Deccan Education Society's

Navinchandra Mehta Institute of Technology and Development

C E R T I F I C A T E

This is to certify that Mr. **Nishant Deepak Desai** of M.C.A. Semester II with Roll No.**C23029** has completed All practical's of **Advanced Web Technology** under supervision of **Mr. Ganesh Bhagwat** in this college during the year 2023 -2024.

CO	R1 (Attendance)	R2 (Performance during lab session)	R3 (Innovation in problem solving technique)	R4 (Mock Viva)	R5 (Variation in implementation of learnt topics on projects)
CO1					
CO2					
CO3					
CO4					

Practical-in-charge **External Examiner** **Head of Department**

MCA Department
(NMITD)

MCAL24 Advanced Web Technologies Lab

INDEX

Name of the faculty: Ganesh Bhagwat

Name of the experiment	Date	CO	Sign
Experiment Number 1: C# 1. C# application to read and display user information. 2. C# application to demonstrate the use of various numeric data types. 3. C# application to Calculate addition of two numbers. 4. C# application to demonstrate String functions & properties. 5. C# application to display date in various formats. 6. C# application to Display numbers from 1 to 20 except multiples of 3. 7. C# application to create a class Rectangle and write public function (Area) which returns the area of rectangle. 8. C# application which SWAPs two numbers 9. C# application with Employee class with some public properties 10. C# application to create a class Student to demonstrate the use of parametrized constructors. 11. C# application to make square a partial class. 12. C# application to derive Employee class from Person class. 13. C# application to demonstrate the use of Interfaces.		CO1	
Experiment Number 2: ASP.NET 1. Web Server Controls 2. Validation Controls 3. Design an e-Commerce Website using Master Page, Theme and Skins. 4. Use AJAX controls to change Banner ADs after every second, make use of Ad rotator Control.		CO1 CO2	

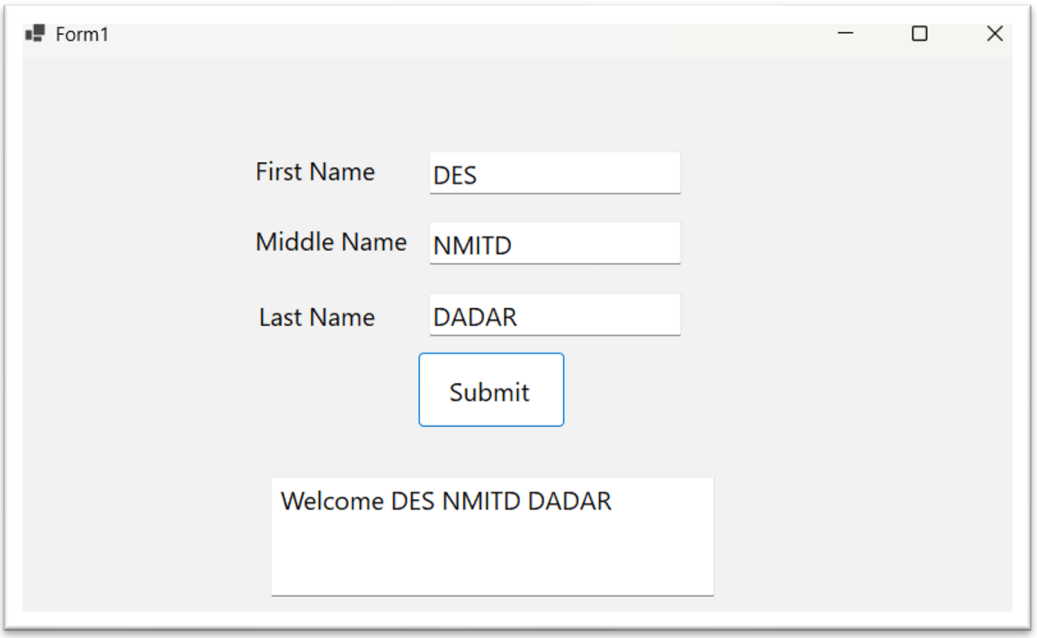
Experiment Number 3: ADO.NET 1. Design a webpage to demonstrate a connection-oriented architecture. 2. Design a webpage to demonstrate a disconnected architecture. 3. Create a webpage that demonstrates the use of data bound controls of ASP.NET. 4. Design a webpage to demonstrate the working of a simple stored procedure. 5. Design a webpage to demonstrate the working of parameterized stored procedure.		**CO2**	
Experiment Number 4: STATE MANAGEMENT 1. View State Example 2. Cookies with multiple keys 3. Login Application with Cookies 4. Login Application with Session 5. Out-of-process Session State 6. Application State		**CO3**	
Experiment Number 5: WEB SERVICES AND WCF 1. Create an application to show the use of web services. 2. Create an application to show the use of WCF.		**CO4**	
Experiment Number 6: MVC and LINQ 1. Create an MVC based application. 2. Create an application to show the use of LINQ		**CO4**	
Assignment Number 1		**CO1** **CO2**	
Assignment Number 2		**CO3**	
Assignment Number 3		**CO4**	

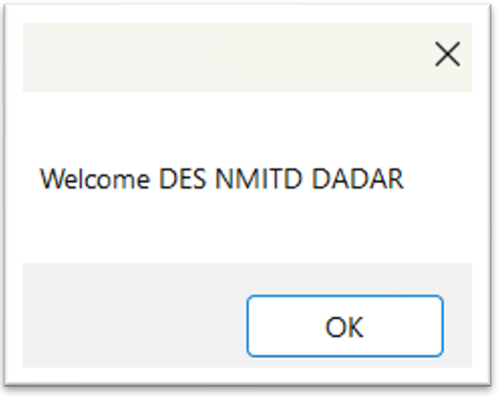
Experiment Number 1: C#

1. C# application to read and display user information.

```
namespace WinFormsApp2
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void label1_Click(object sender, EventArgs e)
      {

      }

      private void button1_Click(object sender, EventArgs e)
      {

         MessageBox.Show("Welcome " + textBox1.Text + " " + textBox2.Text + " "
        + textBox3.Text);
         textBox4.Text = "Welcome " + textBox1.Text + " " + textBox2.Text + " "
        + textBox3.Text;

      }

      private void label3_Click(object sender, EventArgs e)
      {

      }
   }
}
```

DESIGN

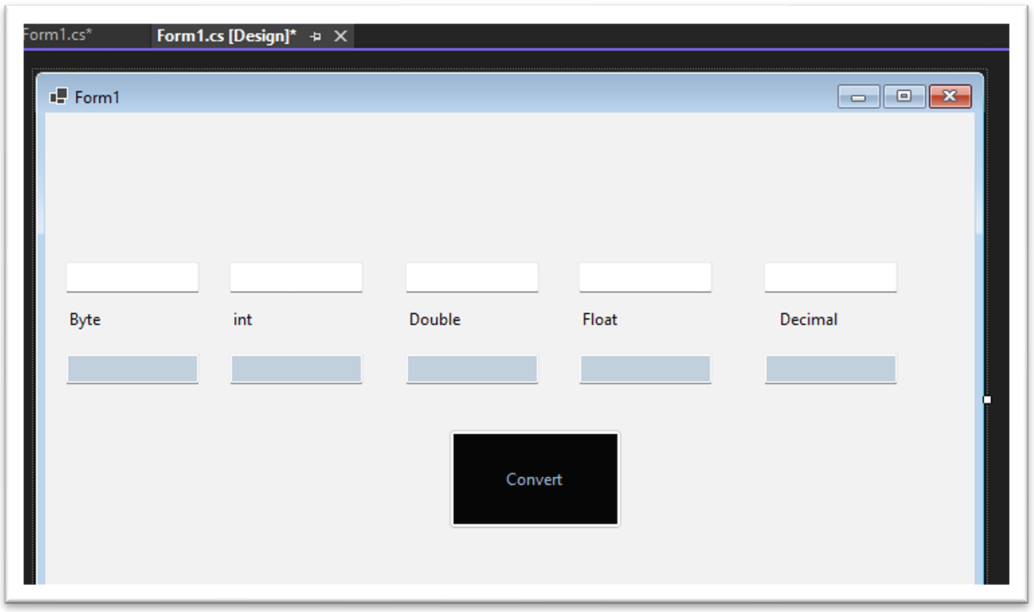

Form1
First Name
DES
Middle Name
NMITD
Last Name
DADAR
Submit
Welcome DES NMITD DADAR


Welcome DES NMITD DADAR
OK

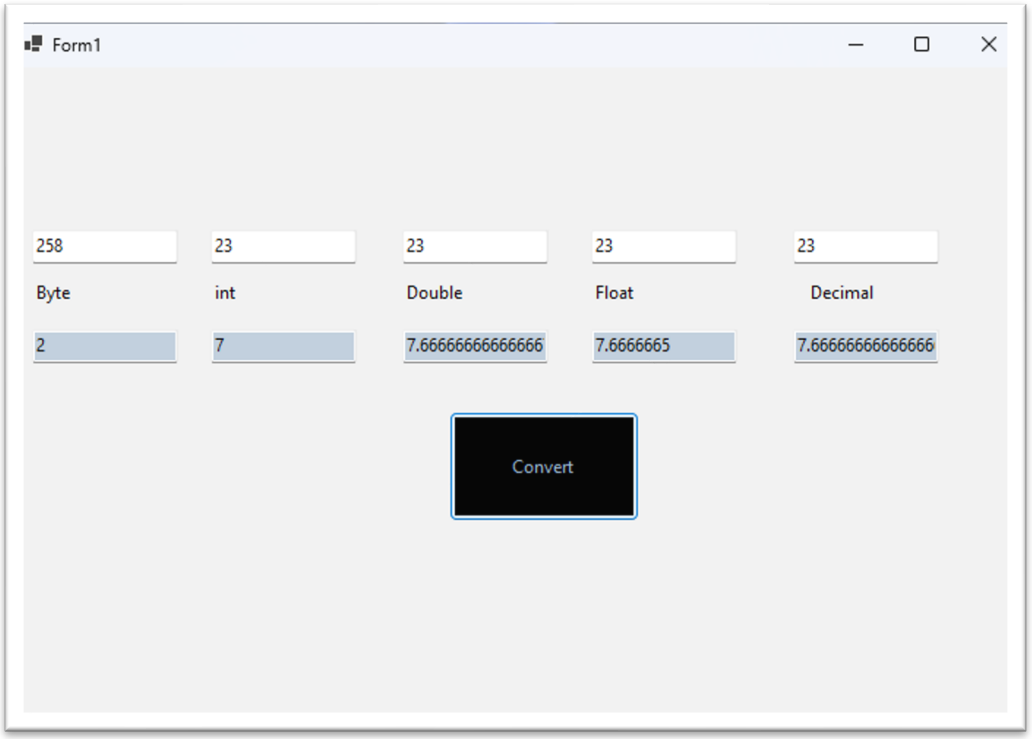
2. C# application to demonstrate the use of various numeric data types.

```
namespace WinFormsApp3
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click_1(object sender, EventArgs e)
      {
         if (Convert.ToInt16(textBox1.Text) > 255)
         {
            textBox10.Text =
           ((byte)(Convert.ToInt16(textBox1.Text))).ToString();
         }
         else
         {
            textBox10.Text = (Convert.ToByte(textBox1.Text)).ToString();
         }
         textBox9.Text = (Convert.ToInt32(textBox2.Text) / 3).ToString();
         textBox8.Text = (Convert.ToDouble(textBox3.Text) / 3).ToString();
         textBox7.Text = (float.Parse(textBox4.Text) / 3).ToString();
         textBox6.Text = (Convert.ToDecimal(textBox5.Text) /
3).ToString();
      }

   }
}
```

DESIGN



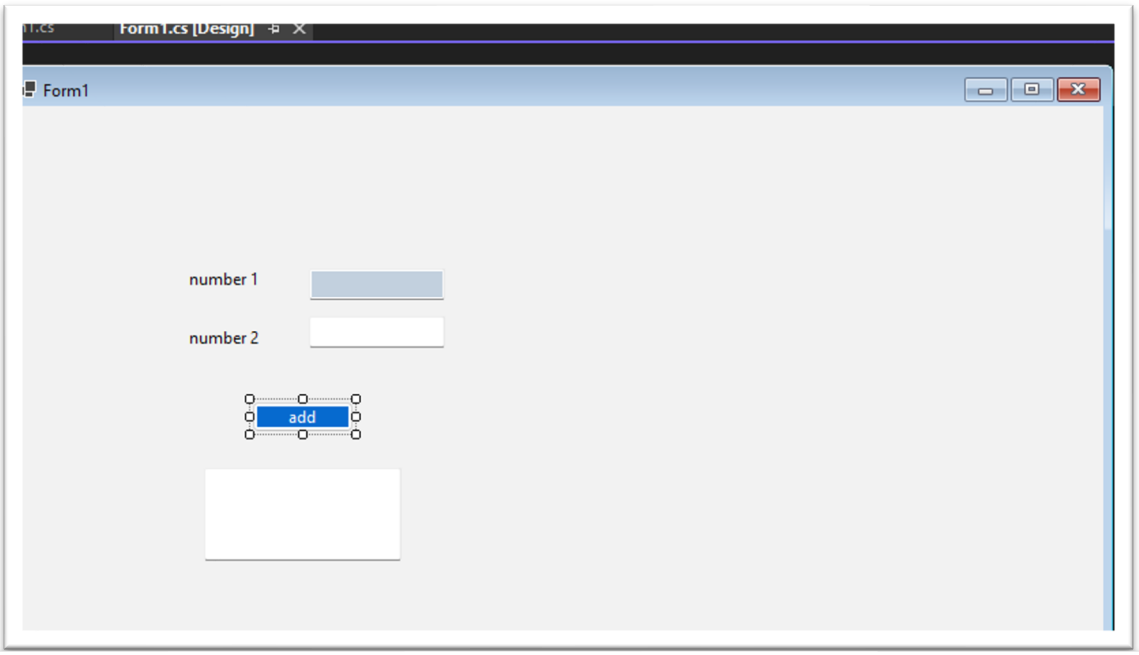
OUTPUT

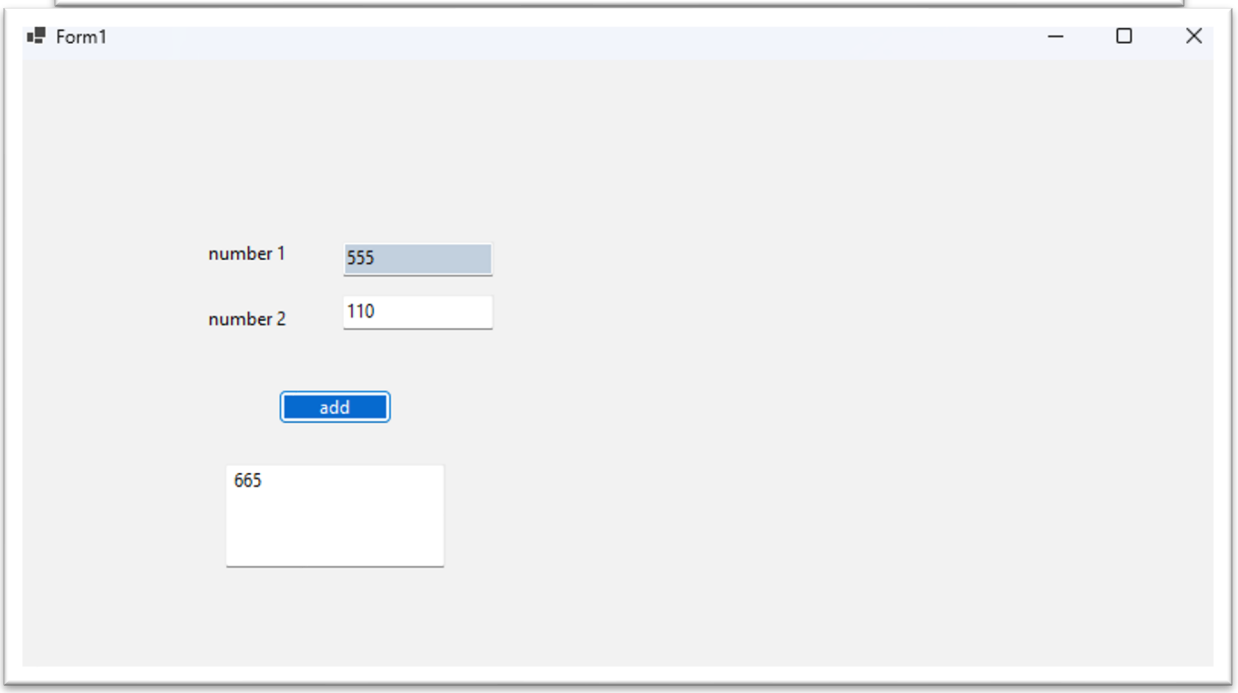


3. C# application to Calculate addition of two numbers.

```
namespace WinFormsApp10
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void Form1_Load(object sender, EventArgs e)
      {

      }

      private void textBox3_TextChanged(object sender, EventArgs e)
      {

      }

      private void button1_Click(object sender, EventArgs e)
      {
         int a = (Convert.ToInt16(textBox1.Text));
         int b = (Convert.ToInt16(textBox2.Text));
         textBox3.Text = (a + b).ToString();

      }
   }
}
```

