tiempo de ejecución.

En un formulario se pueden asociar más de un control de validación a un control. Por ejemplo, se podría especificar que se requiera un control y que además contenga un intervalo de valores específico.

Existe un control relacionado, ValidationSummary, que no realiza ningún tipo de validación pero suele utilizarse con otros controles de validación para mostrar los mensajes de error de todos los controles de validación de la página juntos.





Validación no-intrusiva en ASP.NET 4.5 y errores de validadores

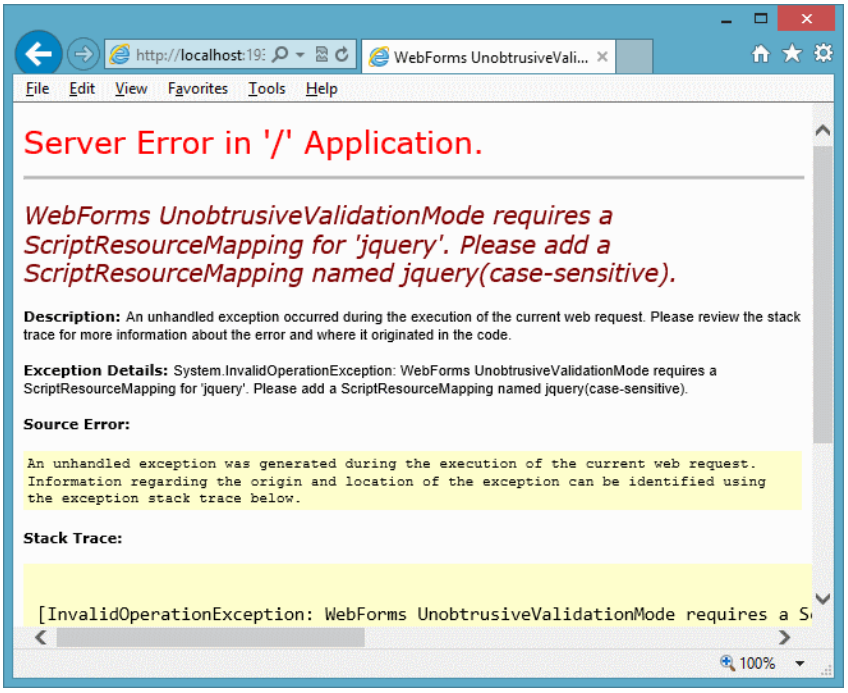
Por defecto, *ASP.NET 4.5 (desde Visual Studio 2012) cambió la manera en la que funciona la validación de controles*. Desde fuera parece que nada ha cambiado, pero por debajo ahora se utiliza por omisión un *nuevo tipo de validación no intrusiva* basada en jQuery, en lugar de los scripts anteriores.

Ahora, la validación de lado cliente se consigue de una manera más sencilla usando el plugin jQuery validation, y decorando los diferentes controles de validación usando **atributos "data-val"**, en lugar de llenar la página de scripts de validación.

Por ejemplo, este es el HTML resultante de un control RequiredValidator cuando se está usando el modo de validación no-intrusiva:

```
1: <span
2:    id="RequiredFieldValidator1"
3:    data-val-controltovalidate="TextBox1"
4:    data-val-focusOnError="t"
5:    data-val-errormessage="Required!"
6:    data-val-display="Dynamic"
7:    data-val="true"
8:    data-val-evaluationfunction="RequiredFieldValidatorEvaluateIsValid"
9:    data-val-initialvalue=""
10:   style="color:Red;display:none;">Required!</span>
```

Sin embargo, si creas una nueva aplicación web vacía con Visual Studio 2012 o posterior y añades un control de validación a alguna página, cuando ejecutes la aplicación verás esta página de error al intentar hacer cualquier validación:



Obtienes este error porque los scripts de validación necesitan un recurso de script registrado en el sistema con el nombre "jquery", y no lo encuentran en tu aplicación. Para corregir el error necesitamos deshabilitar la validación no-intrusiva para toda la aplicación. Para conseguirlo tenemos que agregar una línea a nuestro archivo de configuración *web.config*, colocándola en el nodo *appsettings*, así:

```
1: <appSettings>
2:     <add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />
3: </appSettings>
```

Controles de usuario
Además de utilizar controles de servidor Web en las páginas Web ASP.NET, puede crear sus propios controles personalizados reutilizables con las mismas técnicas que para crear páginas Web ASP.NET. Estos controles se denominan controles de usuario.
Un control de usuario es un tipo de control compuesto que funciona de forma similar a la de una página Web ASP.NET: se pueden agregar controles de servidor Web y marcado a un control de usuario, así como definir propiedades y métodos para el control. A continuación, puede incrustarlos en páginas Web ASP.NET, donde actúan como una unidad.

