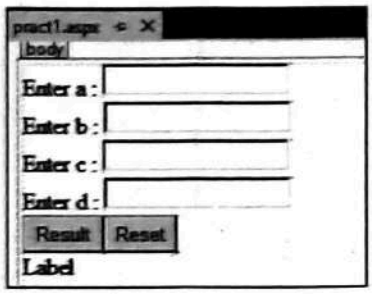
PRACTICAL JOURNAL

Practical 1 : Working with basic C# and ASP .NET

Practical 1(a) : Create an application that obtains four int values from the user and displays the product.

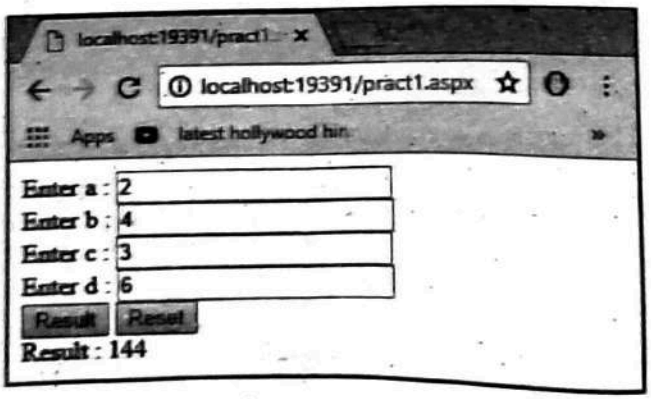
pract1.aspx



pract1.aspx.cs

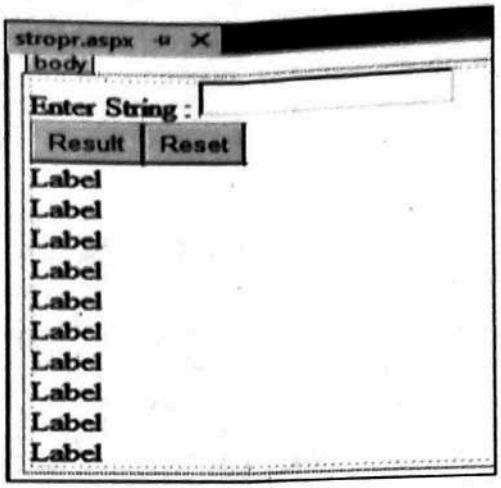
```
public partial class pract1 : System.Web.UI.Page
{
protected void btnResult_Click(object sender, EventArgs e)
  {
    int r;
    r = Convert.ToInt32(TextBox1.Text) * Convert.ToInt32(TextBox2.Text) *
Convert.ToInt32(TextBox3.Text) * Convert.ToInt32(TextBox4.Text);
    Label1.Text = "Result : "+r.ToString( );
  }

  protected void btnReset_Click(object sender, EventArgs e)
  {
    TextBox1.Text = "";
    TextBox2.Text = "";
    TextBox3.Text = "";
    TextBox4.Text = "";
    Label1.Text = "";
  }
}
```

OUTPUT



Practical 1(b) : Create an application to demonstrate string operations.

stropr.aspx

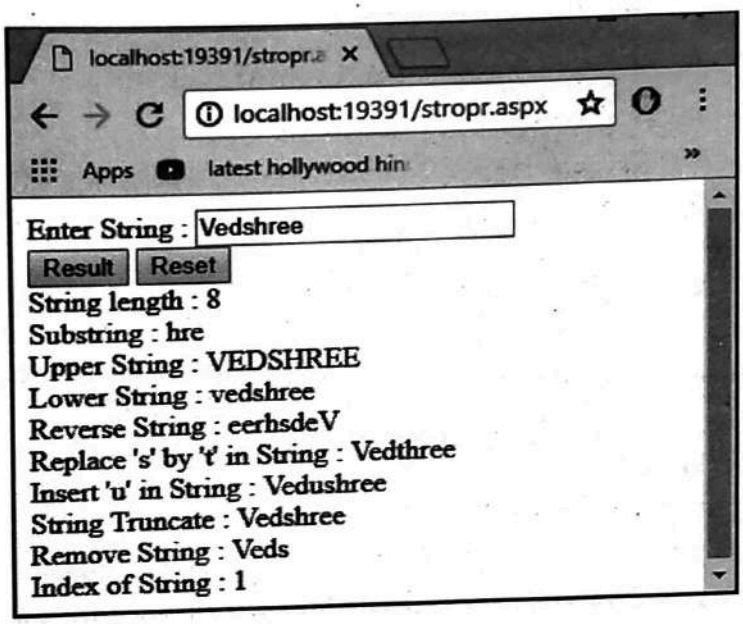


stropr.aspx.cs

```
public partial class stropr : System.Web.UI.Page
{
protected void btnResult_Click(object sender, EventArgs e)
  {
     string s = TextBox1.Text;
     Label1.Text = "String length : "+s.Length;
     Label2.Text = "Substring : "+s.Substring(4, 3);
     Label3.Text = "Upper String : "+s.ToUpper( );
     Label4.Text = "Lower String : "+s.ToLower( );
     string  rev = "";
     for (int i = s.Length - 1; i >= 0; i--)
     {
        rev = rev + s[i];
     }
     Label5.Text = "Reverse String : "+ rev.ToString( );
     Label6.Text = "Replace 's' by 't' in String : "+s.Replace('s','t');
     Label7.Text = "Insert 'u' in String : "+s.Insert(3,"u");
     Label8.Text = "String Truncate : "+s.Trim( );
     Label9.Text = "Remove String : "+s.Remove(4);
     Label10.Text = "Index of String : "+s.IndexOf('e');
  }
  protected void btnReset_Click(object sender, EventArgs e)
  {
     Label1.Text = "";
     Label2.Text = "";
     Label3.Text = "";
     Label4.Text = "";
     Label5.Text = "";
```

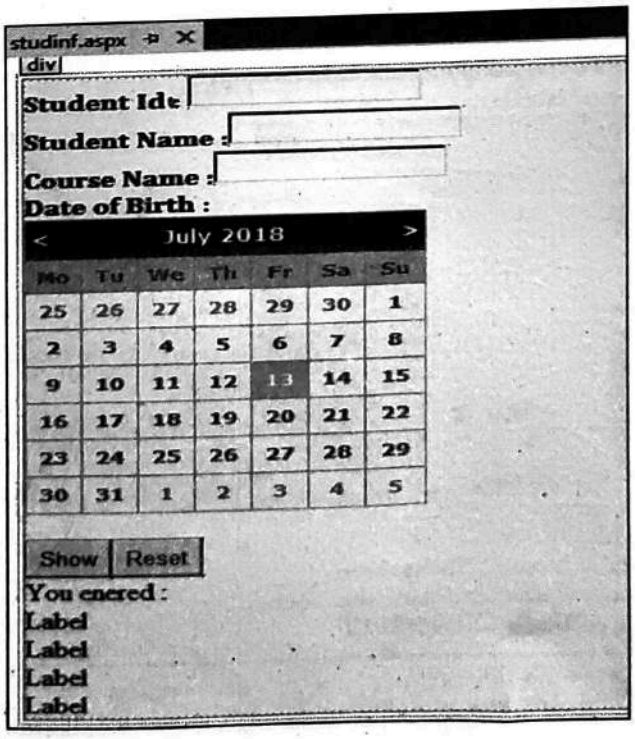
```
        Label6.Text = "";
        Label7.Text = "";
        Label8.Text = "";
        Label9.Text = "";
        Label10.Text = "";
        TextBox1.Text = "";
    }
}
```

OUTPUT



Practical 1(c) : Create an application that receives the (Student Id, Student Name, Course Name, Date of Birth) information from a set of students. The application should also display the information of all the students once the data entered.

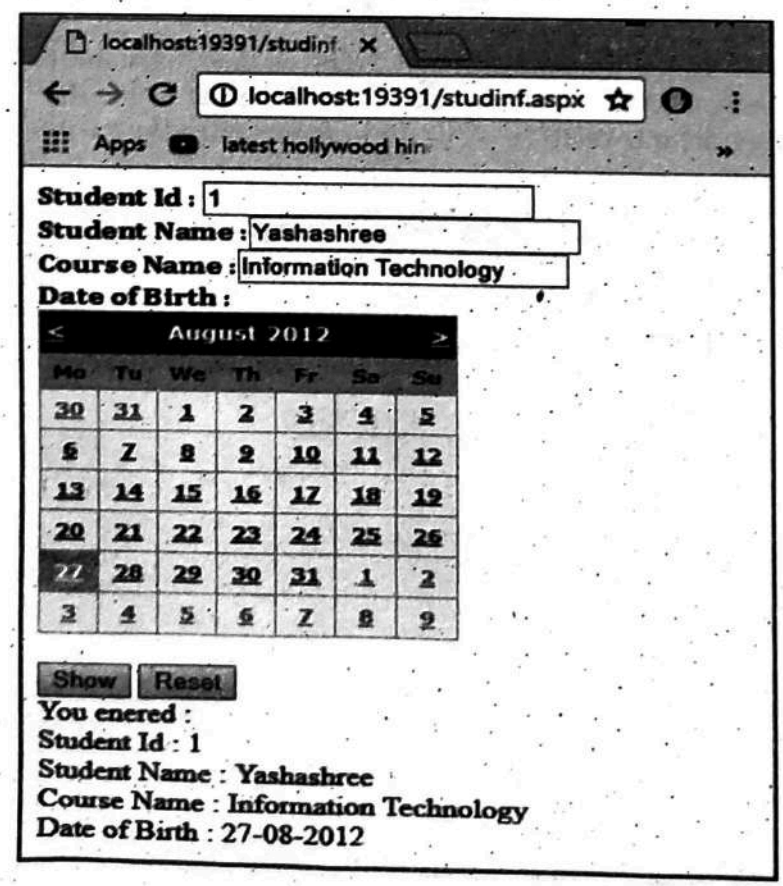
studinf.aspx



studinf.aspx.cs

```
public partial class studinf : System.Web.UI.Page
{
protected void btnShow_Click(object sender, EventArgs e)
  {
    Label1.Text = "Student Id : " + TextBox1.Text;
    Label2.Text = "Student Name : " + TextBox2.Text;
    Label3.Text = " Course Name : " + TextBox3.Text;
    Label4.Text= " Date of Birth : " + Calendar1.SelectedDate.ToShortDateString( );
  }
  protected void btnReset_Click(object sender, EventArgs e)
  {
    Label1.Text = "";
    Label2.Text = "";
    Label3.Text = "";
    Label4.Text = "";
    TextBox1.Text = "";
    TextBox2.Text = "";
    TextBox3.Text = "";
    Calendar1.SelectedDates.Clear( ) ;
  }
}
```

Output



Practical 1(d) : Create an application to demonstrate following operations :

i) **Generate Fibonacci series.**

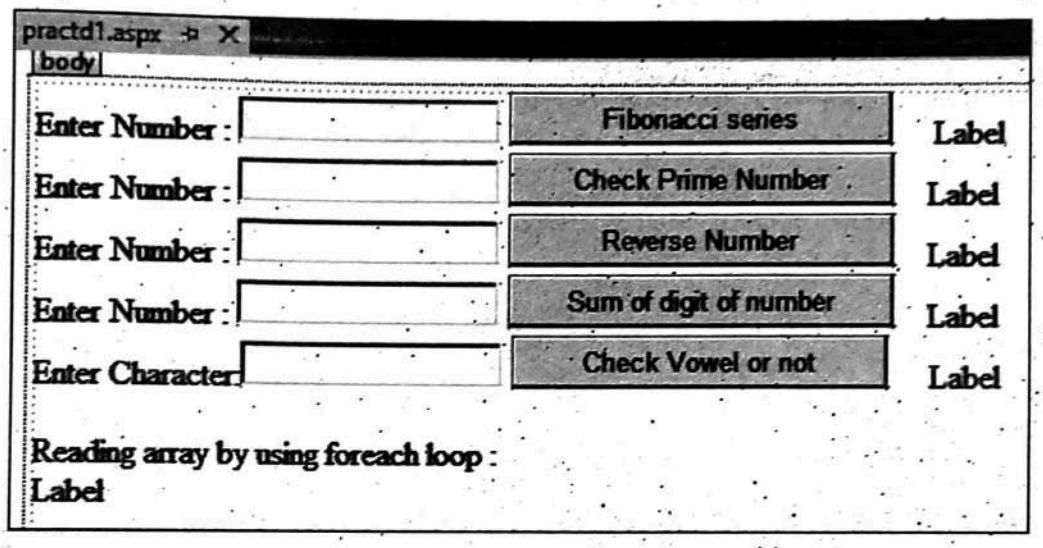
ii) **Test for prime numbers.**

iii) **Test for vowels.**

iv) **Use of foreach loop with arrays**

v) **Reverse a number and find sum of digits of a number.**

practd1.aspx



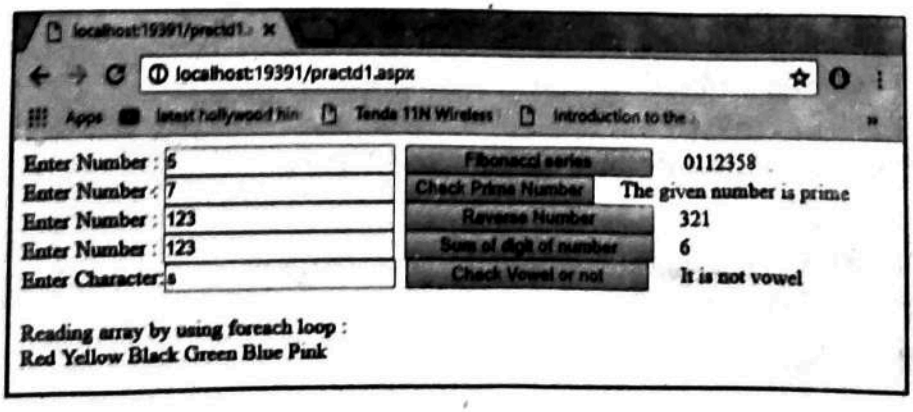
practd1.aspx.cs

```
public partial class practd1 : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
  {
    Label6.Text = "";
    string[] ColorNames = new string[] { "Red", "Yellow", "Black", "Green", "Blue", "Pink"
    };
    foreach (string ColorName in ColorNames)
    {
      Label6.Text = Label6.Text + "  "+ColorName.ToString( );
    }

}

protected void btnFibbo_Click(object sender, EventArgs e)
{
  int a, b, c,i,n;
  a = 0;
  b = 1;
  Label1.Text = a.ToString( )+ b.ToString( );
  n = Convert.ToInt32(TextBox1.Text);
  for (i = 1; i <= n; ++i)
```

```
    {
      c = a + b;
      Label1.Text = Label1.Text + c.ToString( );
      a = b;
      b = c;
    }
  }

  protected void btnPrm_Click(object sender, EventArgs e)
  {
    int i, c = 0, j, num;
    num = Convert.ToInt32(TextBox2.Text);
    for (j = 1; j <= num; j++)
    {
      i = num % j;
      if (i == 0)
      {
        c = c + 1;
      }
    }
    if (c == 2)
      Label2.Text = "The given number is prime";
    else
      Label2.Text = "The given number is not prime";
  }
    protected void btnRev_Click(object sender, EventArgs e)
    {
      long num, i, sum = 0;
      num = Convert.ToInt32(TextBox3.Text);
      while (num > 0)
      {
        i = num % 10;
        sum = i + sum * 10;
        num = num / 10;
      }
      Label3.Text = sum.ToString( );

    }
    protected void btnSmd_Click(object sender, EventArgs e)
    {
      long num, i, sum = 0;
      num = Convert.ToInt32(TextBox4.Text);
      while (num > 0)
```

```
    {
      i = num % 10;
      sum = i + sum;
      num = num / 10;
    }
    Label4.Text = sum.ToString( );
  }
  protected void btnVowel_Click(object sender, EventArgs e)
  {
    char c = Convert.ToChar(TextBox5.Text);
    switch (c)
    {
      case 'a':
        Label5.Text = "a is vowel";
        break;
      case 'e':
        Label5.Text = "e is vowel";
        break;
      case 'i':
        Label5.Text = "i is vowel";
        break;
      case 'o':
        Label5.Text = "o is vowel";
        break;
      case 'u':
        Label5.Text = "u is vowel";
        break;
      default:
        Label5.Text = "It is not vowel";
        break;
    }
  }
}
```

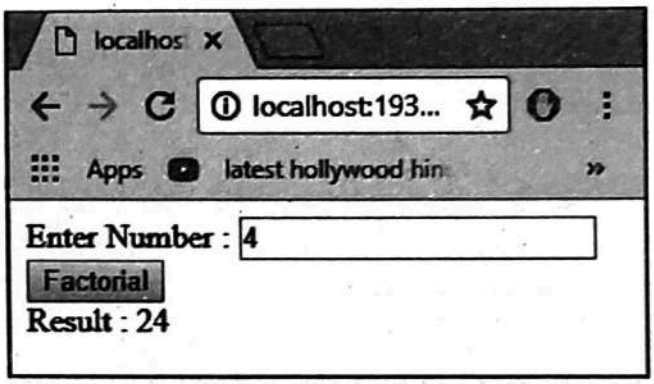
OUTPUT



Practical 2 : Working with Object Oriented C# and ASP .NET

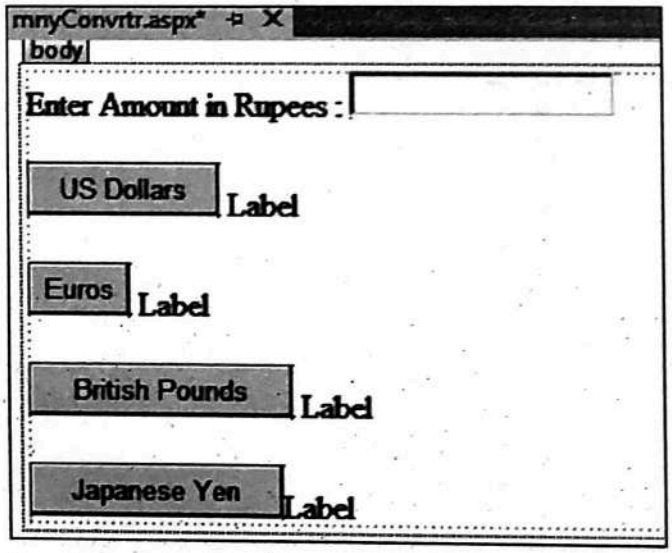
Practical 2(a) : Create simple application to perform following operations :

i) Finding factorial Value



ii) Money Conversion

mnyConvrtr.aspx

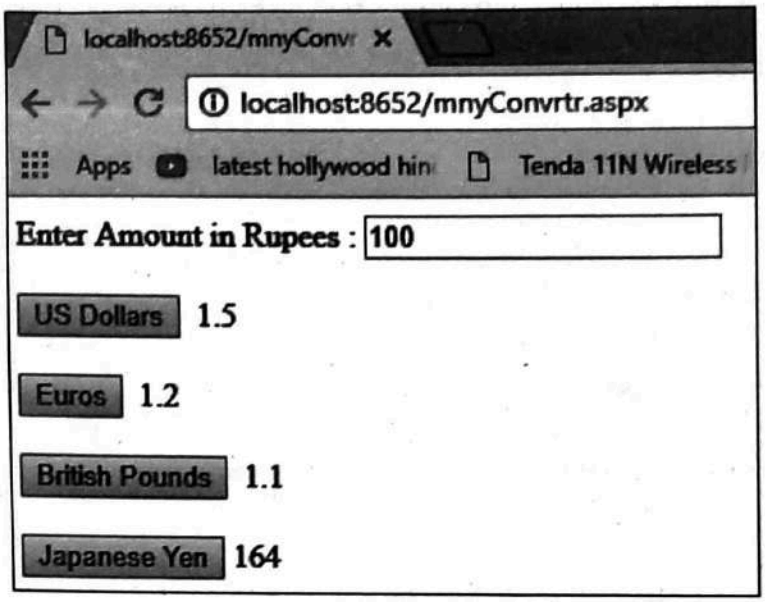


mnyConvrtr.aspx.cs

```
public partial class mnyConvrtr : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
   }
protected void btnDolr_Click(object sender, EventArgs e)
   {
      curConv s = new curConv( );
      double r = Convert.ToDouble(TextBox1.Text);
      double rate = s.Dolr(r);
      Label1.Text = rate.ToString( );
   }
   protected void btnEuros_Click(object sender, EventArgs e)
   {
      curConv s = new curConv( );
```

```
    double r = Convert.ToDouble(TextBox1.Text);
    double rate = s.Euros(r);
    Label2.Text = rate.ToString( );
  }
  protected void btnPound_Click(object sender, EventArgs e)
  {
    curConv s = new curConv( );
    double r = Convert.ToDouble(TextBox1.Text);
    double rate = s.Pound(r);
    Label3.Text = rate.ToString( );
  }
  protected void btnYen_Click(object sender, EventArgs e)
  {
    curConv s = new curConv( );
    double r = Convert.ToDouble(TextBox1.Text);
    double rate = s.Yen(r);
    Label4.Text = rate.ToString( );
  }
}

  public class curConv
  {
    public double Dolr(double r)
    {
      r = r * 0.015;
      return r;
    }
    public double Euros(double r)
    {
      r = r * 0.012;
      return r;
    }
    public double Pound(double r)
    {
      r = r * 0.011;
      return r;
    }
    public double Yen(double r)
    {
      r = r * 1.64;
      return r;
    }
}
```

OUTPUT



iii) Quadratic Equation

The Standard Form of a Quadratic Equation looks like this :

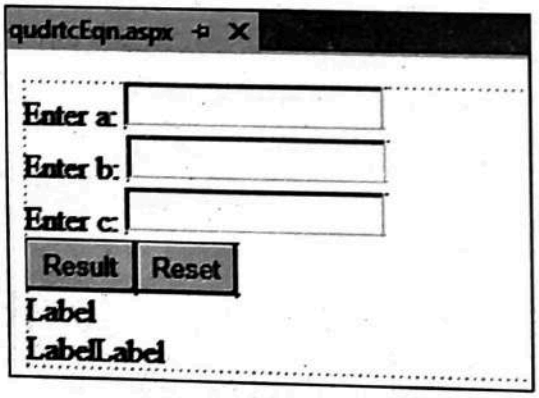
Quadratic Equation: $ax^2 + bx + c = 0$

where

a, b and c are known values. a can't be 0.

"x" is the variable or unknown

qudrtcEqn.aspx

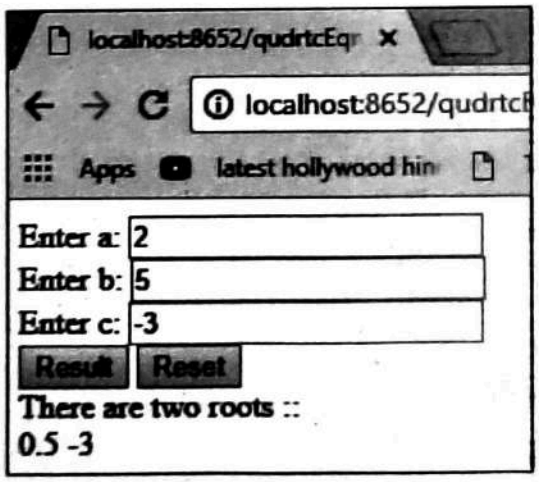


qudrtcEqn.aspx.cs

```
public void demo( )
  {
    double a, b, c, r1, r2, x;
    double det;
    a = Convert.ToInt32(txtBoxa.Text);
    b = Convert.ToInt32(txtBoxb.Text);
    c = Convert.ToInt32(txtBoxc.Text);
    det = (b * b) - (4 * a * c);
    if (det > 0)
    {
      x = Math.Sqrt(det);
```

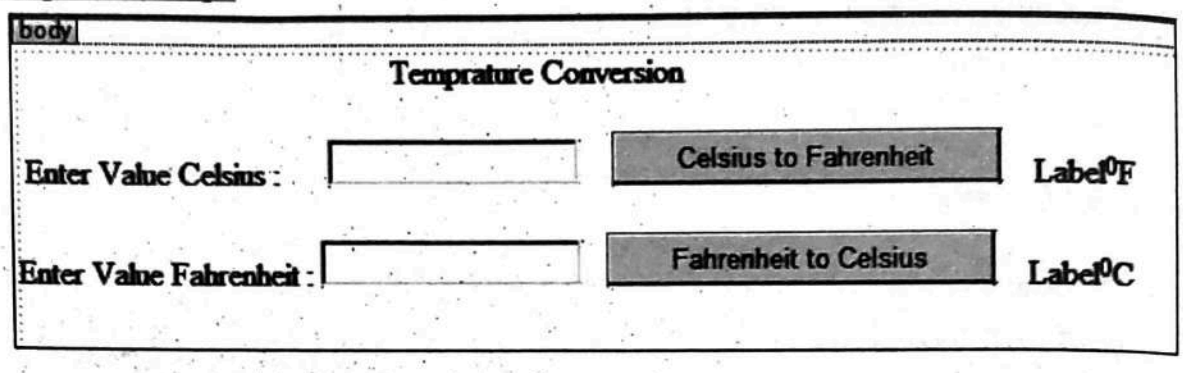
```
        r1 = (-b + x) / (2 * a);
        r2 = (-b - x) / (2 * a);
        Label3.Text = "There are two roots :: ";
        Label1.Text = r1.ToString( );
        Label2.Text = r2.ToString( );
      }
      else if (det == 0)
      {
        x = Math.Sqrt(det);
        r1 = (-b + x) / (2 * a);
        Label1.Text = "There is only one root :: ";
        Label2.Text = r1.ToString( );
      }
      else
      {
        Label1.Text = "There is no root !!!";
      }

  }
  protected void Page_Load(object sender, EventArgs e)
  {

  }
    protected void Button1_Click(object sender, EventArgs e)
  {
     demo( );
  }
```

OUTPUT



iv) Temperature Conversion

tempConvrtr.aspx

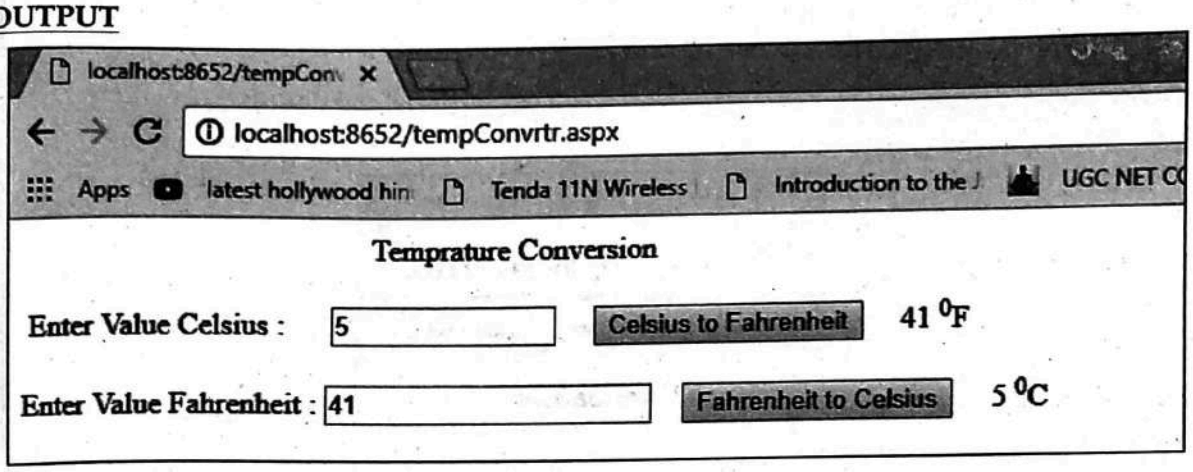


tempConvrtr.aspx.cs

```
public class tempConv
{
   public double ctof(double temp)
   {
      temp = 9.0 / 5.0 * temp + 32;
      return temp;
   }
   public double ftoc(double temp)
   {
      temp = (temp - 32) * 5 / 9;
      return temp;
   }
}
public partial class tempConvrtr : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
   {

   }

   protected void btnCtoF_Click(object sender, EventArgs e)
   {
      tempConv s = new tempConv( );
      double n = Convert.ToDouble(TextBox1.Text);
      double x = s.ctof(n);
      Label1.Text = x.ToString( );
   }

  protected void btnFtoC_Click(object sender, EventArgs e)
  {
     tempConv s = new tempConv( );
```

```
        double n = Convert.ToDouble(TextBox2.Text);
        double x = s.ftoc(n);
        Label2.Text = x.ToString( );
    }
}
```