DESIGN


Form1.cs [Design]
Form1
number 1
number 2
add

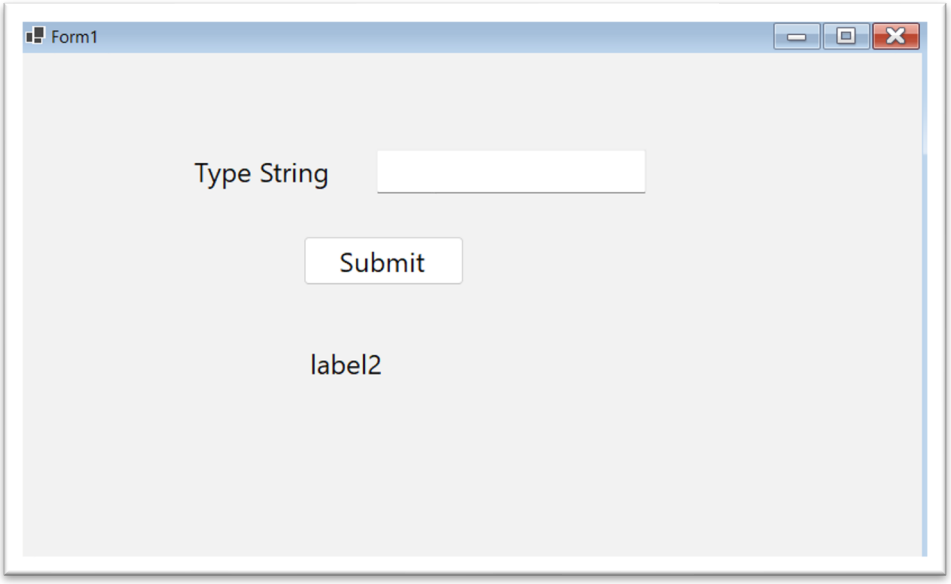

Form1
number 1
555
number 2
110
add
665

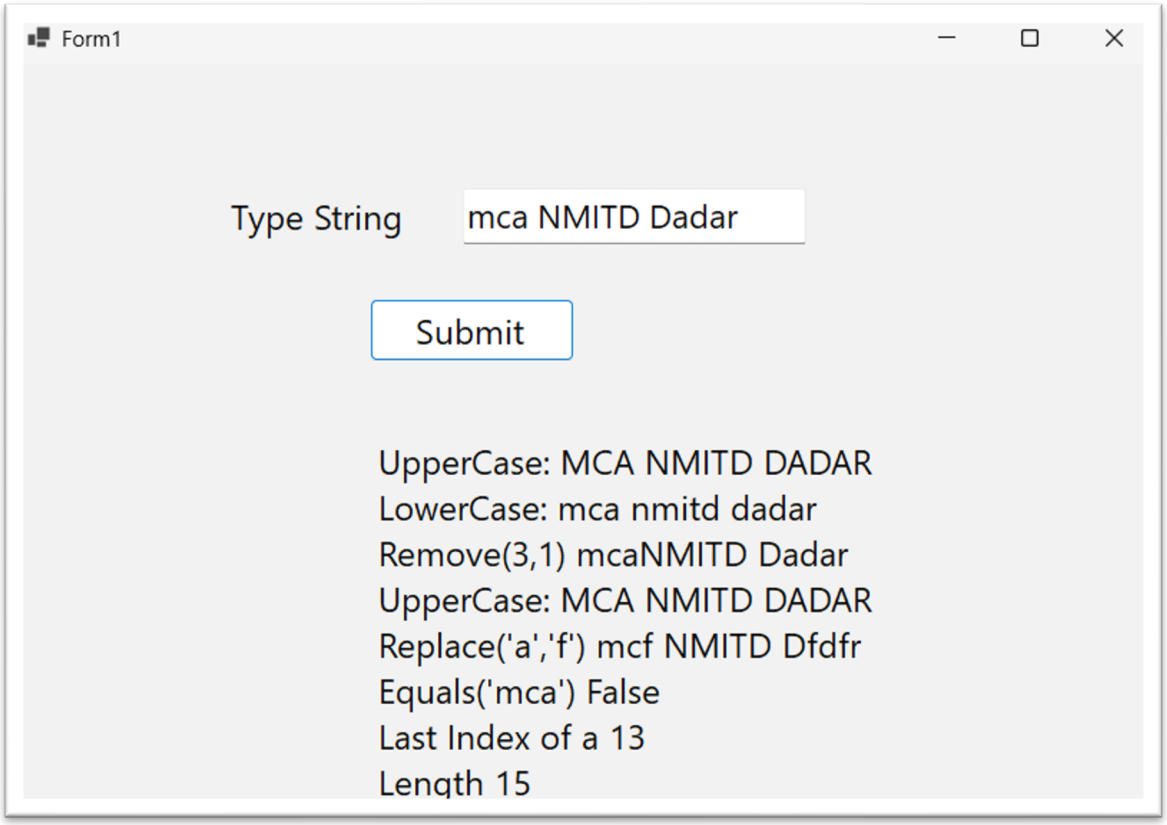
4. C# application to demonstrate String functions & properties.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp11
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void Form1_Load(object sender, EventArgs e)
      {

      }

      private void button1_Click(object sender, EventArgs e)
      {
         string name = textBox1.Text;
         string str2 = @"C:\Program Files\Documents\music.jpg";
         label2.Text = "UpperCase: " + name.ToUpper() +
Environment.NewLine +
         "LowerCase: " + name.ToLower() + Environment.NewLine +
```

```
            "Remove(3,1) " + name.Remove(3, 1) + Environment.NewLine +
            "UpperCase: " + name.ToUpper() + Environment.NewLine +
            "Replace('a','f') " + name.Replace('a', 'f') + Environment.NewLine+
            "Equals('mca') " + name.Equals("mca") + Environment.NewLine +
            "Last Index of a " + name.LastIndexOf('a') +
Environment.NewLine +
             "Length " + name.Length + Environment.NewLine +
str2.ToString();
        }
    }
}
```

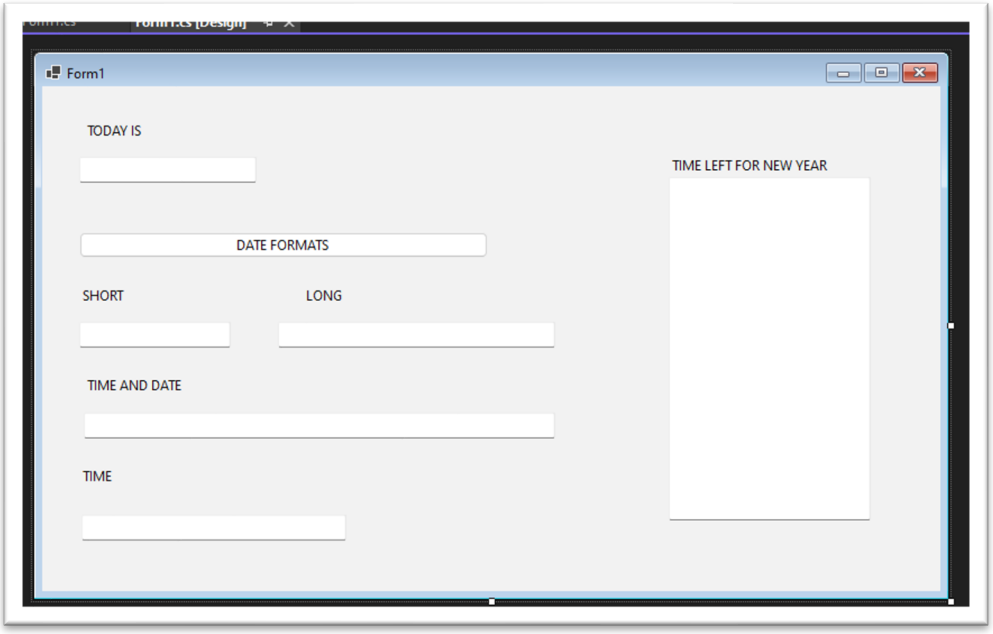
DESIGN



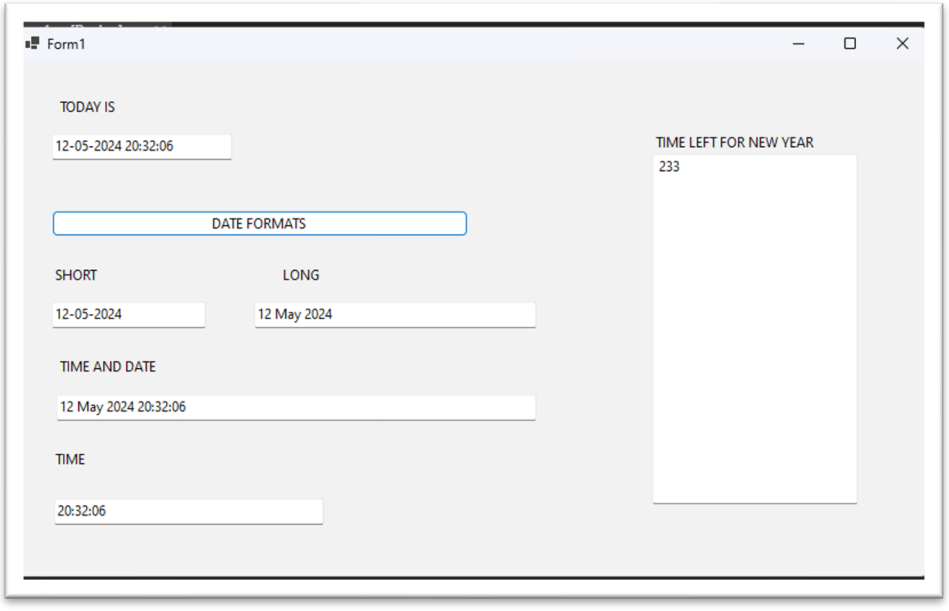

Form1
Type String
mca NMITD Dadar
Submit
UpperCase: MCA NMITD DADAR
LowerCase: mca nmitd dadar
Remove(3,1) mcaNMITD Dadar
UpperCase: MCA NMITD DADAR
Replace('a','f') mcf NMITD Dfdfr
Equals('mca') False
Last Index of a 13
Length 15

5. C# application to display date in various formats.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp7
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }
      private void Form1_Load(object sender, EventArgs e)
      {
         int days;
         DateTime d = new DateTime();
         d = DateTime.Now;
         DateTime newyear = new DateTime(d.Year, 12, 31);
         textBox1.Text = d.ToString(); textBox2.Text =
         d.ToShortDateString(); textBox3.Text =
         d.ToString("D"); textBox4.Text =
         d.ToString("F"); textBox5.Text =
         d.ToLongTimeString();
         days = newyear.DayOfYear - d.DayOfYear;
         textBox6.Text = days.ToString();
      }
   }
}
```

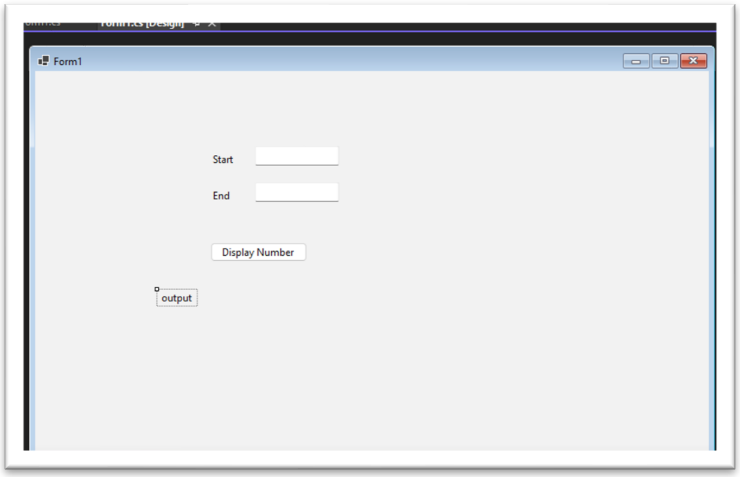

Form1
TODAY IS
TIME LEFT FOR NEW YEAR
DATE FORMATS
SHORT
LONG
TIME AND DATE
TIME

OUTPUT


Form1
TODAY IS
12-05-2024 20:32:06
TIME LEFT FOR NEW YEAR
233
DATE FORMATS
SHORT
LONG
12-05-2024
12 May 2024
TIME AND DATE
12 May 2024 20:32:06
TIME
20:32:06

6. C# application to Display numbers from 1 to 20 except multiples of 3.

DESIGN

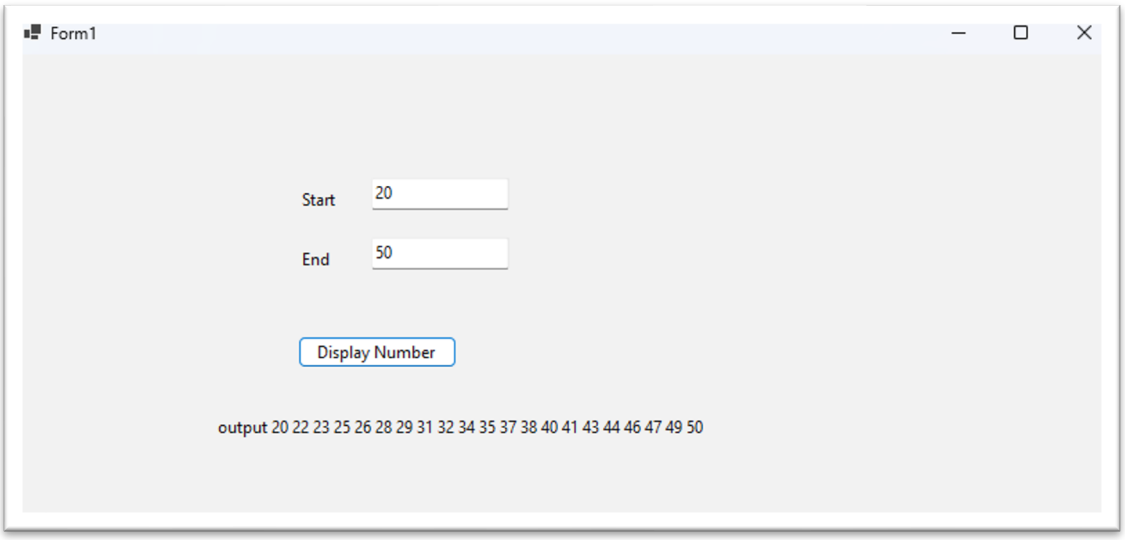


Code

```
namespace WinFormsApp14
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void label1_Click(object sender, EventArgs e)
      {

      }

      private void Form1_Load(object sender, EventArgs e)
      {

      }

      private void button1_Click(object sender, EventArgs e)
```

```
        {
            int start = Convert.ToInt16(textBox1.Text);
            int end = Convert.ToInt16(textBox2.Text);
            for (int i = start; i <= end; i++)
            {
                if (i % 3!=0)
                    {
                    label3.Text+= " "+ i.ToString();
                }

            }
        }
    }
}
```

Output



7. C# application to create a class Rectangle and write public function (Area) which returns the area of rectangle.

Rectangle.cs[Class]

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

namespace WinFormsApp1
{
   internal class Rectangle
   {
      public double Width { get; set; }
      public double Height { get; set; }

      // Constructor to initialize the rectangle dimensions
      public Rectangle(double width, double height)
      {
         Width = width;
         Height = height;
      }

      // Method to calculate the area of the rectangle
      public double Area()
      {
         return Width * Height;
      }
   }
}
```

Form1.cs

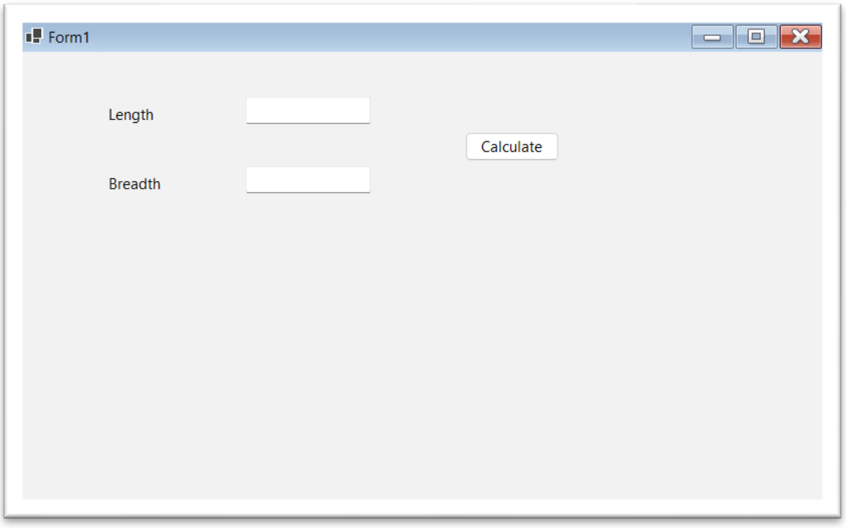
Code:

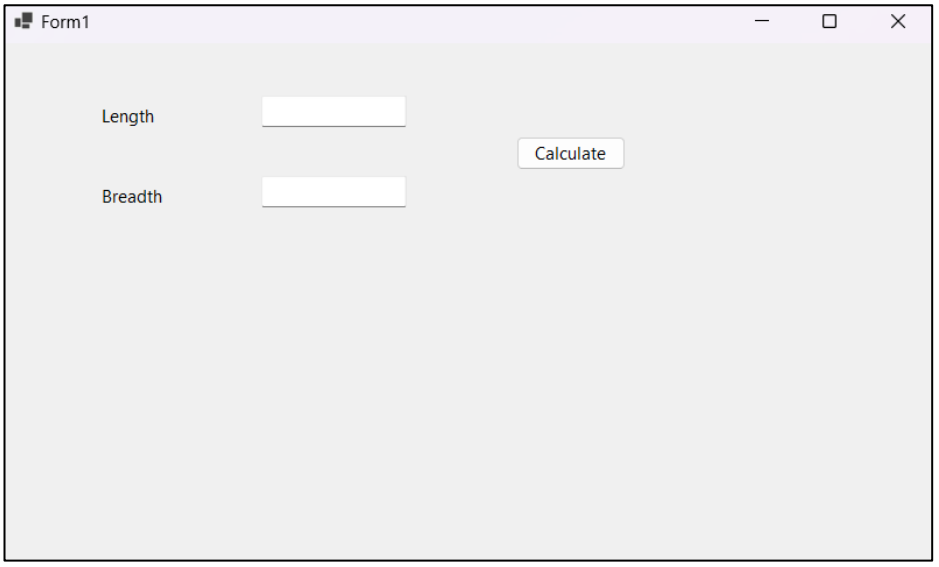
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
```

