Creating User and Custom Controls using ASP.NET

In this lab we will see how ASP.NET allows us to define our own customized controls. These are utility controls needs to be defined and designed based upon requirements.

Exercise 1: Creating ASP.NET User Control.

Task 1: Open VS2010/VS2008 and create new ASP.NET Web Site. Name it as 'ASPNET_User_Custom_Controls'.

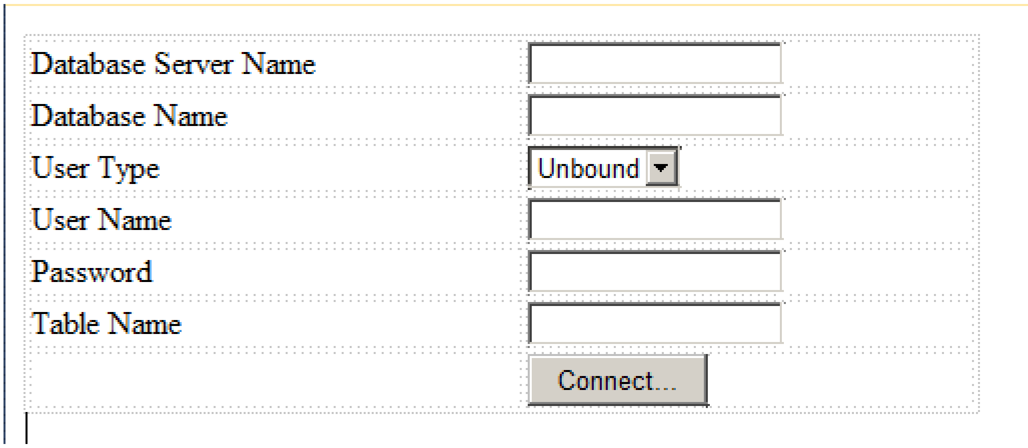
Task 2: In this application add a new folder, name it as 'AddedControls'. On this folder right click and add a new 'Web User Control', name it as 'DataAccessUserControl.ascx'. This user control is defined for connecting to database server and list all rows form the table. The user control class if inherited from 'UserControl' class and the extension of the user control is .ascx. It also uses '@Control' directive.

Task 3: In the 'DataAccessUserControl.ascx' write the following design UI:

```
<table class="style1">
    <tr>
        <td>
            Database Server Name</td>
        <td>
            <asp:TextBox ID="txtDbServer"
runat="server"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td>
            Database Name</td>
        <td>
            <asp:TextBox ID="txtDbName" runat="server"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td>
            User Type</td>
        <td>
            <asp:DropDownList ID="lstUSerType" runat="server"
AutoPostBack="True"

onselectedindexchanged="lstUSerType_SelectedIndexChanged">
            </asp:DropDownList>
        </td>
    </tr>
    <tr>
        <td>
```

```
            User Name</td>
        <td>
            <asp:TextBox ID="txtUName" runat="server"
Enabled="False"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td>
            Password</td>
        <td>
            <asp:TextBox ID="txtPwd" runat="server" Enabled="False"
TextMode="Password"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td>
            Table Name</td>
        <td>
            <asp:TextBox ID="txtTableName"
runat="server"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td>
            &nbsp;</td>
        <td>
            <asp:Button ID="btnConnect" runat="server"
onclick="btnConnect_Click"
                Text="Connect..." />
        </td>
    </tr>
</table>
```

The control will look as below:



Database Server Name	
Database Name	
User Type	Unbound
User Name	
Password	
Table Name	

Task 4: One important thing here is that, when yow create any utility control, you need to consider the following:

- Look of the control:
  - Used to define Web Controls for creation of the control.
- Functionality of the control:
  - Used to define Properties and events in the control.