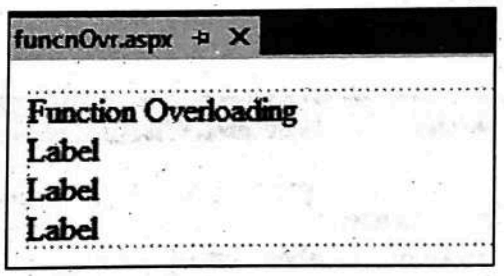
OUTPUT



Practical 2(b) : Create simple application to demonstrate use of following concepts :

i) Function Overloading

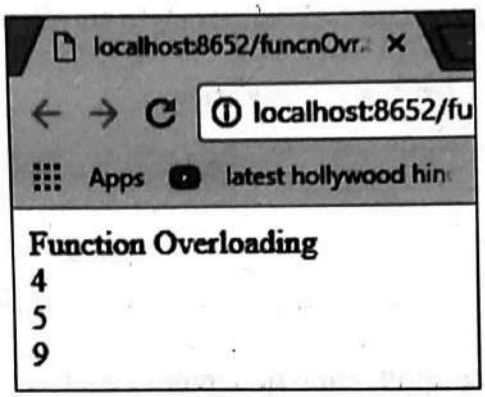
funcnOvr .aspx



funcnOvr .aspx.cs

```
public partial class funcnOvr : System.Web.UI.Page
{
public int add(int a)
  {
    return a + a;
  }
  public int add(int a, int b)
  {
    return a + b;
  }
  public int add(int a, int b, int c)
  {
    return a + b + c;
  }
  protected void Page_Load(object sender, EventArgs e)
  {
```

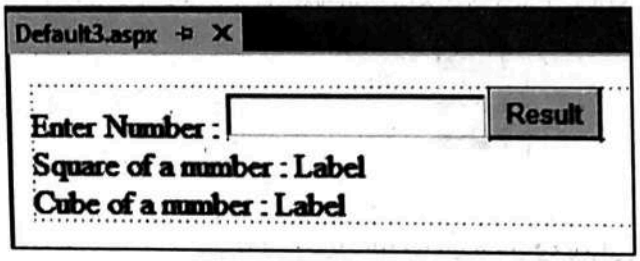
```
    int x, y, z;
    x = add(2);
    y = add(2, 3);
    z = add(2, 3, 4);
    Label1.Text = x.ToString( );
    Label2.Text = y.ToString( );
    Label3.Text = z.ToString( );
  }
}
```



ii) Inheritance (all types)

1. Single Inheritance

Default3.aspx



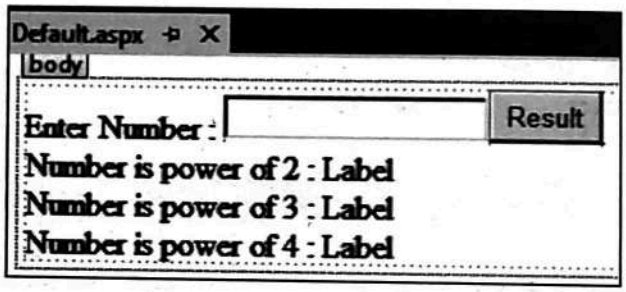
Default3.aspx.cs

```
public partial class Default3 : System.Web.UI.Page
{
    protected void btnResult_Click(object sender, EventArgs e)
    {
        B s = new B( );
        int n =Convert.ToInt32(TextBox1.Text);
        int x = s.sqr(n);
        int y = s.cub(n);
        Label1.Text = x.ToString( );
        Label2.Text = y.ToString( );
    }
}

    public class A
```

```
    {
      public int sqr(int Val1)
      {
       return Val1 * Val1;
      }
    }

    public class B : A
    {
      public int cub(int Val1)
      {
        int v1 = sqr(Val1);
        return v1 * Val1;
      }
    }
```

2. Multilevel Inheritance

Default.aspx

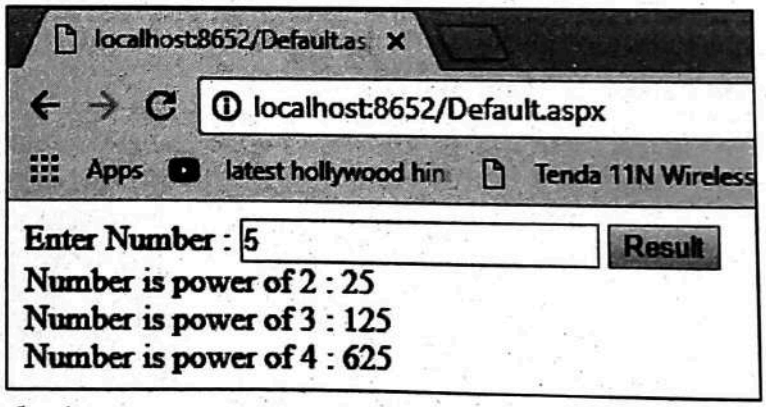


Default.aspx.cs

```
public partial class Default: System.Web.UI.Page
{
    protected void btnResult_Click(object sender, EventArgs e)
    {
      C s = new C( );
      int n = Convert.ToInt32(TextBox1.Text);
      int x = s.pow2(n);
      int y = s.pow3(n);
      int z= s.pow4(n);
      Label1.Text = x.ToString( );
      Label2.Text = y.ToString( );
      Label3.Text = z.ToString( );
    }
}

  public class A
    {
```

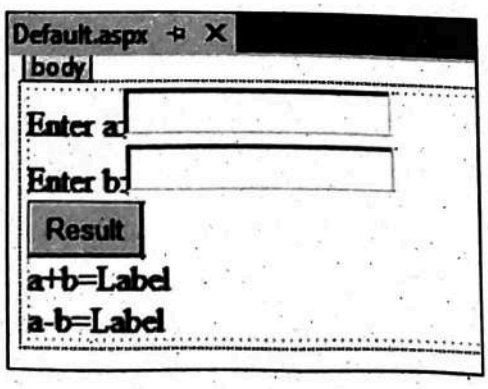
```
    public int pow2(int Val1)
    {
      return Val1 * Val1;
    }
  }

  public class B : A
  {
    public int pow3(int Val1)
    {
      int v1 = pow2(Val1);
      return v1 * Val1;
    }
  }

  public class C : B
  {
    public int pow4(int Val1)
    {
      int v1 = pow3(Val1);
      return v1 * Val1;
    }
  }
```

OUTPUT



3. Hierarchical Inheritance

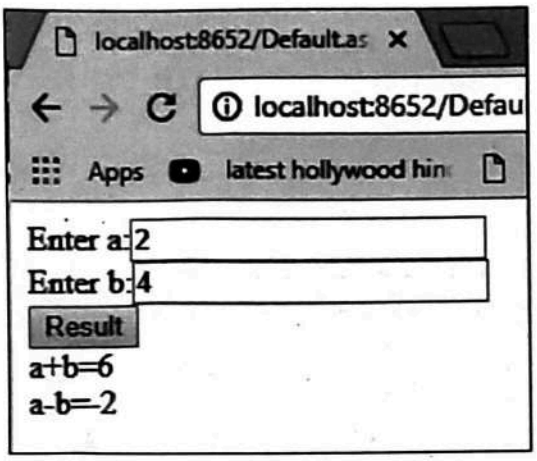
Default.aspx



Default.aspx.cs

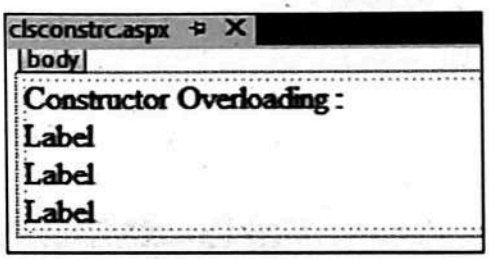
```
public class A
  {
     public int a;
     public int b;
  }
  public class B : A
  {
     public int add(int Val1, int Val2)
     {
        a = Val1;
        b = Val2;
        return a+b;
     }
  }
  public class C : A
  {
     public int sub(int Val1, int Val2)
     {
        a = Val1;
        b = Val2;
        return a - b;
     }
  }
public partial class  Default : System.Web.UI.Page
{
 protected void btnResult_Click1(object sender, EventArgs e)
  {
     B s1 = new B( );
     C s2 = new C( );
     int m = Convert.ToInt32(TextBox1.Text);
     int n = Convert.ToInt32(TextBox2.Text);
     int x = s1.add(m, n);
     int y = s2.sub(m, n);
     Label1.Text = x.ToString( );
     Label2.Text = y.ToString( );
  }
}
```

OUTPUT



iii) Constructor overloading

clsconstrc.aspx

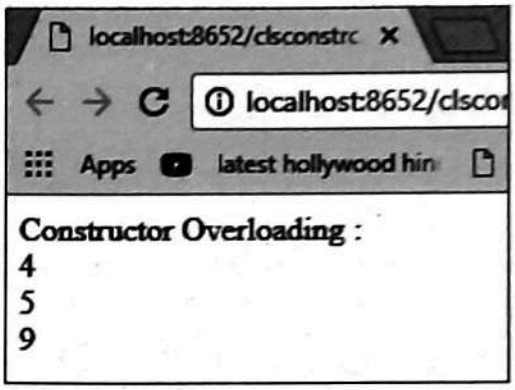


clsconstrc.aspx.cs

```
public partial class clsconstrc : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
   {
      add obj1 = new add(2);
      add obj2 = new add(2,3);
      add obj3 = new add(2,3,4);
      Label1.Text= obj1.r.ToString( );
      Label2.Text = obj2.r.ToString( );
      Label3.Text = obj3.r.ToString( );
   }
}
   public class add
   {
      public int r;
      public add(int a)
      {
         r = a + a;
      }
      public add(int a, int b)
      {
         r = a + b;
```

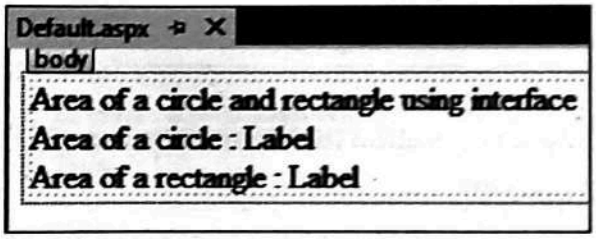
```
    }
    public add(int a, int b,int c)
    {
       r = a + b+c;
    }
  }
```

OUTPUT



iv) Interfaces

Default.aspx

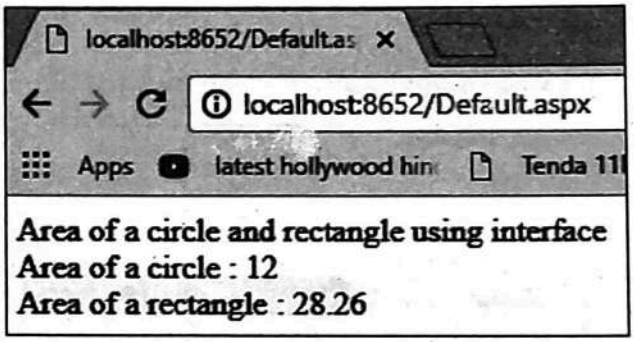


Default.aspx.cs

```
interface Area
  {
     double show(double s, double t);
  }
  class Rect : Area
  {
     public double show(double s, double t)
     {
        return s * t;
     }
  }
  class Circle :Area
  {
     public double show(double s, double t)
     {
     return (3.14 * s * s);
     }
  }
```

```
public partial class  Default : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      Rect r1 = new Rect( );
      double x=r1.show(3, 4);
      Circle c1 = new Circle( );
      double y=c1.show(3, 4);
      Label1.Text = x.ToString( );
      Label2.Text = y.ToString( );
   }
}
```

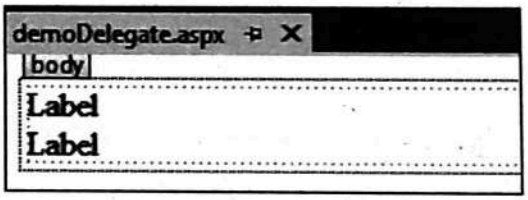
OUTPUT



Practical 2(c) Create simple application to demonstrate use of following concepts

i) Using Delegates and events

demoDelegate.aspx

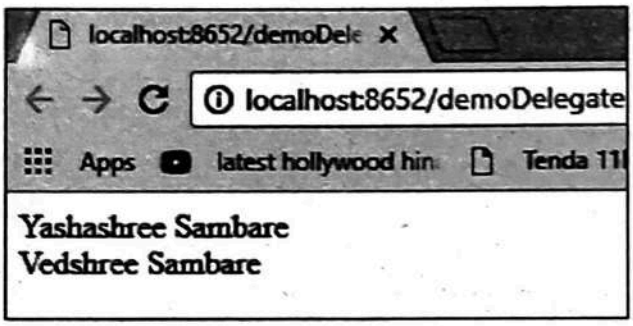


demoDelegate.aspx.cs

```
public delegate string dele( );
   public static string display1( )
   {
      string s1="Yashashree Sambare";
      return s1;
   }
   public static string display2( )
   {
      string s2 = "Vedshree Sambare";
      return s2;
   }
     protected void Page_Load(object sender, EventArgs e)
   {
```

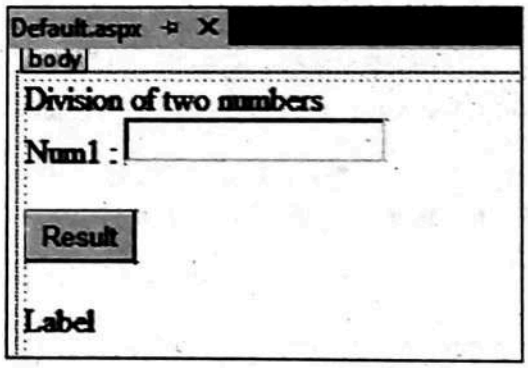
```
    dele d1 = new dele(display1);
    d1( );
    dele d2 = new dele(display2);
    d2( );
    Label1.Text = d1( );
    Label2.Text = d2( );
  }
```

OUTPUT



ii) Exception handling

Default.aspx

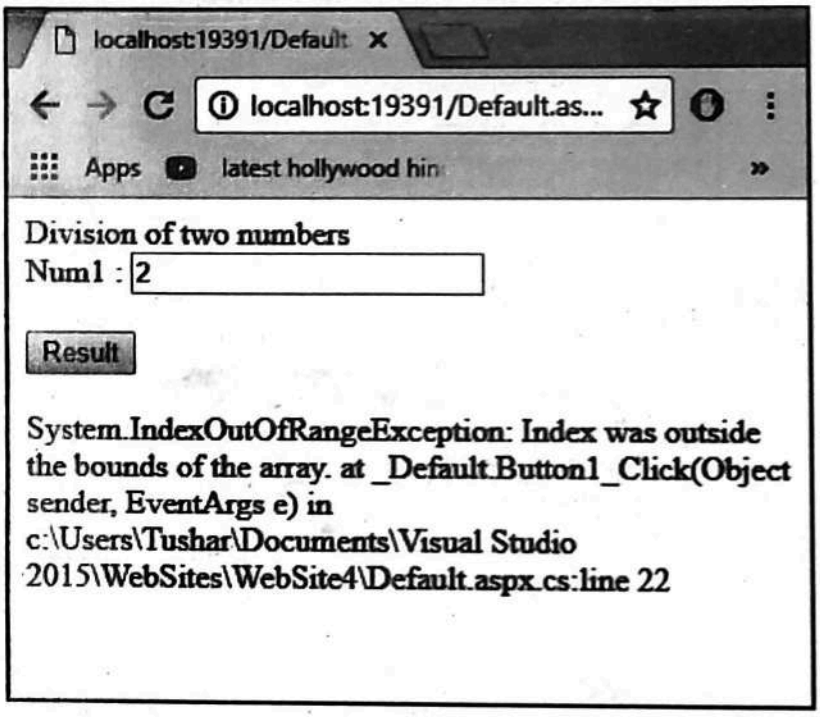


Default.aspx.cs

```
public partial class _Default : System.Web.UI.Page
{

protected void Button1_Click(object sender, EventArgs e)
  {
    try
    {
      int a = Convert.ToInt32(TextBox1.Text);
      int[] b ={ 12, 23, 33 };
      int resultVal;
      resultVal = (b[3]/a);
      Label1.Text = "The result is : " + resultVal.ToString( );
    }
    catch (System.DivideByZeroException  ex)
```

```
    {
      Label1.Text = ex.ToString( );
    }
    catch (System.IndexOutOfRangeException ex)
    {
      Label1.Text = ex.ToString( );
    }
  }
}
```

OUTPUT :

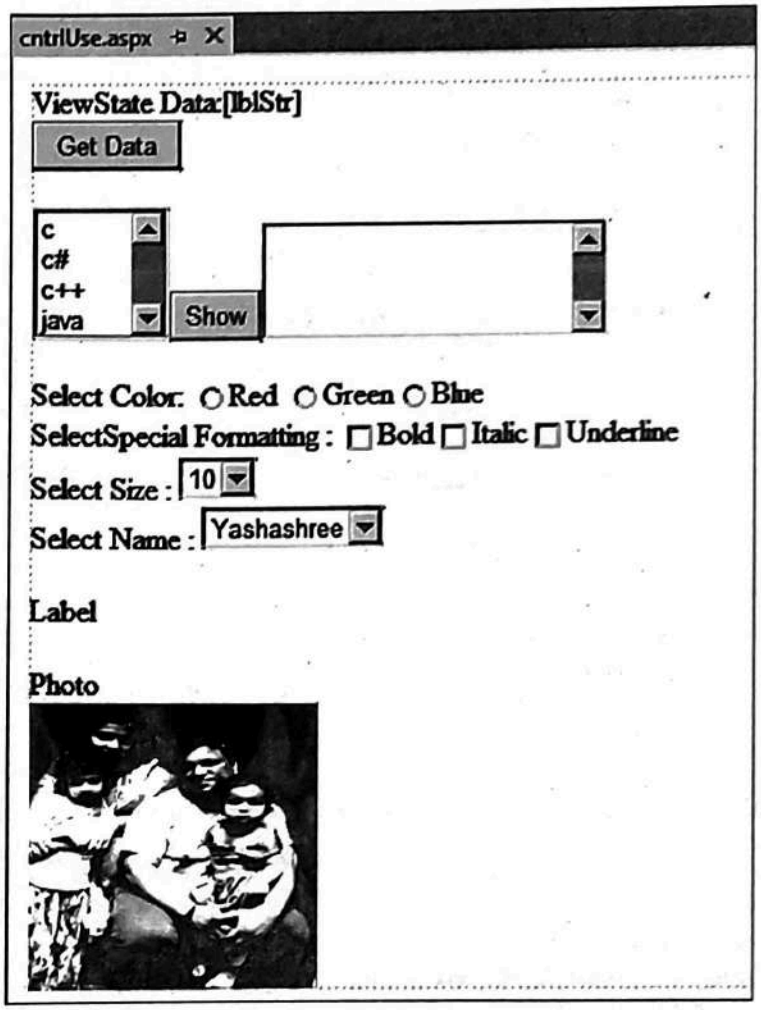


Practical 3 : Working with Web Forms and Controls

Practical 3(a) : Create a simple web page with various sever controls to demonstrate setting and use of their properties. (Example : AutoPostBack)

1. On click of a button control display the selected items from the list in a textbox. Also in the same webpage display the name of the selected item from the DropDownList1 in a label. Also change the font size of the same label according to the font size selected from the dropdownlist2.
2. Image control for photo.
3. Check Boxes provides special formatting and Radio Buttons provides colour.
4. Use of AutoPostBack property.

cntrlUse.aspx

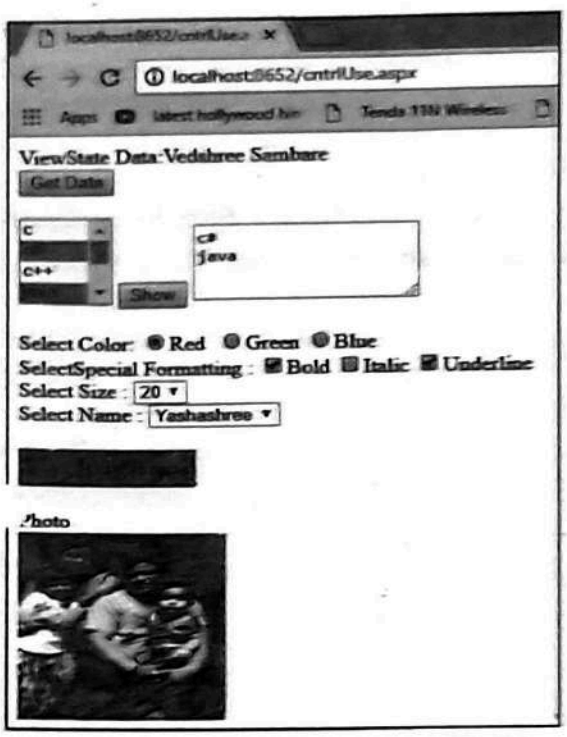


cntrlUse.aspx.cs

```
public partial class _Default : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
  {
     if (!IsPostBack)
     {
        string str = "Vedshree Sambare";
        if (ViewState["name"] == null)
        {
           ViewState["name"] = str;
        }
     }

  }
  protected void btn_Click(object sender, EventArgs e)
  {
     lblStr.Text = ViewState["name"].ToString( );
```

```
}
protected void Button1_Click(object sender, EventArgs e)
{
  TextBox1.Text = "";
  for (int i = 0; i < ListBox1.Items.Count; i++)
  {
    if (ListBox1.Items[i].Selected == true)
    {
      TextBox1.Text = TextBox1.Text + "" + ListBox1.Items[i].Text + "\n";
    }
  }
}
protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
{
  Label1.Text = DropDownList1.SelectedItem.Text;
}
protected void DropDownList2_SelectedIndexChanged(object sender, EventArgs e)
{
  Label1.Font.Size = int.Parse(DropDownList2.SelectedItem.Text);
}
protected void RadioButton1_CheckedChanged(object sender, EventArgs e)
{
  Label1.BackColor = System.Drawing.Color.Red;
}
protected void RadioButton2_CheckedChanged(object sender, EventArgs e)
{
  Label1.BackColor = System.Drawing.Color.Green;
}
protected void RadioButton3_CheckedChanged(object sender, EventArgs e)
{
  Label1.BackColor = System.Drawing.Color.Blue;
}
protected void CheckBox1_CheckedChanged(object sender, EventArgs e)
{
  Label1.Font.Bold  = true;
}
protected void CheckBox2_CheckedChanged(object sender, EventArgs e)
{
  Label1.Font.Italic = true;
}
protected void CheckBox3_CheckedChanged(object sender, EventArgs e)
{
  Label1.Font.Underline = true;
}
}
```

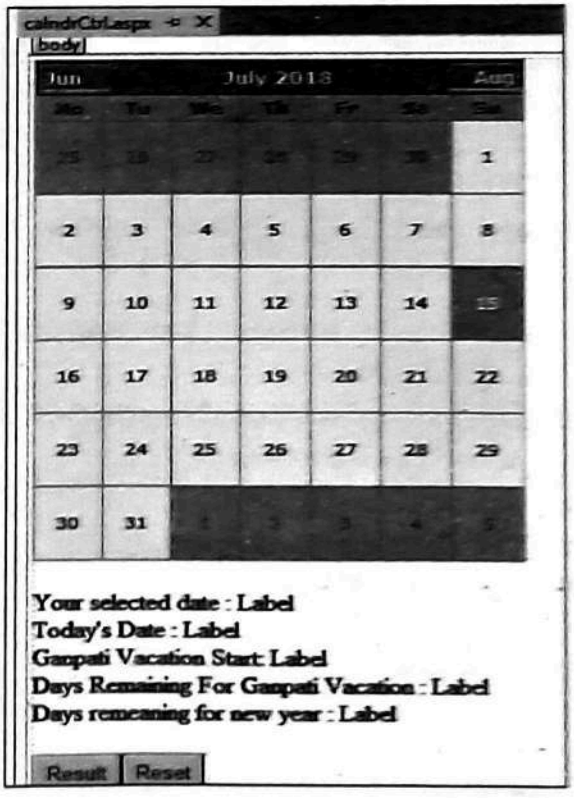
OUTPUT



Practical 3(b) : Demonstrate the use of Calendar control to perform following operations.

a) **Display messages in a calendar control**
b) **Display vacation in a calendar control**
c) **Selected day in a calendar control using style**
d) **Difference between two calendar dates**

calndrCtrl.aspx



Calender properties set for this example :

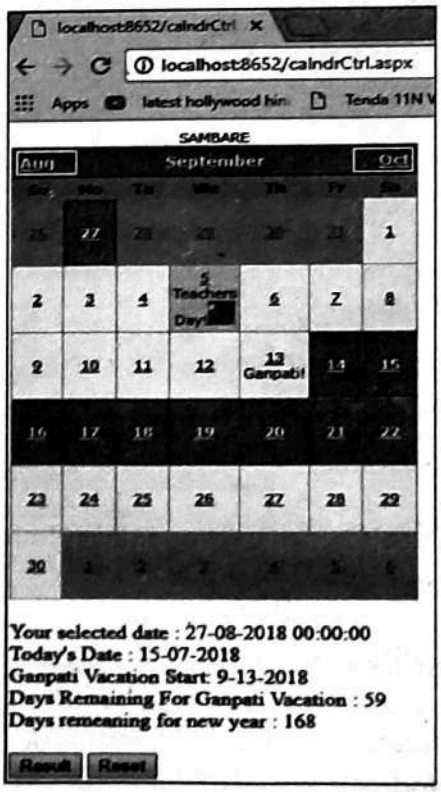
```
<asp:Calendar ID="Calendar1" runat="server" BackColor="#FFFFCC"
BorderColor="#FFCC66" BorderWidth="1px" DayNameFormat="Shortest" Font-
Names="Verdana" Font-Size="8pt" ForeColor="#663399" Height="200px"
NextPrevFormat="ShortMonth" OnDayRender="Calendar1_DayRender"
ShowGridLines="True" Width="300px">
        <DayHeaderStyle BackColor="#FFCC66" Font-Bold="True" Height="1px" />
        <NextPrevStyle BorderStyle="Solid" BorderWidth="2px" Font-Size="9pt"
ForeColor="#FFFFCC" />
        <OtherMonthDayStyle BackColor="#FFCC99" BorderStyle="Solid"
ForeColor="#CC9966" />
        <SelectedDayStyle BackColor="Red" Font-Bold="True" />
        <SelectorStyle BackColor="#FFCC66" />
        <TitleStyle BackColor="#990000" Font-Bold="True" Font-Size="9pt"
ForeColor="#FFFFCC" />
        <TodayDayStyle BackColor="#FFCC66" ForeColor="White" />
        <WeekendDayStyle Height="50px" />
      </asp:Calendar>
```

calndrCtrl.aspx.cs

```
public partial class _Default : System.Web.UI.Page
{
protected void btnResult_Click(object sender, EventArgs e)
   {
      Calendar1.Caption = "SAMBARE";
      Calendar1.FirstDayOfWeek = FirstDayOfWeek.Sunday;
      Calendar1.NextPrevFormat = NextPrevFormat.ShortMonth;
      Calendar1.TitleFormat = TitleFormat.Month;

      Label1.Text = Calendar1.SelectedDate.Date.ToString( );
      Label2.Text = Calendar1.TodaysDate.ToShortDateString( );
      Label3.Text = "9-13-2018";
      TimeSpan d = new DateTime(2018, 9, 13)- DateTime.Now;
      Label4.Text = d.Days.ToString( );
      TimeSpan d1 = new DateTime(2018, 12, 31)-DateTime.Now;
      Label5.Text = d1.Days.ToString( );

      if (Calendar1.SelectedDate.ToShortDateString( ) == "9-13-2018")
         Label3.Text = "<b>Ganpati Festival Start</b>";
      if (Calendar1.SelectedDate.ToShortDateString( ) == "9-23-2018")
         Label3.Text = "<b>Ganpati Festival End</b>";
   }
   protected void Calendar1_DayRender(object sender,
System.Web.UI.WebControls.DayRenderEventArgs e)
   {
      if (e.Day.Date.Day == 5 && e.Day.Date.Month == 9)
      {
         e.Cell.BackColor = System.Drawing.Color.Yellow;
         Label lbl = new Label( );
```

```
        lbl.Text = "<br>Teachers Day!";
        e.Cell.Controls.Add(lbl);
        Image g1 = new Image( );
        g1.ImageUrl = "td.jpg";
        g1.Height = 20;
        g1.Width = 20;
        e.Cell.Controls.Add(g1);
      }
      if (e.Day.Date.Day == 13 && e.Day.Date.Month == 9)
      {
        Calendar1.SelectedDate = new DateTime(2018, 9, 12);
        Calendar1.SelectedDates.SelectRange(Calendar1.SelectedDate,
Calendar1.SelectedDate.AddDays(10));
        Label lbl1 = new Label( );
        lbl1.Text = "<br>Ganpati!";
        e.Cell.Controls.Add(lbl1);
      }
    }
    protected void btnReset_Click(object sender, EventArgs e)
    {
      Label1.Text = "";
      Label2.Text = "";
      Label3.Text = "";
      Label4.Text = "";
      Label5.Text = "";
      Calendar1.SelectedDates.Clear( );
    }
}
```

OUTPUT



Practical 3(c) : Demonstrate the use of Treeview control perform following operations :

a) Treeview control and datalist

b) Treeview operations

Add XML File

Website -> Add -> XML File and Name it 'stdetail'.