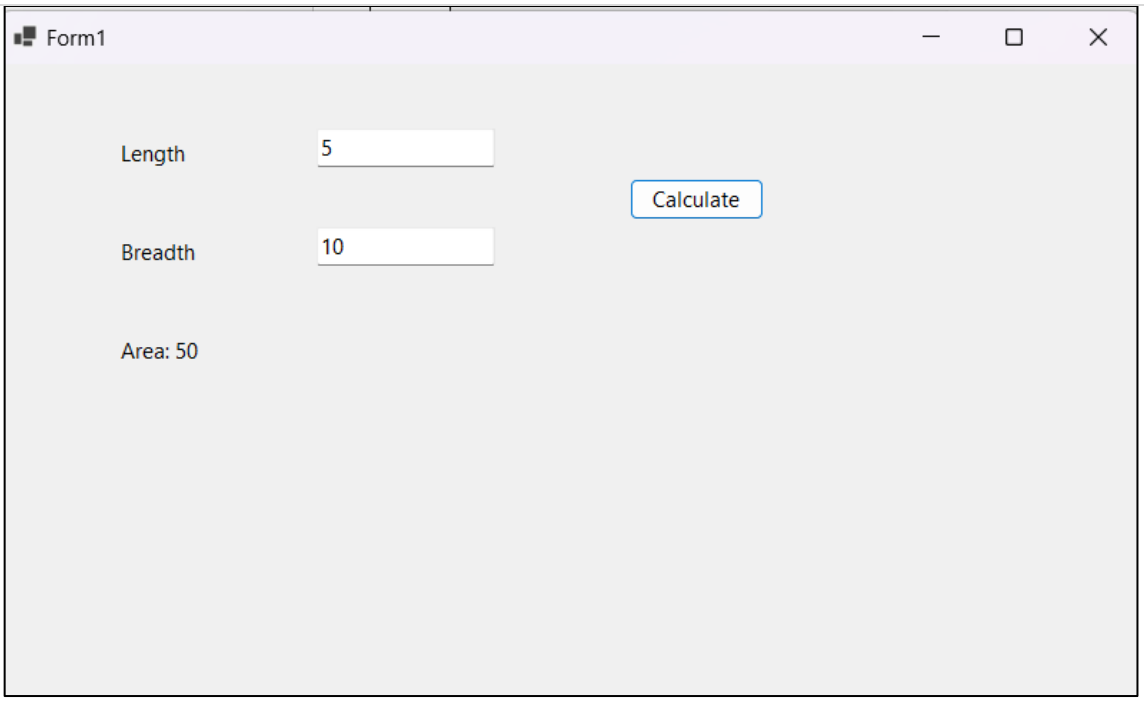
```
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp1
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }




      private void label1_Click_1(object sender, EventArgs e)
      {

      }

      private void button1_Click_1(object sender, EventArgs e)
      {
         double length = double.Parse(textBox1.Text);
         double breadth = double.Parse(textBox2.Text);

         // Create a Rectangle object
         Rectangle rectangle = new Rectangle(length, breadth);

         // Calculate the area
         double area = rectangle.Area();

         // Display the area in a label
```

```
            label3.Text = "Area: " + area.ToString();
        }
    }
}
```

designer



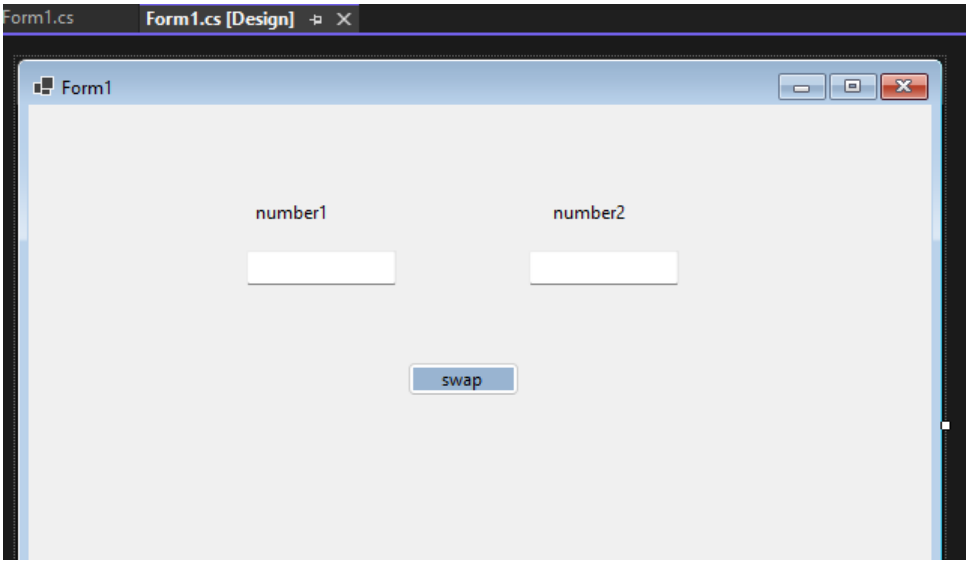



Form1
Length
5
Calculate
Breadth
10
Area: 50

8. C# application which SWAPs two numbers

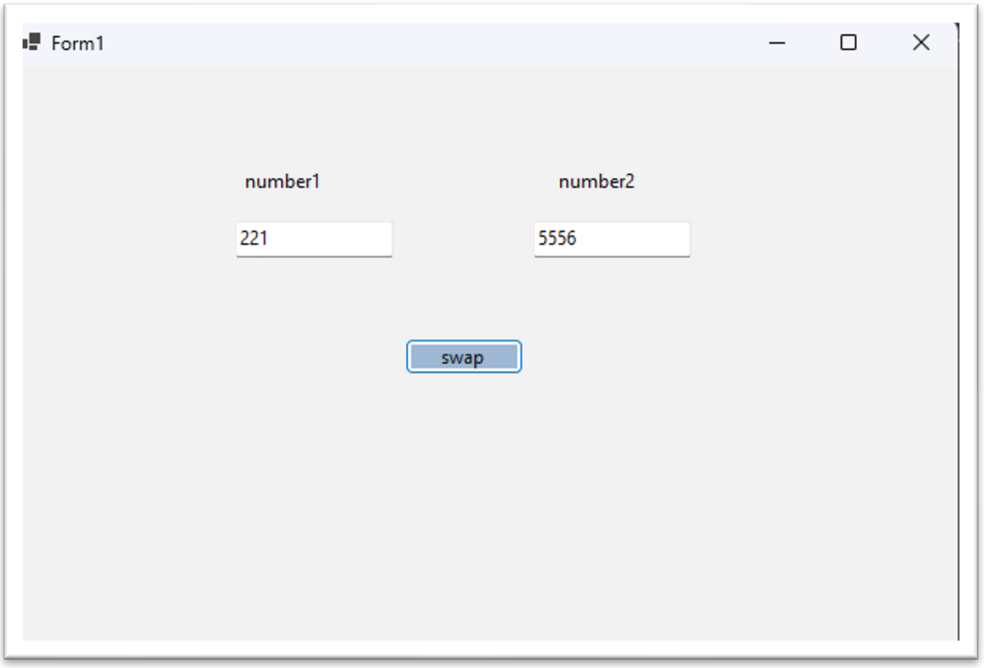
```
namespace WinFormsApp15
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         int num1 = int.Parse(textBox1.Text);
         int num2 = int.Parse(textBox2.Text);

         (num1, num2) = (num2, num1);

         textBox1.Text = num1.ToString();
         textBox2.Text = num2.ToString();
      }
   }
}
```

DESIGN

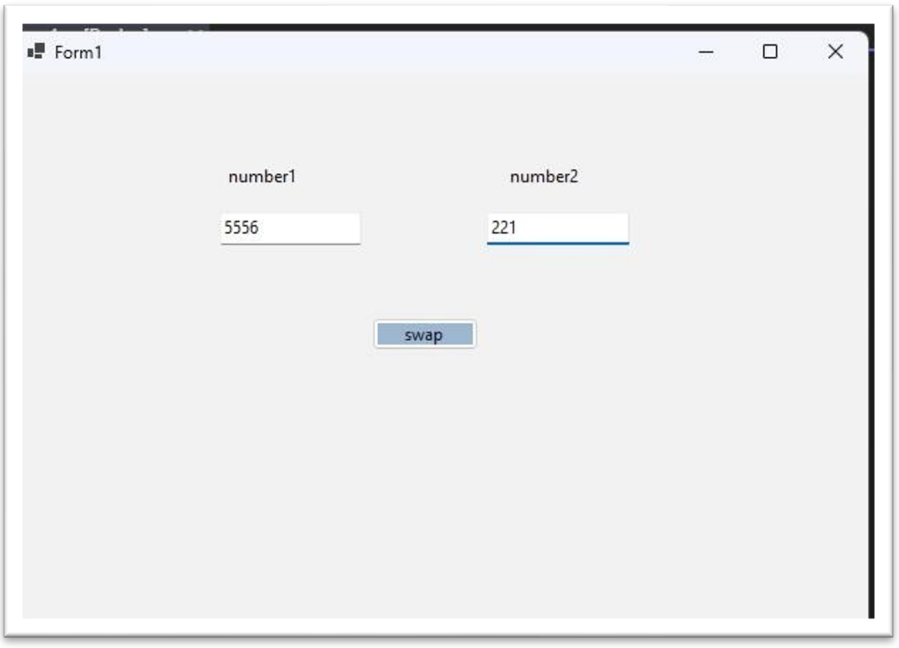


OUTPUT

Before swapping



After swapping

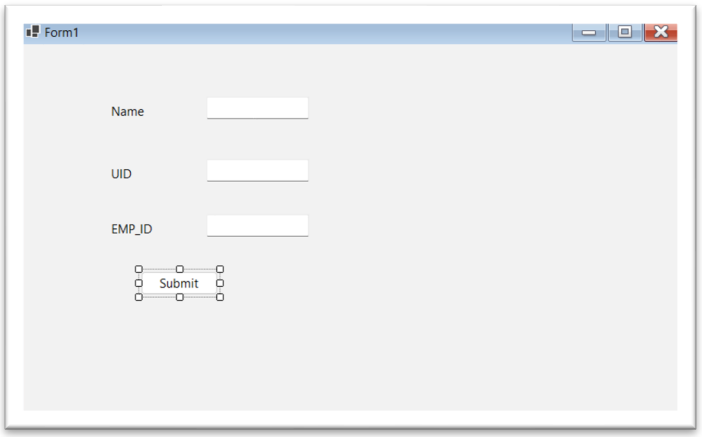


9. C# application with Employee class with some public properties.

Form1.cs

Code:

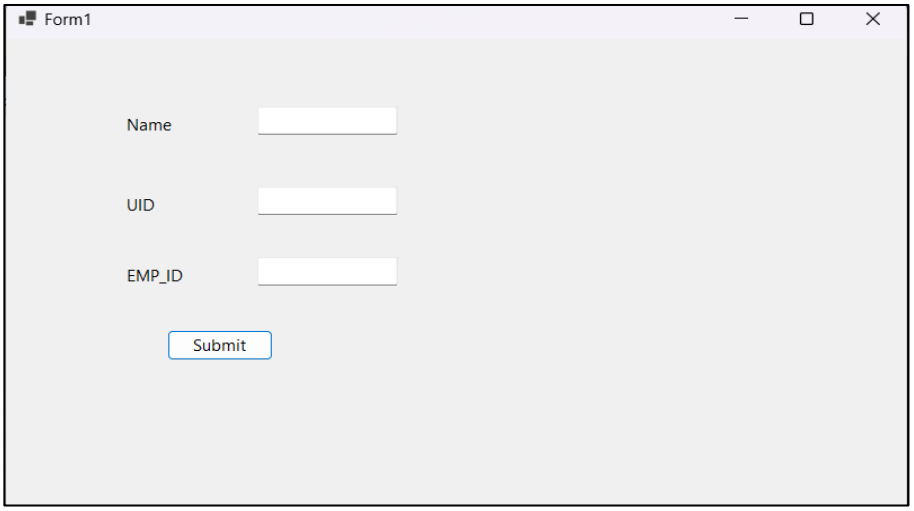
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp1
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         Person P = new Person(textBox1.Text, textBox2.Text);
         P.getInfo();
      }
   }
}
```

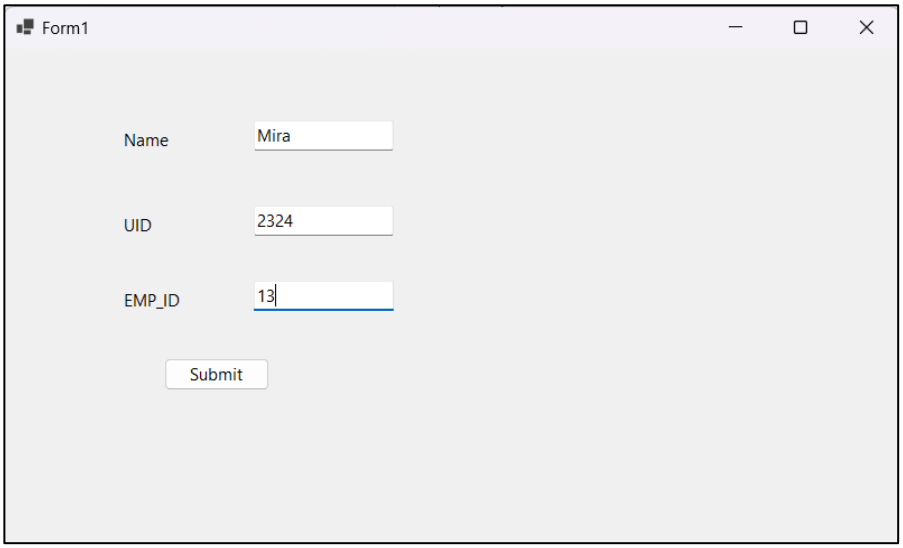


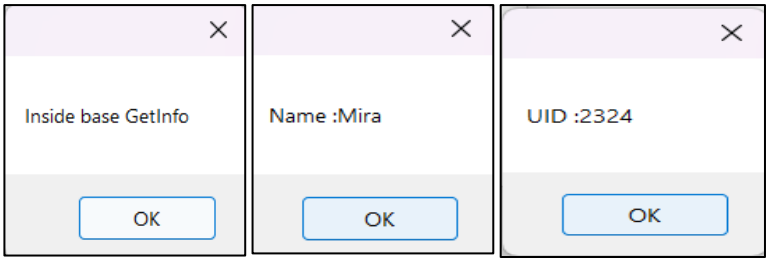
Person.cs[Class]

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

namespace WinFormsApp1
{
   internal class Person
   {
      protected string uid; protected string sname;
      public Person(string name, string uid)
      {
         this.sname = name; this.uid = uid;
      }
      public virtual void getInfo()
      {
         System.Windows.Forms.MessageBox.Show("Inside base GetInfo");
         System.Windows.Forms.MessageBox.Show("Name :" + sname);
         System.Windows.Forms.MessageBox.Show("UID :" + uid);
      }
   }
   class empl : Person
   {
      public string eid;
      public empl(string sname, string uid, string eid) : base(sname, uid)
```

```
        {
            this.eid = eid;
        }
        public override void getInfo()
        {
            base.getInfo();
            System.Windows.Forms.MessageBox.Show("Employee id:" + eid);
        }
    }
}
```






Inside base GetInfo
OK
Name :Mira
OK
UID :2324
OK

10. C# application to create a class Student to demonstrate the use of parametrized constructors.

Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp1
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         string sname = textBox1.Text;
         string course = textBox2.Text;

         Student s;
         //s=new Student(sname,course);

         if (course.Equals(" "))
            s = new Student(sname);
         else
```

```
            s = new Student(sname, course);
        s.GetInfo();