¿Cómo los Web Controls se procesan?
El servidor Web ejecuta el código VB/CS encontrado detrás de la página Web ASP.NET
Cuando un navegador solicita una página Web .aspx:

1. El servidor lee/interpreta los controles Web en la página
2. Las declaraciones de VB/CS en el archivo de código subyacente son ejecutadas
3. Se crea una página web de etiquetas HTML estándar y controles creados con controles Web y código VB/CS
4. La página Web HTML resultante es enviada de vuelta al navegador

ViewState o estado de la vista
El estado de vista es el método que utiliza el marco de las páginas ASP.NET para conservar los valores de página y control entre viajes de ida y vuelta. Cuando se representa el marcado HTML de la página, el estado actual de la página y los valores que se deben conservar durante la devolución de datos se serializan en cadenas codificadas en base64. Esta información se coloca a continuación en el campo o campos ocultos del estado de vista. El marco de la página ASP.NET usa automáticamente el estado para conservar información que se debe mantener entre las devoluciones de datos. Esta información incluye cualquier valor no predeterminado de los controles. También puede usar el estado de vista para almacenar datos de aplicación específicos de una página.

PostBack
PostBack es el nombre dado al proceso de presentación o envío de una página ASP.NET al servidor para su procesamiento. El Postback se realiza si ciertas credenciales de la página se van a comprobar en contra de algunas fuentes (como la verificación de nombre de usuario y la contraseña utilizando la base de datos). Esto es algo que una máquina cliente no es capaz de lograr y por lo tanto estos detalles tienen que ser 'publicado de nuevo' en el servidor.

III. MATERIALES Y EQUIPO

Para la realización de la guía de práctica se requerirá lo siguiente:

No.	Requerimiento	Cantidad
1	Guía de práctica #3: Controles Web y Validación de Formularios Web en ASP.NET	1
2	Computadora con Visual Studio 2017 instalado	1
3	Memoria USB	1

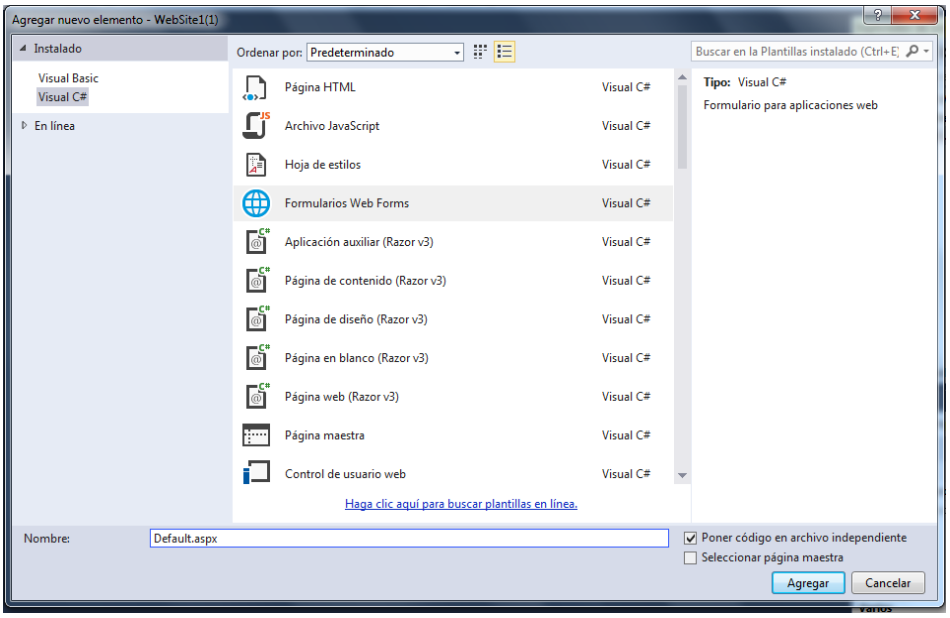
IV. PROCEDIMIENTO

Ejercicio #1 – Ejercicio único que muestra un sitio web donde se simula una bolsa de trabajo básica en la que los es posible el registro y validación de nuevos candidatos y posteriormente su presentación en un listado general

1. Proceda a ejecutar Visual Studio en su computadora. Una vez cargado el IDE, proceda a crear un nuevo sitio web. Para eso, seleccione del menú Archivo la opción "Nuevo sitio web".
2. Seleccione el nodo de Visual C# a la izquierda de la pantalla. Luego seleccione "Sitio web vacío de ASP.NET". Colóquele el nombre de "BolsaTrabajo". Seleccione como ubicación web la opción "Sistema de archivos". Una vez realizado lo siguiente de clic en "Aceptar".
3. Una vez creado nuestro sitio web, procederemos a configurar las condiciones de trabajo del mismo mediante la creación de carpetas y archivos. Para ello cree 4 carpetas en la raíz del sitio según el detalle de la siguiente tabla. El contenido de cada carpeta se encuentra como recurso en esta guía por lo que deberá adicionar posteriormente su contenido tal y como lo realizó en la guía 2.