Since the control, is used to connect to database and return data-table, we need to define property for exposing Table to the control consumer and also event when this table property will be exposed. Open ‘DataAccessUserControl.ascx.cs’ the code behind for the control and write the following code:

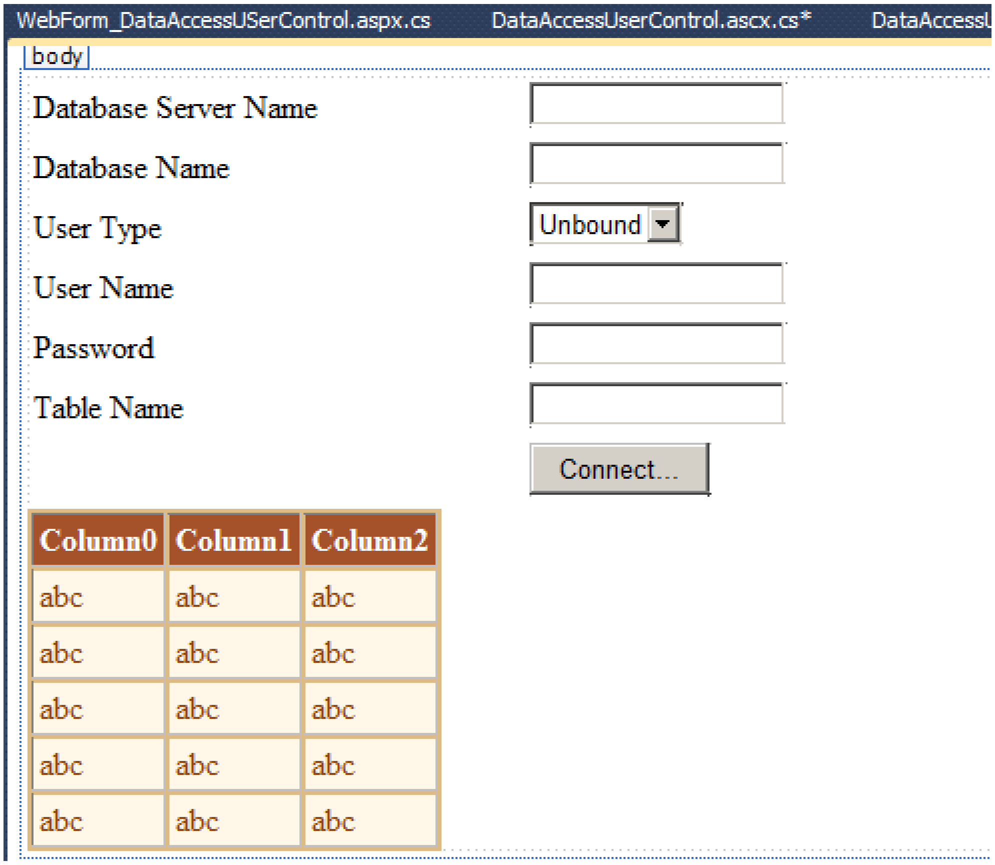
```
using System;
using System.Data;
using System.Data.SqlClient;

namespace Web_User_Custom_Controls.AddedControls
{
    public partial class DataAccessUserControl : 
System.Web.UI.UserControl
    {

        SqlConnection Conn;
        SqlDataAdapter AdTable;
        DataSet Ds;

        public event EventHandler ConnectOperationClick;

        public DataTable TableName { get; set; }
        public Exception DbException { get; set; }


        protected void Page_Load(object sender, EventArgs e)
        {
            if (this.IsPostBack == false)
            {
                lstUSerType.Items.Add("Windows Authentication");
                lstUSerType.Items.Add("Sql Server AUthentication");
            }
        }

        protected void btnConnect_Click(object sender, EventArgs e)
        {

            Conn = new SqlConnection(ViewState["ConnStr"].ToString());
            AdTable = new SqlDataAdapter("Select * from " +
txtTableName.Text,Conn);

            Ds = new DataSet();

            AdTable.Fill(Ds, txtTableName.Text);

            TableName = Ds.Tables[txtTableName.Text];

            if (ConnectOperationClick != null)
            {
                ConnectOperationClick(this, EventArgs.Empty);
            }
        }

        protected void lstUSerType_SelectedIndexChanged(object sender,
EventArgs e)
        {
            if (lstUSerType.SelectedIndex == 0)
            {
                txtUName.Text = "Administrator";
                ViewState["ConnStr"] = "Data Source = " +
txtDbServer.Text.Trim() + ";Initial Catalog=" + txtDbName.Text.Trim()
+ ";Integrated Security=SSPI";
            }
            if (lstUSerType.SelectedIndex == 1)
            {
                txtUName.Enabled = true;
                txtPwd.Enabled = true;
                ViewState["ConnStr"] = "Data Source = " +
txtDbServer.Text.Trim() + ";Initial Catelog=" + txtDbName.Text.Trim()
+ ";User Id=" + txtUName.Text.Trim() + ";Password=" +
txtPwd.Text.Trim();
```

```
                    }
                }
            }
}
```