    }
  }
}
```

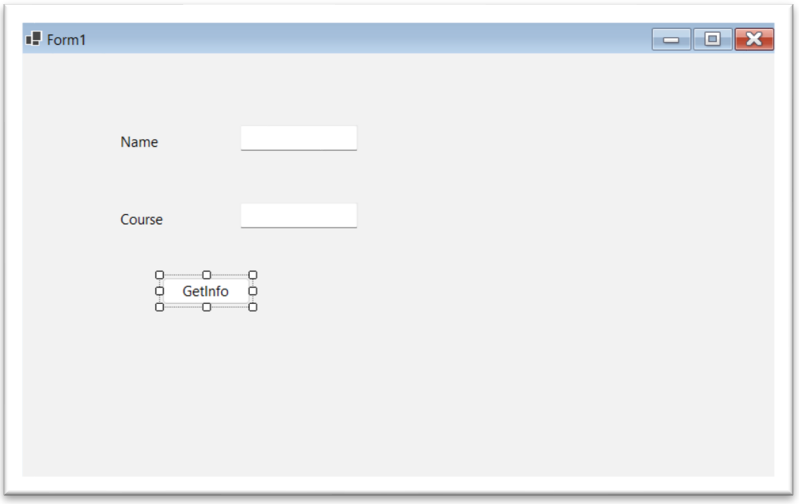
Student.cs[Class]

Code:

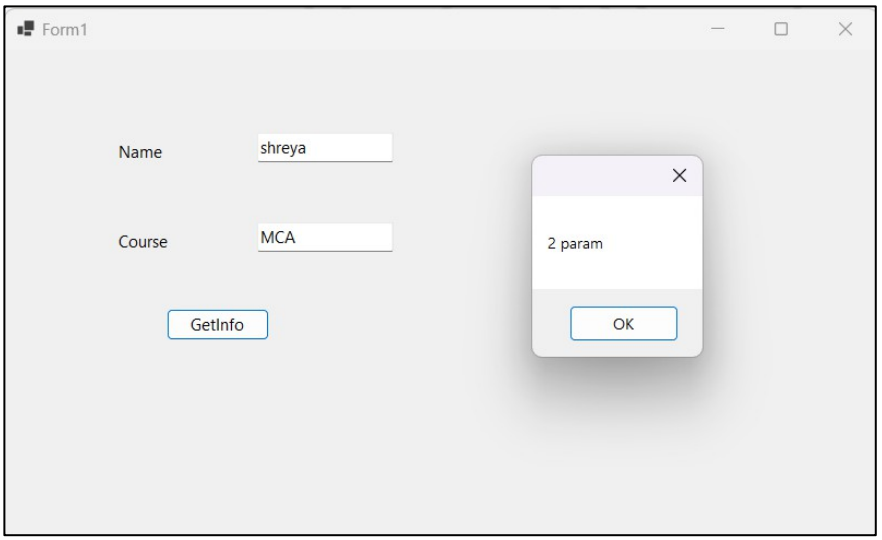
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WinFormsApp1
{
  internal class Student
  {
    public string course;
    public string sname;
    public Student(string sname, string course)
    {
      this.sname = sname;
      this.course = course;
      MessageBox.Show("2 param");
    }
    public Student(string sname) : this(sname, "MCA")
    {
      //this.sname=name;
      //this.course="Mca";
      MessageBox.Show("1 param");
    }
    public void GetInfo()
    {
      MessageBox.Show(sname + " " + course);
    }
```

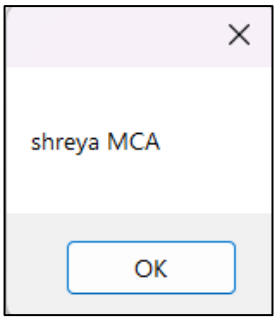
```
    }
}
```

Design



Output


Form1
Name
shreya
Course
MCA
GetInfo
2 param
OK


shreya MCA
OK

11. C# application to make square a partial class.

Form1.cs

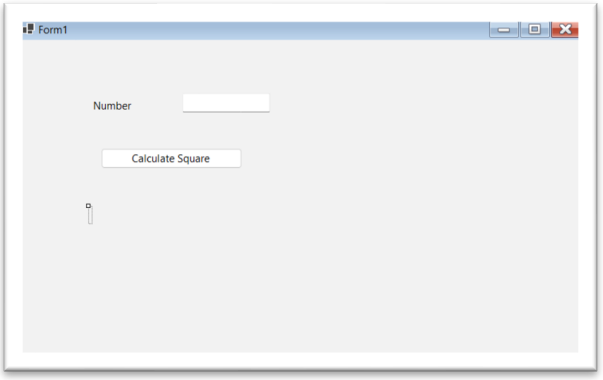
Code:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp1
{
   public partial class Form1 : Form
   {


      public Form1()
      {
         InitializeComponent();

      }

      private void button1_Click(object sender, EventArgs e)
      {
         if (double.TryParse(textBox1.Text, out double number))
         {
            double square = number * number;
            label2.Text = $"Result: {square}";
         }
         else
         {
            MessageBox.Show("Please enter a valid number.");
```

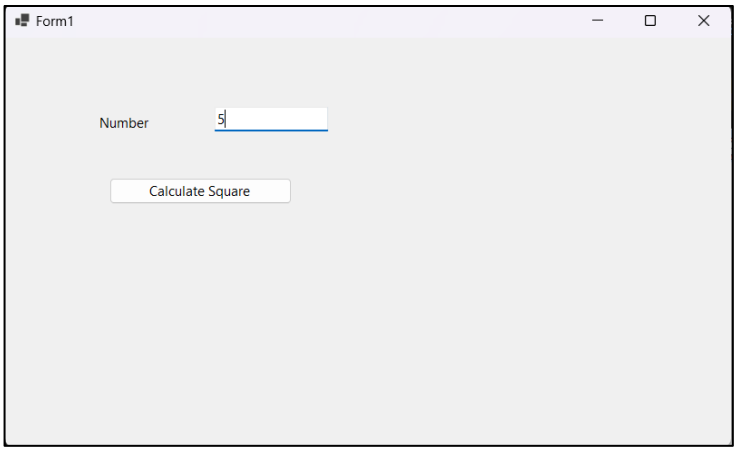
```
        }
```

```
    }
}
}
```

Design



Output



12. C# application to derive Employee class from Person class

Form1.cs

Code:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp1
{
   public partial class Form1 : Form
   {


      public Form1()
      {
         InitializeComponent();

      }

      private void label3_Click(object sender, EventArgs e)
      {

      }



      private void button1_Click_1(object sender, EventArgs e)
      {
         string firstName = textBox1.Text;
         string lastName = textBox2.Text;
```

```
            string employeeID = textBox3.Text;
            string department = textBox4.Text;

            Employee employee = new Employee(firstName, lastName,
employeeID, department);
            listBox1.Items.Add(employee.ToString());

            // Clear the input fields after submission
            textBox1.Clear();
            textBox2.Clear();
            textBox3.Clear();
            textBox4.Clear();
        }
    }
}
```

Employee.cs[Class]

Code:

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

namespace WinFormsApp1
{
    internal class Employee : Person
    {
        public string EmployeeID { get; set; }
        public string Department { get; set; }

        public Employee(string firstName, string lastName, string
employeeID, string department)
            : base(firstName,
                  lastName)
        {
```

```
            EmployeeID = employeeID;
            Department = department;
        }

        public override string ToString()
        {
            return $"{base.ToString()}, ID: {EmployeeID}, Department:
{Department}";
        }
    }
}
```

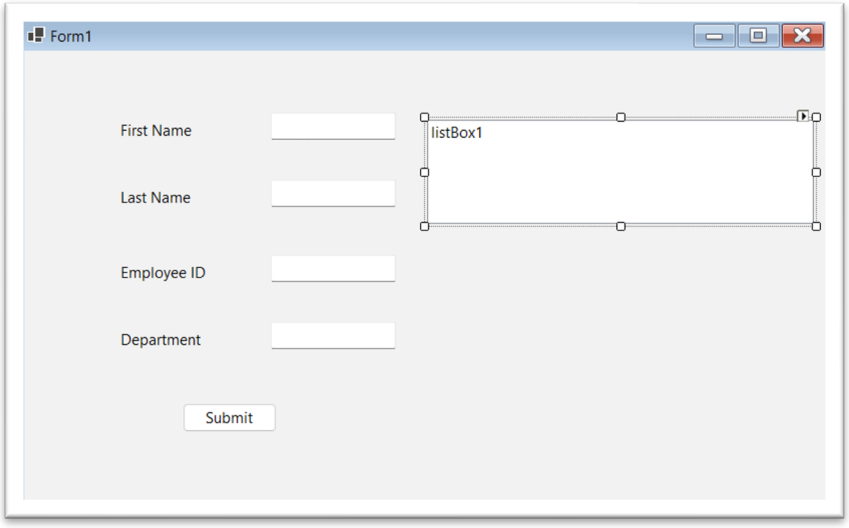
Person.cs[Class]

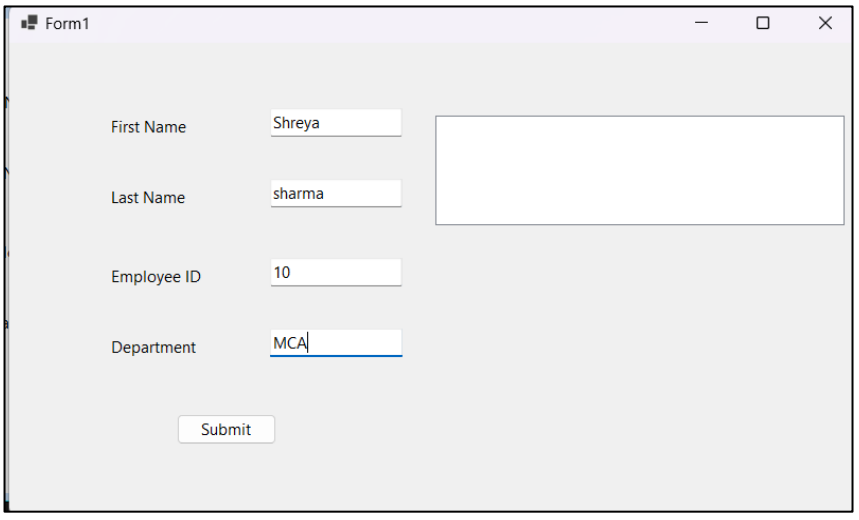
Code:

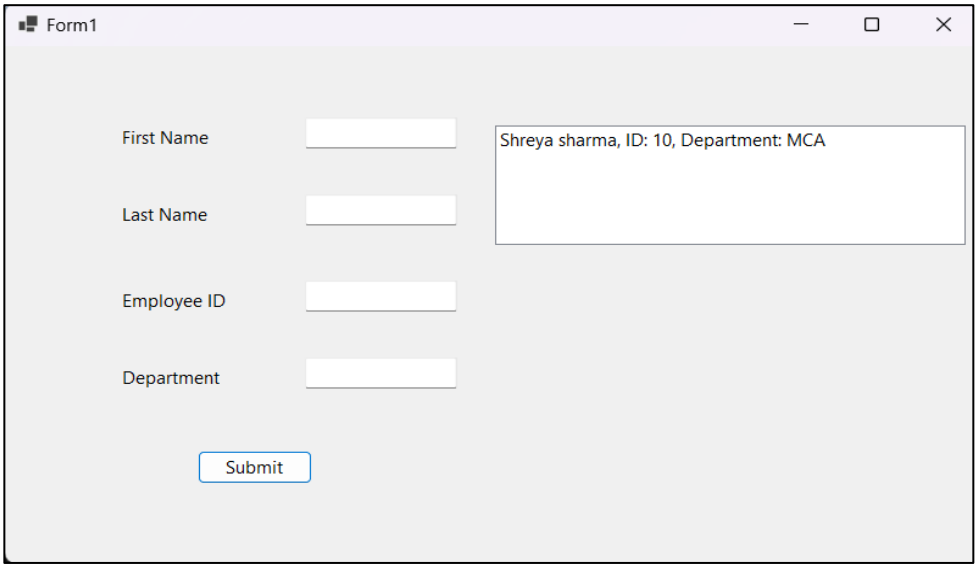
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WinFormsApp1
{
    internal class Person
    {
        public string FirstName { get; set; }
        public string LastName { get; set; }

        public Person(string firstName, string lastName)
        {
            FirstName = firstName;
            LastName = lastName;
        }

        public override string ToString()
        {
            return $"{FirstName} {LastName}";
```

```
        }
    }
}
```

design


Form1
First Name
Last Name
Employee ID
Department
Submit
listBox1


Form1
First Name
Shreya
Last Name
sharma
Employee ID
10
Department
MCA
Submit


Form1
First Name
Last Name
Employee ID
Department
Submit
Shreya sharma, ID: 10, Department: MCA

13. C# application to demonstrate the use of Interfaces.

Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WinFormsApp1
{
   public partial class Form1 : Form
   {
      private ICalculator calculator;


      public Form1()
      {
         InitializeComponent();
         calculator = new SimpleCalculator();

      }

      private void label3_Click(object sender, EventArgs e)
      {

      }



      private void button1_Click_1(object sender, EventArgs e)
      {
```

```
    int a = int.Parse(textBox1.Text);
    int b = int.Parse(textBox2.Text);
    int result = calculator.Add(a, b);
    textBox3.Text = result.ToString();
}

private void label1_Click(object sender, EventArgs e)
{

}

private void button2_Click(object sender, EventArgs e)
{
    int a = int.Parse(textBox1.Text);
    int b = int.Parse(textBox2.Text);
    int result = calculator.Subtract(a, b);
    textBox3.Text = result.ToString();
}

private void button3_Click(object sender, EventArgs e)
{
    int a = int.Parse(textBox1.Text);
    int b = int.Parse(textBox2.Text);
    int result = calculator.Multiply(a, b);
    textBox3.Text = result.ToString();
}

private void button4_Click(object sender, EventArgs e)
{
    int a = int.Parse(textBox1.Text);
    int b = int.Parse(textBox2.Text);
    try
    {
        double result = calculator.Divide(a, b);
        textBox3.Text = result.ToString();
    }
    catch (DivideByZeroException ex)
```

```
            {
                MessageBox.Show(ex.Message);
            }
        }
    }
}
```

SimpleCalculator.cs[Class]

Code:

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

namespace WinFormsApp1
{
    internal class SimpleCalculator : ICalculator
    {
        public int Add(int a, int b)
        {
            return a + b;
        }

        public int Subtract(int a, int b)
        {
            return a - b;
        }

        public int Multiply(int a, int b)
        {
            return a * b;
        }

        public double Divide(int a, int b)
        {
```

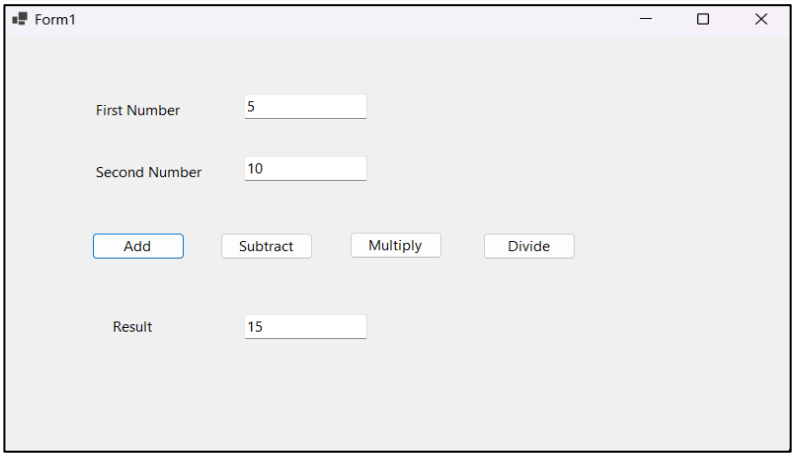
```
            if (b == 0)
            {
                throw new DivideByZeroException("Division by zero is not
allowed.");
            }
            return (double)a / b;
        }
    }
}
```

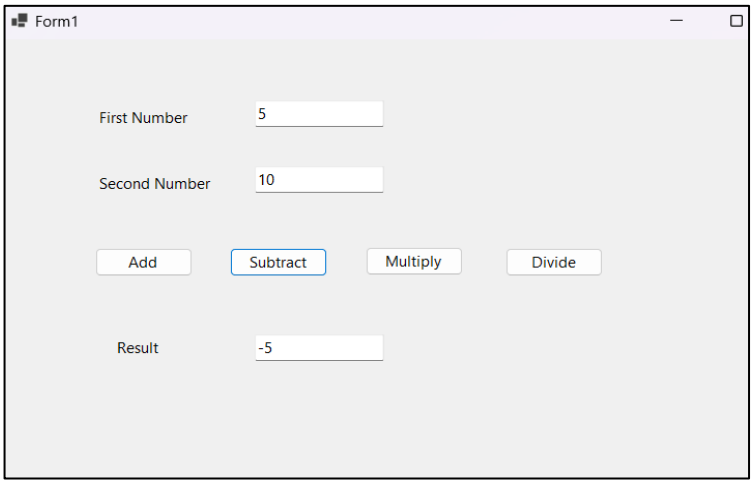
ICalculator.cs[Interface]