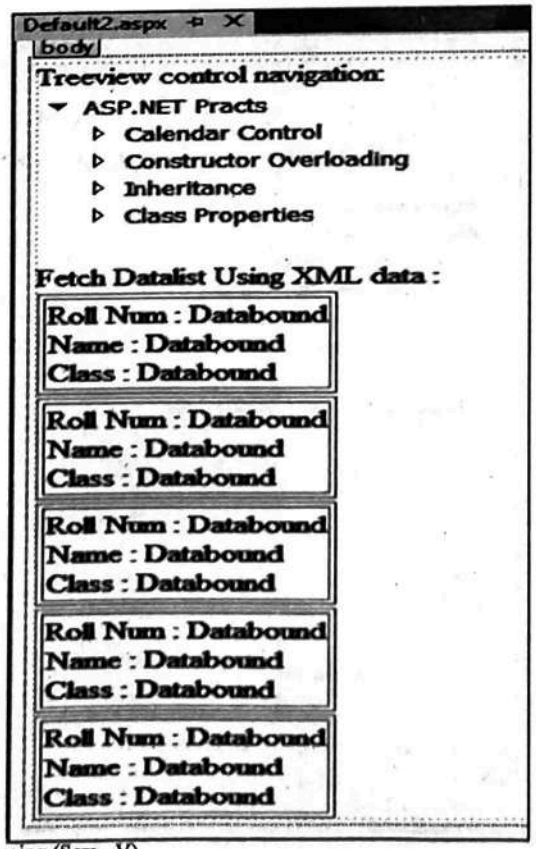
stdetail.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<studentdetail>
 <student>
  <sid>1</sid>
  <sname>Tushar</sname>
  <sclass>TYIT</sclass>
 </student>
 <student>
  <sid>2</sid>
  <sname>Sonali</sname>
  <sclass>TYCS</sclass>
 </student>
 <student>
  <sid>3</sid>
  <sname>Yashashree</sname>
  <sclass>TYIT</sclass>
 </student>
 <student>
  <sid>4</sid>
  <sname>Vedshree</sname>
  <sclass>TYCS</sclass>
 </student>
</studentdetail>
```

Default2.aspx

```
<form id="form1" runat="server">
  <div>
    Treeview control navigation:<asp:TreeView ID = "TreeView1" runat = "server" Width
= "150px" ImageSet="Arrows">
    <HoverNodeStyle Font-Underline="True" ForeColor="#5555DD" />
<Nodes>
<asp:TreeNode Text = "ASP.NET Practs" Value = "New Node">
<asp:TreeNode Text = "Calendar Control" Value = "RED"
NavigateUrl="~/calndrCtrl.aspx"> </asp:TreeNode>
<asp:TreeNode Text = "Constructor Overloading" Value = "GREEN"
NavigateUrl="~/clsconstrc.aspx"> </asp:TreeNode>
  <asp:TreeNode NavigateUrl="~/singleInh.aspx" Text="Inheritance"
Value="BLUE"></asp:TreeNode>
```

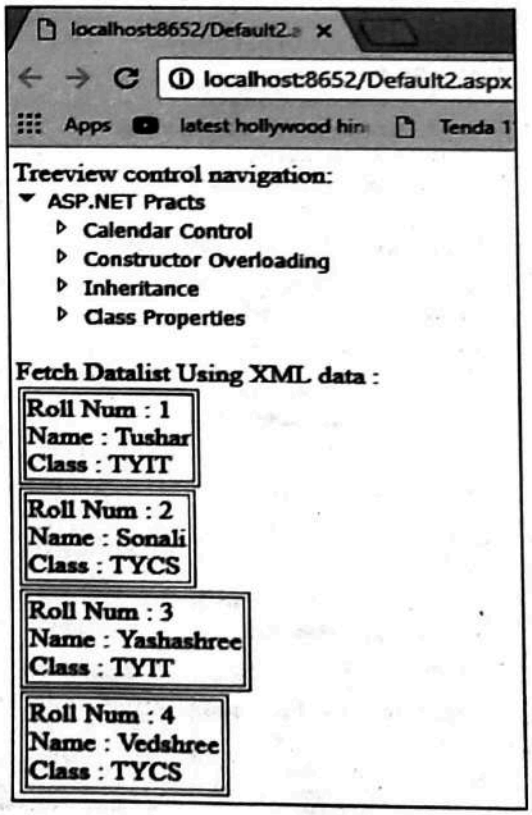
```
  <asp:TreeNode NavigateUrl="~/clsProp.aspx" Text="Class Properties" Value="Class
Properties"></asp:TreeNode>
</asp:TreeNode>
</Nodes>
    <NodeStyle Font-Names="Tahoma" Font-Size="10pt" ForeColor="Black"
HorizontalPadding="5px" NodeSpacing="0px" VerticalPadding="0px" />
    <ParentNodeStyle Font-Bold="False" />
    <SelectedNodeStyle Font-Underline="True" ForeColor="#5555DD"
HorizontalPadding="0px" VerticalPadding="0px" />
</asp:TreeView>
    <br />
    Fetch Datalist Using XML data : </div>
<asp:DataList ID="DataList1" runat="server">
      <ItemTemplate>
    <table class = "table" border="1">
       <tr>
        <td>Roll Num : <%# Eval("sid") %><br />
          Name : <%# Eval("sname") %><br />
          Class : <%# Eval("sclass")%>
        </td>
       </tr>
    </table>
  </ItemTemplate>
</asp:DataList>
```



Default2.aspx.cs

```
using System.Data;
public partial class _Default : System.Web.UI.Page
{
  protected void Page_Load(object sender, EventArgs e)
  {
    if (!IsPostBack)
    {
      BindData( );
    }
  }
  protected void BindData( )
  {
    DataSet ds = new DataSet( );
    ds.ReadXml(Server.MapPath("stdetail.xml"));
    if (ds != null && ds.HasChanges( ))
    {
        DataList1.DataSource = ds;
        DataList1.DataBind( );
    }
    else
    {
       DataList1.DataBind( );
    }
  }
}
```

OUTPUT



Practical 4 : Working with Form Controls

Practical 4(a) : Create a Registration form to demonstrate use of various Validation controls.

ValidationPract.aspx

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

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
  <script runat="server">
    void ValidateBtn_OnClick(object sender, EventArgs e)
    {
      if (Page.IsValid)
      {
        lbl1.Text = "Thank You!";
      }
      else
      {
        lbl1.Text = "The text must be exactly 8 characters long!";
      }
    }
    void ServerValidation (object source, ServerValidateEventArgs e)
    {

      if (e.Value.Length == 8)
        e.IsValid = true;
      else
        e.IsValid = false;

    }
  </script>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  Your name:<br />
<asp:TextBox runat="server" id="txtName" />
<asp:RequiredFieldValidator runat="server" id="reqName" ForeColor="Red" controltovalidate="txtName" errormessage="Please enter your name!" />
<br /><br />
```

```
Enter age:<br />
<asp:TextBox runat="server" id="txt1" />
<asp:RangeValidator ID="RangeValidator2" Type="Integer" runat="server"
ForeColor="Red" ControlToValidate="txt1" MinimumValue="18" MaximumValue="100"
ErrorMessage="Not valid age">Not valid age</asp:RangeValidator>
<br /><br />

Password:<br />
<asp:TextBox runat="server" id="txt11" TextMode="Password" /><br />
ReEnter Password:<br />
<asp:TextBox runat="server" id="txt12" TextMode="Password" /><br />
<asp:CompareValidator runat="server" id="cmpNumbers" ForeColor="Red"
controltovalidate="txt12" controltocompare="txt11" operator="LessThan" type="Integer"
errormessage="Password should match!" >Password should
match!</asp:CompareValidator><br />
<br />

Email ID:<br />
<asp:TextBox runat="server" id="txtNumber" />
<asp:RegularExpressionValidator runat="server" ForeColor="Red" id="rexNumber"
controltovalidate="txtNumber" errormessage="Please enter valid email address!"
ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*" >Please enter
valid email address!</asp:RegularExpressionValidator>
<br />

Custom text:<br />
<asp:TextBox runat="server" id="txtCustom" />
<asp:CustomValidator id="CustomValidator1" ControlToValidate="txtCustom"
ClientValidationFunction="ServerValidation" Display="Static" ForeColor="Red"
runat="server"/>
<asp:Label id="lbl1" runat="server" Font-Name="Verdana" Font-Size="10pt" Font-
Names="Verdana" ForeColor="Red" /><br />
<asp:Button id="Button1" Text="Validate" OnClick="ValidateBtn_OnClick"
runat="server"/>

<asp:ValidationSummary ID="ValidationSummary1" runat="server" Style="top: 452px;
left: 24px; position: absolute; height: 38px; width: 625px; right: 487px;" />
</div>
</form>
</body>
</html>
```

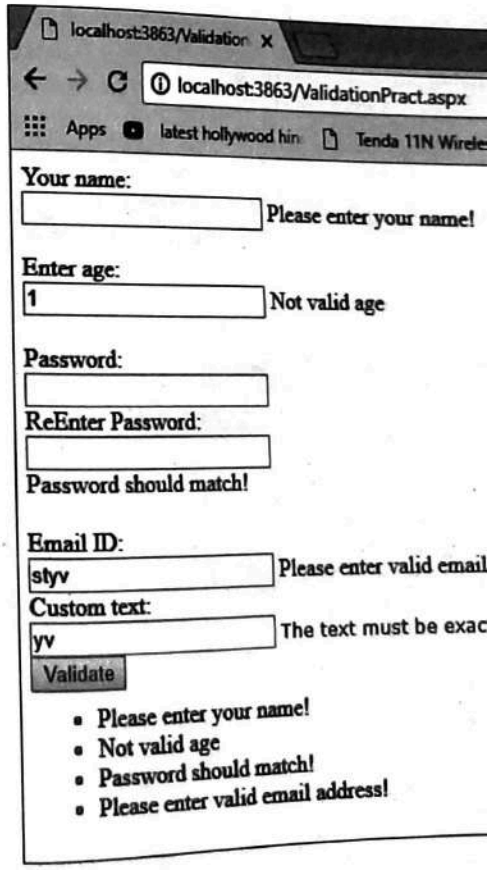
Web.config

```
<configuration>
 <appSettings>
```



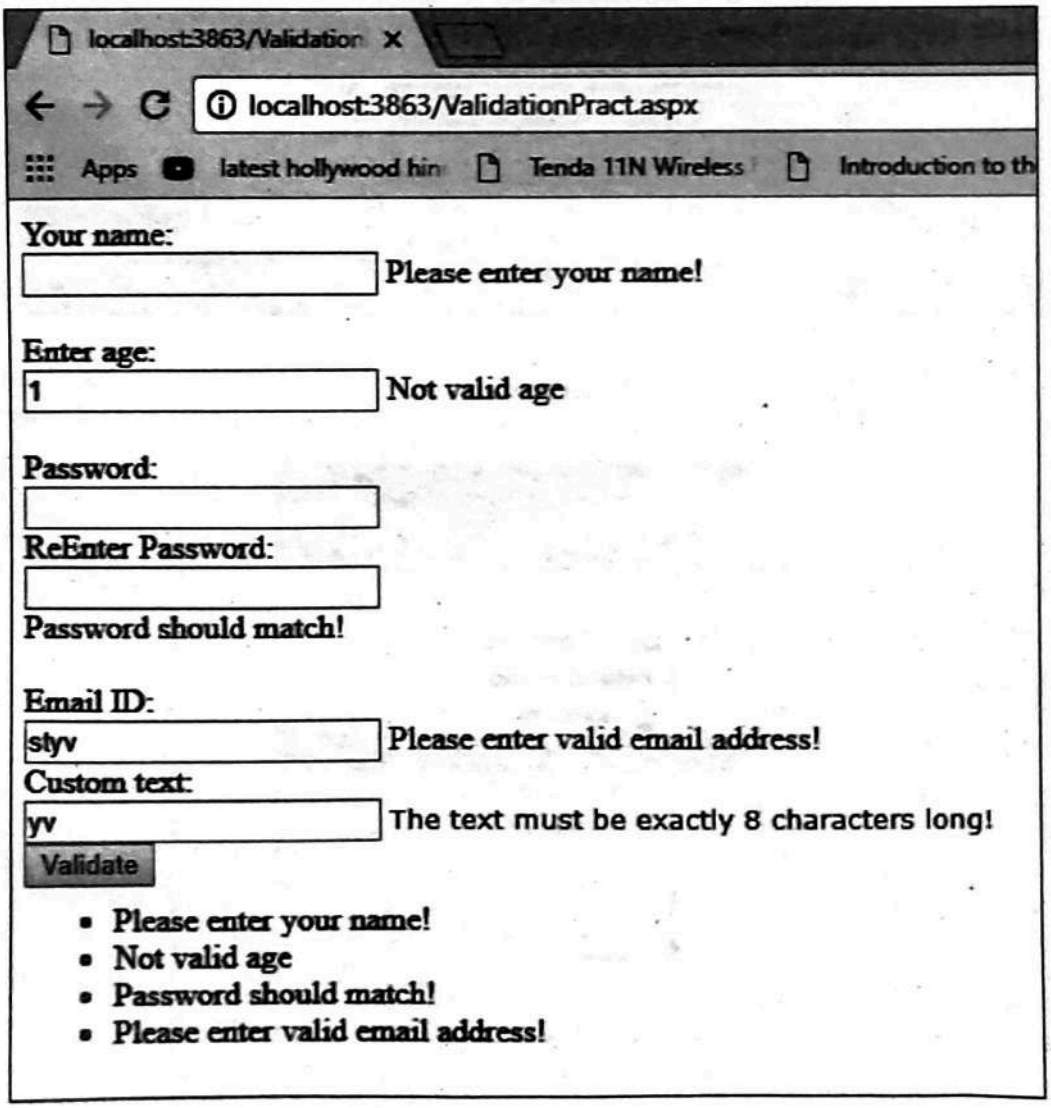
```
  <add key="ValidationSettings:UnobtrusiveValidationMode
 </appSettings>
 <system.web>
  <compilation debug="true" targetFramework="4.5.2" />
  <httpRuntime targetFramework="4.5.2" />
 </system.web>

</configuration>
```

OUTPUT



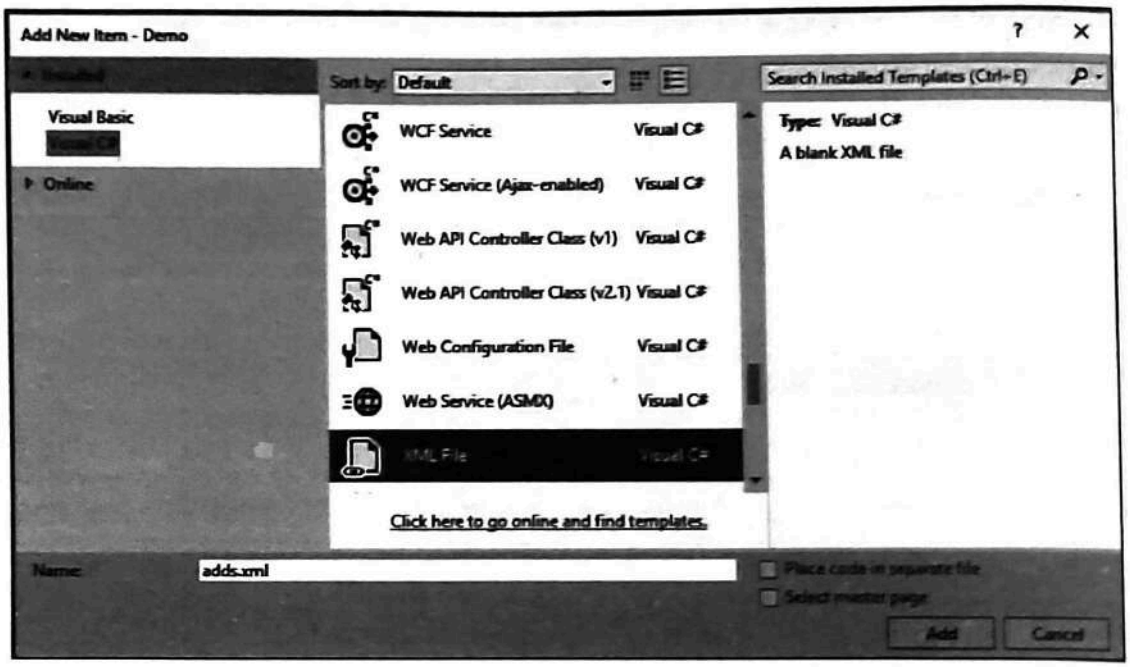
```
    <add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />
  </appSettings>
  <system.web>
    <compilation debug="true" targetFramework="4.5.2" />
    <httpRuntime targetFramework="4.5.2" />
  </system.web>

</configuration>
```

OUTPUT

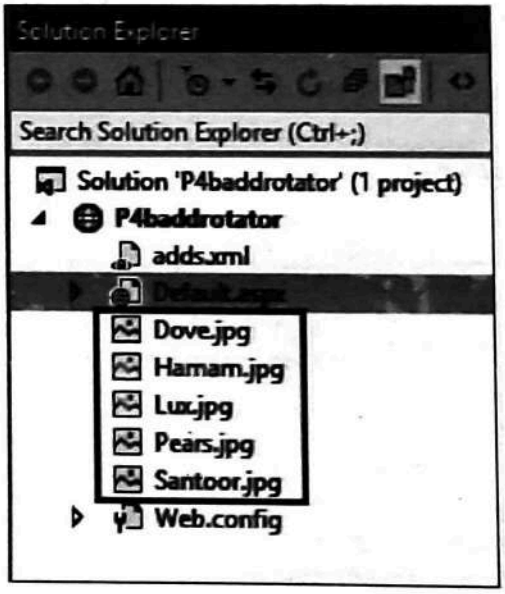


Practical 4 (b) : Create Web Form to demonstrate use of Adrotator Control.

Add a XML file, name it "adds.xml"



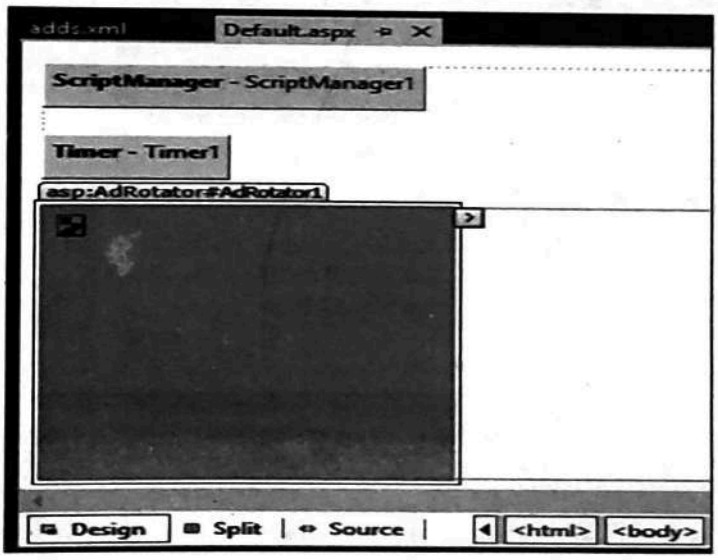
Add images to test out the adrotator functionality.



Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs
Inherits="_Default" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
```

```
<body>
  <form id="form1" runat="server">
  <div>
  <asp:ScriptManager ID="ScriptManager1" runat="server">
</asp:ScriptManager>
<br />
<asp:Timer ID="Timer1" interval="2000" runat="server">
</asp:Timer>
<br />
<asp:UpdatePanel ID="UpdatePanel1" runat="server">
<Triggers>
<asp:AsyncPostBackTrigger ControlID="Timer1" EventName="Tick" />
</Triggers>
<ContentTemplate>
<asp:AdRotator ID="AdRotator1" runat="server" AdvertisementFile="~/adds.xml"
Height="200px" Width="200px" />
</ContentTemplate>
</asp:UpdatePanel>
  </div>
  </form>
</body>
</html>
```

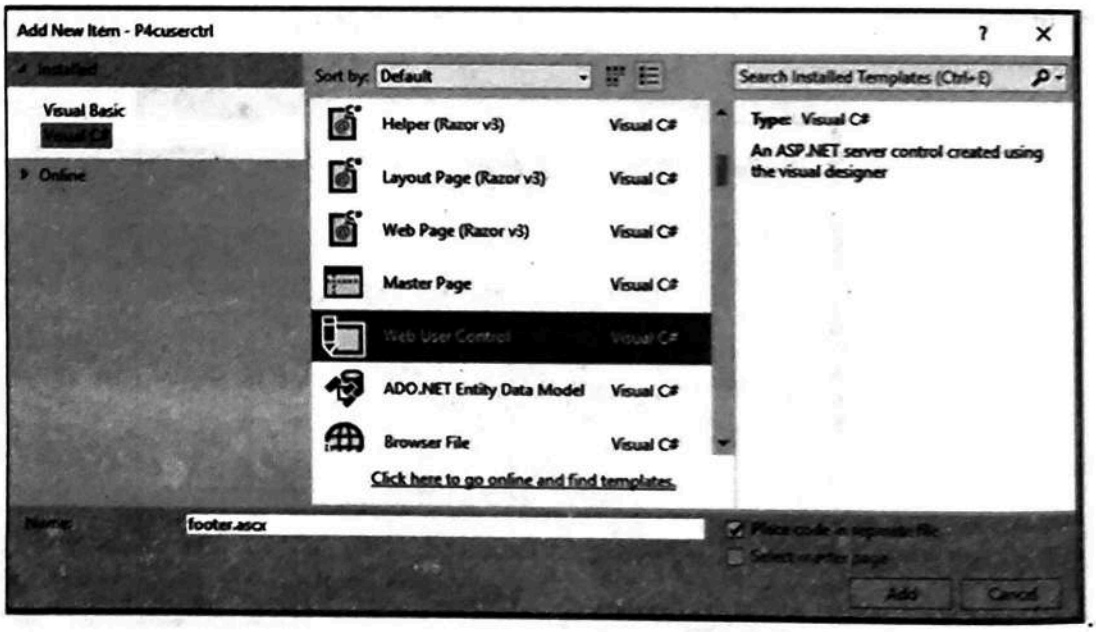


OUTPUT

Practical 4 (c) : Create Web Form to demonstrate use User Controls.

Add Web User Control

Website -> Add -> Web User Control and Name it 'footer'.



footer.ascx

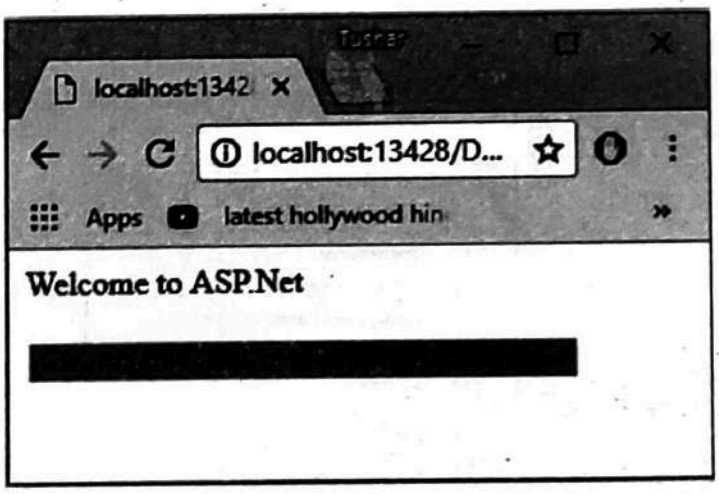
```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="footer.ascx.cs" Inherits="footer" %>
<table>
  <tr>
    <td align="center" style="font-family: Cambria; background-color: #00CCFF; font-size: 14px; text-decoration: blink; color: #FF0000; font-weight: bold;">
Advanced Web programming by Sambare</td>
```

```
  </tr>
</table>
```

Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"
Inherits="_Default" %>
<%@ Register Src="~/footer.ascx" TagName="footer" TagPrefix="STfooter" %>
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
  <asp:Label ID="Label1" runat="server" Text="Welcome to ASP.Net ">
</asp:Label><br /> <br />
    </div>
    <STfooter:footer ID="footer1" runat="server" />
  </form>
</body>
</html>
```

OUTPUT :

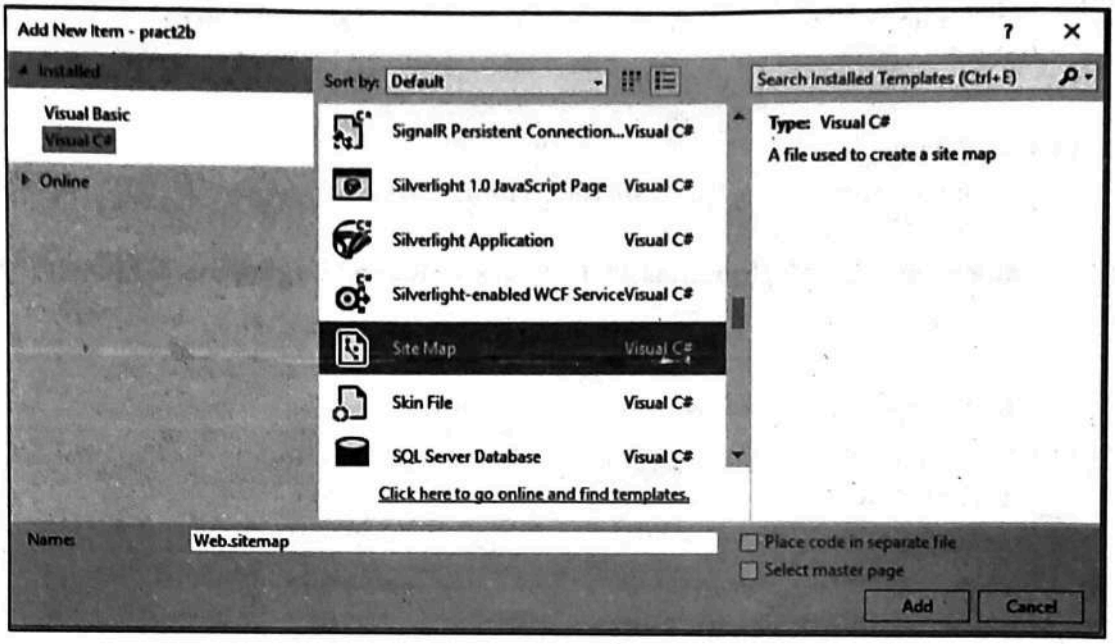


Practical 5 : Working with Navigation, Beautification and Master page.