The above code defines, ‘TableName’ property of the type ‘DataTable’ which will be expose to the consumer of this user control and also the ‘ConnectOperationClick’ event which will be raised when ‘Connect’ button is clicked. The ‘User Type’ drop-down list is used to select the user type for the database server i.e. whether Windows Authenticated user or not and based upon the user the connectivity is established to the database server.

Task 5: Build the control and make sure that it is error free.

Task 6: Rename ‘Default.aspx’ to ‘WebForm_DataAccessUSerControl.aspx’ and Drag the user control created from the solution explorer and drop it on the ‘WebForm_DataAccessUSerControl.aspx’ also Drag-Drop the GridView on the page form the toolbox. The page will look as below:



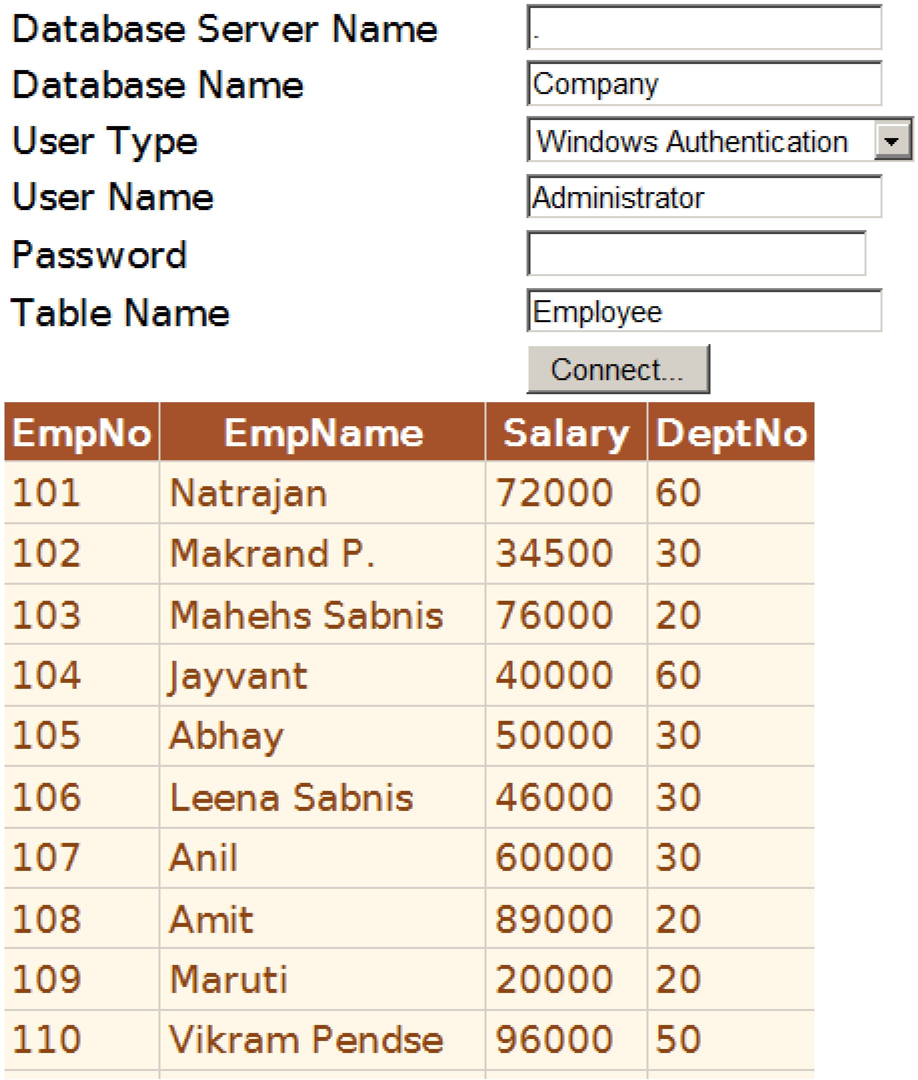
Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

Name the GrdiView as ‘gdvData’.