Carpeta
Css
Fonts
images
Js

4. Cree una carpeta adicional al mismo nivel de las 4 carpetas anteriores con el nombre de "archivos". En esta carpeta se almacenarán los archivos subidos por el usuario por lo que es importante nombrarla de esta manera. Inicialmente esta carpeta estará vacía.
5. Cree una carpeta especial de ASP.NET para almacen de datos, para ello de clic derecho sobre el nombre del proyecto en la ventana del explorador de soluciones y luego seleccione "Agregar", luego "Agregar carpeta ASP.NET" y seleccione el tipo de carpeta "App_Data". En esta carpeta deberá alojar los archivos "Paises.xml" y "Municipios.xml" que son proporcionados en los recursos de esta guía.
6. Agregue el archivo "candidatos.txt" dentro de la carpeta "App_Data" el cual nos servirá para ir almacenando un registro de los candidatos registrados en el sitio web. Con esto estamos listo para iniciar el diseño de las páginas del sitio web. El archivo se encuentra proporcionado en los recursos.
7. Procederemos ahora a agregar un nuevo elemento WebForm o Formulario WebForm, para ello en el menú Archivo seleccione la opción "Nuevo Archivo". Se nos desplegará la siguiente ventana. Dentro de esta ventana seleccionamos la opción de "Formulario Web Forms", y colocamos como nombre de archivo "Default.aspx". Finalmente damos clic en el botón de Aceptar.



8. Ahora, procederemos a digitar el marcado relacionado con la página inicial, para ello proceda a digitar el siguiente marcado en la vista de código del diseñador. Con esto construiremos el contenido de la página web que será visualizada por los usuarios. En esta página no existen controles de servidor ya que no se procesará información desde este punto de la aplicación

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default"
%>

<!DOCTYPE HTML>
<html>
	<head>
		<title>Bolsa de Trabajo en Línea - Universidad Don Bosco</title>
		<meta http-equiv="content-type" content="text/html; charset=utf-8" />
		<!--[if lte IE 8]><script src="css/ie/html5shiv.js"></script><![endif]-->
		<script src="js/jquery.min.js"></script>
		<script src="js/jquery.scrolly.min.js"></script>
		<script src="js/jquery.dropotron.min.js"></script>
		<script src="js/jquery.scrollex.min.js"></script>
		<script src="js/skel.min.js"></script>
		<script src="js/skel-layers.min.js"></script>
		<script src="js/init.js"></script>
		<noscript>
			<link rel="stylesheet" href="css/skel.css" />
			<link rel="stylesheet" href="css/style.css" />
			<link rel="stylesheet" href="css/style-xlarge.css" />
		</noscript>
		<!--[if lte IE 9]><link rel="stylesheet" href="css/ie/v9.css" /><![endif]-->
		<!--[if lte IE 8]><link rel="stylesheet" href="css/ie/v8.css" /><![endif]-->
	</head>
	<body class="landing">

		<!-- Header -->
			<header id="header" class="skel-layers-fixed">
				<h1 id="logo"><a href="/">Página Principal</a></h1>
				<nav id="nav">
					<ul>
						<li><a href="/" class="button
special">Inicio</a></li>
						<li><a href="ListadoCandidatos.aspx" class="button
special">Aspirantes</a></li>
						<li><a href="NuevoCandidato.aspx" class="button
special">Nuevo Aspirante</a></li>
					</ul>
				</nav>
			</header>

		<!-- Banner -->
			<section id="banner">
				<div class="content">
					<header>
						<h2>Bolsa de Trabajo - Universidad Don Bosco</h2>
						<p>Tu oportunidad esta con nosotros</p>
					</header>
					<span class="image" style="height:auto"><img
src="images/udb.png" alt="" /></span>
				</div>
				<a href="#one" class="goto-next scrolly">Next</a>
```

```
                    </section>

                <!-- One -->
                    <section id="one" class="spotlight style1 bottom">
                        <span class="image fit main"><img src="images/IMG_9418.jpg"
/></span>
                        <div class="content">
                            <div class="container">
                                <div class="row">
                                    <div class="4u 12u$(medium)">
                                        <header>
                                            <h2>Proyección Social
Permanente</h2>
                                        </header>
                                    </div>
                                    <div class="4u 12u$(medium)">
                                        <p>Forma Parte de nuestro Programa de
Bolsa de Trabajo, el cual tiene como objetivo promover la inserción laboral de sus
Estudiantes, Egresados y Graduados en áreas afines a su formación académica a través de
Empresas e instituciones potencialmente empleadoras.</p>
                                    </div>
                                    <div class="4u$ 12u$(medium)">
                                        <p>La idea del Programa de  Bolsa de
Trabajo surge en respuesta a su interés de generar un impacto de inserción laboral de sus
profesionales en la realidad nacional; a lo que se suma el hecho de estar ubicada en una zona
eminentemente industrial.</p>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </section>

                <!-- Footer -->
                    <footer id="footer">
                        <ul class="icons">
                            <li><a href="#" class="icon alt fa-twitter"><span
class="label">Twitter</span></a></li>
                            <li><a href="#" class="icon alt fa-facebook"><span
class="label">Facebook</span></a></li>
                            <li><a href="#" class="icon alt fa-envelope"><span
class="label">Email</span></a></li>
                        </ul>
                        <ul class="copyright">
                            <li>&copy; 2018. All rights reserved.</li>
                        </ul>
                    </footer>