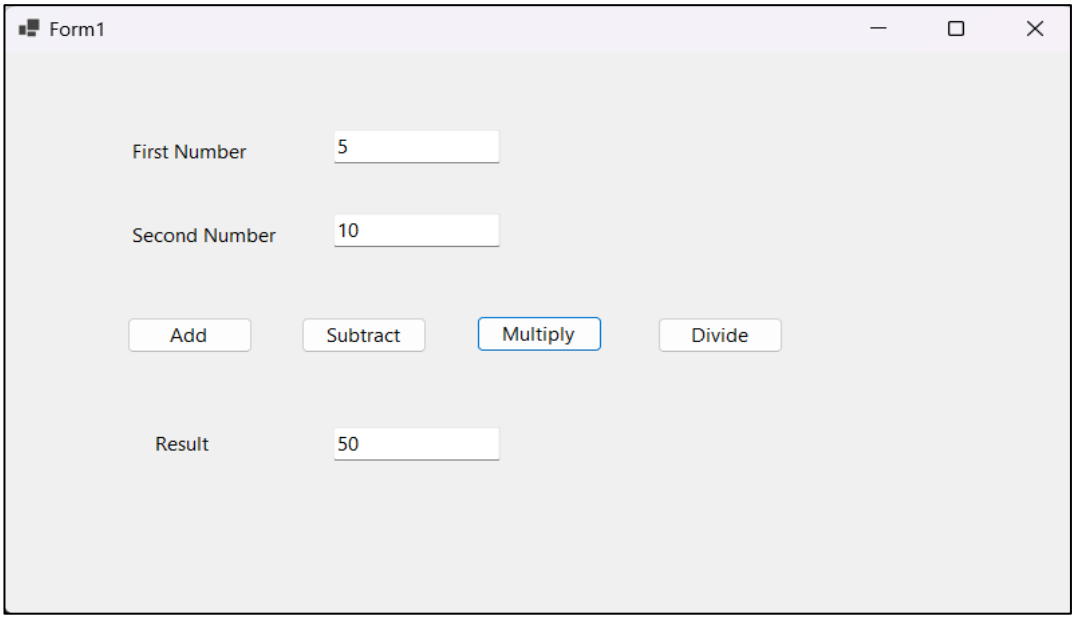
Code:

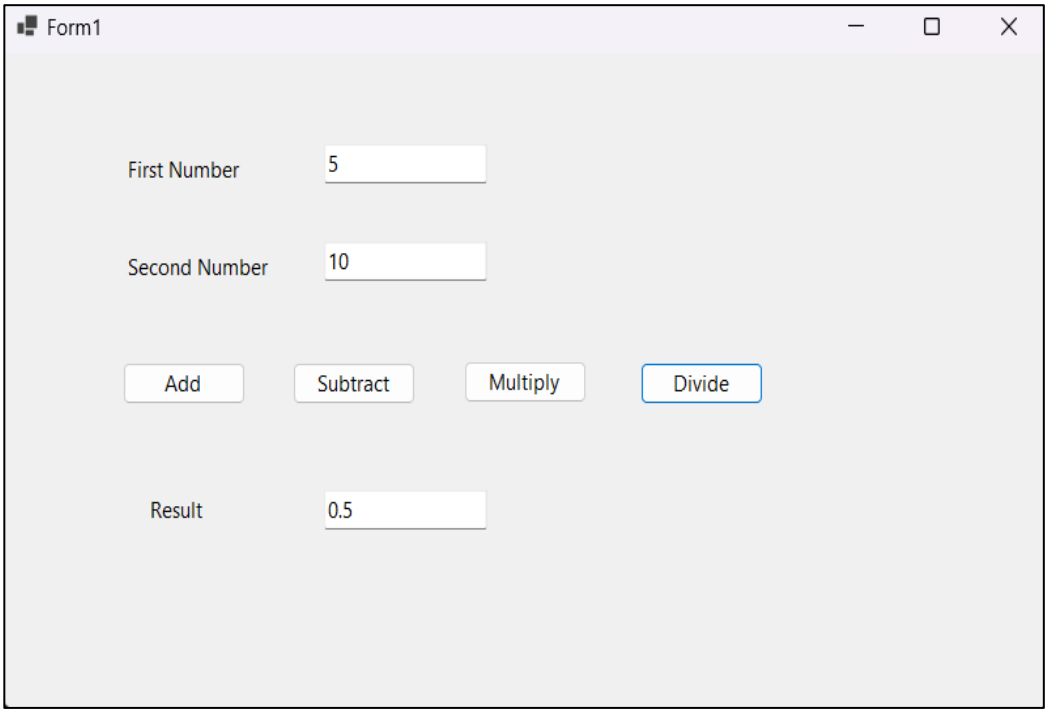
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WinFormsApp1
{
    internal interface ICalculator
    {
        int Add(int a, int b);
        int Subtract(int a, int b);
        int Multiply(int a, int b);
        double Divide(int a, int b);
    }
}
```

DESIGN


Form1
First Number
5
Second Number
10
Add
Subtract
Multiply
Divide
Result
15


Form1
First Number
5
Second Number
10
Add
Subtract
Multiply
Divide
Result
-5


Form1
First Number
5
Second Number
10
Add
Subtract
Multiply
Divide
Result
50


Form1
First Number
5
Second Number
10
Add
Subtract
Multiply
Divide
Result
0.5

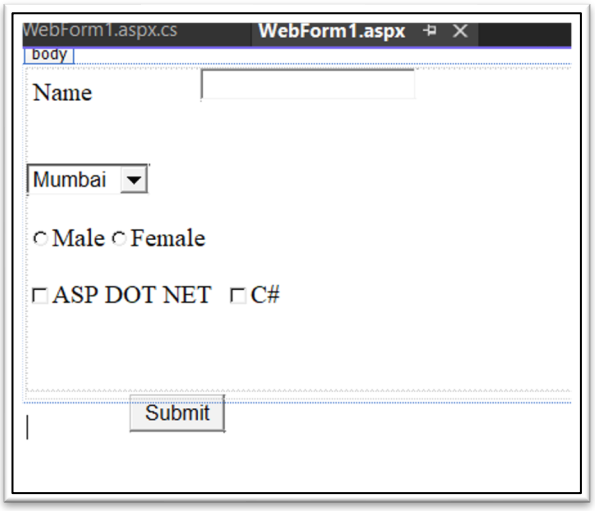
Experiment Number 2: ASP.NET

1. Web Server Controls

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Xml.Linq;

namespace WebApplication3
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         Response.Write(TextBox1.Text + "</br>");
         Response.Write(DropDownList1.SelectedItem.Text + "</br>");

         if (RadioButton1.Checked)
         {
            Response.Write(RadioButton1.Text + "<br>");
         }
         else if (RadioButton2.Checked)
         {
            Response.Write(RadioButton2.Text + "<br>");
         }

         if (CheckBox1.Checked)
         {
            Response.Write(CheckBox1.Text + "</br>");
```

```
            }
            if (CheckBox2.Checked)
            {
               Response.Write(CheckBox2.Text + "</br>");
            }

            TextBox1.Visible = false;
            Label1.Visible = false;
            DropDownList1.Visible = false;
            CheckBox1.Visible = false;
            CheckBox2.Visible = false;
            RadioButton1.Visible = false;
            RadioButton2.Visible = false;
            Button1.Visible = false;
        }
    }
}
```

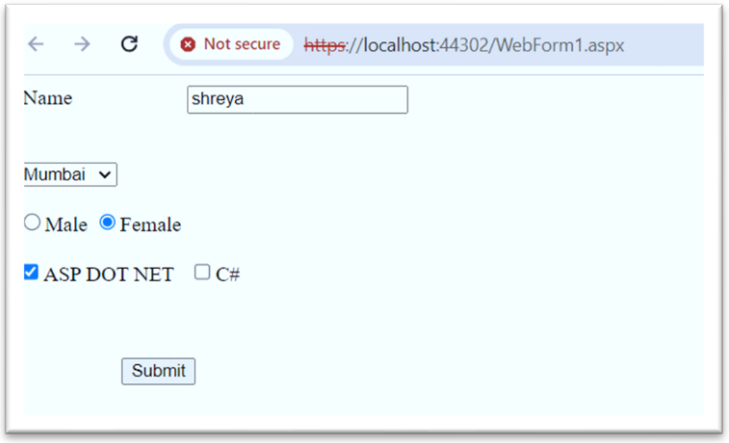
Design

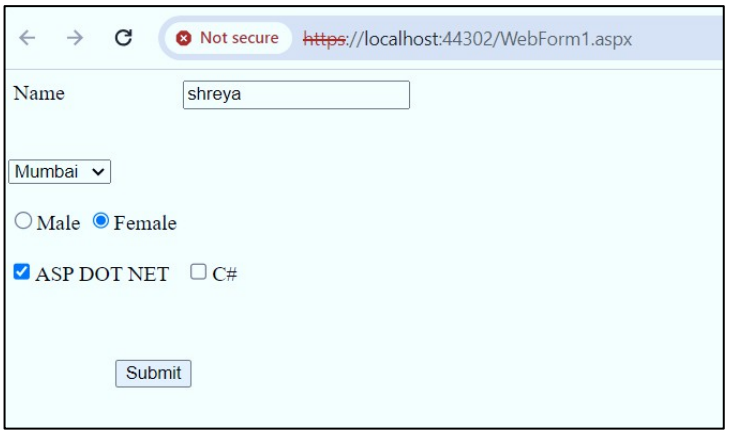



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Xml.Linq;
```

```
namespace WebApplication3
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         Response.Write(TextBox1.Text + "</br>");
         Response.Write(DropDownList1.SelectedItem.Text + "</br>");

         if (RadioButton1.Checked)
         {
            Response.Write(RadioButton1.Text + "<br>");
         }
         else if (RadioButton2.Checked)
         {
            Response.Write(RadioButton2.Text + "<br>");
         }

         if (CheckBox1.Checked)
         {
            Response.Write(CheckBox1.Text + "</br>");
         }
         if (CheckBox2.Checked)
         {
            Response.Write(CheckBox2.Text + "</br>");
         }