Practical 5 (a) : Create Web Form to demonstrate use of Website Navigation controls and Site Map.

Add Site Map File

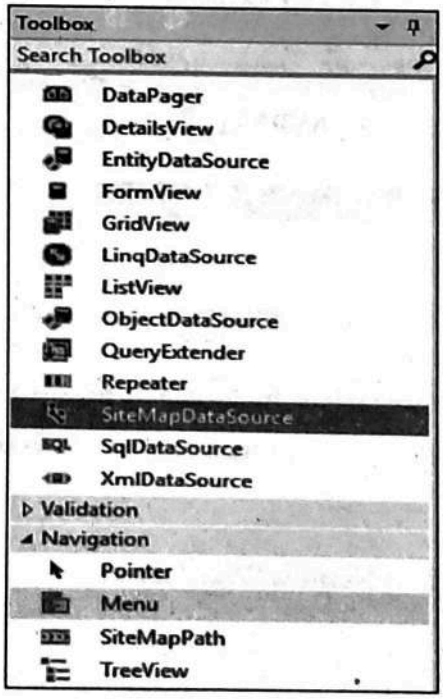
Website -> Add -> Site Map and Name it 'Web.Sitemap'.



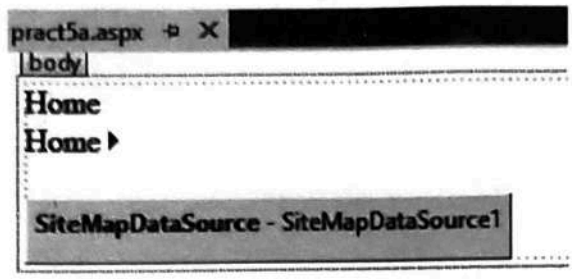
Web.Sitemap

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="pract5a.aspx"  title="Home"  description="Home page of our web site">
    <siteMapNode url="clsProp.aspx" title="Page2"  description="Page2" />
    <siteMapNode url="Default3.aspx" title="Page3"  description="Page3" />
    <siteMapNode url="Default2.aspx" title="Page4"  description="Page4" />
  </siteMapNode>
</siteMap>
```

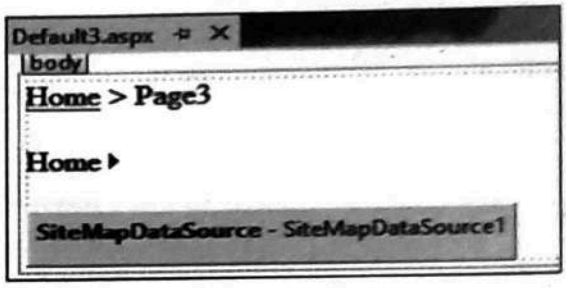
Now, Drag and Drop a SiteMapDataSource control and Menu control from Toolbox to the page.



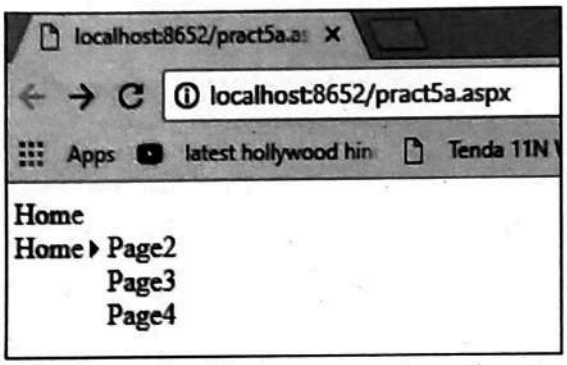
Pract5a.aspx

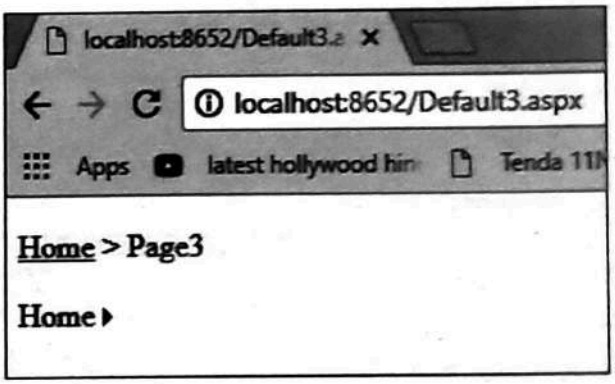


Default3.aspx



OUTPUT





Practical 5 (b) : Create a web application to demonstrate use of Master Page with applying Styles and Themes for page beautification.

SkinFile1.skin

```
<asp:Label runat="server" ForeColor="red" Font-Size="14pt" Font-Names="Verdana" />
<asp:button runat="server" Borderstyle="Solid" Borderwidth="2px" Bordercolor="Blue" Backcolor="Magenta"/>
```

demostyle.css

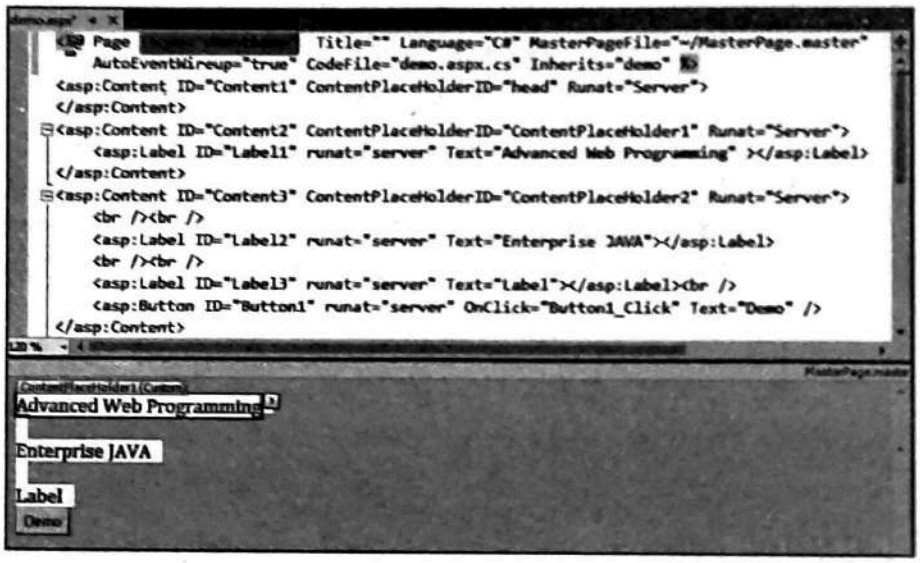
```
body
{
  background-color:yellow ;
  font-family:Cambria;
  font-size:18px;
}
```

MasterPage.master

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs" Inherits="MasterPage" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
  <asp:ContentPlaceHolder id="head" runat="server">
  </asp:ContentPlaceHolder>
```

```
  <link href = "demostyle.css" rel = "stylesheet" type = "text/css" />
</head>
<body>
  <form id="form1" runat="server">
  <div>
    <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
        </asp:ContentPlaceHolder>
    <asp:ContentPlaceHolder id="ContentPlaceHolder2" runat="server">
        </asp:ContentPlaceHolder>
  </div>
  </form>
</body>
</html>
```

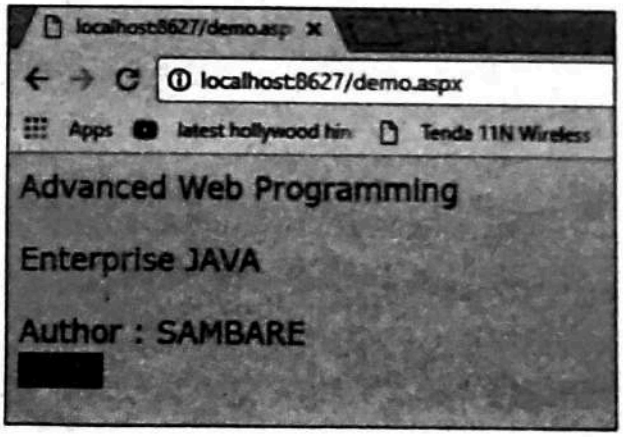
demo.aspx.



demo.aspx.cs

```
protected void Button1_Click(object sender, EventArgs e)
  {
    Label3.Text = "Author : SAMBARE";
  }
```

OUTPUT



Practical 5 (c).cs .ce a web application to demonstrate various States of ASP.NET Pages.

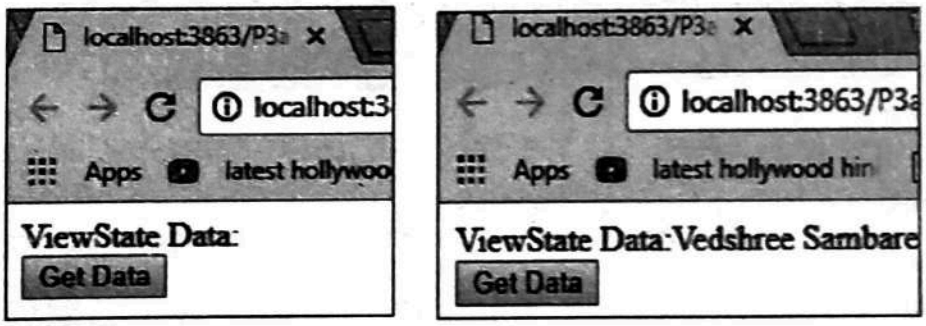
1. View State

P3a.aspx

P3a.aspx.cs

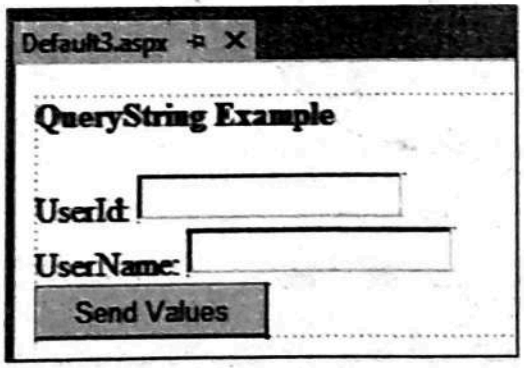
```
public partial class P3a : System.Web.UI.Page
{
  protected void Page_Load(object sender, EventArgs e)
  {
    if (!IsPostBack)
    {
      string str = "Vedshree Sambare";
      if (ViewState["nam"] == null)
      {
        ViewState["nam"] = str;
      }
    }
  }

  protected void btn_Click(object sender, EventArgs e)
  {
    lblStr.Text = ViewState["nam"].ToString( );
  }
}
```

OUTPUT



2. Query String

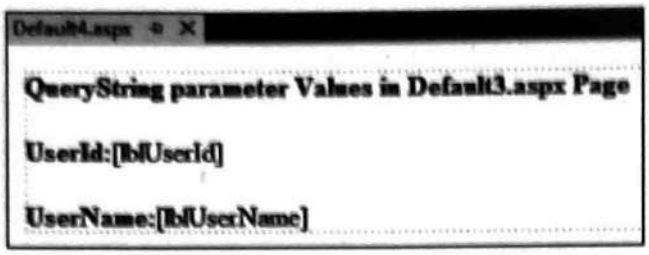
Default3.aspx



Default3.aspx.cs

```
protected void btnSend_Click(object sender, EventArgs e)
  {
    Response.Redirect("Default4.aspx?UserId=" + txtUserId.Text + "&UserName=" +
txtUserName.Text);
  }
```

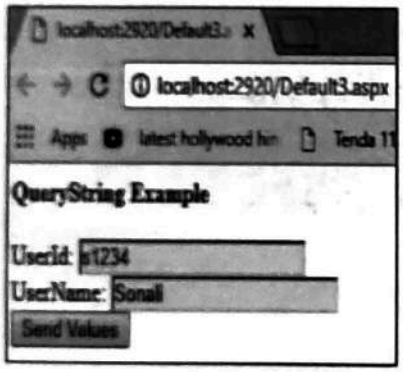
Default4.aspx

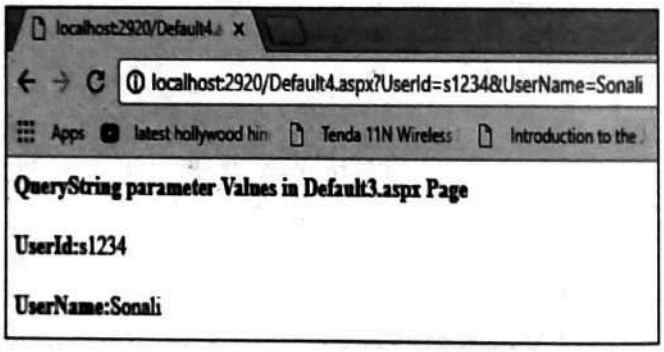


Default4.aspx.cs

```
protected void Page_Load(object sender, EventArgs e)
  {
    if (!IsPostBack)
    {
      lblUserId.Text = Request.QueryString["UserId"];
      lblUserName.Text = Request.QueryString["UserName"];
    }
  }
```

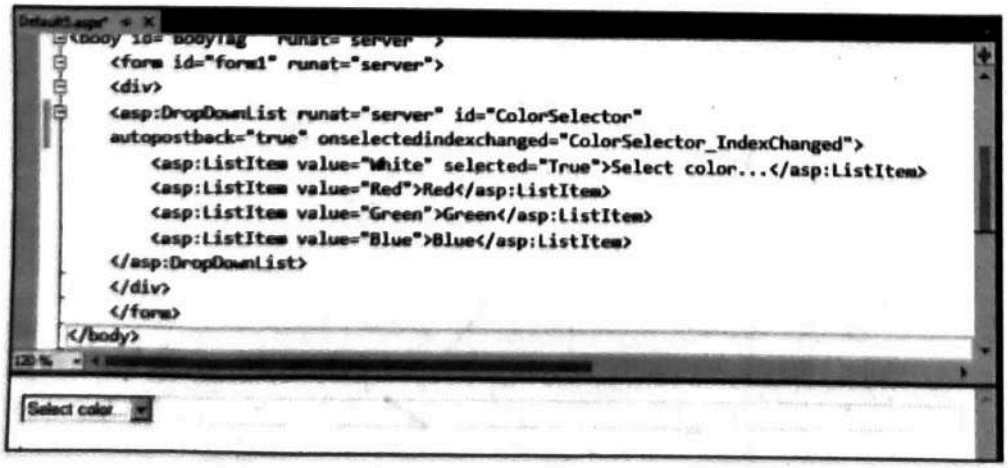
OUTPUT





3. Cookies

Default5.aspx



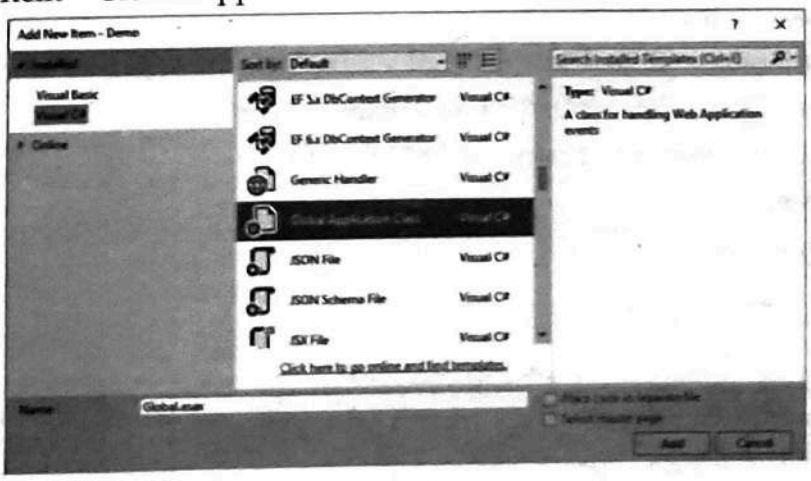
Default5.aspx.cs

```
using System.Data;
public partial class Default5 : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      if (Request.Cookies["BackgroundColor"] != null)
      {
         ColorSelector.SelectedValue = Request.Cookies["BackgroundColor"].Value;
         BodyTag.Style["background-color"] = ColorSelector.SelectedValue;
      }
   }
   protected void ColorSelector_IndexChanged(object sender, EventArgs e)
   {
      BodyTag.Style["background-color"] = ColorSelector.SelectedValue;
      HttpCookie cookie = new HttpCookie("BackgroundColor");
      cookie.Value = ColorSelector.SelectedValue;
      cookie.Expires = DateTime.Now.AddMilliseconds(20);
      Response.SetCookie(cookie);
   }
}
```

OUTPUT:

4. Session and Application State

1. Create website and add webform.
2. Adding a Global.asax to web application.
3. Add New Item > Global Application Class > Add.



Global.asax

```
<%@ Application Language="C#" %>
<script runat="server">
  void Application_Start(object sender, EventArgs e)
  {
        Application["OnlineUsers"] = 0;  //application variable
  }
  void Application_End(object sender, EventArgs e)
  {
    //  Code that runs on application shutdown
  }
  void Application_Error(object sender, EventArgs e)
  {
    // Code that runs when an unhandled error occurs
  }
  void Session_Start(object sender, EventArgs e)
  {
    Application.Lock( );
    Application["OnlineUsers"] = (int)Application["OnlineUsers"] + 1;
    Application.UnLock( );
  }
  void Session_End(object sender, EventArgs e)
  {
    Application.Lock( );
    Application["OnlineUsers"] = (int)Application["OnlineUsers"] - 1;
    Application.UnLock( );
  }
</script>
```

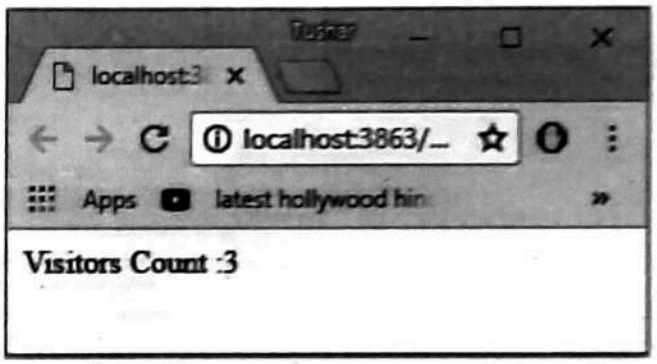
Web.config

```
<?xml version="1.0"?>
<configuration>
  <system.web>
   <sessionState mode="InProc" cookieless="false" timeout="1"/>
   <compilation debug="true" targetFramework="4.5.2" />
   <httpRuntime targetFramework="4.5.2" />
  </system.web>
</configuration>
```

Default.aspx

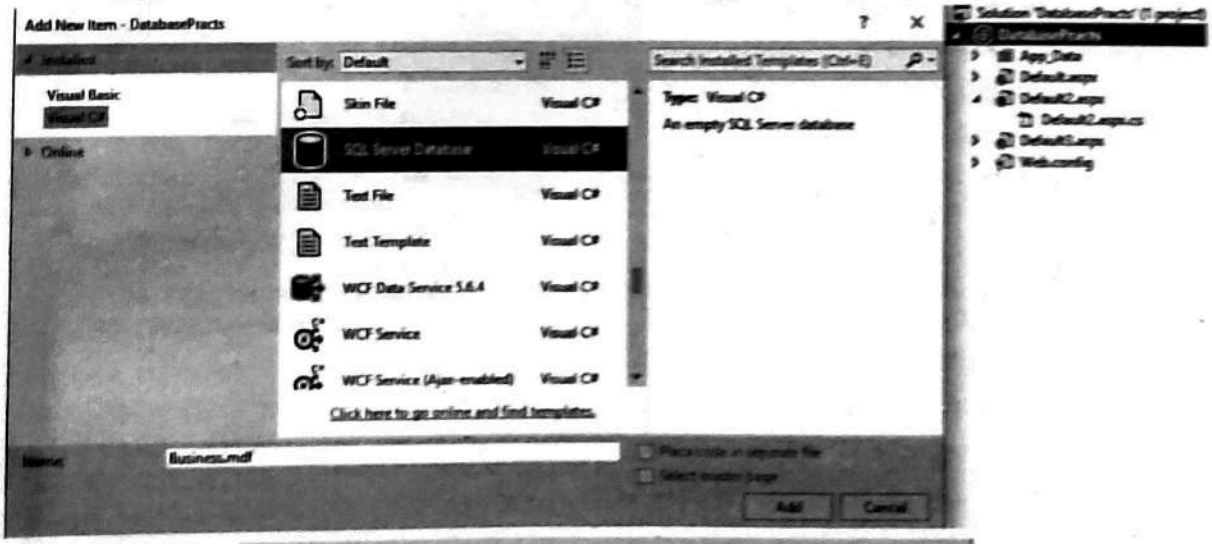
```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"
Inherits="_Default" %>
<!DOCTYPE html>
```