        </body>
</html>
```

9. Ahora, como puede observar es necesario crear las páginas restantes. Proceda a crear las páginas "NuevoCandidato.aspx" y "ListadoCandidatos.aspx". A continuación mostramos el marcado y el código de C# para cada una de ellas.
10. Proceda a escribir el código de la página "NuevoCandidato.aspx" dentro de la sección de código del diseñador, tomando en cuenta los nuevos controles existentes.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="NuevoCandidato.aspx.cs"
```

```
Inherits="_Default" %>

<!DOCTYPE HTML>
<html>
	<head>
		<title>Bolsa de Trabajo en Línea - Universidad Don Bosco</title>
		<meta http-equiv="content-type" content="text/html; charset=utf-8" />
		<!--[if lte IE 8]><script src="css/ie/html5shiv.js"></script><![endif]-->
		<script src="js/jquery.min.js"></script>
		<script src="js/jquery.scrolly.min.js"></script>
		<script src="js/jquery.dropotron.min.js"></script>
		<script src="js/jquery.scrollex.min.js"></script>
		<script src="js/skel.min.js"></script>
		<script src="js/skel-layers.min.js"></script>
		<script src="js/init.js"></script>
		<noscript>
			<link rel="stylesheet" href="css/skel.css" />
			<link rel="stylesheet" href="css/style.css" />
			<link rel="stylesheet" href="css/style-xlarge.css" />
		</noscript>
		<!--[if lte IE 9]><link rel="stylesheet" href="css/ie/v9.css" /><![endif]-->
		<!--[if lte IE 8]><link rel="stylesheet" href="css/ie/v8.css" /><![endif]-->
	</head>
	<body>

		<!-- Header -->
			<header id="header" class="skel-layers-fixed">
				<h1 id="logo"><a href="/">Página Principal</a></h1>
				<nav id="nav">
					<ul>
						<li><a href="/" class="button
special">Inicio</a></li>
						<li><a href="ListadoCandidatos.aspx" class="button
special">Aspirantes</a></li>
						<li><a href="NuevoCandidato.aspx" class="button
special">Nuevo Aspirante</a></li>
					</ul>
				</nav>
			</header>

		<!-- Main -->
			<div id="main" class="wrapper style1">
				<div class="container">
					<header class="major">
						<h2>Bolsa de Trabajo en Línea</h2>
						<p>Universidad Don Bosco</p>
					</header>

					<!-- Text -->
						<section>
							<p>
Forma Parte de nuestro Programa de Bolsa de Trabajo, el cual
tiene como objetivo promover la inserción laboral de sus Estudiantes, Egresados y Graduados
en áreas afines a su formación académica a través de Empresas e instituciones potencialmente
empleadoras.
<br />
La idea del Programa de  Bolsa de Trabajo surge en respuesta
a su interés de generar un impacto de inserción laboral de sus profesionales en la realidad
```

```
nacional; a lo que se suma el hecho de estar ubicada en una zona eminentemente industrial.
                                        </p>


                                    </section>
                                <!-- Formulario de Ingreso de Información -->
                                    <section>
                                        <h3>Registrate gratis en Bolsa de Trabajo
UDB</h3>
                                        <blockquote><asp:Label ID="lblResultado"
runat="server" Text="" Visible="false"></asp:Label></blockquote>
                            <h4>Datos Personales</h4>

                            <form id="form1" runat="server">
                                                <div class="row uniform 50%">
                                                    <div class="6u 12u$(xsmall)">
                                    <asp:TextBox ID="txtNombre" runat="server"
placeholder="Nombre"></asp:TextBox>
                                    <asp:RequiredFieldValidator
ID="RequiredFieldValidator1" Display="Dynamic" runat="server" ErrorMessage="Nombre es
Requerido" ControlToValidate="txtNombre"></asp:RequiredFieldValidator>
                                                    </div>
                                <div class="6u 12u$(xsmall)">
                                    <asp:TextBox ID="txtApellidos" runat="server"
placeholder="Apellidos"></asp:TextBox>
                                    <asp:RequiredFieldValidator
ID="RequiredFieldValidator2" Display="Dynamic" runat="server" ErrorMessage="Apellido es
Requerido" ControlToValidate="txtApellidos"></asp:RequiredFieldValidator>
                                                    </div>
                                <div class="6u$ 12u$(xsmall)">
                                                        <div class="select-
wrapper">
                                        <asp:DropDownList ID="drlNacionalidad"
runat="server"></asp:DropDownList>
                                                        </div>
                                                    </div>
                                                    <div class="6u 12u$(xsmall)">
                                    <asp:TextBox ID="txtEmail1" runat="server"
placeholder="E-mail"></asp:TextBox>
                                    <asp:RequiredFieldValidator Display="Dynamic"
ID="RequiredFieldValidator3" runat="server" ErrorMessage="Email es Requerido"
ControlToValidate="txtEmail1"></asp:RequiredFieldValidator>
                                    <asp:RegularExpressionValidator Display="Dynamic"
ID="RegularExpressionValidator1" runat="server" ErrorMessage="El dato ingresado no es un
Email" ControlToValidate="txtEmail1" ValidationExpression="\w+([-+.']\w+)*@\w+([-
.]\w+)*\.\w+([-.]\w+)*"></asp:RegularExpressionValidator>
                                </div>
                                <div class="6u 12u$(xsmall)">
                                    <asp:TextBox ID="txtEmail2" runat="server"
placeholder="E-mail alternativo"></asp:TextBox>
                                                    </div>
                                                    <div class="6u 12u$(medium)">
                                    <asp:RadioButton ID="radio_male" runat="server"
GroupName="radioGenero" Checked="true" />
                                                        <label
for="radio_male">Hombre</label>
                                    <asp:RadioButton ID="radio_female" runat="server"
GroupName="radioGenero" />
```

```
                                                                <label