         TextBox1.Visible = false;
         Label1.Visible = false;
         DropDownList1.Visible = false;
         CheckBox1.Visible = false;
         CheckBox2.Visible = false;
```

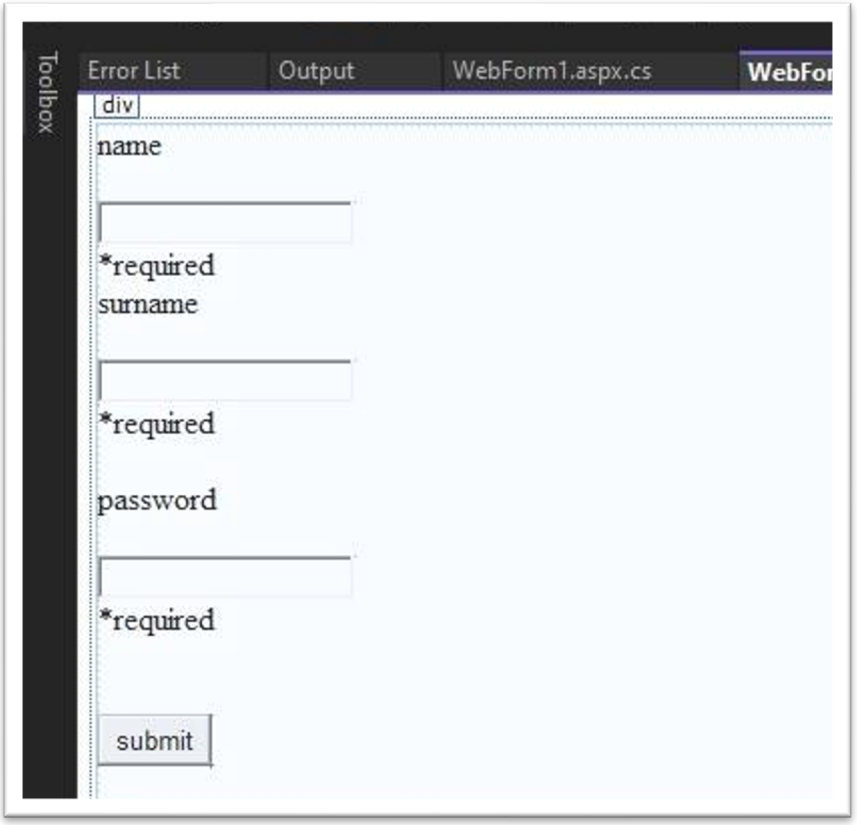
```
            RadioButton1.Visible = false;
            RadioButton2.Visible = false;
            Button1.Visible = false;
        }
    }
}
```



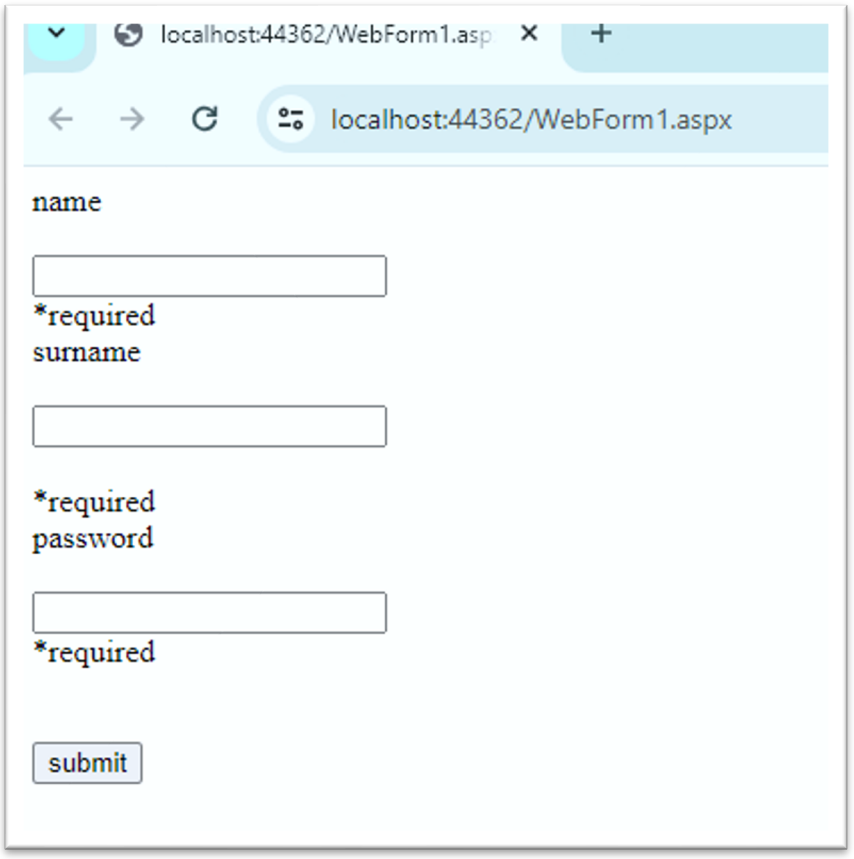


2. Validation Controls

design



If all columns left empty

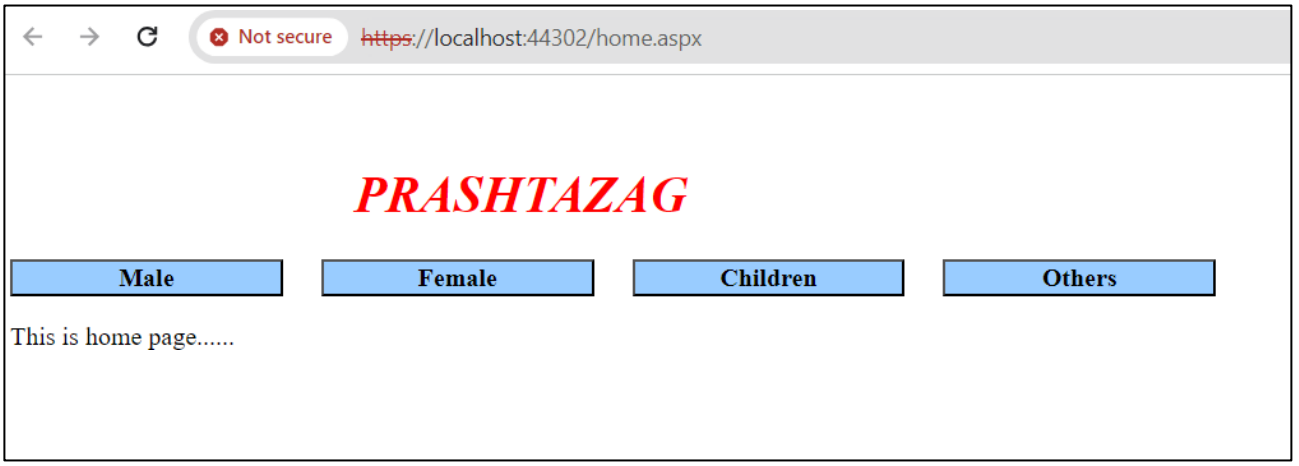


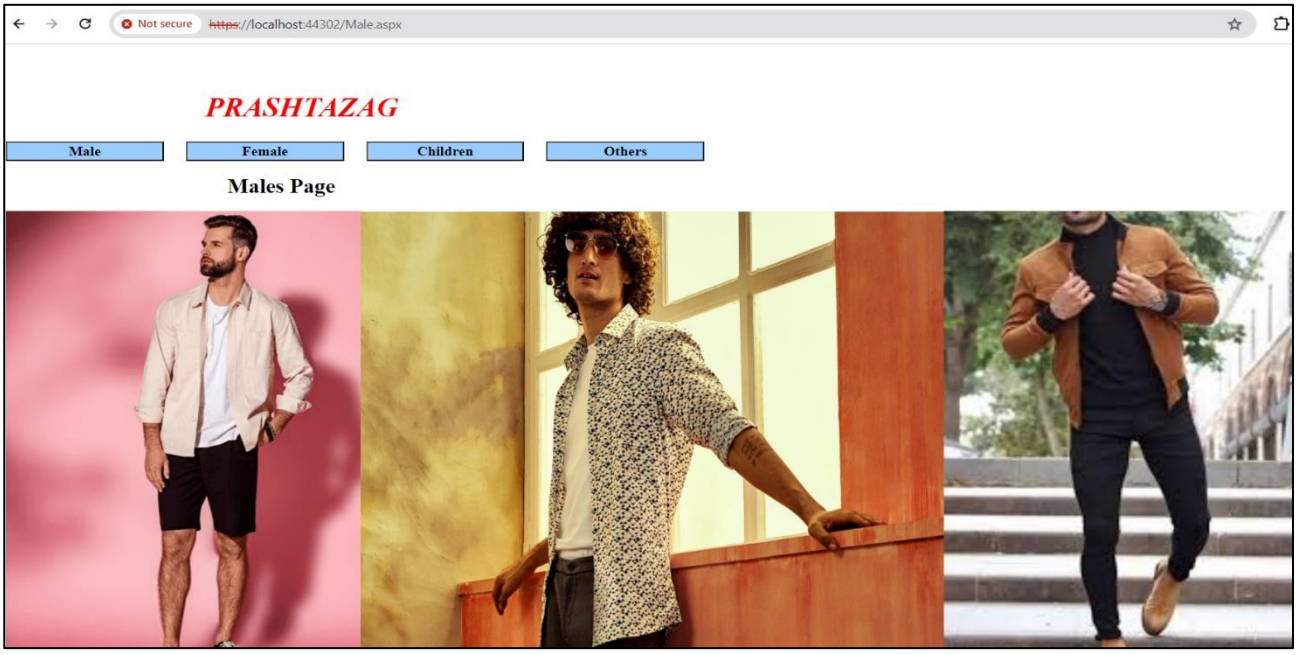
After all columns are field

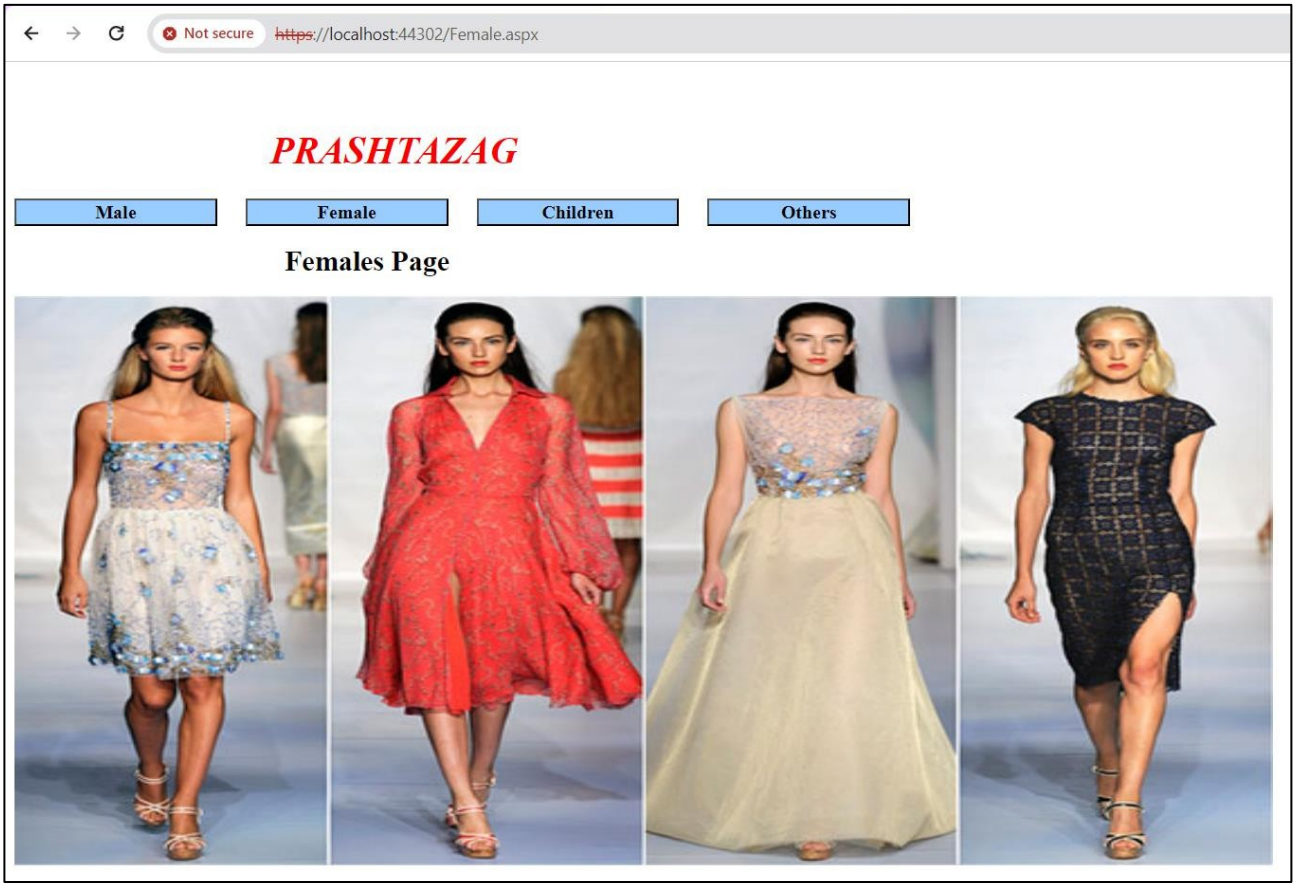
3. Design an e-Commerce Website using Master Page, Theme and Skins.

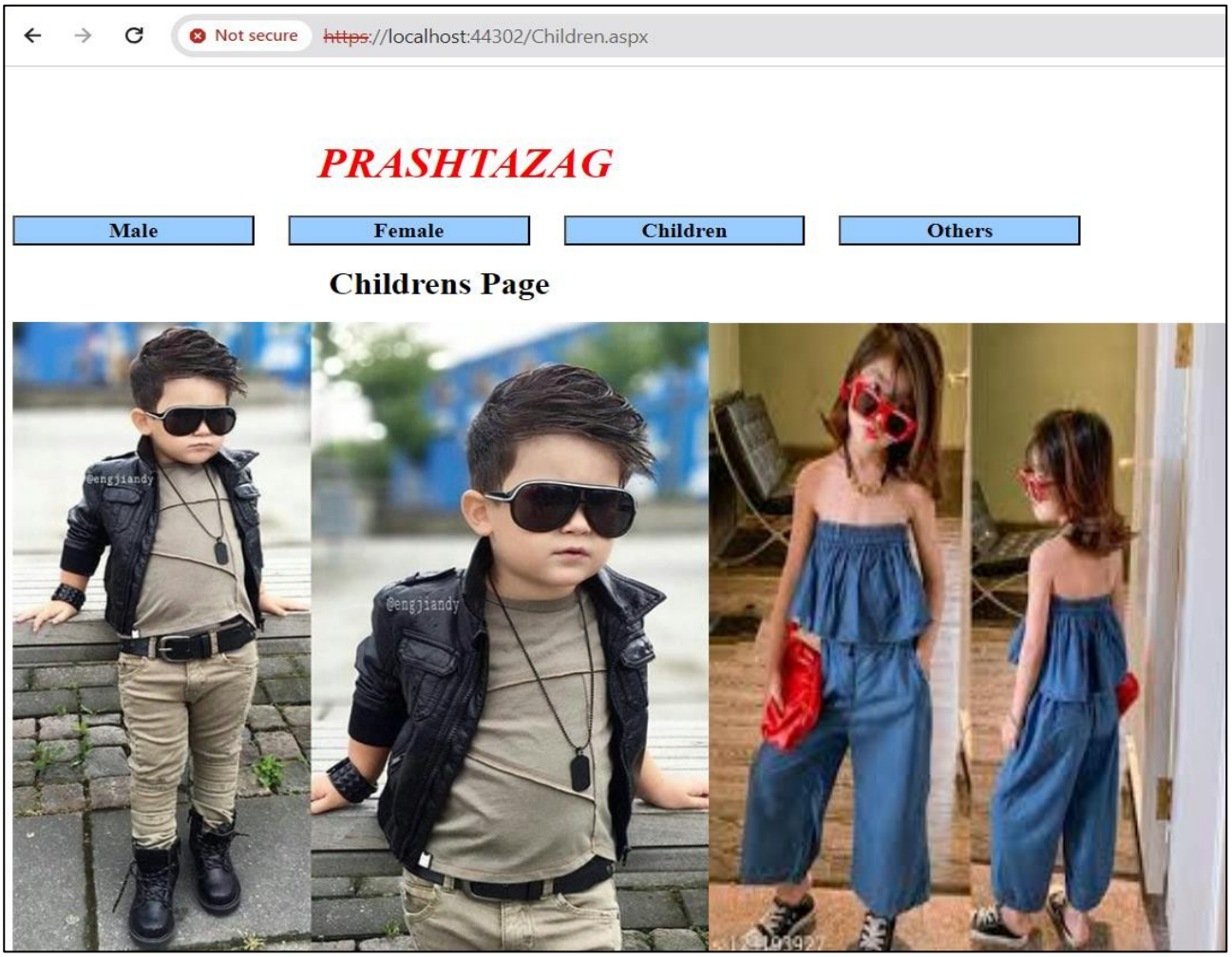
Master Page

Output:




Not secure https://localhost:44302/Male.aspx
PRASHTAZAG
Male
Female
Children
Others
Males Page


Not secure https://localhost:44302/Female.aspx
PRASHTAZAG
Male
Female
Children
Others
Females Page


Not secure
https://localhost:44302/Children.aspx
PRASHTAZAG
Male
Female
Children
Others
Childrens Page

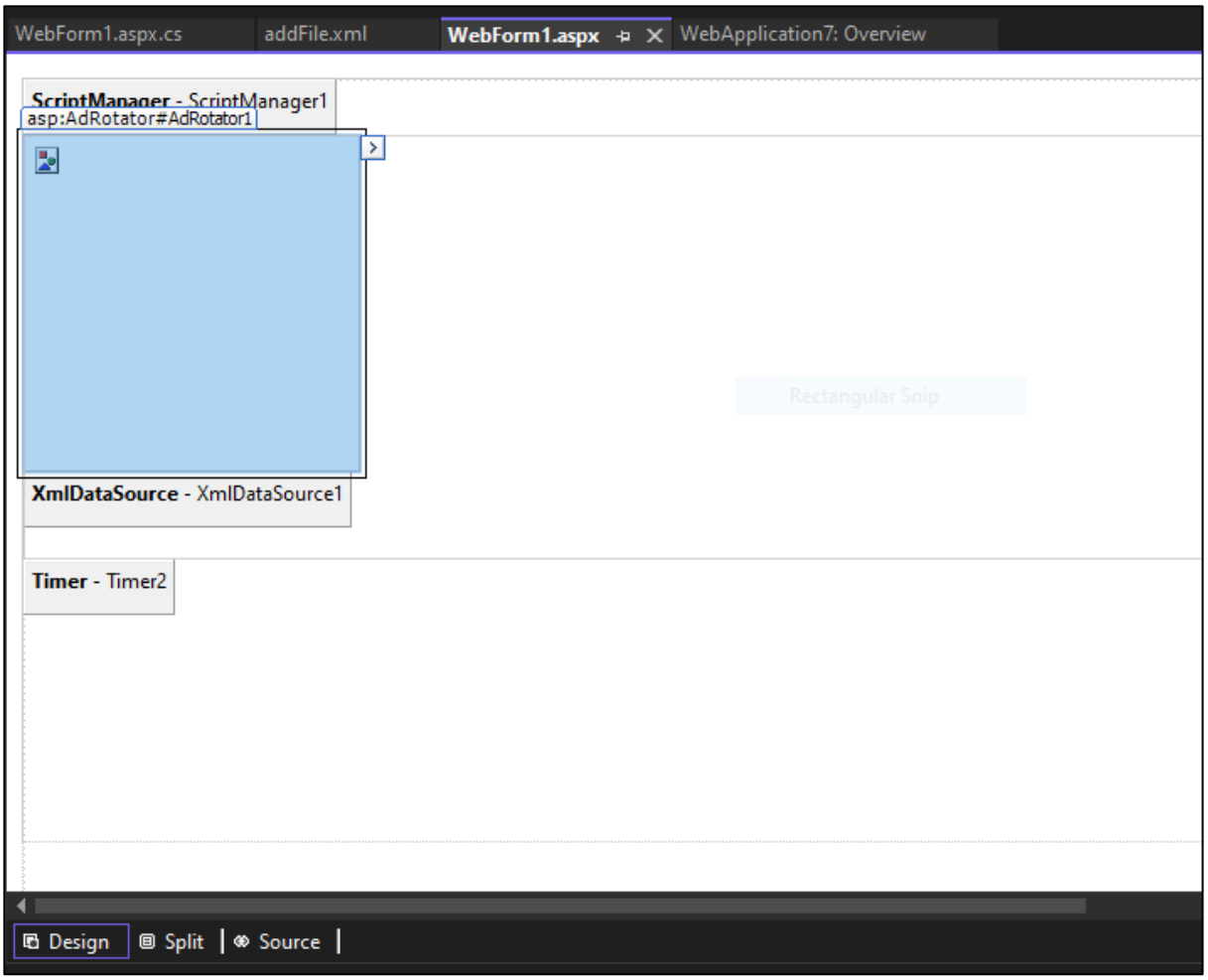
2.Use AJAX controls to change Banner ADs after every second, make use of Ad rotator Control.

addFile.xml

Code:

```
<Advertisements>
	<Ad>
		<ImageUrl>Astronaut-2.png</ImageUrl>
		<NavigateUrl> https://www.geeksforgeeks.org/html-vs-xml </NavigateUrl>
		<AlternateText>
			Random photos
		</AlternateText>
		<Impressions>20</Impressions>
		<Keyword>Astronaut</Keyword>
	</Ad>
	<Ad>
		<ImageUrl>images1.png</ImageUrl>
		<NavigateUrl>https://www.w3schools.com/xml/xml_whatis.asp </NavigateUrl>
		<AlternateText>
			Random photos
		</AlternateText>
		<Impressions>20</Impressions>
		<Keyword>images1</Keyword>
	</Ad>
	<Ad>
		<ImageUrl>images2.jpg</ImageUrl>
		<NavigateUrl> https://www.geeksforgeeks.org/sgml-vs-xml/?ref=ml_lbp </NavigateUrl>
		<AlternateText>
			Random photos
		</AlternateText>
		<Impressions>20</Impressions>
		<Keyword>images2</Keyword>
	</Ad>
	<Ad>
		<ImageUrl>image3.jpg</ImageUrl>
		<NavigateUrl> https://en.wikipedia.org/wiki/XML </NavigateUrl>
		<AlternateText>
			Random photos
		</AlternateText>
		<Impressions>20</Impressions>
		<Keyword>images3</Keyword>
	</Ad>
```

</Advertisements>



UpdatePanel → properties → conditional

Timer must be outside updatePanel

Webform.apsx:

Code:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication7.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        #form1 {
            height: 483px;
        }
    </style>
</head>
<body style="height: 452px">
    <form id="form1" runat="server">
```

```
    <asp:ScriptManager ID="ScriptManager1" runat="server">

    </asp:ScriptManager>
    <asp:UpdatePanel ID="UpdatePanel1" runat="server" ClientIDMode="Predictable"
UpdateMode="Conditional">
      <ContentTemplate>
        <asp:AdRotator ID="AdRotator1" runat="server" DataSourceID="XmlDataSource1"
Height="200px" Width="200px" />
        <asp:XmlDataSource ID="XmlDataSource1" runat="server"
DataFile="~/addFile.xml"></asp:XmlDataSource>
        <br />
      </ContentTemplate>

      <Triggers>
        <asp:AsyncPostBackTrigger ControlID="Timer2" EventName="Tick" />
      </Triggers>

      </asp:UpdatePanel>

    <asp:Timer ID="Timer2" runat="server" OnTick="Timer2_Tick" Interval="1000">
    </asp:Timer>
  </form>
</body>
</html>
```