Task 7: Open ‘WebForm_DataAccessUSerControl.aspx.cs’ code behind and register the event of the user control created as below:

```
protected void Page_Load(object sender, EventArgs e)
        {
            ctrlDataAccess.ConnectOperationClick += new 
EventHandler(ctrlDataAccess_ConnectOperationClick);
        }

        void ctrlDataAccess_ConnectOperationClick(object sender, 
EventArgs e)
        {
            gdvData.DataSource = ctrlDataAccess.TableName;
            gdvData.DataBind();
        }
```

Task 8: Run the Page and enter the data, following result will be displayed:



Database Server Name	.
Database Name	Company
User Type	Windows Authentication
User Name	Administrator
Password	
Table Name	Employee

EmpNo	EmpName	Salary	DeptNo
101	Natrajan	72000	60
102	Makrand P.	34500	30
103	Mahehs Sabnis	76000	20
104	Jayvant	40000	60
105	Abhay	50000	30
106	Leena Sabnis	46000	30
107	Anil	60000	30
108	Amit	89000	20
109	Maruti	20000	20
110	Vikram Pendse	96000	50

Exercise 2: Creating Custom Control

In this exercise we will create a ASP.NET Custom control. Here we can extend the existing control or we can create the new composite custom control write from the scratch. This custom control is inherited from ‘CompositControl’ base class and overrides its methods. In this lab we will also see how to manage control state for the control instead of using ViewState. Typically functionality of the control can be improved using ‘ConstrolState’.

Task 1: In the solution of the Web Site we have created in the Exercise 1, right click and add a new ‘ASP.NET Server Control’ project. Name it as ‘ASPNET_CustomControlLibrary’.

Task 2: Rename ‘ServerControl1.cs’ to ‘ConnectDataControl.cs’. Change the base class from ‘WebControl’ to ‘CompositControl’ This base class allows us to define functionality for the child controls contained into the composite custom control.

Task 3: On the class you will find the below attribute:

```
[ToolboxData("<{0}:ServerControl1 runat=server></{0}:ServerControl1>")]
```

This is used to display the custom control in the toolbox. Change the above attribute as below:

```
[ToolboxData("<{0}:ConnectDataControl
runat=server></{0}:ConnectDataControl>")]
```

The control will now be displayed in the tool box with name ‘ConnectDataControl’.

Remove the default ‘TextProperty’ from the class.

Task 4: Use the following namespaces for the database programming

```
using System.Data;
using System.Data.SqlClient;
```

Task 5: Define the following UI objects, properties to expose and Event for the control class:

```
  private TextBox txtDbServerName;
        private TextBox txtDbName;
        private DropDownList lstUserType;
        private TextBox txtUserName;
        private TextBox txtPassword;
        private TextBox txtTableName;
        private Button btnConnect;

        private Label lblDbServerName;
        private Label lblDbName;
        private Label lblUsertype;
        private Label lblUserName;
        private Label lblPassword;
        private Label lblTableName;


        public DataTable TableName { get; set; }

        public event EventHandler ConnectOperationClick;

        SqlConnection Conn;
        SqlDataAdapter AdTable;
        DataSet Ds;


        string _ConnstringInControlState;

        public string ConnstringInControlState
        {
            get { return _ConnstringInControlState; }
            set { _ConnstringInControlState = value; }
        }
```

TextBoxes, Labels are used to define UI of the control. ‘TableName’ property is used to expose datatable type to the container of the control. ‘ConnectOperationClick’ event is used to expose ‘TableName’ property to the container of the control.