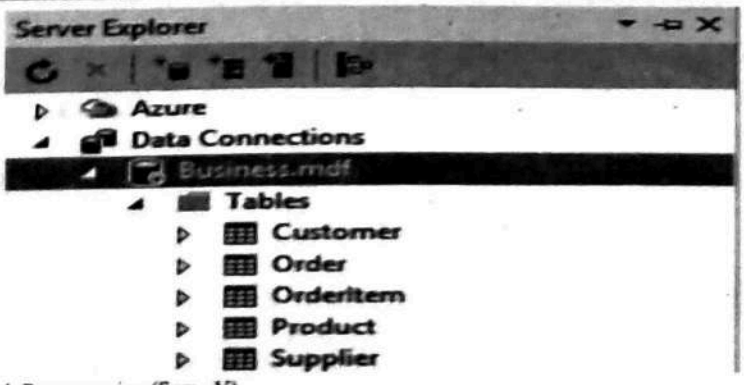
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  Visitors Count :<%=Application["OnlineUsers"].ToString( ) %>
  </div>
    </form>
</body>
</html>
```



Note : For Database practical's we have used SQL Server 2014 version.

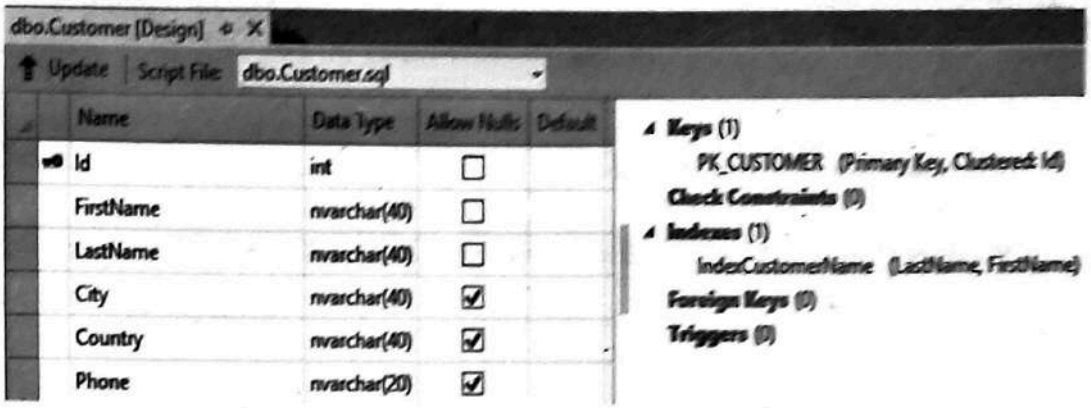
Here we to add new database in our website, as shown below. Add this database inside App_Data folder.



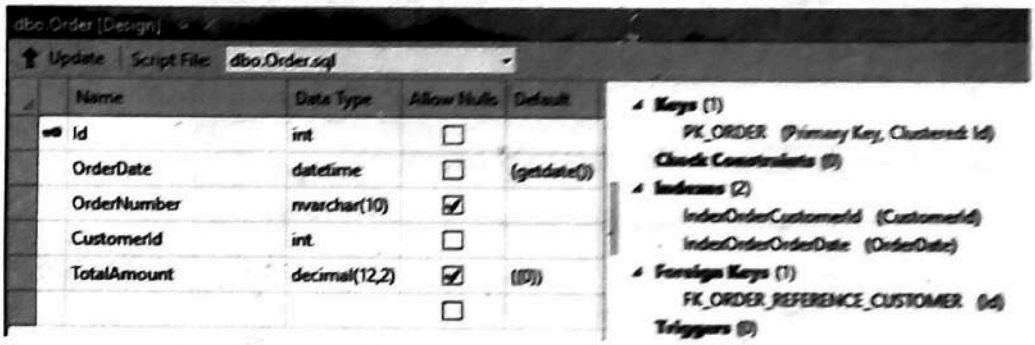


The structure of the tables are displayed below :

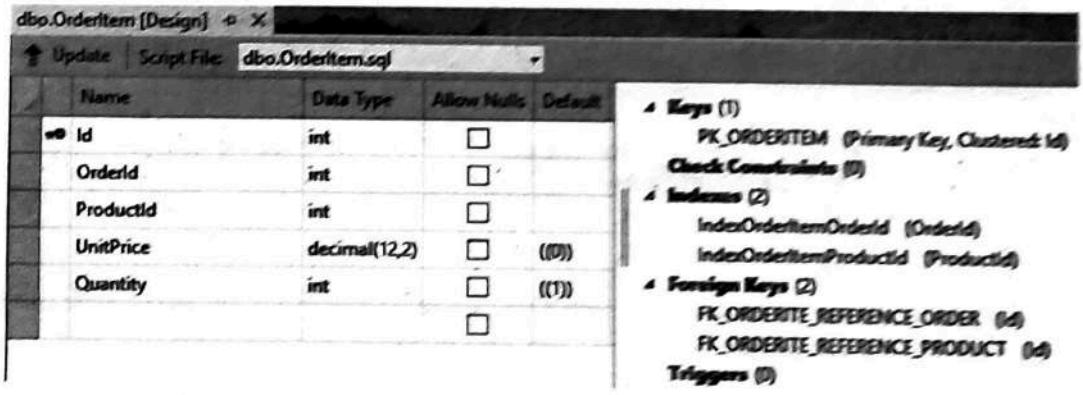
Customer Table



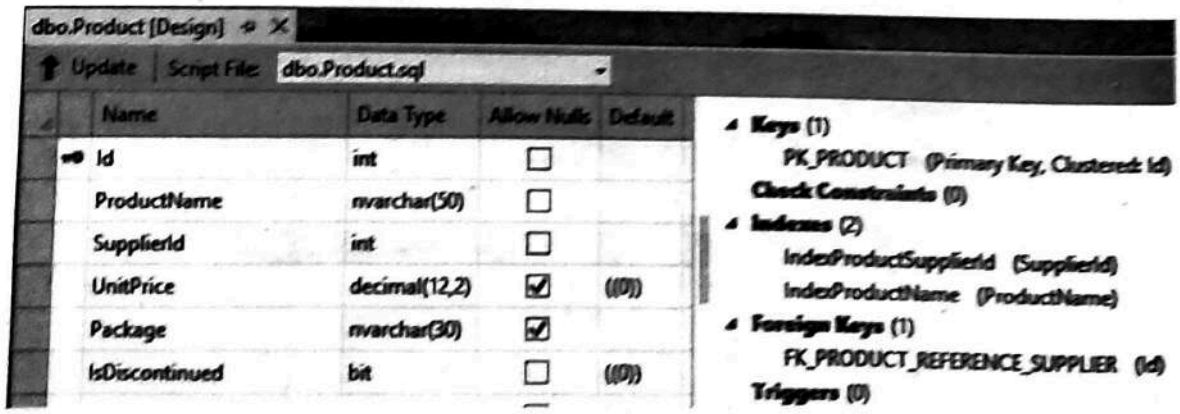
Order Table



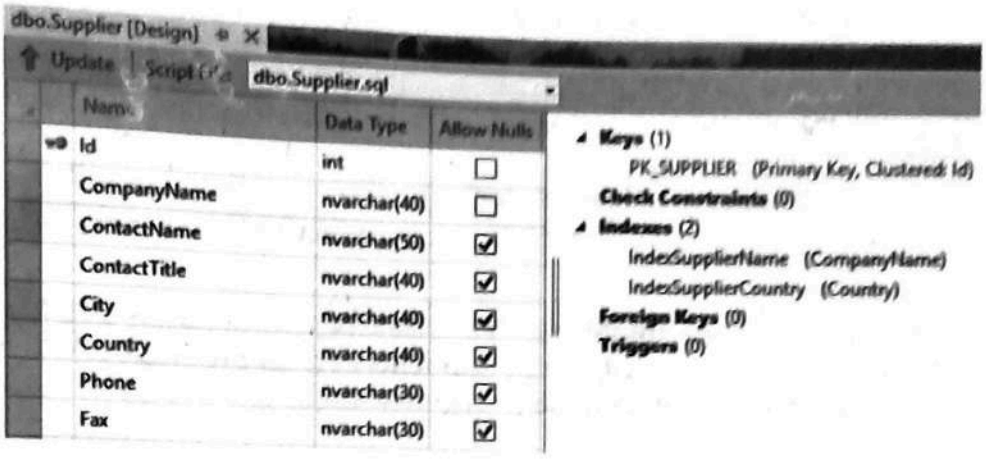
OrderItem Table



Product Table



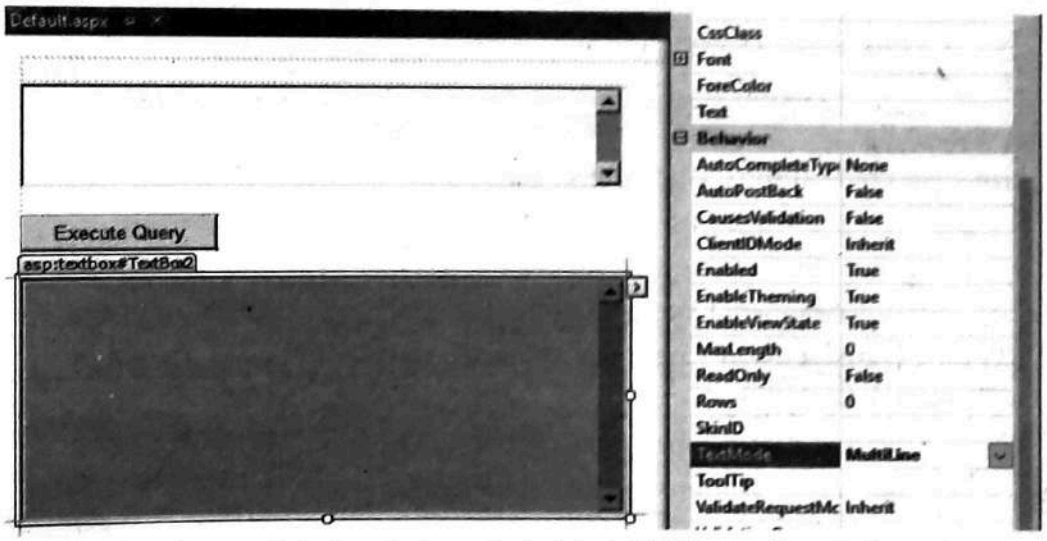
Supplier Table



Practical 6

Practical 6 (a) : Create a web application to bind data in a multiline textbox by querying in another textbox.

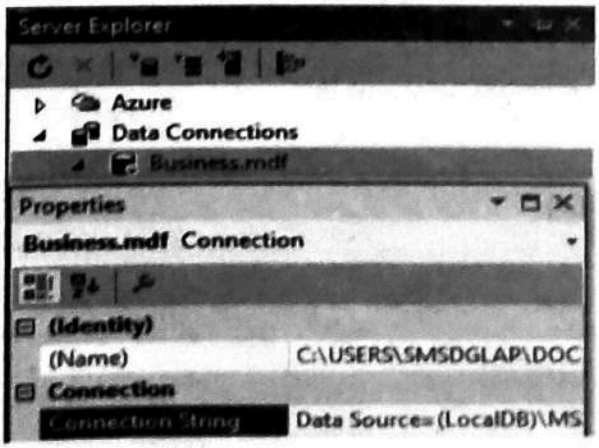
1. Create a webpage with one **Button** and two Multiline TextBox with setting **TextMode** Property of them to **Multiline** as shown below.



2. Write the Database related code in code behind C# file as given below.

Note : **The users have to use their own system connection string in place of connection string given in following code.**

The connection string is available in Server Explorer (Right click on Database Name and Select Properties) as displayed below. User can copy this connection string and can use in code.



3. Add this string to configuration file (web.config) as given below.

Web.confing

```
<?xml version="1.0"?>
<configuration>
  <system.web>     <compilation debug="true" targetFramework="4.5.2" />
   <httpRuntime targetFramework="4.5.2" />    </system.web>
 <connectionStrings>
  <add name="connStr" connectionString="Data Source=(LocalDB)\MSSQLLocalDB;
AttachDbFilename ='C:\Users\mydglap\Documents\Visual Studio 2015\WebSites\
DatabasePracts\App_Data\Business.mdf';Integrated Security=True" />
 </connectionStrings>
</configuration>
```

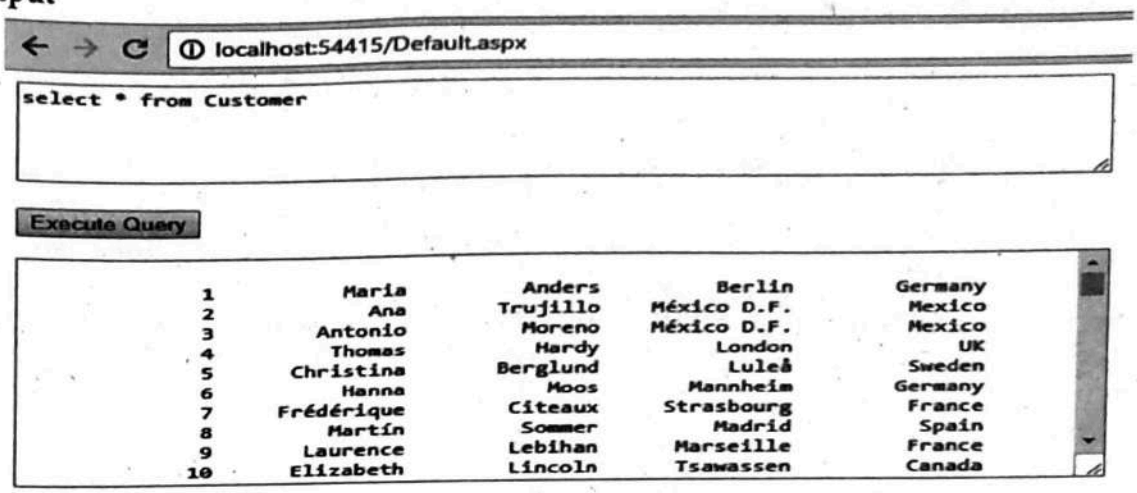
4. Now use the following code C# in Default.aspx.cs (**Note** : First write following using statements at the top of file)

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

5. After that in Button click event write the following code :

```
protected void Button1_Click(object sender, EventArgs e)
  {
    string connStr =
ConfigurationManager.ConnectionStrings["connStr"].ConnectionString;
    SqlConnection con = new SqlConnection(connStr);
    con.Open( );
    SqlCommand cmd = new SqlCommand(TextBox1.Text, con);
    SqlDataReader reader = cmd.ExecuteReader( );
    TextBox2.Text = "";
     while(reader.Read( ))
     {
     //To add new blank line in the text area
     TextBox2.Text += Environment.NewLine;
     for(int i =0; i<reader.FieldCount-1;i++)
     {
         TextBox2.Text += reader[i].ToString( ).PadLeft(15);
       }
     }
    reader.Close( );
    con.Close( );
}
```

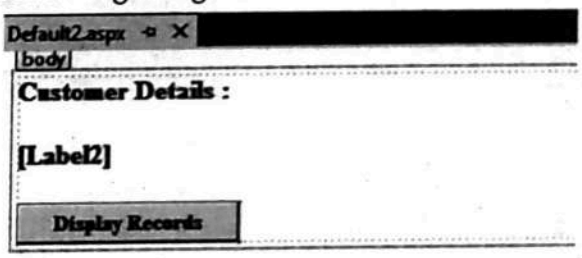
Output



Note : For all Database related practicals demonstrated in this book uses same database, tables and connection string setting in the web.config file and using Database related namespaces. Readers are requested to follow same setting as demonstrated in Practical 6 (a).

Practical 6 (b) : Create a web application to display records by using database.

Create a web page with following design :

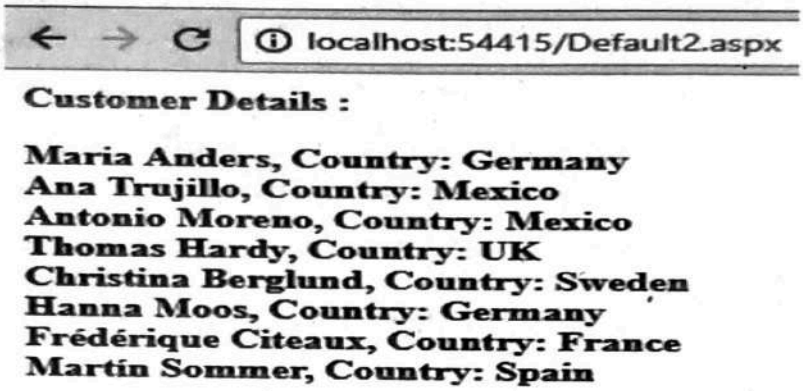


Add the following code on Button click event in C# Code behind file.

```
protected void Button1_Click(object sender, EventArgs e)
 {
   string connStr =
ConfigurationManager.ConnectionStrings["connStr"].ConnectionString;
   SqlConnection con = new SqlConnection(connStr);
   SqlCommand cmd = new SqlCommand("Select FirstName, LastName, Country
from
                                    Customer", con);
   con.Open( );
   SqlDataReader reader = cmd.ExecuteReader( );

   while (reader.Read( ))
   {

     Label2.Text += reader["FirstName"].ToString( )+" " +
reader["LastName"].ToString( ) +
                    ", Country: " + reader["Country"].ToString( ) + "<br>";
```

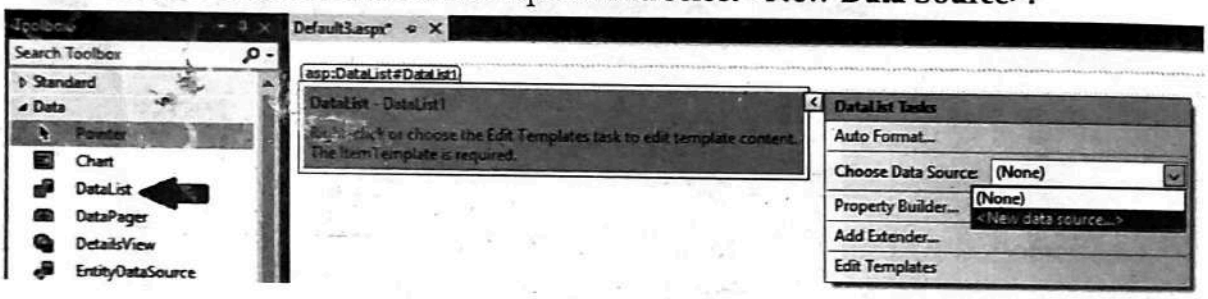
```
    }
    reader.Close( );
    con.Close( );
  }
```

Output

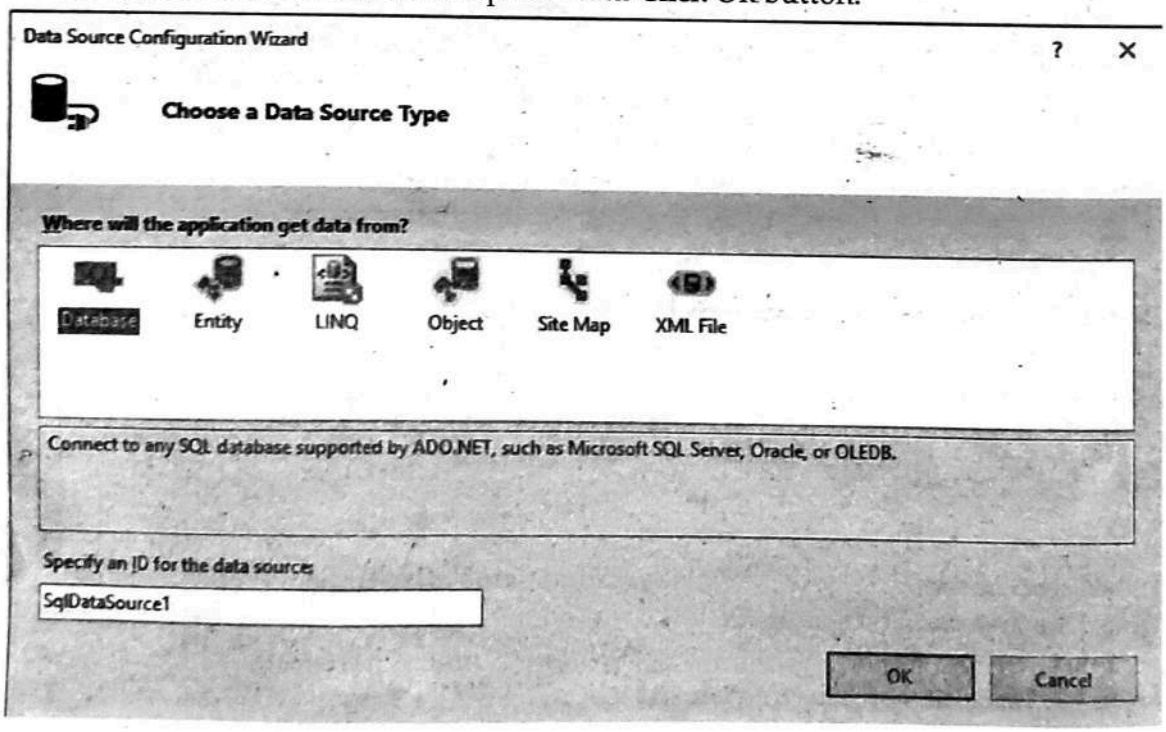


Practical 6 (c) : Demonstrate the use of Datalist link control.

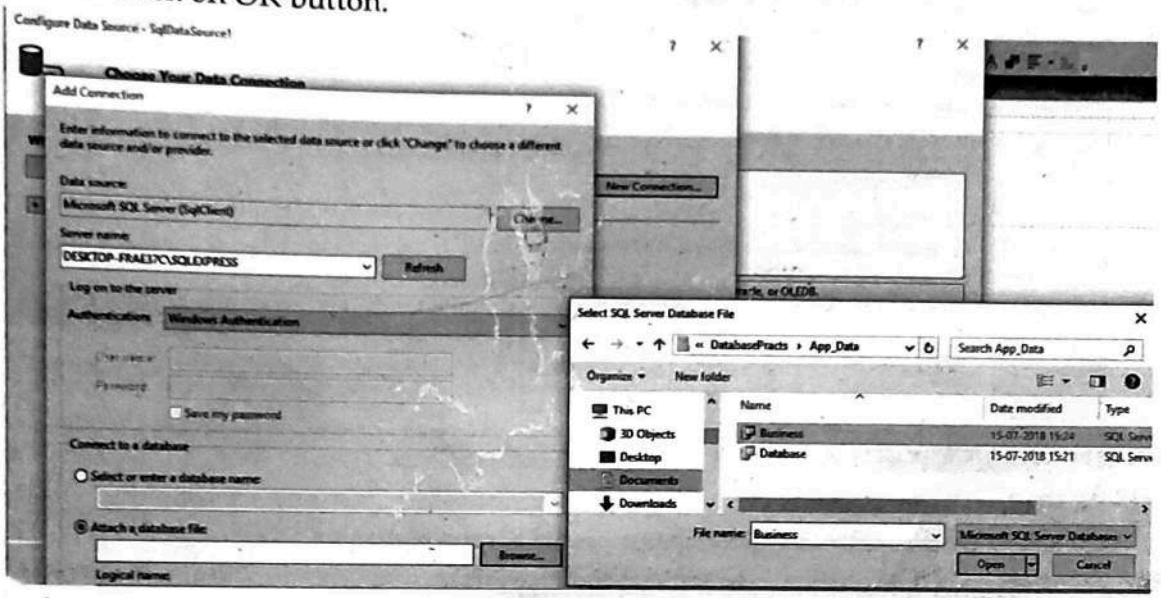
1. Drag the Datalist control to our web page form toolbox->Data-> Datalist.
2. Then select **Choose Data Source** Option and select **<New Data Source>.**

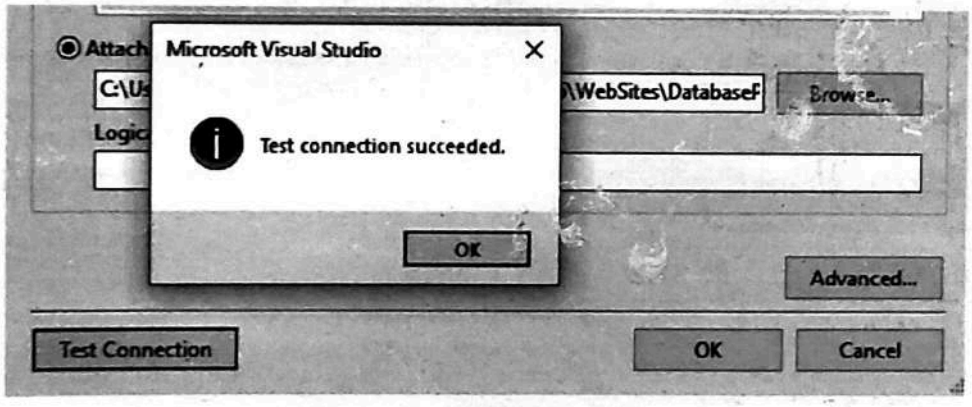


3. Now Select SQL Database from options and Click Ok button.

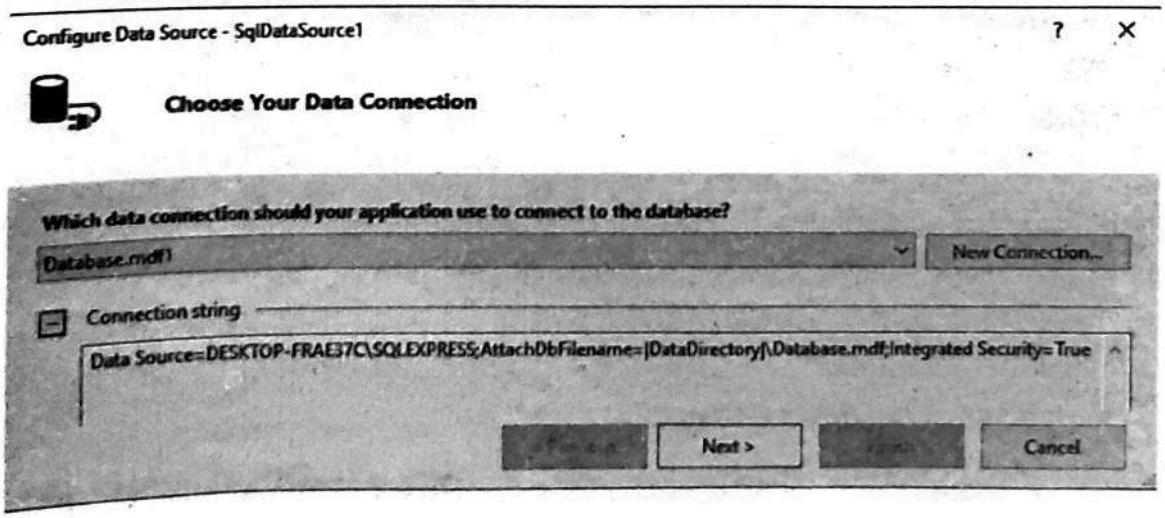


1. In next window click on New Connection button.
2. In add connection window Select the available SQL Server Name
3. Keep the Authentication as Windows Authentication.
4. After that select Attach a Database file radio button. Here we have to select the database that we have created in our application. (Usually it will be in Documents folder under Visual Studio 2015/ Websites).
5. After selection of Database file. We can also Test the connection.
6. Then Click on OK button.



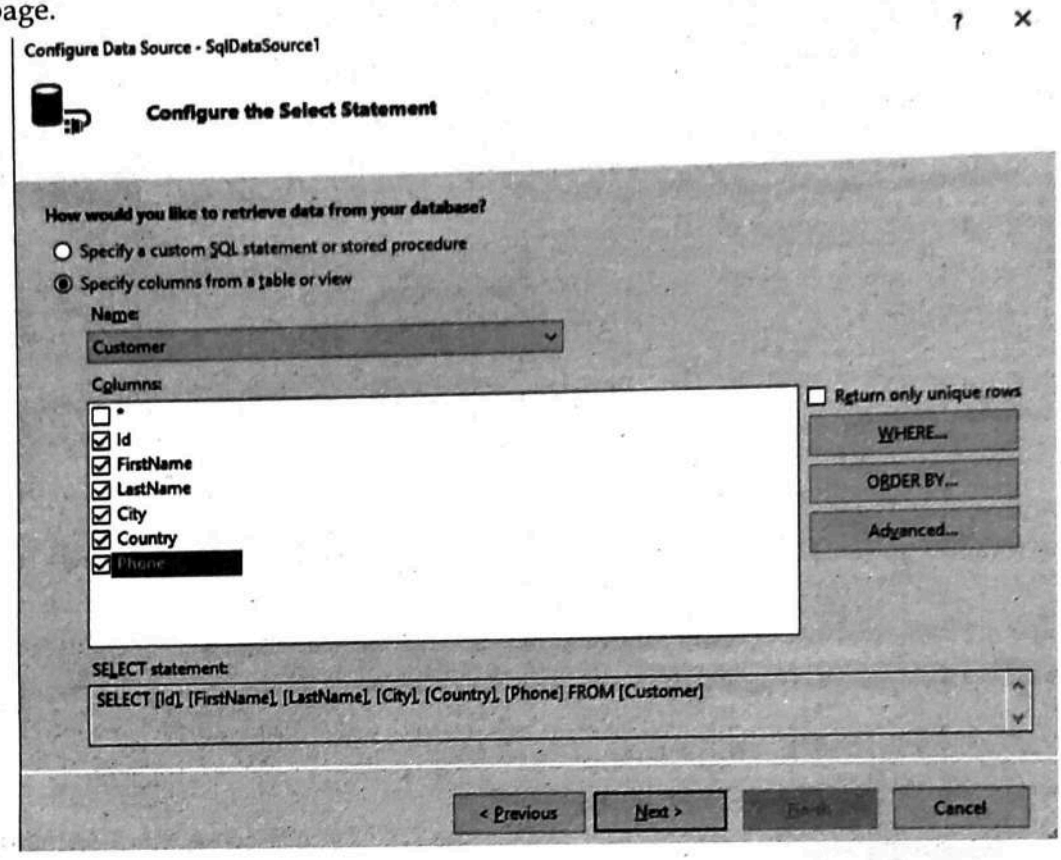


7. Once the Connection is made then click on Next button from Data Source Wizard.

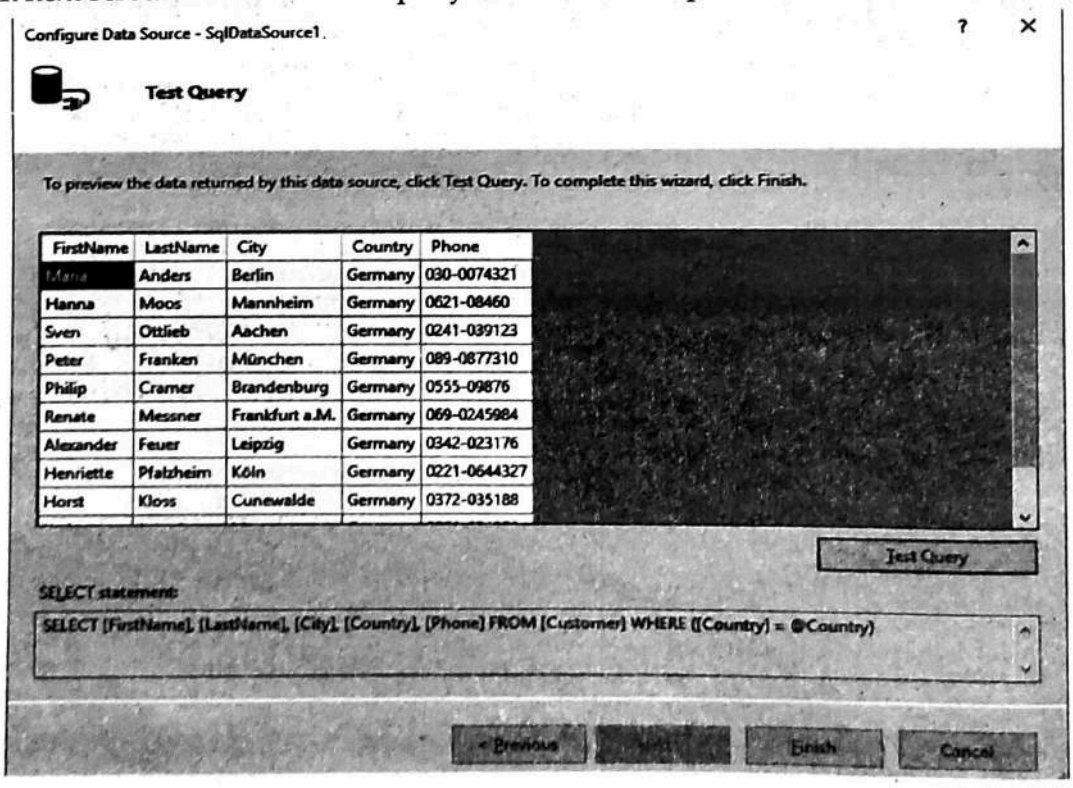


8. Then wizard ask for saving the connection string in configuration file. If you already stored it web.config file then uncheck check box, if you haven't, then select the checkbook. Then click on next button.

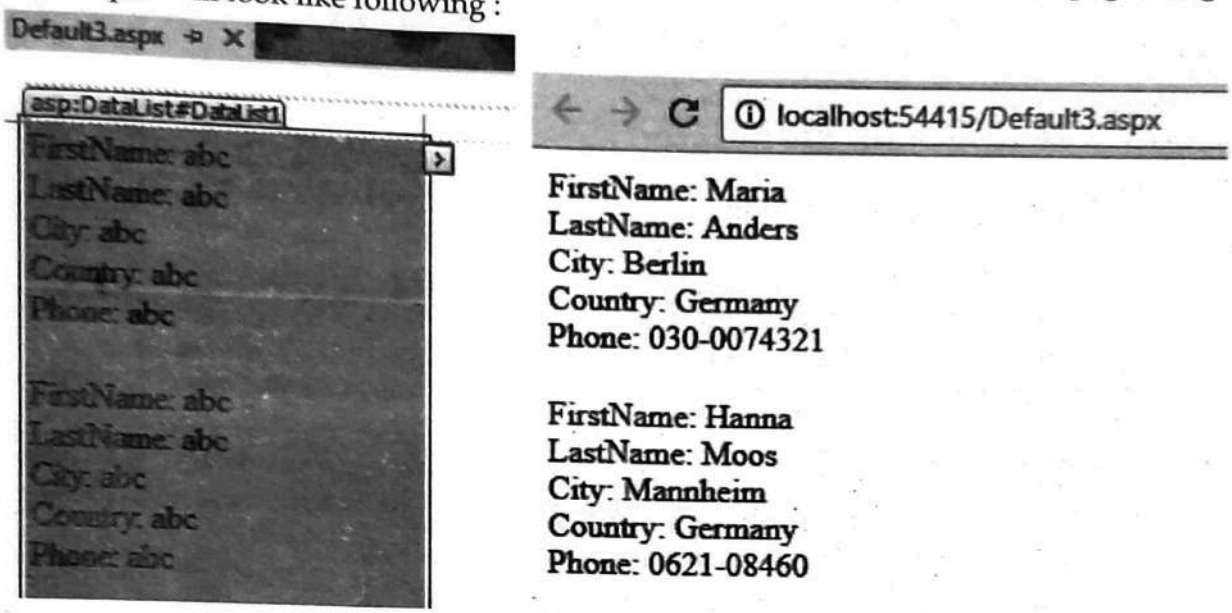
9. The next screen gives option to configure the select statement. Here we can choose the table as well as configure the select statement as we need to display the data on web page.



10. In next screen we can test our query to check the output. Then Click on finish.



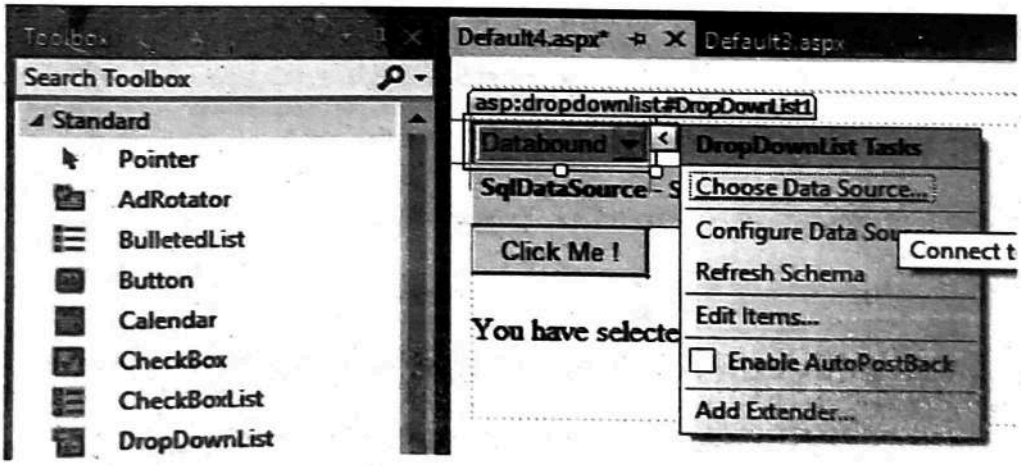
After successful steps form the Datalist controls option wizard our web page design and output will look like following :



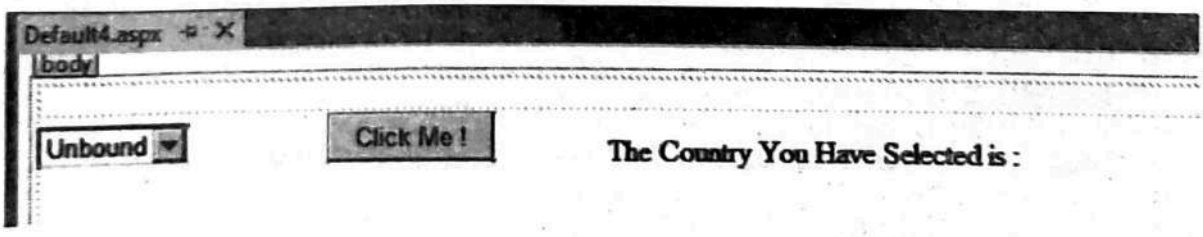
Practical 7

Practical 7 (a) : Create a web application to display Databinding using Dropdownlist control.

1. Create a web page with DropDownList control, one Button and one Label control.
2. The Databinding to DropDownList can be done using Visual Studio options to select the Data Source, the steps will be almost similar as performed in practical 6 (C). We can also use the similar data source available already or we can create new Data source with appropriate select statement to fetch and bound the data.
3. However in following example we have provided code also to bind the data to DropDownList.



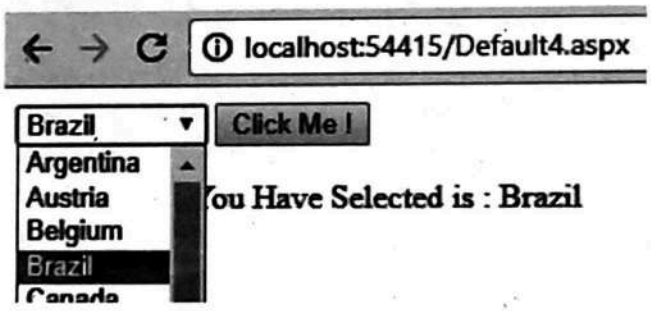
Or with Code also we can achieve the same thing.



Code of C# Code behind file