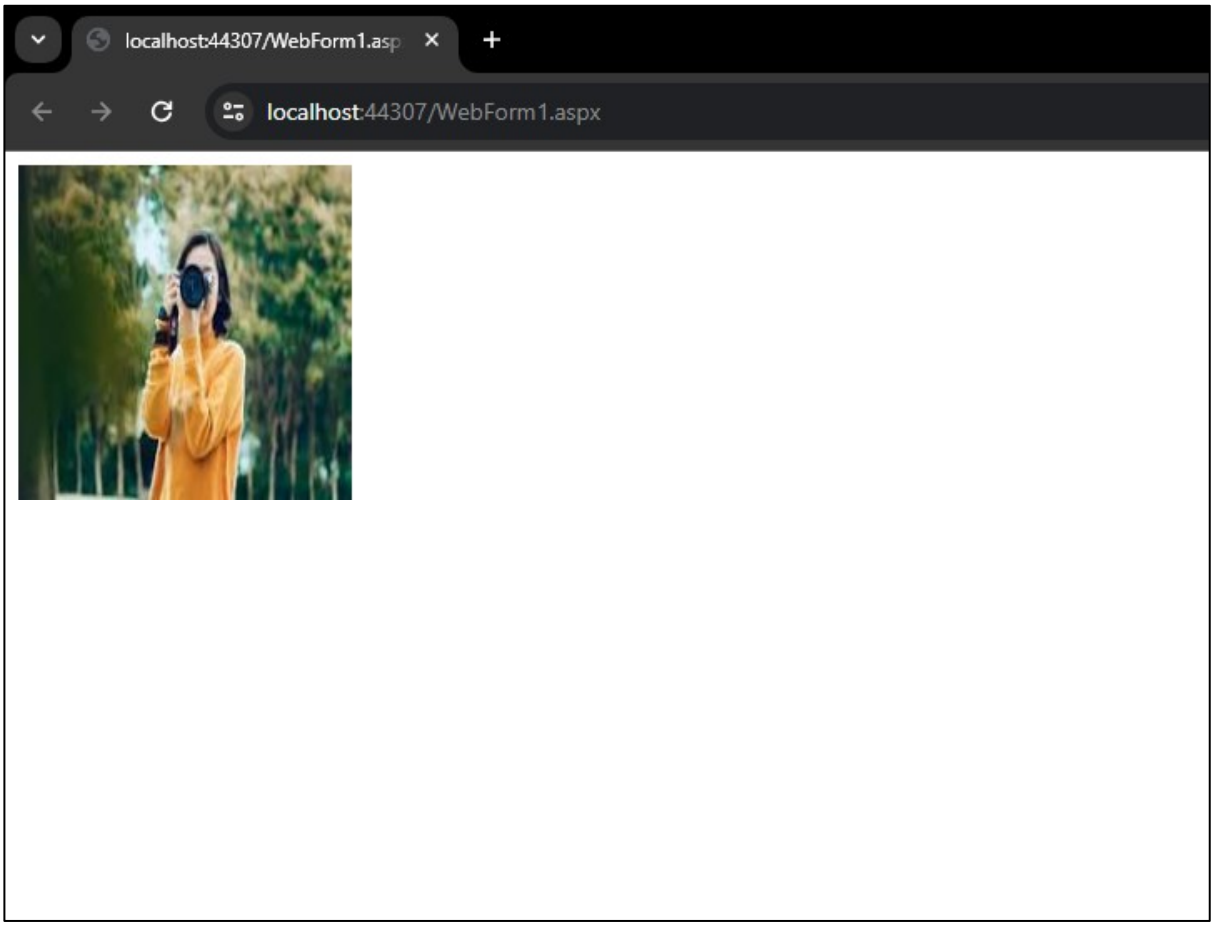
Webform.aspx.cs:

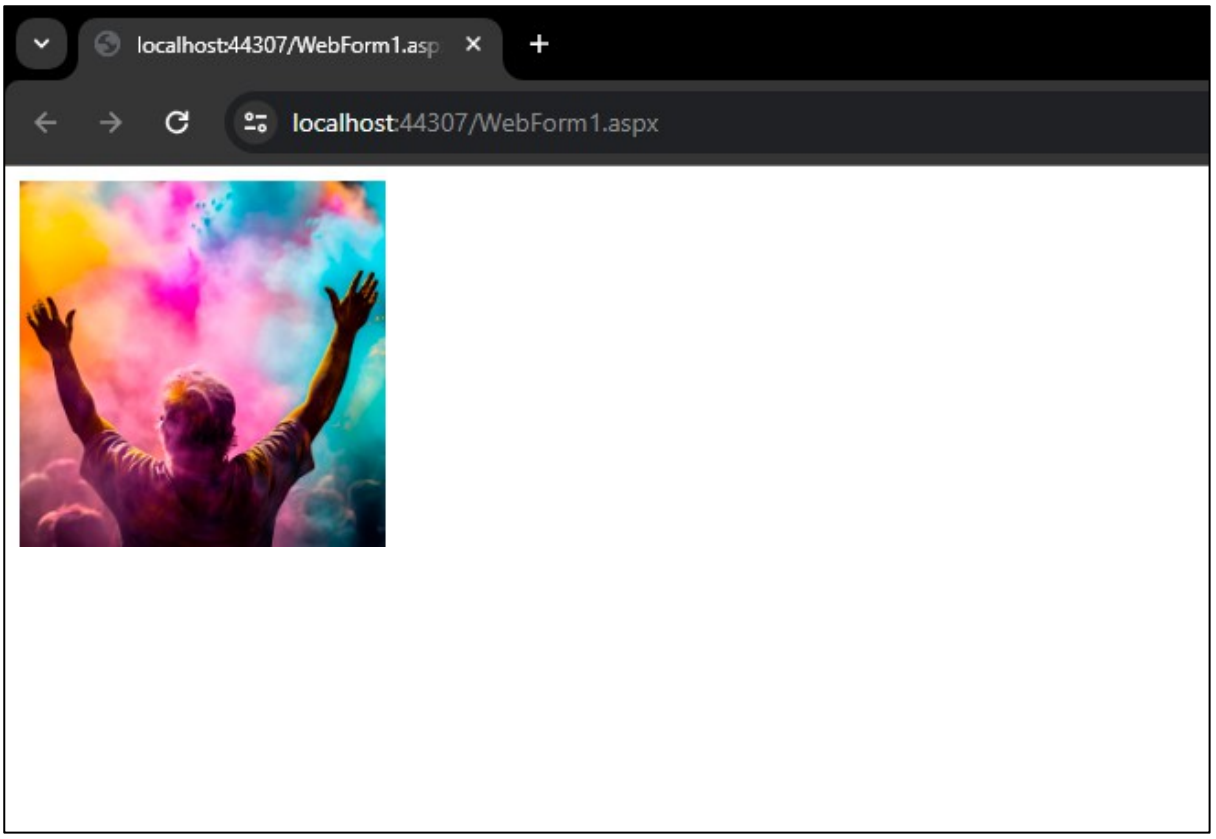
Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication7
{
  public partial class WebForm1 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {
      if (!IsPostBack)
      {
        //Start the timer on the initial page load
        Timer2.Enabled = true;
      }
    }