Task 6: Override the ‘CreateChildControls()’ method of the base class. In this method we will be defining objects of UI type defined above and set its various properties.

```
    protected override void CreateChildControls()
        {
            txtDbServerName = new TextBox();
            txtDbServerName.ID = "txtDBServerName";
            this.Controls.Add(txtDbServerName);

            txtDbName = new TextBox();
            txtDbName.ID = "txtDbName";
            this.Controls.Add(txtDbName);

            lstUserType = new DropDownList();
            lstUserType.ID = "lstUserType";

            lstUserType.AutoPostBack = true;

            lstUserType.Items.Add("");
            lstUserType.Items.Add("Windows Authentication");
           // lstUserType.Items.Add("Sql Server AUthentication");
```

```
            lstUserType.SelectedIndexChanged += new
EventHandler(lstUserType_SelectedIndexChanged);

            this.Controls.Add(lstUserType);

            txtUserName = new TextBox();
            txtUserName.ID= "txtUserName";
            //txtUserName.Text = "Administrator";
            this.Controls.Add(txtUserName);

            txtPassword = new TextBox();
            txtPassword.ID = "txtPassword";
            txtPassword.TextMode = TextBoxMode.Password;
            this.Controls.Add(txtPassword);

            txtTableName = new TextBox();
            txtTableName.ID = "txtTableName";
            this.Controls.Add(txtTableName);

            btnConnect = new Button();
            btnConnect.ID = "btnConnect";
            btnConnect.Text = "Connect To Database";

            btnConnect.Click += new EventHandler(btnConnect_Click);

            this.Controls.Add(btnConnect);



            lblDbServerName = new Label();
            lblDbServerName.Text = "Database Server Name";
            this.Controls.Add(lblDbServerName);

            lblDbName = new Label();
            lblDbName.Text = "Database Name";
            this.Controls.Add(lblDbName);

            lblUsertype = new Label();
            lblUsertype.Text = "User Type";
            this.Controls.Add(lblUsertype);

            lblUserName = new Label();
            lblUserName.Text = "User Name";
            this.Controls.Add(lblUserName);
            lblPassword = new Label();
            lblPassword.Text = "Password";
            this.Controls.Add(lblPassword);

            lblTableName = new Label();
            lblTableName.Text = "Table Name";
            this.Controls.Add(lblTableName);
        }
```

Write the implementation of the 'SelectedIndexChanged' of the drop-down list as below:

```
void lstUserType_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (lstUserType.SelectedIndex == 1)
            {
                txtUserName.Text = "Administrator";
                _ConnstringInControlState = "Data Source = " +
txtDbServerName.Text.Trim() + ";Initial Catalog=" +
txtDbName.Text.Trim() + ";Integrated Security=SSPI";
            }

        }
```

The code above defines the connection string based upon the user selection for the user type drop down list.

Write the click event on the 'Connect' button where the database will be connected and the 'ConnectOperationClick' event will be clicked.

```
  void btnConnect_Click(object sender, EventArgs e)
        {

            Conn = new SqlConnection(ConnstringInControlState);
            AdTable = new SqlDataAdapter("Select * from " +
txtTableName.Text, Conn);

            Ds = new DataSet();

            AdTable.Fill(Ds, txtTableName.Text);

            TableName = Ds.Tables[txtTableName.Text];

            if (ConnectOperationClick != null)
            {
                ConnectOperationClick(this, EventArgs.Empty);
            }
        }
```

Now we will write the rendering for the UI, in the form of Html. Rendering is taken place using ‘HtmlTextWriter’ class. This class contains methods for Html rendering of tags. In the RenderContents() method we will write the code below:

```
protected override void RenderContents(HtmlTextWriter writer)
        {
            //The COde for Rendering Design
            writer.RenderBeginTag(HtmlTextWriterTag.Table); //Begin
Table

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //First Row

            //For Displaying "database Server Name Label And Text Box"

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first
Cell in First Row

            lblDbServerName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in First Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second
Cell in First Row

            txtDbServerName.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in First Row

            writer.RenderEndTag(); //End of First Row



            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Second Row


            //For Database Name Label and Text Box

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first
Cell in Second Row

            lblDbName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Second Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second
```

```
Cell in Second Row

            txtDbName.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Second Row

            //Ends Here

            writer.RenderEndTag(); //End of Second Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Third Row

            //For User Type
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first
Cell in Third Row

            lblUsertype.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Third Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second
Cell in Third Row

            lstUserType.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Third Row
            //Ends Here

            writer.RenderEndTag(); //End of Third Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Fourth Row

            //For User Name
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first
Cell in Fourth Row

            lblUserName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Fourth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second
Cell in Fourth Row

            txtUserName.RenderControl(writer);
            writer.RenderEndTag(); //End of Second Cell in Fourth Row
            //Ends Here

            writer.RenderEndTag(); //End of Fourth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Fifth Row

            //FOr Password
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first
Cell in Fifth Row

            lblPassword.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Fifth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second
Cell in Fifth Row

            txtPassword.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Fifth Row
            //Ends Here

            writer.RenderEndTag(); //End of Fifth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Sixth Row

            //For Table Name
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first
Cell in Sixth Row

            lblTableName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Sixth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second
Cell in Sixth Row

            txtTableName.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Sixth Row
            //Ends Here

            writer.RenderEndTag(); //End of Sixth Row
            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Seventh Row

            //FOr Button
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first
Cell in Seventh Row

            writer.RenderEndTag(); //End of First Cell  in Seventh Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second
Cell in Seventh Row

            btnConnect.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Seventh Row
            //Ends Here

            writer.RenderEndTag(); //End of Seventh Row

            writer.RenderEndTag(); //Table
            //base.RenderContents(writer);
        }
```