for="radio_female">Mujer</label>
                                                            </div>
                                    <div class="6u 12u$(medium)">
                                        <asp:TextBox ID="txtFechaNac" runat="server"
placeholder="Fecha de Nacimiento"></asp:TextBox>
                                        <asp:RequiredFieldValidator
ID="RequiredFieldValidator4" Display="Dynamic" runat="server" ErrorMessage="Fecha de
Nacimiento es Requerido" ControlToValidate="txtFechaNac"></asp:RequiredFieldValidator>
                                        <asp:RegularExpressionValidator
ID="RegularExpressionValidator2" runat="server" ErrorMessage="(dd/mm/aa)"
ControlToValidate="txtFechaNac" ValidationExpression="^(0?[1-9]|[12][0-9]|3[01])[\/](0?[1-
9]|1[012])[\/](19|20)\d{2}$"></asp:RegularExpressionValidator>
                                    </div>
                                </div>
                                <h4>Datos de Contacto</h4>
                                <div class="row uniform 50%">
                                    <div class="6u 12u$(xsmall)">
                                        <label>País de Residencia</label>
                                                                <div class="select-
wrapper">
                                            <asp:DropDownList ID="ddlResidencia"
AutoPostBack="true" runat="server"></asp:DropDownList>
                                                                </div>
                                                            </div>
                                    <div class="6u$ 12u$(xsmall)">

      <label>Departamento</label>
                                        <asp:XmlDataSource ID="XmlDataSource1"
XPath="/Municipios/Municipio[ @Pais='1']" runat="server"
DataFile="~/App_Data/Municipios.xml"></asp:XmlDataSource>
                                                                <div class="select-
wrapper">
                                            <asp:DropDownList ID="ddlDepartamento"
EnableViewState="true" runat="server" DataSourceID="XmlDataSource1" DataTextField="Nombre"
DataValueField="Id"></asp:DropDownList>
                                                                </div>
                                                            </div>
                                    <div class="6u 12u$(xsmall)">
                                        <asp:TextBox ID="txtTelefonoCasa" runat="server"
placeholder="Teléfono Fijo"></asp:TextBox>
                                        <asp:RequiredFieldValidator
ID="RequiredFieldValidator5" Display="Dynamic" runat="server" ErrorMessage="Teléfono es
requerido" ControlToValidate="txtTelefonoCasa"></asp:RequiredFieldValidator>
                                                            </div>
                                    <div class="6u 12u$(xsmall)">
                                        <asp:TextBox ID="txtTelefonoCelular" runat="server"
placeholder="Celular"></asp:TextBox>
                                                            </div>
                                    <div class="6u 12u$(xsmall)">
                                        <asp:TextBox ID="txtDireccion" TextMode="multiline"
Rows="6" runat="server" placeholder="Dirección de domicilio"></asp:TextBox>
                                                            </div>
                                    <div class="6u$ 12u$(xsmall)">
                                        <label>Cargar Curriculum</label>
                                        <asp:FileUpload ID="FileUpload1" runat="server"
/><br />
                                        <asp:RequiredFieldValidator
```

```
ID="RequiredFieldValidator6" Display="Dynamic" runat="server"
ControlToValidate="FileUpload1" ErrorMessage="Debe cargar un
archivo"></asp:RequiredFieldValidator>
                                                                    </div>
                                                                    <div class="6u$ 12u$(medium)">
                                                <asp:CheckBox ID="chkHumano" runat="server" />
                                                                        <label
for="chkHumano">Soy humano y no un robot</label>
                                                                    </div>
                                                                    <div class="12u$">
                                                                        <ul class="actions">
                                                                            <li>
                                                        <asp:Button ID="btnEnviar" runat="server"
Text="Registrar Información" CssClass="special"/>
                                                                            </li>
                                                                            <li><input
type="reset" value="Limpiar" /></li>
                                                                        </ul>
                                                                    </div>
                                                                </div>
                                    </form>
                                                </section>

                                </div>
                            </div>

                    <!-- Footer -->
                            <footer id="footer">
                                    <ul class="icons">
                                            <li><a href="#" class="icon alt fa-twitter"><span
class="label">Twitter</span></a></li>
                                            <li><a href="#" class="icon alt fa-facebook"><span
class="label">Facebook</span></a></li>
                                            <li><a href="#" class="icon alt fa-envelope"><span
class="label">Email</span></a></li>
                                    </ul>
                                    <ul class="copyright">
                                            <li>&copy; 2018. All rights reserved.</li>
                                    </ul>
                            </footer>

        </body>
</html>
```