```
protected void Button1_Click(object sender, EventArgs e)
  {
    Label1.Text = "The Country You Have Selected is : " +

DropDownList1.SelectedValue;
  }
 protected void Page_Load(object sender, EventArgs e)
{
   if (IsPostBack == false)
   {
string connStr =
ConfigurationManager.ConnectionStrings["connStr"].ConnectionString;
SqlConnection con = new SqlConnection(connStr);
SqlCommand cmd = new SqlCommand("Select Distinct Country from Customer",
con);
con.Open( );
SqlDataReader reader = cmd.ExecuteReader( );
DropDownList1.DataSource = reader;
DropDownList1.DataTextField = "Country";
DropDownList1.DataBind( );
reader.Close( );
con.Close( );
    }
}
```

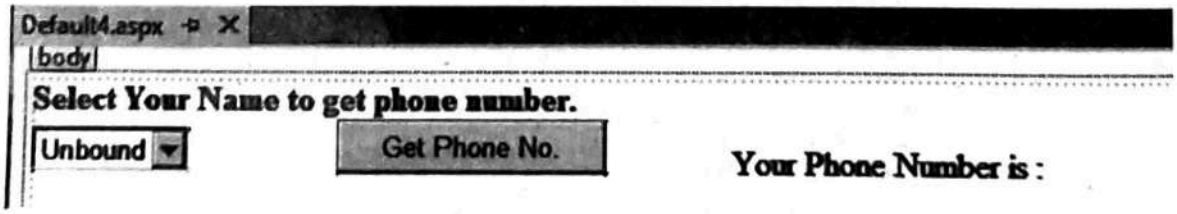
Output



Practical 7 (b): Create a web application for to display the phone no of Customer using database.

We need modify above example 7 (a) little bit to implement our example.

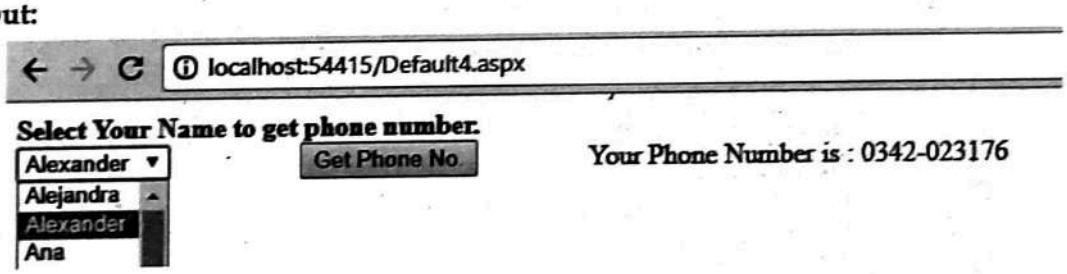
Create a web page with DropDownList, Button and with Label control as shown below :



Code of C# Code behind file

```
protected void Button1_Click(object sender, EventArgs e)
  {
    Label1.Text = "Your Phone Number is : " + DropDownList1.SelectedValue;
  }
  protected void Page_Load(object sender, EventArgs e)
  {
    if (IsPostBack == false)
    {
      string connStr =
       ConfigurationManager.ConnectionStrings["connStr"].ConnectionString;
      SqlConnection con = new SqlConnection(connStr);
      SqlCommand cmd = new SqlCommand("Select Distinct FirstName,LastName,
       Phone
                                                from Customer", con);
      con.Open( );
      SqlDataReader reader = cmd.ExecuteReader( );
      DropDownList1.DataSource = reader;
      DropDownList1.DataTextField = "FirstName";
      DropDownList1.DataValueField = "Phone";
      DropDownList1.DataBind( );
      reader.Close( );
      con.Close( );
    }
   }
```

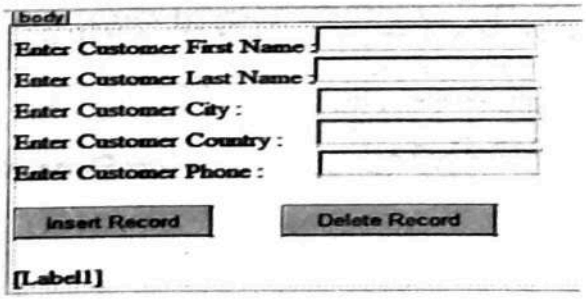
Output:



Practical 7 (c) : Create a web application for inserting and deleting record from a database. (Using Execute-Non Query).

Create a web page with TextBox, and Two Button and one Label control as shown below :

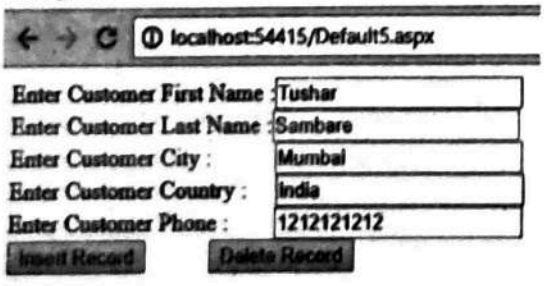
And follow the database related steps same as it is in previous examples.

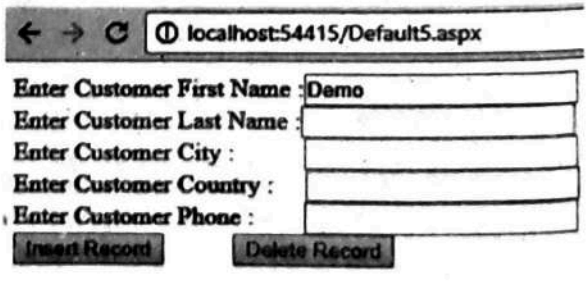


Code of C# Code behind file

```
protected void Button1_Click(object sender, EventArgs e)
  {
     string connStr =
ConfigurationManager.ConnectionStrings["connStr"].ConnectionString;
     SqlConnection con = new SqlConnection(connStr);
     string InsertQuery = "insert into customer values(@fname, @lname, @city, @country,
                                                        @phone)";
     SqlCommand cmd = new SqlCommand(InsertQuery, con);
     cmd.Parameters.AddWithValue("@fname", TextBox1.Text);
     cmd.Parameters.AddWithValue("@lname", TextBox2.Text);
     cmd.Parameters.AddWithValue("@city", TextBox3.Text);
     cmd.Parameters.AddWithValue("@country", TextBox4.Text);
     cmd.Parameters.AddWithValue("@phone", TextBox5.Text);
     con.Open( );
     cmd.ExecuteNonQuery( );
     Label1.Text = "Record Inserted Successfuly.";
     con.Close( );
  }

  protected void Button2_Click(object sender, EventArgs e)
  {
     string connStr =
ConfigurationManager.ConnectionStrings["connStr"].ConnectionString;
     SqlConnection con = new SqlConnection(connStr);
     string InsertQuery = "delete from customer where FirstName=@fname";
     SqlCommand cmd = new SqlCommand(InsertQuery, con);
     cmd.Parameters.AddWithValue("@fname", TextBox1.Text);
     con.Open( );
     cmd.ExecuteNonQuery( );
     Label1.Text = "Record Deleted Successfuly.";
     con.Close( );
  }
```

Output:

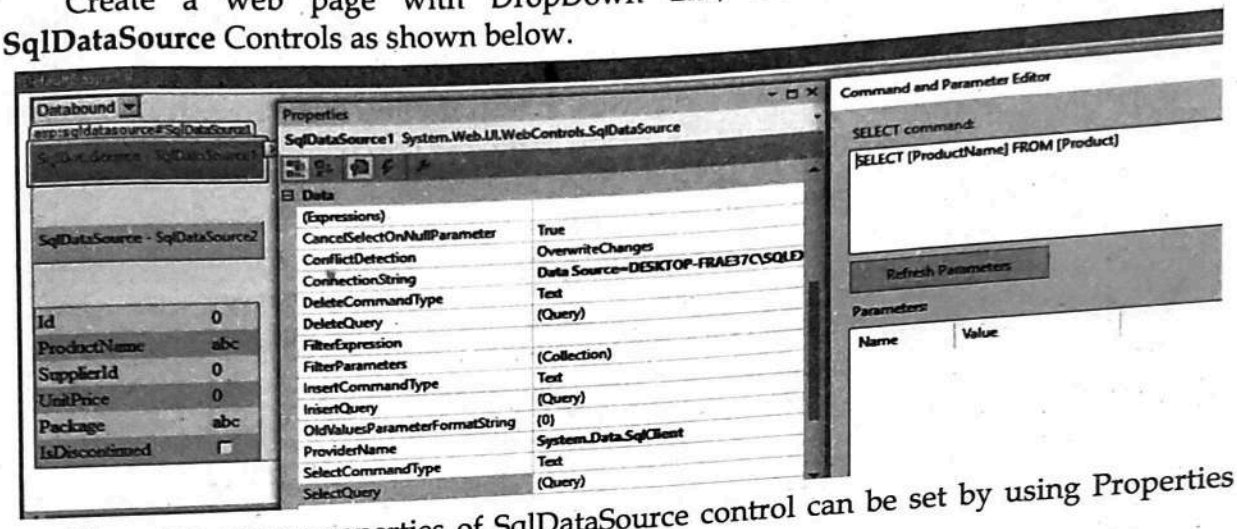




Practical 8

Practical 8 (a) : Create a web application to demonstrate various uses and properties of SqlDataSource.

Create a web page with DropDown List, Details View Control and two **SqlDataSource** Controls as shown below.



There too many properties of SqlDataSource control can be set by using Properties window or by code.

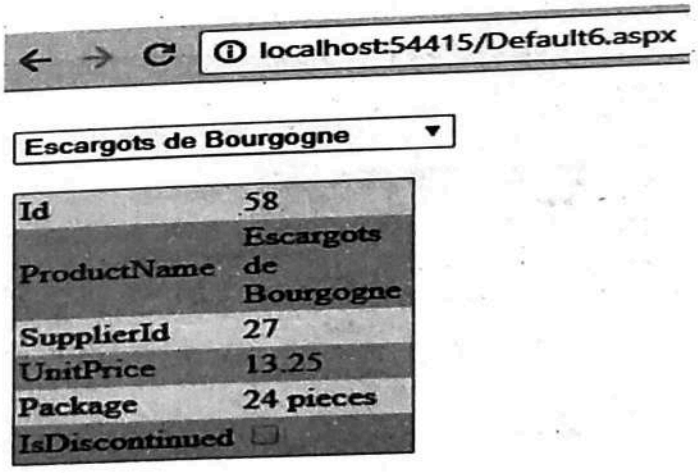
Here in following example we have used a dropdown list which is bind with SqlDataSource1 control to get the Product Name from product table. While the SqlDataSource2 control is used to get the details of the Product selected in Dropdown list. The list and details view are bind to data source as well as the the properties are also set for selecting the templates and view of data.

Note : Kindly refer the practical no **6 (C)** for the detail steps of data binding process with SqlDataSource.

Code of C# Code behind file:

```
protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs
e)
  {
SqlDataSource2.SelectCommand = "Select * from Product where ProductName='"
+                                    DropDownList1.SelectedValue + "'";

  }
```

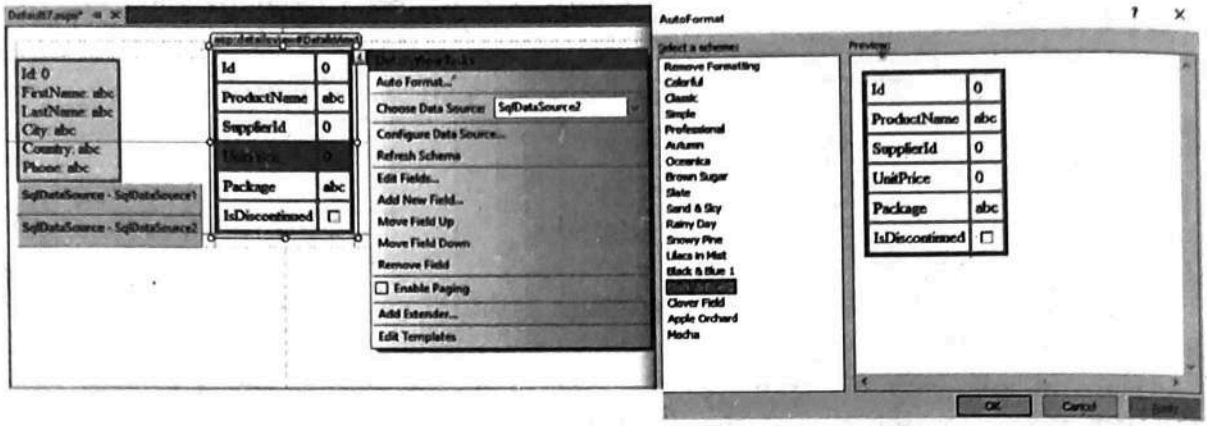
Output



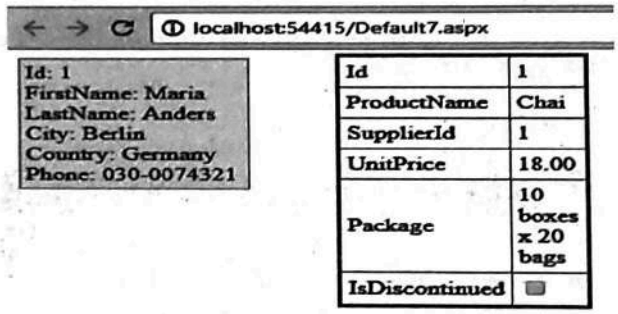
Practical 8 (b) : Create a web application to demonstrate data binding using DetailsView and FormView Control.

1. Add new web page with **DetailsView and FormView** Controls. These controls are available under Data controls in Toolbox.
2. After adding these controls, configure the Data Source property of these controls in similar manner as explained in **Practical 6 (a).**
3. After finishing the Data Source work, we can also use the Auto Format option and Edit template option to configure the display setting of the controls.

   And now just run the application.

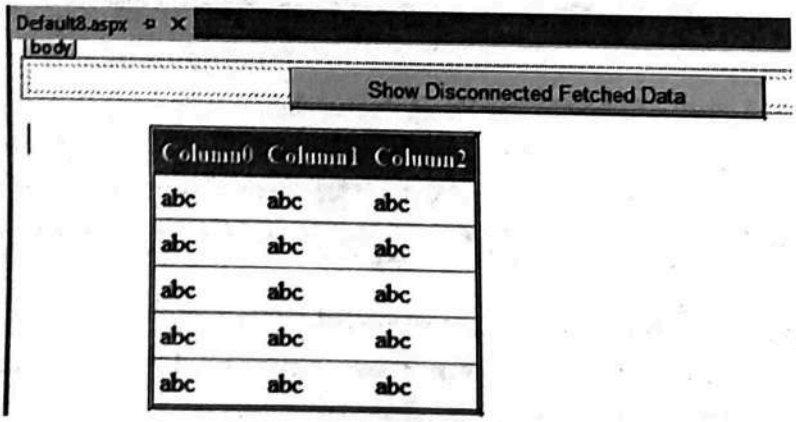


Output



Practical 8 (c) : Create a web application to display Using Disconnected Data Access and Databinding using GridView.

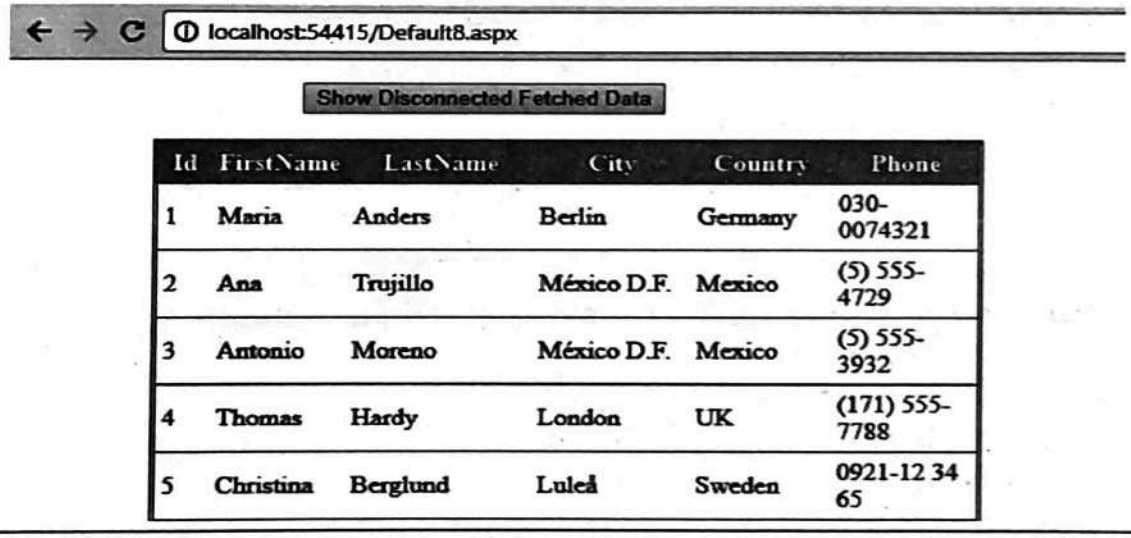
1. Create a webpage using Button and GridView Control.
2. Modify the display setting of GridView using Auto Format option.
3. Change the Text of Button.
4. Write the code provided below in C# code behind file inside the Button click event code segment.
5. Then run the web application.



Code of C# Code behind file