Now the most important is, to main the control state we need to override, ‘OnInit’, ‘LoadControlState()’ and ‘SaveControl()’ methods as below:

```
protected override void OnInit(EventArgs e)
        {
            Page.RegisterRequiresControlState(this);
            base.OnInit(e);
        }

        protected override object SaveControlState()
        {
            return _ConnstringInControlState;
            //return base.SaveControlState();
        }

        protected override void LoadControlState(object savedState)
        {
            _ConnstringInControlState = (string)savedState;
            //base.LoadControlState(savedState);
        }
```

The Complete Code of the control is as below:

```
using System;
using System.Web.UI;
using System.Web.UI.WebControls;

using System.Data;
using System.Data.SqlClient;

namespace ASPNET_CustomControlLibrary
{

    [ToolboxData("<{0}:ConnectDataControl
runat=server></{0}:ConnectDataControl>")]
    public class ConnectDataControl : CompositeControl /
    {
        private TextBox txtDbServerName;
        private TextBox txtDbName;
        private DropDownList lstUserType;
        private TextBox txtUserName;
        private TextBox txtPassword;
        private TextBox txtTableName;
        private Button btnConnect;

        private Label lblDbServerName;
        private Label lblDbName;
        private Label lblUsertype;
        private Label lblUserName;
        private Label lblPassword;
        private Label lblTableName;


        public DataTable TableName { get; set; }

        public event EventHandler ConnectOperationClick;

        SqlConnection Conn;
        SqlDataAdapter AdTable;
        DataSet Ds;


        string _ConnstringInControlState;

        public string ConnstringInControlState
        {
            get { return _ConnstringInControlState; }
            set { _ConnstringInControlState = value; }
        }

        protected override void CreateChildControls()
        {
            txtDbServerName = new TextBox();
            txtDbServerName.ID = "txtDBServerName";
            this.Controls.Add(txtDbServerName);

            txtDbName = new TextBox();
            txtDbName.ID = "txtDbName";
            this.Controls.Add(txtDbName);

            lstUserType = new DropDownList();
            lstUserType.ID = "lstUserType";

            lstUserType.AutoPostBack = true;

            lstUserType.Items.Add("");
            lstUserType.Items.Add("Windows Authentication");
           // lstUserType.Items.Add("Sql Server AUthentication");

            lstUserType.SelectedIndexChanged += new
EventHandler(lstUserType_SelectedIndexChanged);

            this.Controls.Add(lstUserType);

            txtUserName = new TextBox();
            txtUserName.ID= "txtUserName";
            //txtUserName.Text = "Administrator";
            this.Controls.Add(txtUserName);

            txtPassword = new TextBox();
            txtPassword.ID = "txtPassword";
            txtPassword.TextMode = TextBoxMode.Password;
            this.Controls.Add(txtPassword);

            txtTableName = new TextBox();
            txtTableName.ID = "txtTableName";
            this.Controls.Add(txtTableName);

            btnConnect = new Button();
            btnConnect.ID = "btnConnect";
            btnConnect.Text = "Connect To Database";

            btnConnect.Click += new EventHandler(btnConnect_Click);

            this.Controls.Add(btnConnect);



            lblDbServerName = new Label();
            lblDbServerName.Text = "Database Server Name";
            this.Controls.Add(lblDbServerName);

            lblDbName = new Label();
            lblDbName.Text = "Database Name";
            this.Controls.Add(lblDbName);

            lblUsertype = new Label();
            lblUsertype.Text = "User Type";
            this.Controls.Add(lblUsertype);

            lblUserName = new Label();
            lblUserName.Text = "User Name";
            this.Controls.Add(lblUserName);

            lblPassword = new Label();
            lblPassword.Text = "Password";
            this.Controls.Add(lblPassword);

            lblTableName = new Label();
            lblTableName.Text = "Table Name";
            this.Controls.Add(lblTableName);
        }

        void lstUserType_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (lstUserType.SelectedIndex == 1)
            {
                txtUserName.Text = "Administrator";

                _ConnstringInControlState = "Data Source = " +
txtDbServerName.Text.Trim() + ";Initial Catalog=" + txtDbName.Text.Trim() +
";Integrated Security=SSPI";
            }

        }


        protected override void OnInit(EventArgs e)
        {
            Page.RegisterRequiresControlState(this);
            base.OnInit(e);
        }

        protected override object SaveControlState()
        {
            return _ConnstringInControlState;