11. Como puede observar, en la página anterior existen fragmentos de marcado relacionados a nuevos controles de servidor. Llene la siguiente tabla con los controles nuevos que encuentre en el marcado anterior. Si tiene dudas consulte a su docente.

Control	Posibles funciones
XmlDataSource	
RegularExpressionValidator	
RequiredFieldValidator	
FileUpload	

12. Ahora procederemos a escribir el código respectivo para la aplicación, para ello, en el evento Page_Load digite el siguiente código.

```
protected void Page_Load(object sender, EventArgs e)
    {
       //Esto bloque de código se ejcuta cada vez que la página no ha sido producto de un
postback
        //Qué es un postback?
        if (!Page.IsPostBack)
        {
            drlNacionalidad.Items.Clear();
            drlNacionalidad.Items.Add("Guatemala");
            drlNacionalidad.Items.Add("El Salvador");
            drlNacionalidad.Items.Add("Honduras");
            drlNacionalidad.Items.Add("Nicaragua");
            drlNacionalidad.Items.Add("Costa Rica");
            drlNacionalidad.Items.Add("Panamá");
            drlNacionalidad.Items.Add("Otro país");

            //Cargamos el dropdownlist con los datos del archivo XML para paises
            string myXMLfile = Server.MapPath("App_Data/Paises.xml");
            DataSet dsPaises = new DataSet();
            try
            {
                dsPaises.ReadXml(myXMLfile);
                ddlResidencia.DataSource = dsPaises;
                ddlResidencia.DataValueField = "Id";
                ddlResidencia.DataTextField = "Nombre";
                ddlResidencia.DataBind();
            }
            catch (Exception ex)
            {
                Response.Write(ex.ToString());
            }
        }

    }
```

13. Ahora, en la vista de diseñador, de doble clic sobre el DropDownList con ID "ddlResidencia" para que se nos abra el método de cambio de índice asociado. Proceda a digitar el siguiente código.

```
protected void ddlResidencia_SelectedIndexChanged(object sender, EventArgs e)
    {
        ddlDepartamento.SelectedIndex = -1;
        string country = ddlResidencia.SelectedItem.Value;
```

```
        //Filtramos el dropdownlist de departamentos dependiendo la opción de
        //pais seleccionada por el usuario, obteniendo los datos desde el XML para
        //departamentos
        if (country != string.Empty)
        {
            XmlDataSource1.XPath = "/Municipios/Municipio[ @Pais='" + country + "']";
        }
        else
        {
            XmlDataSource1.XPath = "/Municipios/Municipio[ @Pais='1']";
        }
    }
```

14. Finalmente procederemos a escribir el código asociado al botón con ID "btnEnviar" por lo que debe proceder a dar doble clic sobre el botón y escribir el siguiente código.

```
protected void btnEnviar_Click(object sender, EventArgs e)
    {
        try
        {
            //Verificamos si la pagina enviada ha superado los procesos de validación
            //y el chechbox de seguridad esta checkeado
            if (Page.IsValid && chkHumano.Checked)
            {
                //Recuperamos los datos de información básica
                String nombreCandidato = txtNombre.Text;
                String apellidoCandidato = txtApellidos.Text;
                String paisNacimiento = drlNacionalidad.SelectedItem.Text;
                String email = txtEmail1.Text;
                String emailAlternativo = txtEmail2.Text;
                String generoCandidato = (radio_male.Checked) ? "Masculino" : "Femenino";
                String fechaNacimiento = txtFechaNac.Text;

                //Recuperamos los datos de contacto y residencia
                String paisResidencia = ddlResidencia.SelectedItem.Text;
                String municipioResidencia = ddlDepartamento.SelectedItem.Text;
                String telefonoCasa = txtTelefonoCasa.Text;
                String telefonoCelular = txtTelefonoCelular.Text;
                String domicilioCandidato = txtDireccion.Text;
                String cv = (FileUpload1.HasFile) ? FileUpload1.FileName : "";

                //Procedemos a guardar el archivo cargado en una carpeta de nuestro servidor
                //por lo que "mapeamos" una carpeta de nuestro servidor llamada archivos
                //es ahi donde se guardaran los nuevos archivos
                String path = Server.MapPath("~/archivos/");
                FileUpload1.PostedFile.SaveAs(path + FileUpload1.FileName);

                //Procedemos finalmente a salvar la información ingresada en un archivo de
texto
                //alojado en nuestro servidor, y por cada vez que se ingrese un nuevo
registro
                //se añade una nueva linea al archivo y a la vez separa cada campo mediante
una coma
                using (StreamWriter w =
File.AppendText(Server.MapPath("~/App_Data/candidatos.txt")))
                {
                    w.WriteLine("{0},{1},{2},{3},{4},{5},{6},{7},{8},{9},{10},{11},{12}",
```

```
                        nombreCandidato,
                        apellidoCandidato,
                        paisNacimiento,
                        email,
                        emailAlternativo,
                        generoCandidato,
                        fechaNacimiento,
                        paisResidencia,
                        municipioResidencia,
                        telefonoCasa,
                        telefonoCelular,
                        domicilioCandidato,
                        cv);
                }

            }
            else
            {
                lblResultado.Text = "Error: <br/>Existe un error en la página o no ha
indicado que es una persona";
            }
        }
        catch (Exception ex)
        {
            lblResultado.Text = "Error: <br/>" + ex.Message;
        }
    }
```