```
protected void Button1_Click(object sender, EventArgs e)
  {
string connStr =
ConfigurationManager.ConnectionStrings["connStr"].ConnectionString;
    SqlConnection con = new SqlConnection(connStr);
    SqlDataAdapter objDa = new SqlDataAdapter( );
    DataSet objDs = new DataSet( );
    using (SqlConnection objConn = new SqlConnection(connStr))
    {
      SqlCommand objCmd = new SqlCommand("Select * from Customer",
objConn);
      objCmd.CommandType = CommandType.Text;
      objDa.SelectCommand = objCmd;
      objDa.Fill(objDs, "Product");
      GridView1.DataSource = objDs.Tables[0];
      GridView1.DataBind( );
    }
  }
```

Output

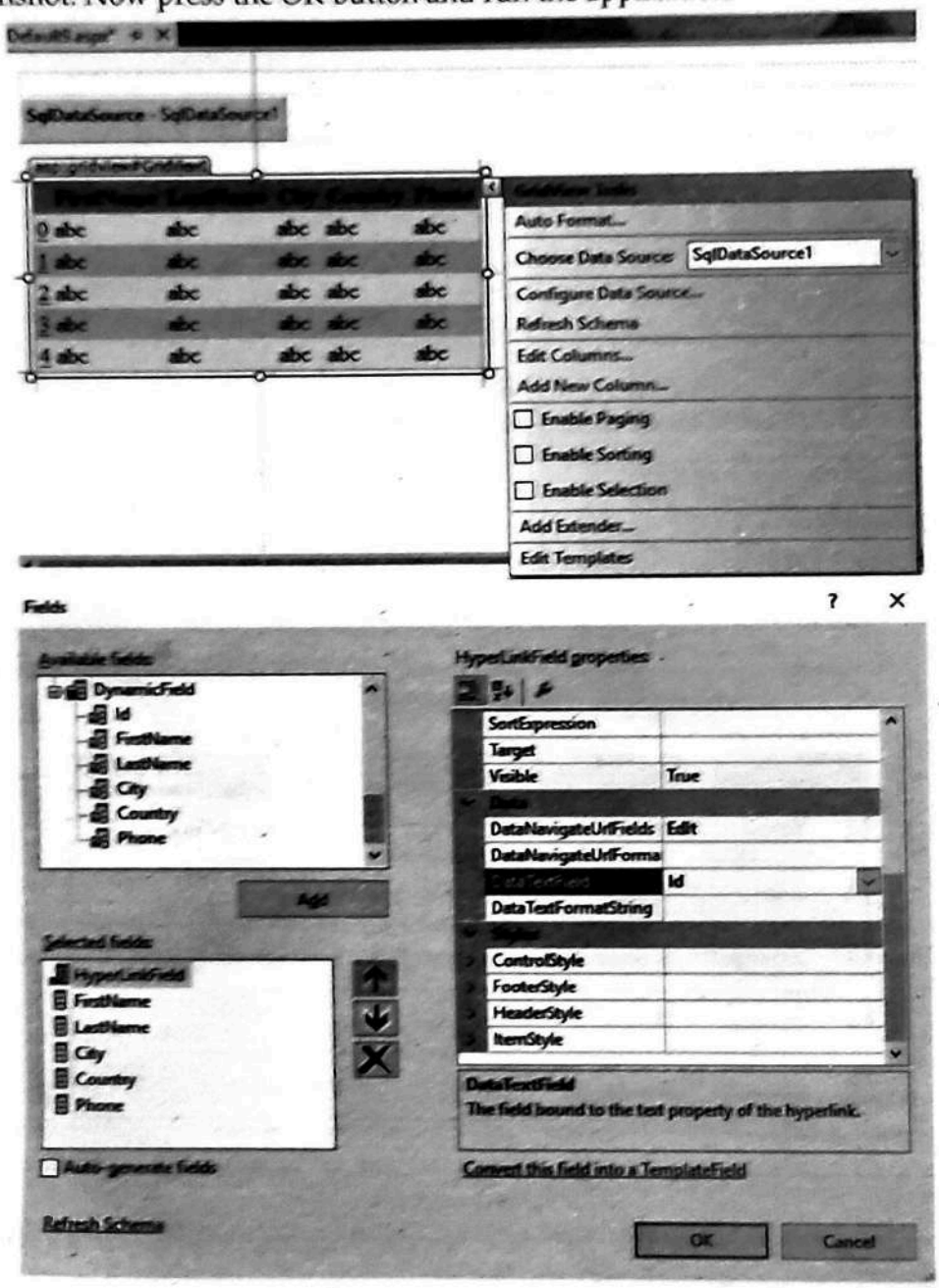


Practical 9

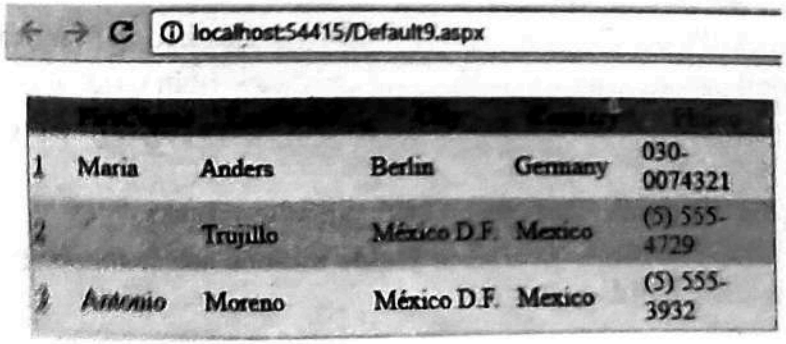
Practical 9 (a) : Create a web application to demonstrate use of GridView control template and GridView hyperlink.

1. Create web page with GridView comtrol.
2. Connect the GridView Control to Data Source as explained in **Practical no 6 (a).**
3. Use the Auto Format options for design of GridView from GridView Tasks.
4. Select Edit templates from GridView Tasks and select **Pager** or **EmptyData** template for GridView.
5. Select the **Edit Columns** form option from GridView Tasks.
6. In the next window from **Available Fields** options select the **HyperLinkField** and **Add** it to **Selected Field**.

7. Navigate the **HyperLinkField** to the top of selected fields using arrow option provided.
8. Remove the repeated or unwanted columns from the selected fields.
9. Select the **HyperLinkField** from the selected fields and change the **DataTextField** Property to **Id** (we are making Id field as hyperlink) as displayed in second Screenshot. Now press the OK button and run the application.



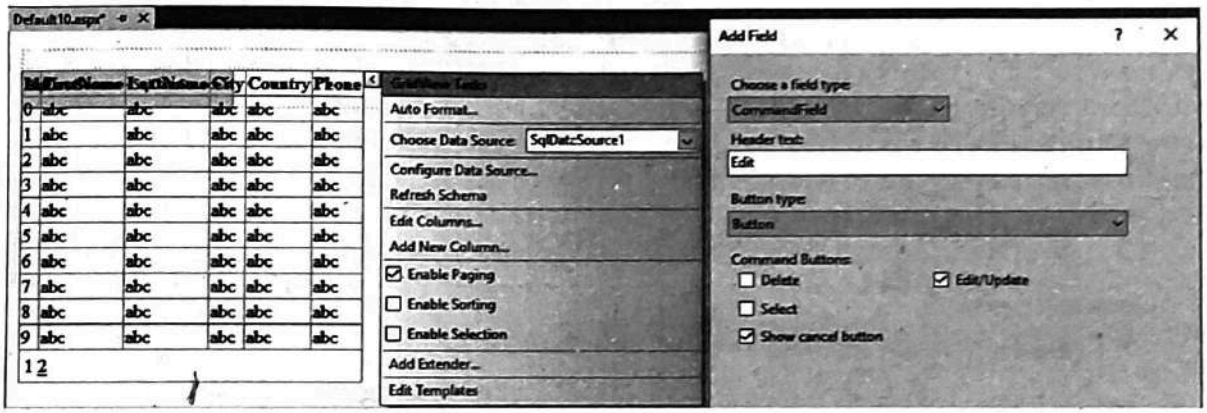
Output



Practical 9 (b) : Create a web application to demonstrate use of GridView button column and GridView events.

Note : This practical intends to demonstrate the capabilities of GridView Control with various options. Following steps and code demonstrates the **Edit** and **Update** Capability of GridView with some additional features in aspx page code, so there is no need of C# code.

1. Add GridView Control to web page.
2. Select **Choose Data Source-> New Data Source** Option from **GridView Tasks**.
3. Configure the new SQLDataSource as explained in **Practical 6(a)**.
4. Select **Add New column** Option from **GridView Tasks**.
5. In **Add Field** Window Choose field type as **CommandField**, Header Text as **Edit**, Button type as **Button**, also select **Edit/Update** CheckBox with **Show Cancel Button** CheckBox.
6. Again Select **Add New column** Option from **GridView Tasks.**
7. In **Add Field** Window Choose field type as **CommandField**, Header Text as **Delete**, Button type as **Button**, also select **Delete** CheckBox.
8. Select **Enable Paging** Checkbox Option from **GridView Tasks.**
9. From **Auto Format** option select the appropriate design for GridView.



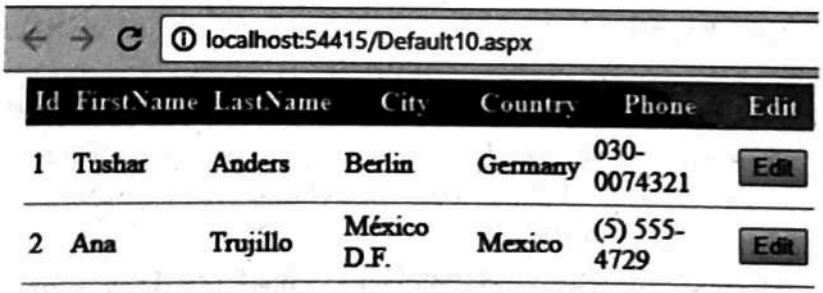
The GridView Code of aspx page.

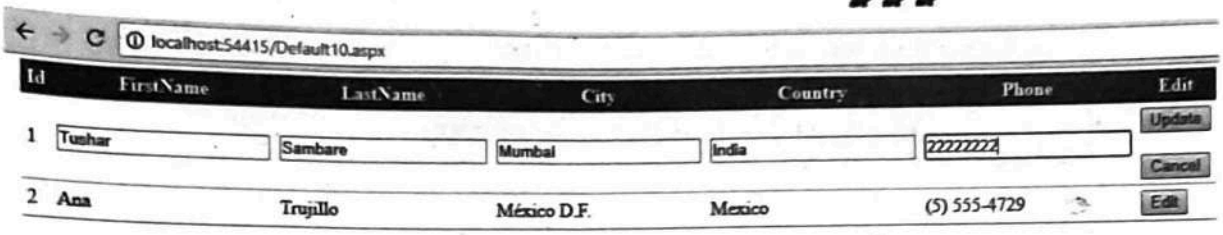
Note : Code that user has to write additionally in SqlDataSource is shown in bold letters.

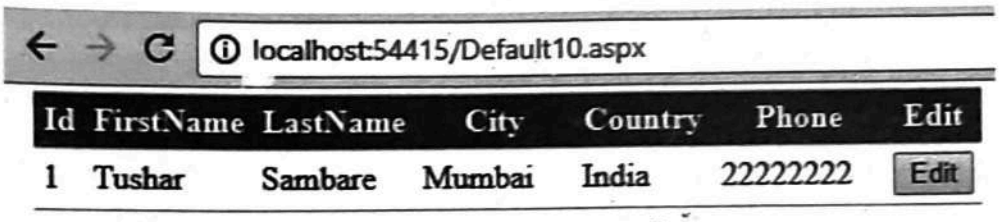
```
<asp:GridView ID="GridView1" runat="server" AllowPaging="True"
AutoGenerateColumns="False" CssClass="auto-style1" DataKeyNames="Id"
DataSourceID="SqlDataSource1" style="z-index: 1" BackColor="White"
BorderColor="#CCCCCC" BorderStyle="None" BorderWidth="1px"
CellPadding="4" ForeColor="Black" GridLines="Horizontal">
    <Columns>
        <asp:BoundField DataField="Id" HeaderText="Id" InsertVisible="False"
ReadOnly="True" SortExpression="Id" />
        <asp:BoundField DataField="FirstName" HeaderText="FirstName"
SortExpression="FirstName" />
        <asp:BoundField DataField="LastName" HeaderText="LastName"
SortExpression="LastName" />
        <asp:BoundField DataField="City" HeaderText="City"
SortExpression="City" />
```

```
            <asp:BoundField DataField="Country" HeaderText="Country"
        SortExpression="Country" />
            <asp:BoundField DataField="Phone" HeaderText="Phone"
        SortExpression="Phone" />
            <asp:CommandField ButtonType="Button" HeaderText="Edit"
        ShowEditButton="True" ShowHeader="True"></asp:CommandField>
          </Columns>
          <FooterStyle BackColor="#CCCC99" ForeColor="Black" />
        <HeaderStyle BackColor="#333333" Font-Bold="True" ForeColor="White" />
          <PagerStyle BackColor="White" ForeColor="Black"
        HorizontalAlign="Right" />
         <SelectedRowStyle BackColor="#CC3333" Font-Bold="True"
        ForeColor="White" />
          <SortedAscendingCellStyle BackColor="#F7F7F7" />
          <SortedAscendingHeaderStyle BackColor="#4B4B4B" />
          <SortedDescendingCellStyle BackColor="#E5E5E5" />
          <SortedDescendingHeaderStyle BackColor="#242121" />
        </asp:GridView>

<asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\Business.mdf;I
ntegrated Security=True" ProviderName="System.Data.SqlClient"
SelectCommand="SELECT * FROM [Customer]"  UpdateCommand="UPDATE
Customer SET FirstName=@FirstName, LastName=@LastName,
City=@City,Country=@Country,Phone=@Phone Where Id=@Id" >

  <UpdateParameters>
        <asp:Parameter Name="Id" Type="Int32" />
        <asp:Parameter Name="FirstName" Type="String" />
        <asp:Parameter Name="LastName" Type="String" />
        <asp:Parameter Name="City" Type="String" />
        <asp:Parameter Name="Country" Type="String" />
        <asp:Parameter Name="Phone" Type="String" />
      </UpdateParameters>
</asp:SqlDataSource>
```

Output



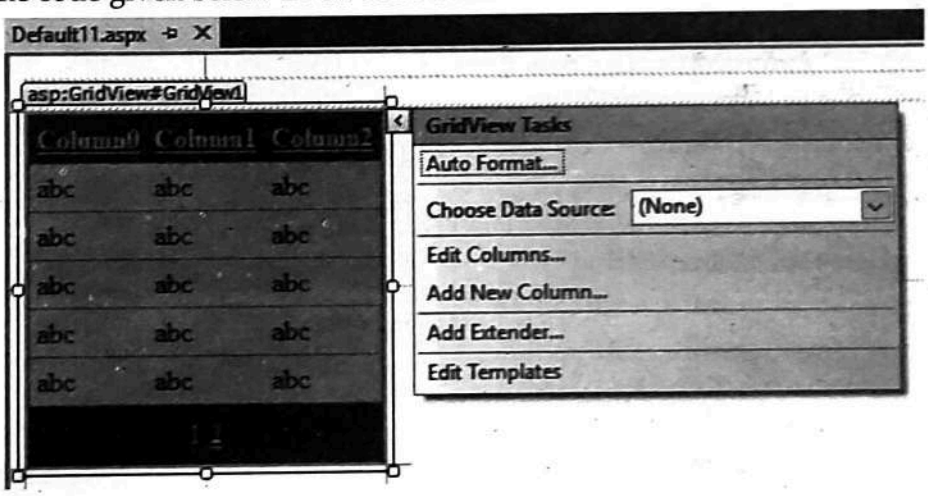




Practical 9 (c) : Create a web application to demonstrate GridView paging and Creating own table format using GridView.

Note : This practical is intended to create an own table and bind data using data grid and also to demonstrate Paging, Sorting functionalities of GridView control.

1. Add GridView Control to web page.
2. From **Auto Format** option select the appropriate design for GridView.
3. Add the code given below in C# code behind File.



Code of GridView in aspx File

```
<asp:GridView ID="GridView1" runat="server" AllowPaging="True" BackColor="White"
        OnSorting="GridView1_Sorting" BorderColor="#336666"
        OnPageIndexChanging="GridView1_PageIndexChanging" BorderStyle="Double"
        BorderWidth="3px"          CellPadding="4"          CssClass="auto-style1"
GridLines="Horizontal"
        PageSize="5" style="z-index: 1" AllowSorting="True">
        <FooterStyle BackColor="White" ForeColor="#333333" />
        <HeaderStyle BackColor="#336666" Font-Bold="True" ForeColor="White" />
        <PagerStyle  BackColor="#336666"  ForeColor="White"  HorizontalAlign="Center"
/>
        <RowStyle BackColor="White" ForeColor="#333333" />
        <SelectedRowStyle BackColor="#339966" Font-Bold="True" ForeColor="White" />
        <SortedAscendingCellStyle BackColor="#F7F7F7" />
```

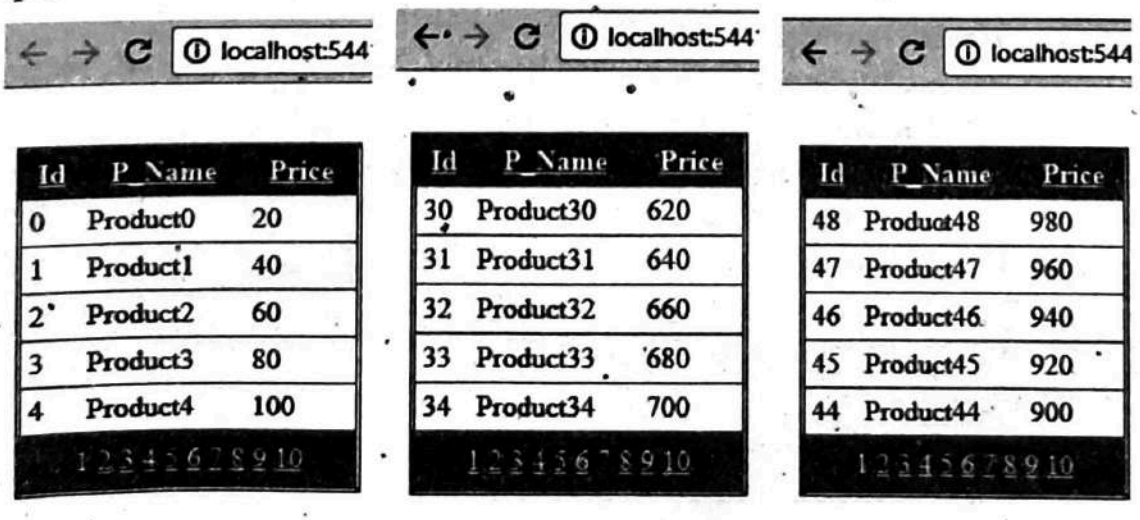
```
    <SortedAscendingHeaderStyle BackColor="#487575" />
    <SortedDescendingCellStyle BackColor="#E5E5E5" />
    <SortedDescendingHeaderStyle BackColor="#275353" />
</asp:GridView>
```

Code of C# Code behind file (add using System.Data):

```
protected void Page_Load(object sender, EventArgs e)
        {
          if (!IsPostBack)
          {
            DemonstrateDataView( );
          }
        }

private void DemonstrateDataView( )
        {
          // Create one DataTable with one column.
          DataTable myTable = new DataTable("myTable");
          DataColumn colItem1 = new DataColumn("Id",
      Type.GetType("System.Int32"));
           DataColumn colItem2 = new DataColumn("P_Name",
      Type.GetType("System.String"));
          DataColumn colItem3 = new DataColumn("Price",
      Type.GetType("System.String"));
          myTable.Columns.Add(colItem1);
          myTable.Columns.Add(colItem2);
          myTable.Columns.Add(colItem3);
          // Add fifty items.
          DataRow NewRow;
          for (int i = 0; i < 50; i++)
          {
            NewRow = myTable.NewRow( );
            NewRow["Id"] =  i;
            NewRow["P_Name"] ="Product"+ i;
            NewRow["Price"] = (20 * (i + 1));
            myTable.Rows.Add(NewRow);
          }

          GridView1.DataSource = myTable;
          GridView1.AllowPaging = true;
          GridView1.AllowSorting = true;
          ViewState["dataTable"] = myTable;
          ViewState["sortdr"] = "Asc";
```

```
        GridView1.DataBind( );
    }

protected void GridView1_PageIndexChanging(object sender, GridViewPageEventArgs
e)
    {
        GridView1.PageIndex = e.NewPageIndex;
        DemonstrateDataView( );
    }


protected void GridView1_Sorting(object sender, GridViewSortEventArgs e)
    {
        DataTable dtrslt = (DataTable)ViewState["dataTable"];
        if (dtrslt.Rows.Count > 0)
        {
            if (Convert.ToString(ViewState["sortdr"]) == "Asc")
            {
                dtrslt.DefaultView.Sort = e.SortExpression + " Desc";
                ViewState["sortdr"] = "Desc";
            }
            else
            {
                dtrslt.DefaultView.Sort = e.SortExpression + " Asc";
                ViewState["sortdr"] = "Asc";
            }
            GridView1.DataSource = dtrslt;
            GridView1.DataBind( );
        }
    }
}
```

Output



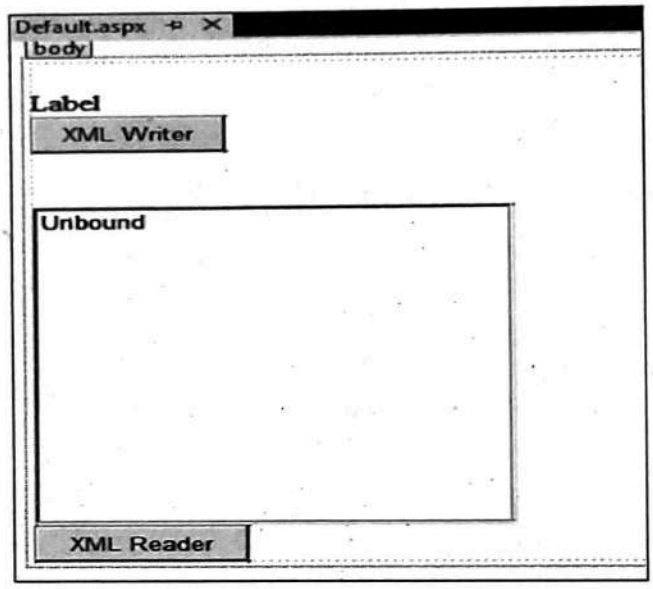
Practical 10 : Working with AJAX and XML

Practical 10(a) : Create a web application to demonstrate reading and writing operation with XML.

demo.xml

```
<?xml version="1.0"?>
```

Default.aspx

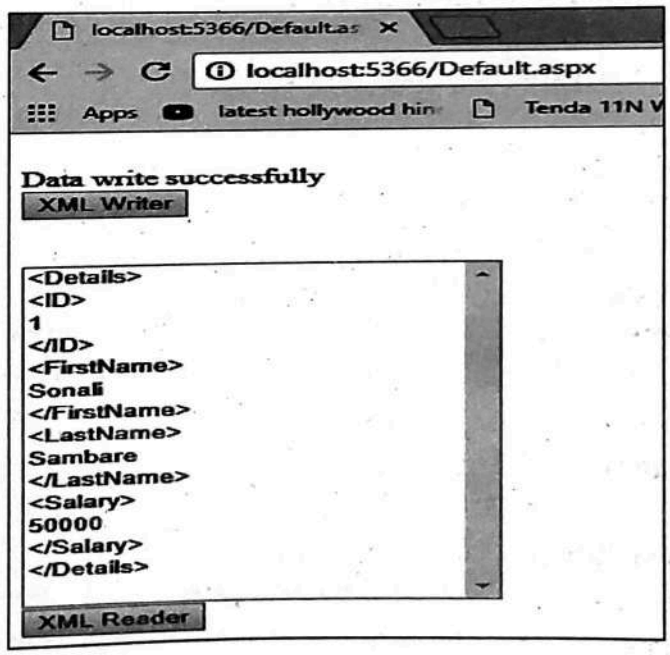


Default.aspx.cs

```
using System;
using System.Xml;
public partial class _Default : System.Web.UI.Page
{
protected void Button1_Click(object sender, EventArgs e)
  {
    XmlTextWriter writer = new
XmlTextWriter("C:\\Users\\Tushar\\Documents\\Visual Studio
2015\\WebSites\\pract10a\\demo.xml", null);
    writer.WriteStartDocument( );
    // Write next element
    writer.WriteStartElement("Details", "");
    writer.WriteElementString("ID", "1");
    writer.WriteElementString("FirstName", "Sonali");
    writer.WriteElementString("LastName", "Sambare");
    writer.WriteElementString("Salary", "50000");
    writer.WriteEndElement( );
    // Ends the document.
    writer.WriteEndDocument( );
    writer.Close( );
    Label1.Text = "Data write successfully";
  }
```

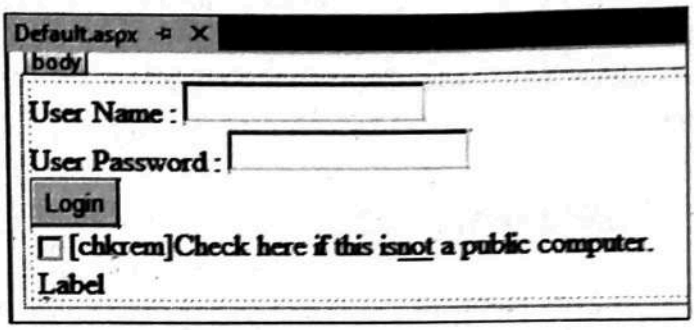
```
protected void Button2_Click(object sender, EventArgs e)
{
    String xmlNode = "C:\\Users\\Tushar\\Documents\\Visual Studio
2015\\WebSites\\pract10a\\demo.xml";
    XmlReader xReader = XmlReader.Create(xmlNode);
    while (xReader.Read())
    {
      switch (xReader.NodeType)
      {
        case XmlNodeType.Element:
          ListBox1.Items.Add("<" + xReader.Name + ">");
          break;
        case XmlNodeType.Text:
          ListBox1.Items.Add(xReader.Value);
          break;
        case XmlNodeType.EndElement:
          ListBox1.Items.Add("</" + xReader.Name + ">");
          break;
      }
    }
  }
}
```

OUTPUT



Practical 10(b) : Create a web applications to demonstrate Form Security and Windows Security with proper Authentication and Authorization properties.

Default.aspx

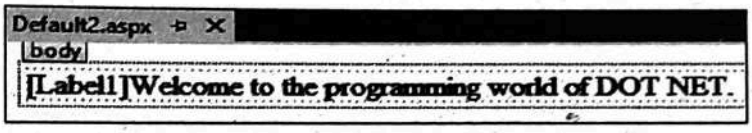


Default.aspx.cs

```
using System;
using System.Web.Security;
public partial class _Default : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
  {

  }
  protected bool authenticate(String uname, String pass)
  {
    if (uname == "Yash")
    {
      if (pass == "yash123")
        return true;
    }

    if (uname == "Ved")
    {
      if (pass == "ved123")
        return true;
    }

    if (uname == "Sam")
    {
      if (pass == "sam123")
        return true;
    }

    return false;
  }
```

```
    protected void Button1_Click(object sender, EventArgs e)
  {
    if (authenticate(txtuser.Text, txtpwd.Text))
    {
      FormsAuthentication.RedirectFromLoginPage(txtuser.Text, chkrem.Checked);
      Session["Username"] = txtuser.Text;
      Response.Redirect("Default2.aspx");
    }
    else
    {
      Response.Write("Invalid user name or password");
    }
  }
}
```

Default2.aspx



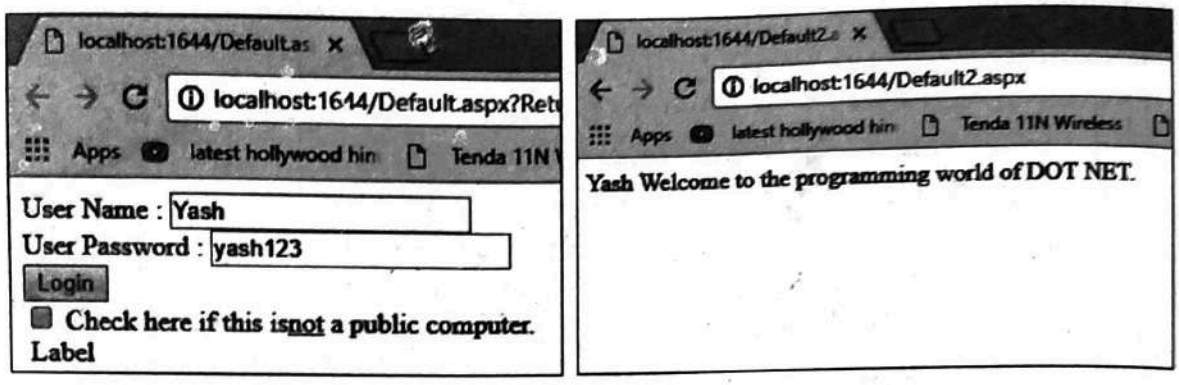
Default2.aspx.cs