            //return base.SaveControlState();
        }

        protected override void LoadControlState(object savedState)
        {
            _ConnstringInControlState = (string)savedState;
            //base.LoadControlState(savedState);
        }

        void btnConnect_Click(object sender, EventArgs e)
        {
            //Conn = new SqlConnection(ViewState["ConnStr"].ToString());
            Conn = new SqlConnection(ConnstringInControlState);
            AdTable = new SqlDataAdapter("Select * from " +
txtTableName.Text, Conn);

            Ds = new DataSet();

            AdTable.Fill(Ds, txtTableName.Text);

            TableName = Ds.Tables[txtTableName.Text];

            if (ConnectOperationClick != null)
            {
                ConnectOperationClick(this, EventArgs.Empty);
            }
        }

        protected override void RenderContents(HtmlTextWriter writer)
        {
            //The COde for Rendering Design
            writer.RenderBeginTag(HtmlTextWriterTag.Table); //Begin Table

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //First Row

            //For Displaying "database Server Name Label And Text Box"

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first Cell in
First Row

            lblDbServerName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in First Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second Cell in
First Row

            txtDbServerName.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in First Row

            writer.RenderEndTag(); //End of First Row



            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Second Row
            //For Database Name Label and Text Box

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first Cell in
Second Row

            lblDbName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Second Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second Cell in
Second Row

            txtDbName.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Second Row

            //Ends Here

            writer.RenderEndTag(); //End of Second Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Third Row

            //For User Type
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first Cell in
Third Row

            lblUsertype.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Third Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second Cell in
Third Row

            lstUserType.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Third Row
            //Ends Here

            writer.RenderEndTag(); //End of Third Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Fourth Row

            //For User Name
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first Cell in
Fourth Row

            lblUserName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Fourth Row
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second Cell in 
Fourth Row

            txtUserName.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Fourth Row
            //Ends Here

            writer.RenderEndTag(); //End of Fourth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Fifth Row

            //FOr Password
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first Cell in 
Fifth Row

            lblPassword.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Fifth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second Cell in 
Fifth Row

            txtPassword.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Fifth Row
            //Ends Here

            writer.RenderEndTag(); //End of Fifth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Sixth Row

            //For Table Name
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first Cell in 
Sixth Row

            lblTableName.RenderControl(writer);

            writer.RenderEndTag(); //End of First Cell  in Sixth Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second Cell in 
Sixth Row

            txtTableName.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Sixth Row
            //Ends Here

            writer.RenderEndTag(); //End of Sixth Row
            writer.RenderBeginTag(HtmlTextWriterTag.Tr); //Seventh Row

            //FOr Button
            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The first Cell in
Seventh Row

            writer.RenderEndTag(); //End of First Cell  in Seventh Row

            writer.RenderBeginTag(HtmlTextWriterTag.Td); //The Second Cell in
Seventh Row

            btnConnect.RenderControl(writer);

            writer.RenderEndTag(); //End of Second Cell in Seventh Row
            //Ends Here

            writer.RenderEndTag(); //End of Seventh Row

            writer.RenderEndTag(); //Table
            //base.RenderContents(writer);
        }
    }
}
```

Task 8: Build the control, make sure that it is error free.