15. Ahora, podemos proceder a probar la página de registro, para ello, proceda a ejecutar la página en el navegador para que se cree una instancia del IIS en localhost. Probablemente se nos muestre el error de "UnobtrusiveValidationMode" abordado en la introducción teórica. Para solucionar este problema proceda a agregar las líneas de código faltante al archivo Web.config de tal manera que luzca de la siguiente manera (ponga especial cuidado a la etiqueta appSettings):

```
<?xml version="1.0" encoding="utf-8"?>

<!--
  Para obtener más información sobre cómo configurar la aplicación de ASP.NET, visite
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->

<configuration>
    <system.web>
      <compilation debug="true" targetFramework="4.5" />
      <httpRuntime targetFramework="4.5" />
    </system.web>
  <appSettings>
    <add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />
  </appSettings>
</configuration>
```

16. Proceda a ejecutar y recargar nuevamente la página desde el IDE de Visual Studio. Una vez cargada la página procederemos a realizar la siguiente prueba. Seleccione un país diferente del primer "select" dentro de la sección de "Datos Personales". ¿Qué sucedió. Ahora seleccione un país de residencia

diferente del primer "select" en la sección de "Datos de contacto". ¿Qué sucedió?. Complete la siguiente tabla.

Control	¿Qué sucedió y por qué?	Propiedad Asociada	Valor (true/false)
Primer select		AutoPostBack	
Segundo select		AutoPostBack	

¿Qué puede concluir de lo anterior?

17. Procedamos ahora a realizar una segunda prueba, para ello, dejemos los campos de nombre y apellido en blanco, en el campo de email introduzcamos texto aleatorio y en el campo fecha escribamos "32/13/2015". Finalmente de clic al botón de "Registrar información". ¿Qué puede concluir del ejercicio?
18. Ahora, procederemos a agregar el marcado correspondiente a la página "ListadoCandidatos.aspx" por lo que debe digitar el siguiente marcado

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="ListadoCandidatos.aspx.cs"
Inherits="ListadoCandidatos" %>

<!DOCTYPE HTML>
<html>
	<head>
		<title>Bolsa de Trabajo en Línea - Universidad Don Bosco</title>
		<meta http-equiv="content-type" content="text/html; charset=utf-8" />
		<!--[if lte IE 8]><script src="css/ie/html5shiv.js"></script><![endif]-->
		<script src="js/jquery.min.js"></script>
		<script src="js/jquery.scrolly.min.js"></script>
		<script src="js/jquery.dropotron.min.js"></script>
		<script src="js/jquery.scrollex.min.js"></script>
		<script src="js/skel.min.js"></script>
		<script src="js/skel-layers.min.js"></script>
		<script src="js/init.js"></script>
		<noscript>
			<link rel="stylesheet" href="css/skel.css" />
			<link rel="stylesheet" href="css/style.css" />
			<link rel="stylesheet" href="css/style-xlarge.css" />
		</noscript>
		<!--[if lte IE 9]><link rel="stylesheet" href="css/ie/v9.css" /><![endif]-->
		<!--[if lte IE 8]><link rel="stylesheet" href="css/ie/v8.css" /><![endif]-->
	</head>
	<body>

		<!-- Header -->
			<header id="header" class="skel-layers-fixed">
				<h1 id="logo"><a href="/">Página Principal</a></h1>
				<nav id="nav">
					<ul>
						<li><a href="/" class="button
special">Inicio</a></li>
						<li><a href="ListadoCandidatos.aspx" class="button
special">Aspirantes</a></li>
						<li><a href="NuevoCandidato.aspx" class="button
special">Nuevo Aspirante</a></li>
					</ul>
				</nav>
			</header>
```

```
            <!-- Main -->
                  <div id="main" class="wrapper style1">
                        <div class="container">
                              <header class="major">
                                    <h2>Bolsa de Trabajo en Línea</h2>
                                    <p>Universidad Don Bosco</p>
                              </header>

                              <!-- Text -->
                                    <section>
                                          <p>
                                    Forma Parte de nuestro Programa de Bolsa de Trabajo, el cual
tiene como objetivo promover la inserción laboral de sus Estudiantes, Egresados y Graduados
en áreas afines a su formación académica a través de Empresas e instituciones potencialmente
empleadoras.
                                    <br />
                                    La idea del Programa de  Bolsa de Trabajo surge en respuesta
a su interés de generar un impacto de inserción laboral de sus profesionales en la realidad
nacional; a lo que se suma el hecho de estar ubicada en una zona eminentemente industrial.
                                                </p>


                                          </section>
                                      <!-- Formulario de Ingreso de Información -->
                                          <section>
                                                <h3>Listado de Candidatos Registrados</h3>

                                    <form id="form1" runat="server">
                                                      <div class="row uniform 50%">