```
protected void Page_Load(object sender, EventArgs e)
  {
    if (Session["Username"] != null)
    {
      //Response.Write(Session["Username"].ToString());
      Label1.Text = Session["Username"].ToString();
    }
  }
```

Web.config

```
<configuration>
  <system.web>
    <authentication mode="Forms">
    <forms loginUrl ="Default.aspx"/>
  </authentication>
    <authorization>
    <deny usefs="?"/>
  </authorization>
    <compilation debug="true" targetFramework="4.5.2" />
    <httpRuntime targetFramework="4.5.2" />
  </system.web>
</configuration>
```

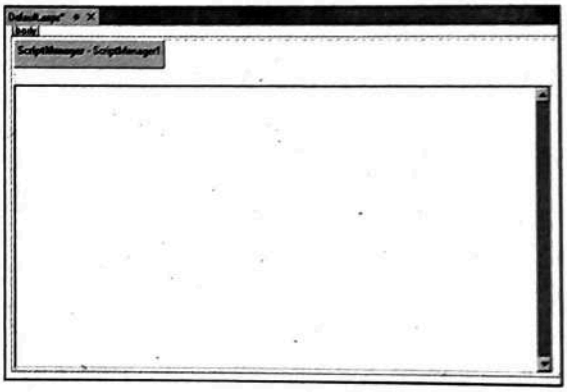
OUTPUT



Practical 10(c) : Create a web application to demonstrate use of various Ajax controls.

1. Add the Script Manager Control on your Page.
2. Add a TextBox (set properties Columns="80" Rows="20" TextMode="MultiLine").
3. Select that TextBox Control.
4. Now add HtmlEditorExtender Control from AJAX Control Toolkit.

Default.aspx

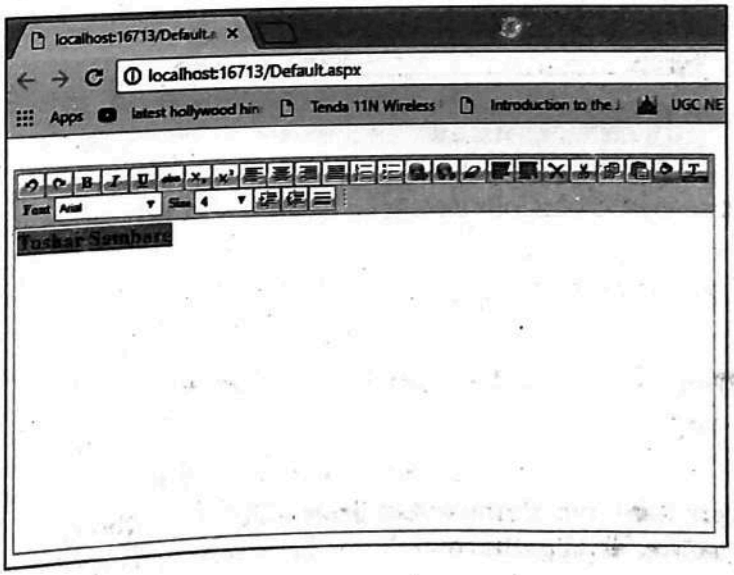


```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"
Inherits="_Default" %>
<%@ Register assembly="AjaxControlToolkit" namespace="AjaxControlToolkit"
tagprefix="ajaxToolkit" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
    <asp:ScriptManager ID="ScriptManager1" runat="server">
    </asp:ScriptManager>
    <br />
  <asp:TextBox runat="server" Columns="80" Rows="20" id="txtBody"
TextMode="MultiLine"></asp:TextBox>
```

```
    <ajaxToolkit:HtmlEditorExtender ID="txtBody_
TargetControlID="txtBody" EnableSanitization="false">
      <Toolbar>
       <ajaxToolkit:Undo/>          <ajaxToolkit:Redo/>
       <ajaxToolkit:Bold/>          <ajaxToolkit:Italic/>
       <ajaxToolkit:Underline/>          <ajaxToolkit:StrikeThrough/>
       <ajaxToolkit:Subscript/>          <ajaxToolkit:Superscript/>
       <ajaxToolkit:JustifyLeft/>          <ajaxToolkit:JustifyCenter/>
       <ajaxToolkit:JustifyRight/>          <ajaxToolkit:JustifyFull/>
       <ajaxToolkit:InsertOrderedList/>
       <ajaxToolkit:InsertUnorderedList/>
       <ajaxToolkit:CreateLink/>          <ajaxToolkit:UnLink/>
       <ajaxToolkit:RemoveFormat/>
       <ajaxToolkit:SelectAll/>          <ajaxToolkit:UnSelect/>
       <ajaxToolkit:Delete/>          <ajaxToolkit:Cut/>
       <ajaxToolkit:Copy/>          <ajaxToolkit:Paste/>
       <ajaxToolkit:BackgroundColorSelector/>
       <ajaxToolkit:ForeColorSelector/>
       <ajaxToolkit:FontNameSelector/>
       <ajaxToolkit:FontSizeSelector/>
       <ajaxToolkit:Indent/>          <ajaxToolkit:Outdent/>
       <ajaxToolkit:InsertHorizontalRule/>
       <ajaxToolkit:HorizontalSeparator/>
      </Toolbar>
    </ajaxToolkit:HtmlEditorExtender>
  </div>
  </form>
</body>
</html>
```

and then add **TSClassLib.dll** by

OUTPUT



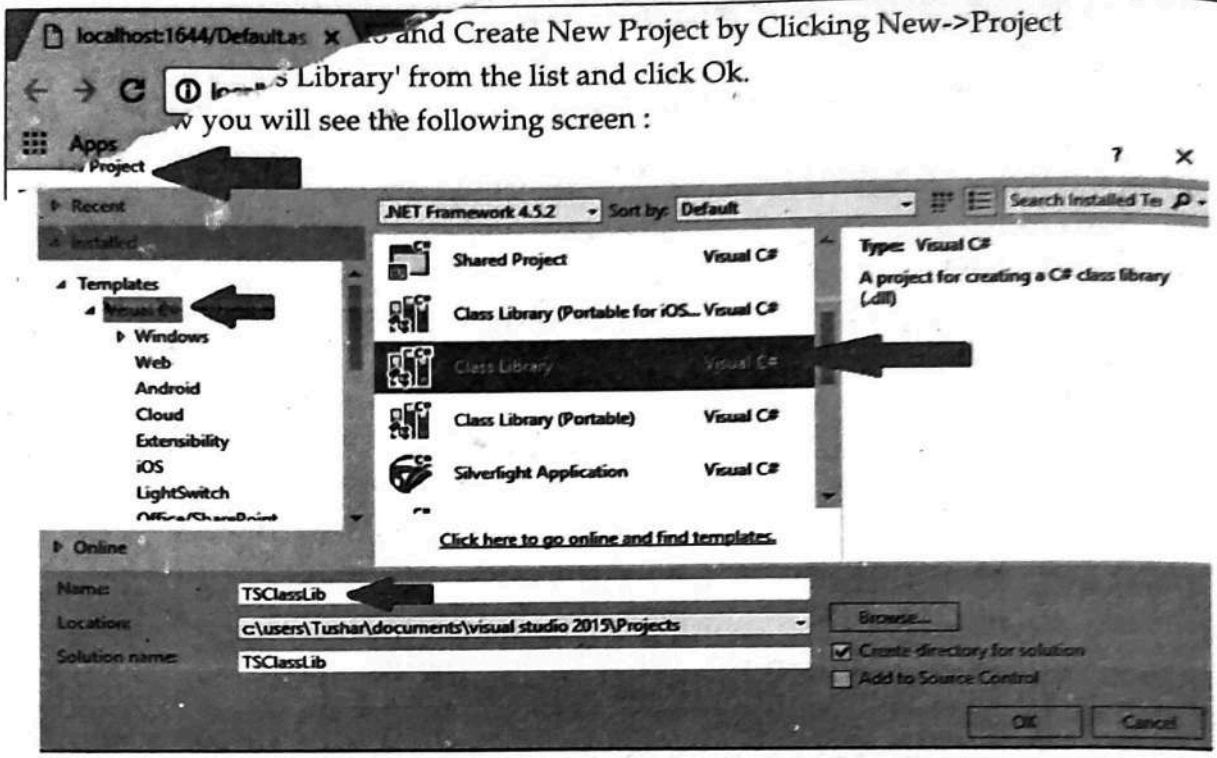
OUTPUT

use DLL

and Create New Project by Clicking New->Project

Library' from the list and click Ok.

you will see the following screen :

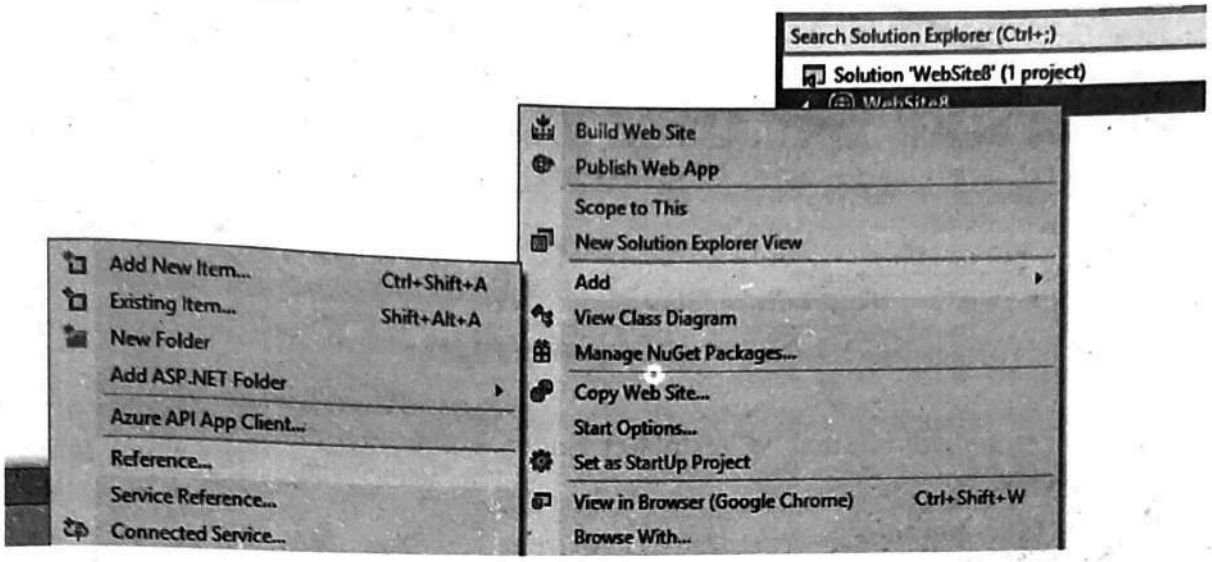


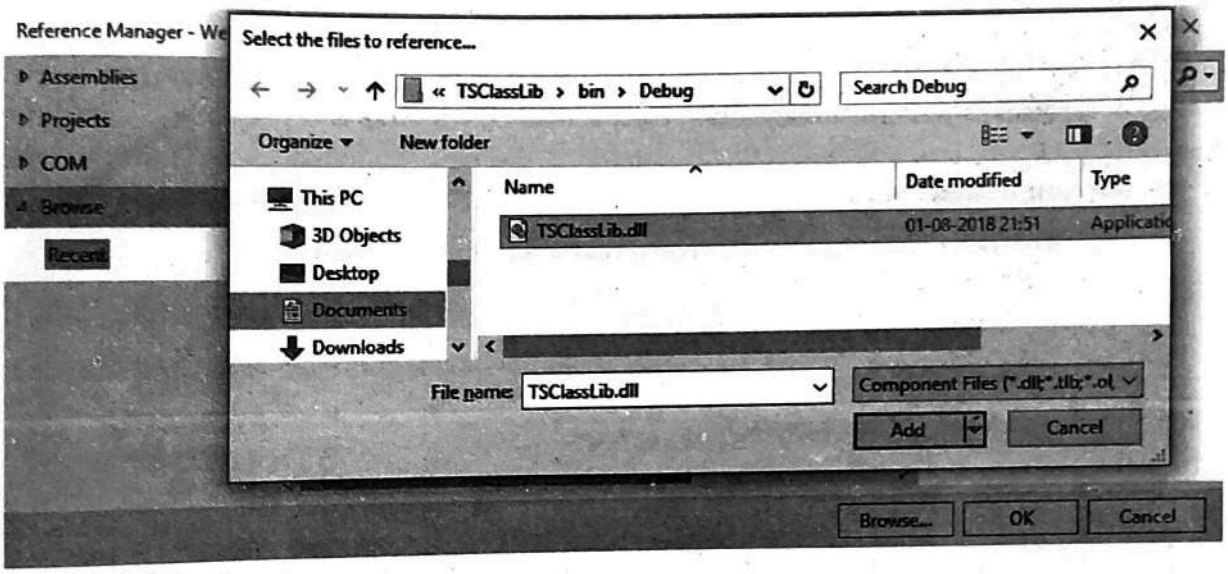
Step 4 : Change the name TS. As I have changed the Name with TS and Write the code following sample code in .cs file. (You can add any other logic also).

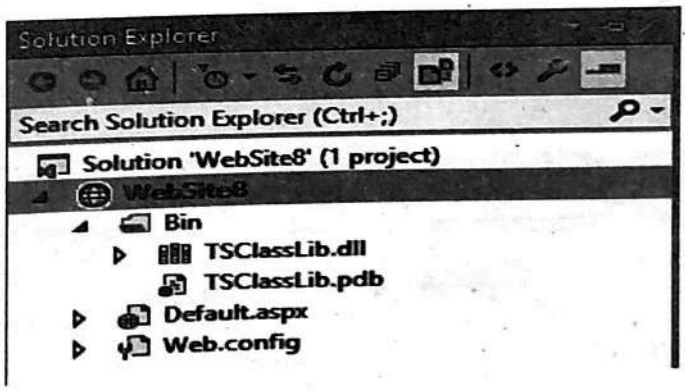
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TSClassLib
{
  public class TS
  {
    public string UpperConvert(string text)
    {
      return text.ToUpper( );
    }
    public string LowerConvert(string text)
    {
      return text.ToLower( );.
    }
  }
}
```

Step 5 : Now Click Build Menu -> Build Solution to complete the process. After doing this Build an application, for generate the TSClassLib.dll. The TSClassLib.dll file automatically generated by compiling the application in Your Application/Bin/Debug Folder.

Step 6 : Now to use the assembly, **create a new web site,** and then add **TSClassLib.dll** by add reference to Bin Folder.

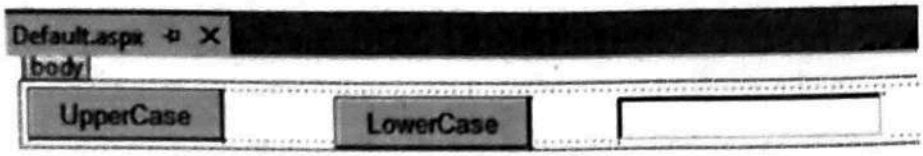






Step 7 : Add new web page in website and add following code in **.aspx** file.

```
<asp:Button ID="Button1" runat="server" onclick="Button1_Click" Text="Upper" />
<asp:Button ID="Button2" runat="server" onclick="Button2_Click" Text="Lower" />
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
```

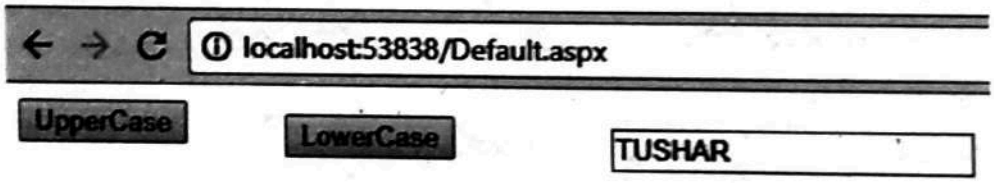


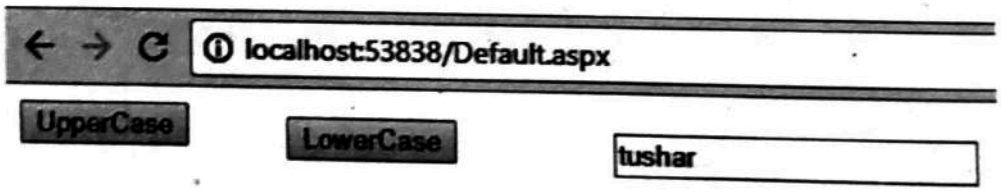
Step 8 : Add the following code in .cs file of your web page for appropriate button handler event.

```
using System;
using TSClassLib;

public partial class _Default : System.Web.UI.Page
{
  protected void Button1_Click(object sender, EventArgs e)
  {
    TS t = new TS( );
    TextBox1.Text=t.UpperConvert(TextBox1.Text);
  }

  protected void Button2_Click(object sender, EventArgs e)
  {
    TS t = new TS( );
    TextBox1.Text=t.LowerConvert(TextBox1.Text);
  }
}
```

Step 9 : Now run the Default.aspx page

Output :





SAMPLE QUESTION PAPER

All questions are compulsory :

1. Attempt *any three* of the following : **(15)**

a) What is .NET Framework? What is in the .NET Framework?

b) What is the CTS, and how does it relate to the CLS?

c) Write a note on Type Conversion.

d) Explain array memory representation with an example.

e) How to write properties in class? Give example.

f) What is constructor overloading? Give example.

2. Attempt *any three* of the following : **(15)**

a) Explain any five common properties of web server controls.

b) Explain Anatomy of a Webform.

c) Write a note on Page class.

d) What is use of autopostback and runat properties.

e) Explain TreeView and Menu site navigation controls.

f) Give the Life Cycle of a Web Page.

3. Attempt *any three* of the following : **(15)**

a) Explain exception handling mechanism in C#.

b) What is custom exception and how to raise it in C#?

c) How to Manage State in ASP.Net?

d) Explain cross page posting with an example.

e) What is CSS? Give its advantages and disadvantages.

f) What is Theme? Explain Global theme.

4. Attempt *any three* of the following : **(15)**

a) What is Database? What are its uses?

b) Give the example of filling ListBox with SQL Database table entries.

c) Write short note on Data Source Controls.

d) How to set Parameter Values in Code?

e) Write short note on selecting a GridView Row.

f) Explain the process of event handling with GridView template in Visual Studio

5. Attempt *any three* of the following : **(15)**

a) What is XML? How we can improve listing with XML?

b) Write short note on XML Namespaces.

c) Explain Windows Authentication.

d) How to Test and Deploy Security Settings in ASP .NET?

e) Explain AJAX with its advantages and Disadvantages.

f) Write short note on Accordion control with appropriate properties.

MUMBAI UNIVERSITY QUESTION PAPER
(November - 2018)

Time: 2 ½ hour **(Marks : 75)**

Note :
1. All questions are compulsory.
2. Make Suitable assumptions wherever necessary and state the assumptions made.
3. Answers to the same question must be written together.
4. Numbers to the right indicate marks.
5. Draw neat labeled diagrams wherever necessary.
6. Use of Non-Programmable calculators is allowed.

1. Attempt *any three* of the following : **(15)**

a) What is namespace? Explain with the help of an example.
b) Explain jagged array with an example.
c) What is .NET Framework? Explain its architecture in brief.
d) Write a program in C# to demonstrate multiple inheritance using interfaces.
e) Explain various types of constructors in C#.
f) What is delegate? Explain multicast delegate with an example.

2. Attempt *any three* of the following : **(15)**

a) What is the difference between ListBox and Drop Down List? List and explain any three common properties of these controls.
b) Explain AdRotator control with an example.
c) List and explain any four types of validation controls used in ASP.NET.
d) Explain Calender control with an example in ASP.NET.
e) Write short note on Page class.
f) Explain SiteMapPath control in ASP.NET.

3. Attempt *any three* of the following : **(15)**

a) What is user-defined exception? Explain with example.
b) What is debugging. Explain the process of debugging in detail.
c) Write short note on cookies in ASP.NET.
d) What is ViewState in ASP.NET? State its Advantages and Disadvantages.
e) Create a web application to demonstrate use of Master Page with applying styles and themes for page beautification. Write necessary steps with code for the same.
f) Explain the four most important selectors present in CSS.

4. Attempt *any three* of the following: **(15)**

a) List and Explain ADO.NET objects
b) What is DataReader in ADO.NET? Explain with an example.
c) Explain SqlDataSource in ADO.NET.
d) What is a GridView control? Explain with an example.
e) What are the application services provided in ASP.NET? Explain.
f) Differentiate between Form View and Details View in ASP.NET.

5. Attempt *any three* of the following : **(15)**

a) Explain XmlTextReader and XmlTextWriter with an example.

b) What is XElement? Explain with an example.

c) What do you mean by authentication? Explain its types.

d) What do you mean by Impersonation in ASP.NET? Explain.

e) Explain ASP.NET AJAX Control Toolkit.

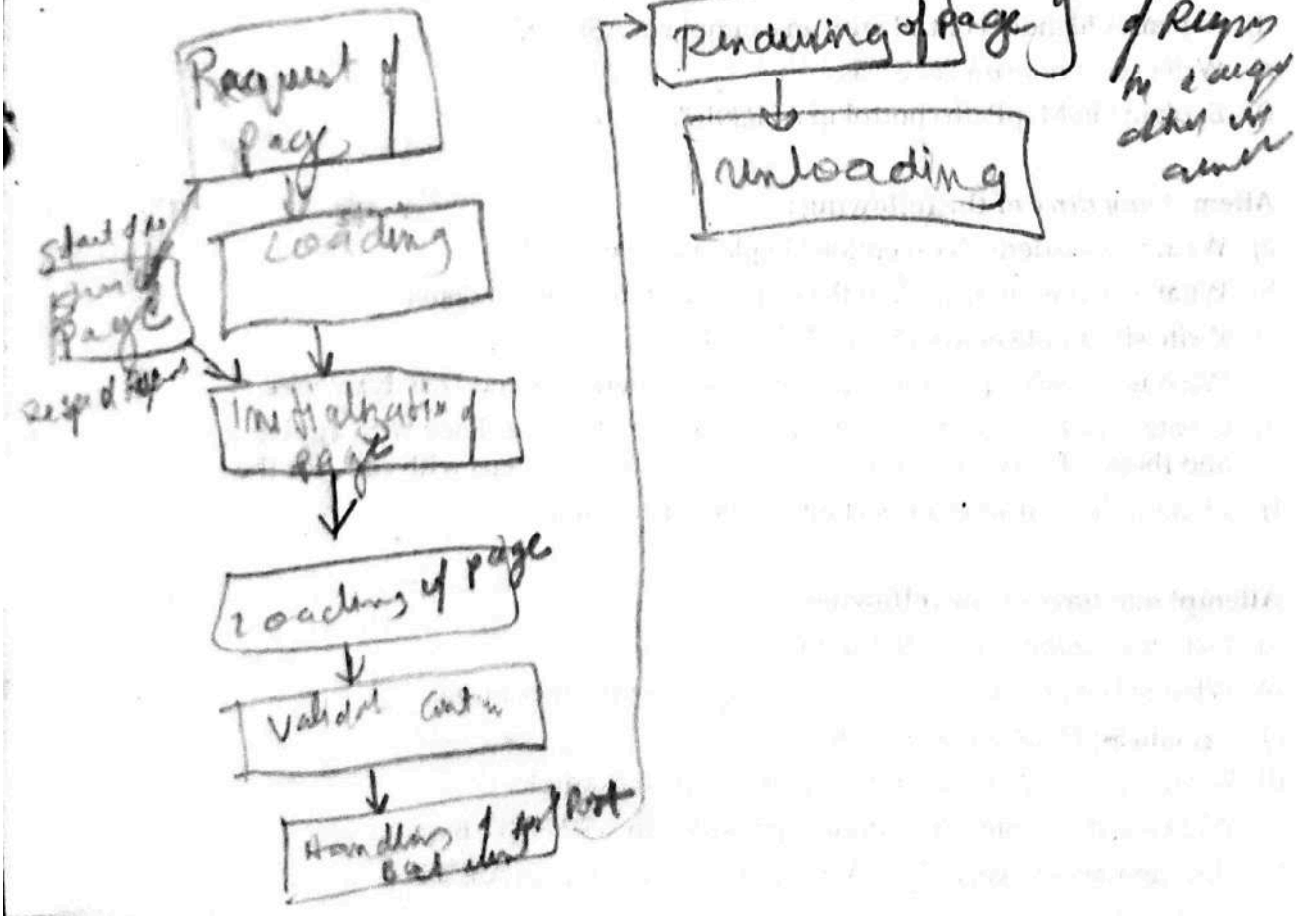
f) Create a web application to demonstrate the use of HTMLEditorExtender Ajax Control.

Write the code of default.aspx and required property settings for the same.

→ client Request for page

App LC

Apps → Object → HTTP App



MUMBAI UNIVERSITY QUESTION PAPER
(April – 2019)

Time: 2 ½ hour (Marks : 75)

Note :
1. All questions are compulsory.
2. Make Suitable assumptions wherever necessary and state the assumptions made.
3. Answers to the same question must be written together.
4. Numbers to the right indicate marks.
5. Draw neat labelled diagrams wherever necessary.
6. Use of Non-Programmable calculators is allowed.

1. **Attempt *any three* of the following :** (15)
   a) Draw and Explain. NET framework architecture.
   b) Elaborate Array memory representation with an example.
   c) Explain any 5 properties/methods of Math class.
   d) Give details about Type Conversion.
   e) Write short note on Value and Reference types.
   f) Explain static members and partial class.

2. **Attempt *any three* of the following :** (15)
   a) List and explain any 5 templates to create ASP. NET applications.
   b) Explain Anatomy of a Webform.
   c) Write a note on page class.
   d) Explain any five properties of List box and Drop-down list controls.
   e) Write a note on AdRotator control.
   f) Brief about Graphics class and it's any 5 methods

3. **Attempt *any three* of the following :** (15)
   a) Explain exception handling mechanism in C# with its key features.
   b) What are state management techniques in ASP.Net?
   c) Elaborate cookies with suitable code snippet.
   d) What is cross page posting? Explain with an example.
   e) Explain master page with its uses and working
   f) What is theme? Explain Global theme.

4. **Attempt *any three* of the following :** (15)
   a) Explain the SQL Data Provider Model.
   b) Give details about DataReader with example.
   c) What is Data Binding? Explain its types.
   d) Write short note on Data Source Controls.
   e) Explain the ways of formatting the GridView Data for display.
   f) Write short note on selecting a GridView Row.

5. **Attempt *any three* of the following :** (15)

a) Write short note on the XML Text Reader class.

b) Explain the reading process from an XML Document with example.

c) Define Authentication and Authorization. Give types of Authentication.

d) What is Windows Authentication? Give details.

e) Explain AJAX with its advantages and disadvantages.

f) Give brief information on Accordion control with appropriate properties.