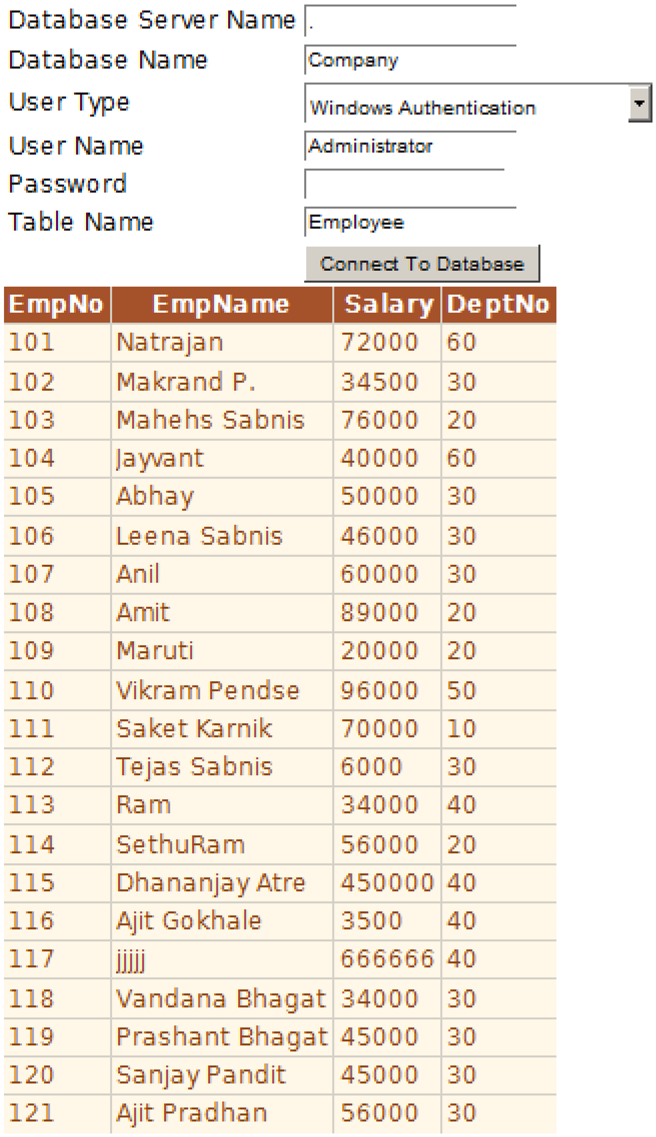
Task 9: In the Web Site project add a new WebForm, name it as, ‘WebForm_DataAccessCustomCOntrol.aspx’.

Task 10: Drag the custom control created above from the toolbox and drop it on this page. Also drag drop the GridView on the page. Name this GridView as ‘gdvData’.

Task 11: Open ‘WebForm_DataAccessCustomCOntrol.aspx.cs’ the code behind file and register for the event of the custom control as below:

```
protected void Page_Load(object sender, EventArgs e)
        {
            ConnectDataControl1.ConnectOperationClick += new
EventHandler(ConnectDataControl1_ConnectOperationClick);
        }

        void ConnectDataControl1_ConnectOperationClick(object sender,
EventArgs e)
        {
            gdvData.DataSource = ConnectDataControl1.TableName;
            gdvData.DataBind();
        }
```

Task 12: Run the page, the following result will be displayed:



Database Server Name	.
Database Name	Company
User Type	Windows Authentication
User Name	Administrator
Password	
Table Name	Employee

EmpNo	EmpName	Salary	DeptNo
101	Natrajan	72000	60
102	Makrand P.	34500	30
103	Mahehs Sabnis	76000	20
104	Jayvant	40000	60
105	Abhay	50000	30
106	Leena Sabnis	46000	30
107	Anil	60000	30
108	Amit	89000	20
109	Maruti	20000	20
110	Vikram Pendse	96000	50
111	Saket Karnik	70000	10
112	Tejas Sabnis	6000	30
113	Ram	34000	40
114	SethuRam	56000	20
115	Dhananjay Atre	450000	40
116	Ajit Gokhale	3500	40
117	jjjjj	666666	40
118	Vandana Bhagat	34000	30
119	Prashant Bhagat	45000	30
120	Sanjay Pandit	45000	30
121	Ajit Pradhan	56000	30