                                            <asp:Table ID="Table1" runat="server" CssClass="alt">
                                                <asp:TableRow>
                                                    <asp:TableCell>Nombres</asp:TableCell>
                                                    <asp:TableCell>Apellidos</asp:TableCell>
                                                    <asp:TableCell>E-mail</asp:TableCell>
                                                    <asp:TableCell>Residencia</asp:TableCell>
                                                    <asp:TableCell>Teléfono </asp:TableCell>
                                                    <asp:TableCell>Celular</asp:TableCell>
                                                    <asp:TableCell>CV</asp:TableCell>
                                                </asp:TableRow>
                                            </asp:Table>

                                        </div>
                                    </form>
                                                </section>

                              </div>
                        </div>

            <!-- Footer -->
                  <footer id="footer">
                        <ul class="icons">
                              <li><a href="#" class="icon alt fa-twitter"><span
class="label">Twitter</span></a></li>
                              <li><a href="#" class="icon alt fa-facebook"><span
class="label">Facebook</span></a></li>
                              <li><a href="#" class="icon alt fa-envelope"><span
```

```
class="label">Email</span></a></li>
                            </ul>
                            <ul class="copyright">
                                  <li>&copy; 2018. All rights reserved.</li>
                            </ul>
                      </footer>

      </body>
</html>
```

19. Ahora, procederemos a programar el evento de carga de la página con el siguiente código.

```
protected void Page_Load(object sender, EventArgs e)
    {
        try
        {
            string line;
            String[] informacion;
            //Procedemos a leer los datos del archivo donde almacenamos la información
            //e iteramos mediante un ciclo while hasta que el puntero de fin de archivo
apunte a null
            StreamReader file = new
System.IO.StreamReader(@Server.MapPath("/App_Data/candidatos.txt"));
            while ((line = file.ReadLine()) != null)
            {
                //Por cada linea que se lea del archivo, se realiza un split o separación de
la misma
                //de tal manera que la cadena original quede dividida en varias subcadenas
almacenadas como arreglos
                informacion = line.Split(',');

                //Agregamos una nueva fila a la tabla
                TableRow tRow = new TableRow();
                Table1.Rows.Add(tRow);
                //Creamos 7 nuevas celdas que serán agregadas a la fila creada anteriormente
                TableCell tNombres = new TableCell();
                TableCell tApellidos = new TableCell();
                TableCell tEmail = new TableCell();
                TableCell tResidencia = new TableCell();
                TableCell tTelefono = new TableCell();
                TableCell tCelular = new TableCell();
                TableCell tCV = new TableCell();

                //Asignamos los respectivos valores para todas las celdas
                tRow.Cells.Add(tNombres);
                tNombres.Text = informacion[0];

                tRow.Cells.Add(tApellidos);
                tApellidos.Text = informacion[1];

                tRow.Cells.Add(tEmail);
                tEmail.Text = informacion[3];

                tRow.Cells.Add(tResidencia);
                tResidencia.Text = informacion[7] + ", " + informacion[8];
```

```
                tRow.Cells.Add(tTelefono);
                tTelefono.Text = informacion[9];

                tRow.Cells.Add(tCelular);
                tCelular.Text = informacion[10];

                //Para agregar un enlace para ver el curriculum se crea un control de tipo
                //HyperLink y se agrega a la celda como control y no como texto
                tRow.Cells.Add(tCV);
                HyperLink verCV = new HyperLink();
                verCV.Text = "Ver CV";
                verCV.NavigateUrl = "archivos/" + informacion[informacion.Length-1];
                tCV.Controls.Add(verCV);
            }

            file.Close();
        }
        catch (Exception ex)
        {
            Response.Write("hola");
        }
    }
```

20. Ahora, proceda a ejecutar toda la aplicación y verifique los datos de ingreso y el listado de candidatos

V. DISCUSIÓN DE RESULTADOS

1. Modifique el archivo de NuevoCandidato.aspx.cs de tal manera que el archivo ingresado corresponda únicamente para aquellos con extensión "*.doc, *.docx, *.pdf" y con un peso no mayor a 2 MB.
2. Modifique el archivo NuevoCandidato.aspx. de tal manera que se valide el teléfono fijo y el celular según el formato salvadoreño.
3. La Universidad Don Bosco solicita que se agregue una página para registro de usuarios y contraseñas por parte de los candidatos, por lo que se solicita crear una página que solicite un correo electrónico como nombre de usuario (username), un campo de ingreso de contraseña, un campo de confirmación de contraseña y un botón de enviar. Usted debe validar que el email ingresado sea un email válido. Además el campo de contraseña y confirmación de contraseña deben de contener la misma información. Todos los campos son requeridos. Finalmente, haga uso de un control "ValidationSumary" para mostrar un resultado general de la validación. Si los datos ingresados pasan el proceso de validación, entonces se debe de redirigir al usuario a la página de "NuevoCandidato.aspx"

VI. BIBLIOGRAFÍA

1. Thierry GROUSSARD. (2013). C# 5: Los fundamentos del lenguaje – Desarrollar con Visual Studio 2012 . Barcelona: ENI.