    protected void Timer2_Tick(object sender, EventArgs e)
    {
      //Force UpdatePanel to refresh
      UpdatePanel1.Update();
    }
  }
```

}

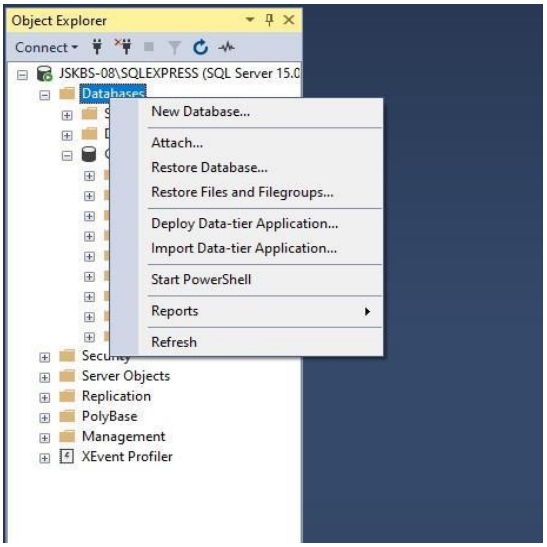
Output:

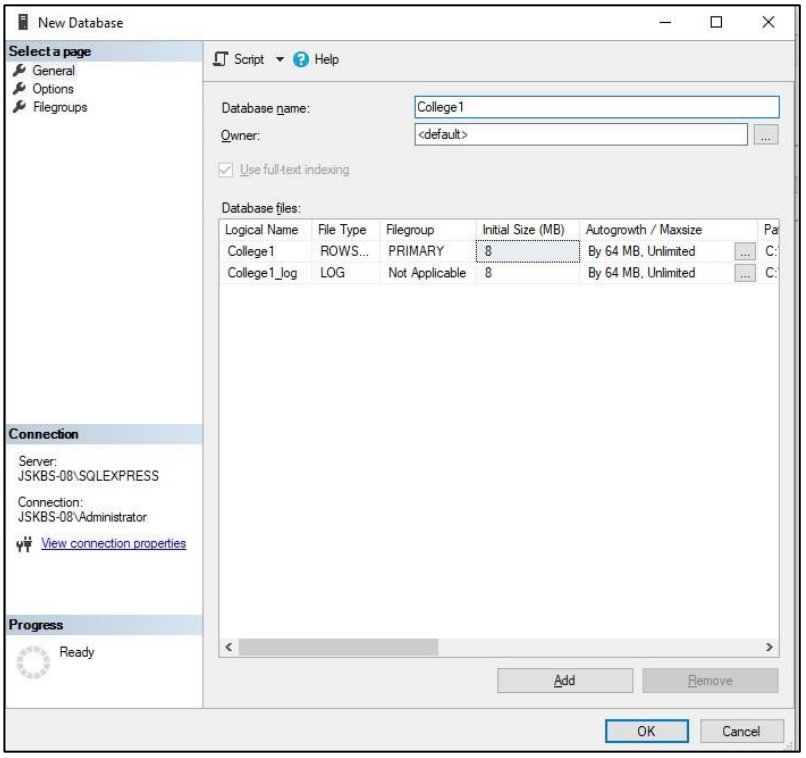

localhost:44307/WebForm1.asp
localhost:44307/WebForm1.aspx

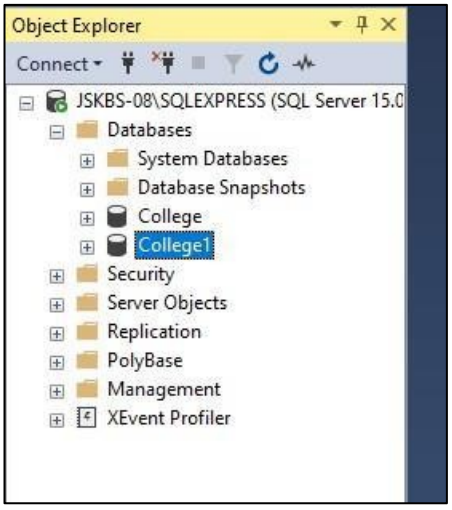

localhost:44307/WebForm1.asp
localhost:44307/WebForm1.aspx

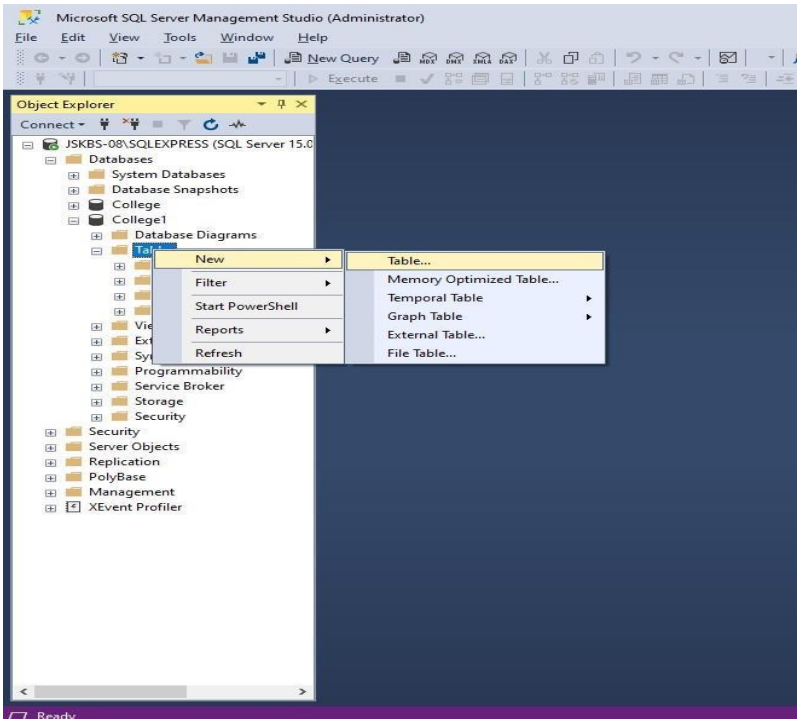
Experiment Number 3: ADO.NET

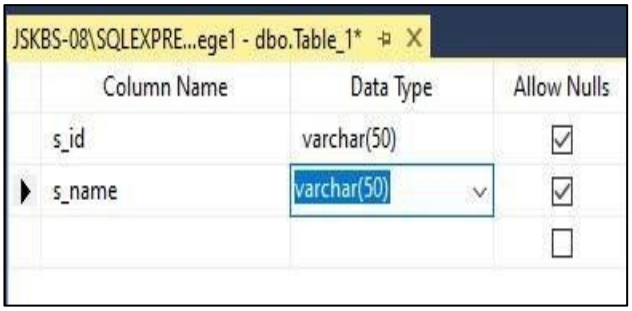
1. Design a webpage to demonstrate a connection-oriented architecture.

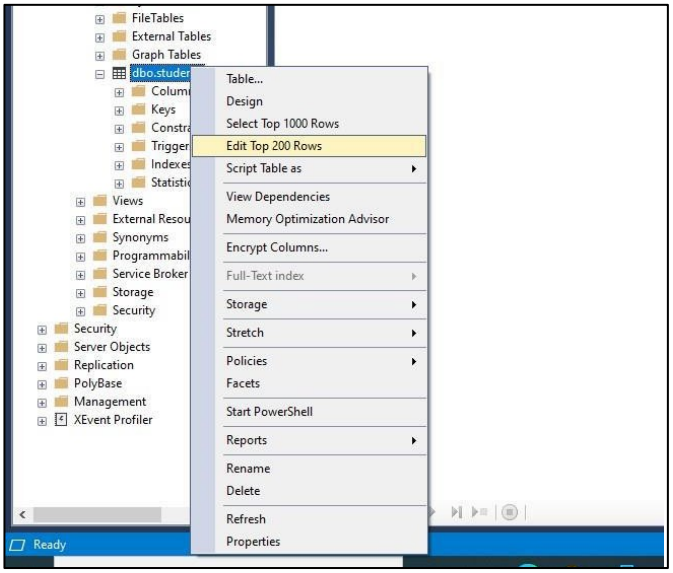



New Database
Select a page
General
Options
Filegroups
Script
Help
Database name:
College1
Owner:
<default>
Use full-text indexing
Database files:
Logical Name
File Type
Filegroup
Initial Size (MB)
Autogrowth / Maxsize
College1
ROWS...
PRIMARY
8
By 64 MB, Unlimited
College1_log
LOG
Not Applicable
8
By 64 MB, Unlimited
Connection
Server:
JSKBS-08\SQLEXPRESS
Connection:
JSKBS-08\Administrator
View connection properties
Progress
Ready
Add
Remove
OK
Cancel

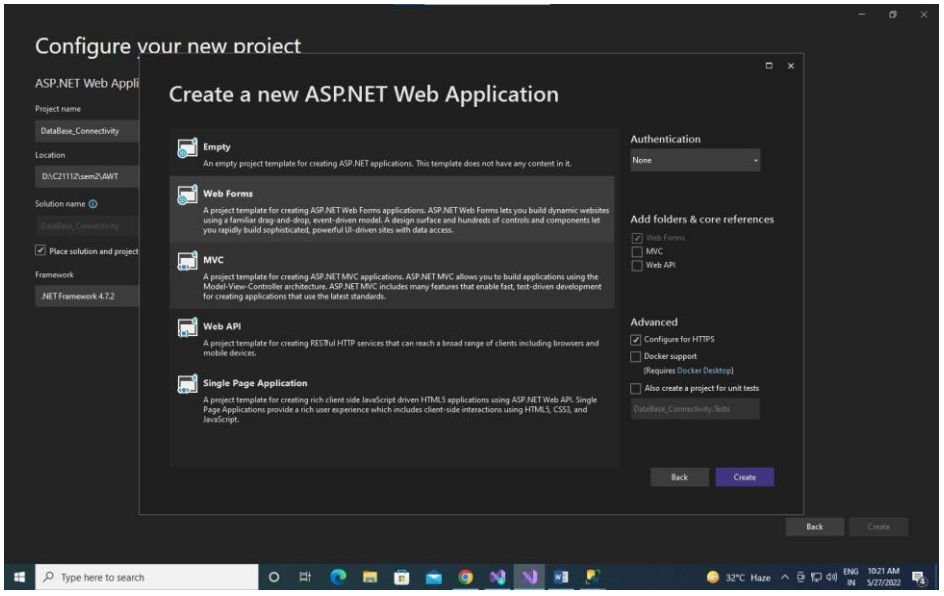

Object Explorer
Connect
JSKBS-08\SQLEXPRESS (SQL Server 15.0
Databases
System Databases
Database Snapshots
College
College1
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler

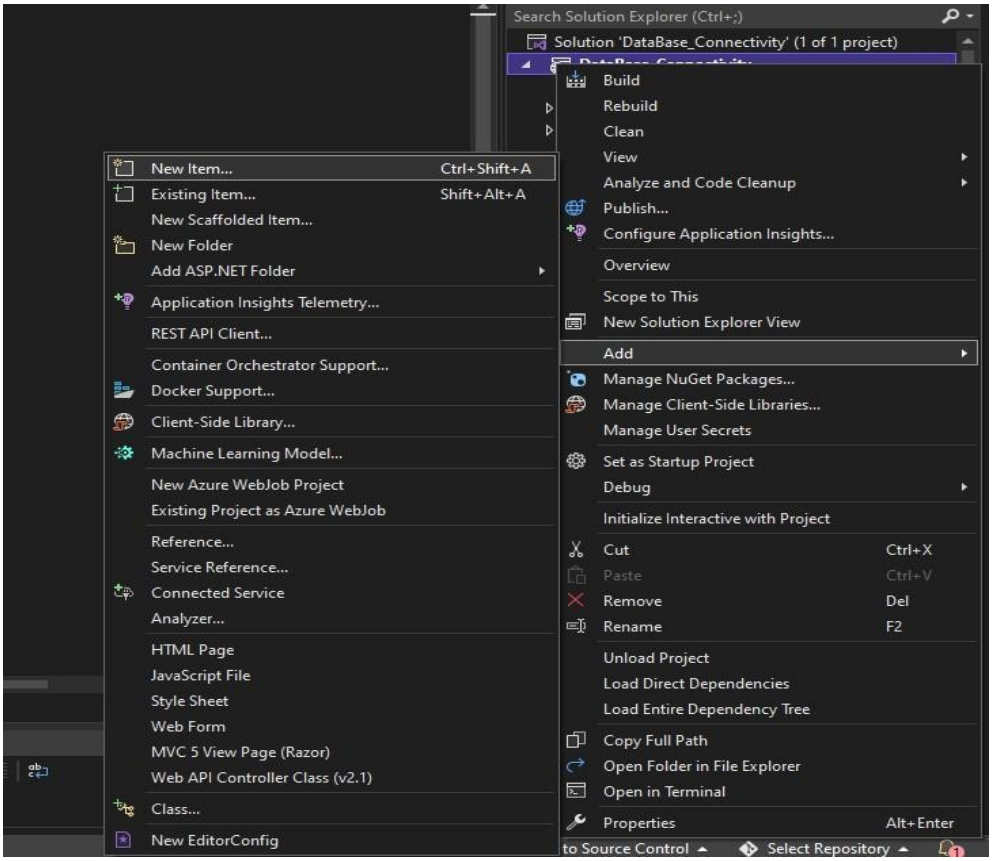

Microsoft SQL Server Management Studio (Administrator)
File Edit View Tools Window Help
New Query
Execute
Object Explorer
Connect
JSKBS-08\SQLEXPRESS (SQL Server 15.0
Databases
System Databases
Database Snapshots
College
College1
Database Diagrams
New
Filter
Start PowerShell
Reports
Refresh
Table...
Memory Optimized Table...
Temporal Table
Graph Table
External Table...
File Table...
Programmability
Service Broker
Storage
Security
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler

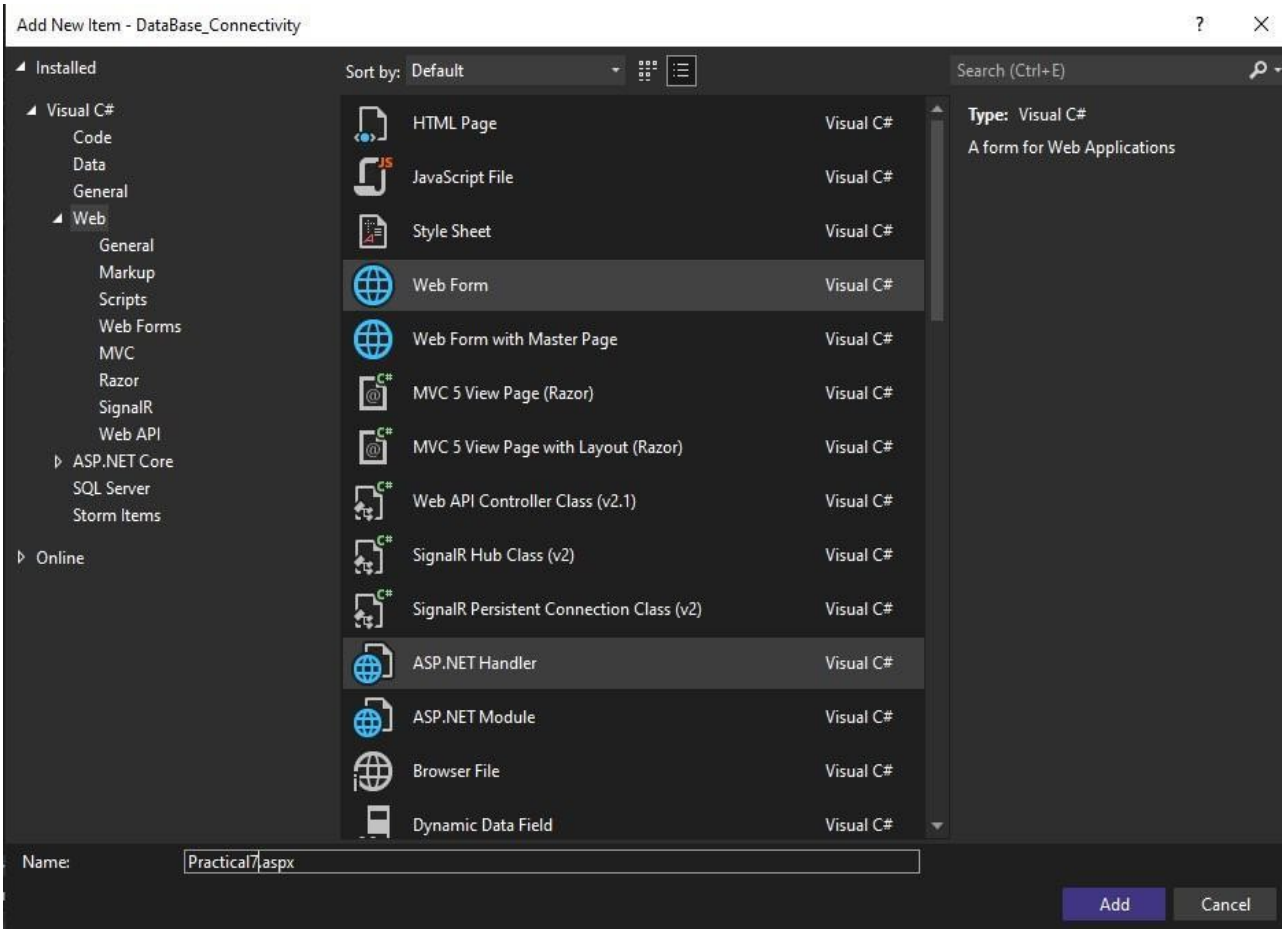

JSKBS-08\SQLEXPRE...ege1 - dbo.Table_1*
Column Name
Data Type
Allow Nulls
s_id
varchar(50)
s_name
varchar(50)

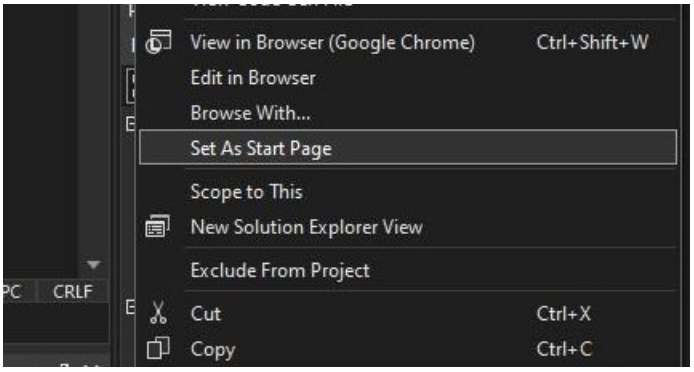

FileTables
External Tables
Graph Tables
Keys
Trigger
Indexes
Views
Synonyms
Service Broker
Storage
Security
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler
Table...
Design
Select Top 1000 Rows
Edit Top 200 Rows
Script Table as
View Dependencies
Memory Optimization Advisor
Encrypt Columns...
Full-Text index
Storage
Stretch
Policies
Facets
Start PowerShell
Reports
Rename
Delete
Refresh
Properties
Ready

Visual Studio



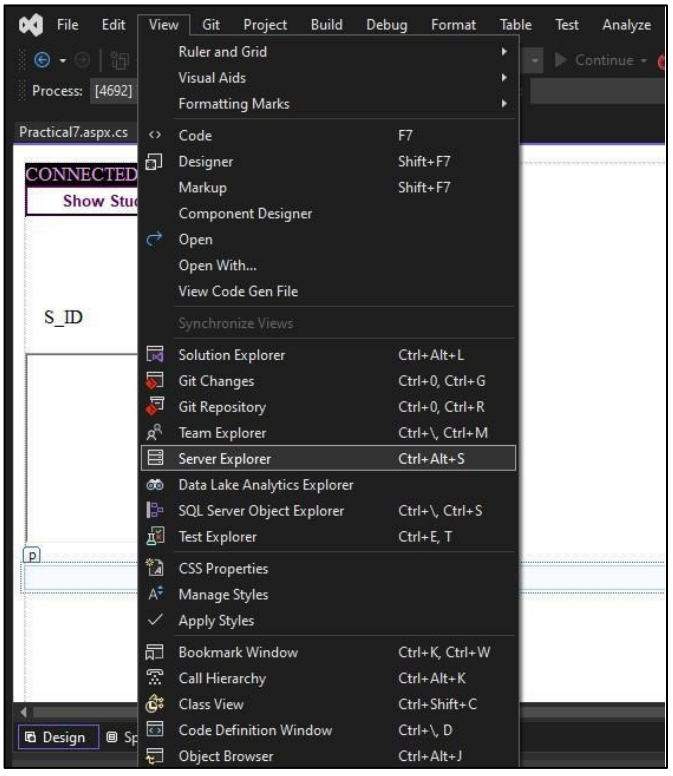


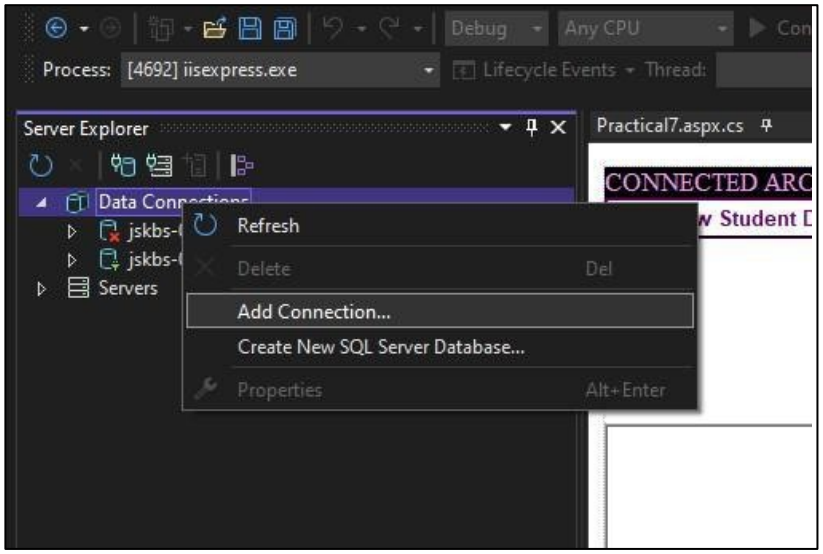

Add New Item - DataBase_Connectivity
Installed
Visual C#
Code
Data
General
Web
General
Markup
Scripts
Web Forms
MVC
Razor
SignalR
Web API
ASP.NET Core
SQL Server
Storm Items
Online
Sort by: Default
Search (Ctrl+E)
HTML Page Visual C#
JavaScript File Visual C#
Style Sheet Visual C#
Web Form Visual C#
Web Form with Master Page Visual C#
MVC 5 View Page (Razor) Visual C#
MVC 5 View Page with Layout (Razor) Visual C#
Web API Controller Class (v2.1) Visual C#
SignalR Hub Class (v2) Visual C#
SignalR Persistent Connection Class (v2) Visual C#
ASP.NET Handler Visual C#
ASP.NET Module Visual C#
Browser File Visual C#
Dynamic Data Field Visual C#
Type: Visual C#
A form for Web Applications
Name: Practical7.aspx
Add
Cancel

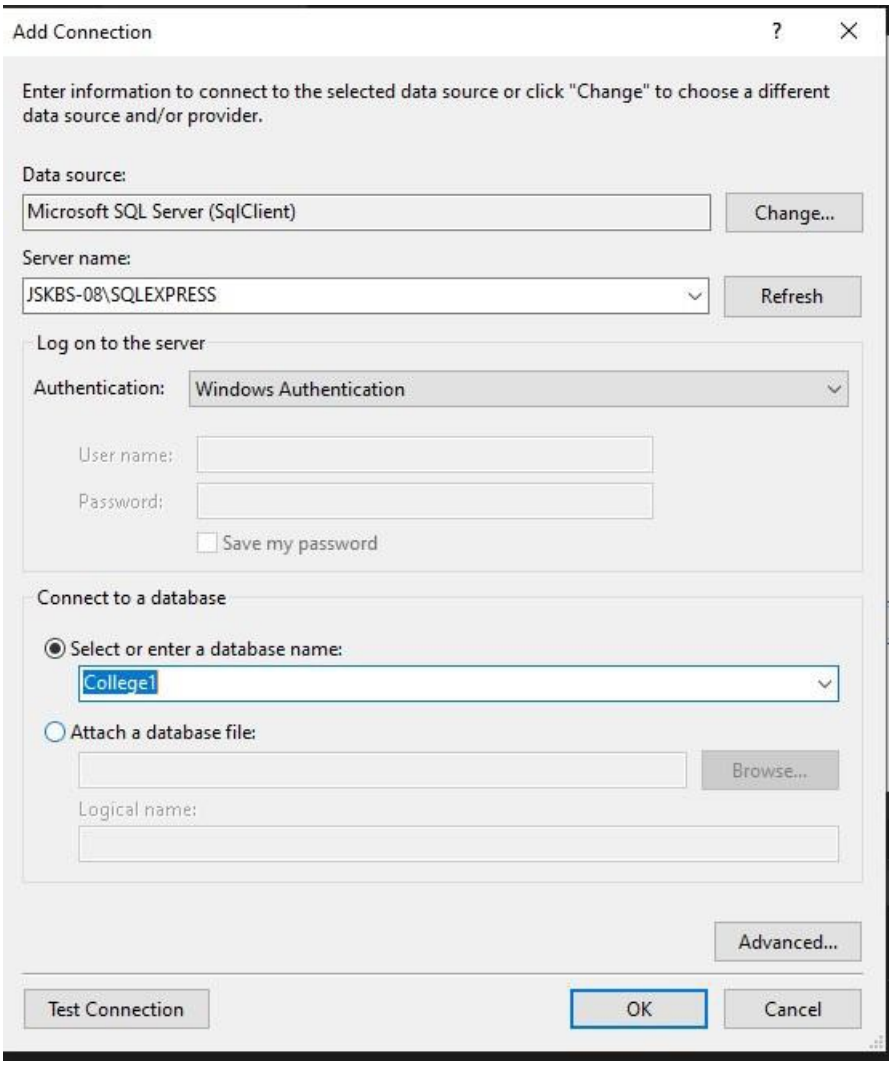

View in Browser (Google Chrome) Ctrl+Shift+W
Edit in Browser
Browse With...
Set As Start Page
Scope to This
New Solution Explorer View
Exclude From Project
Cut Ctrl+X
Copy Ctrl+C
CRLF

DESIGN


CONNECTED ARCHITECTURE DEMO
Show Student Data
S_ID
S_NAME


File Edit View Git Project Build Debug Format Table Test Analyze
Ruler and Grid
Visual Aids
Formatting Marks
Code F7
Designer Shift+F7
Markup Shift+F7
Component Designer
Open
Open With...
View Code Gen File
Synchronize Views
Solution Explorer Ctrl+Alt+L
Git Changes Ctrl+0, Ctrl+G
Git Repository Ctrl+0, Ctrl+R
Team Explorer Ctrl+\, Ctrl+M
Server Explorer Ctrl+Alt+S
Data Lake Analytics Explorer
SQL Server Object Explorer Ctrl+\, Ctrl+S
Test Explorer Ctrl+E, T
CSS Properties
Manage Styles
Apply Styles
Bookmark Window Ctrl+K, Ctrl+W
Call Hierarchy Ctrl+Alt+K
Class View Ctrl+Shift+C
Code Definition Window Ctrl+\, D
Object Browser Ctrl+Alt+J
Process: [4692]
Practical7.aspx.cs
S_ID
Design





Click on your server nameIn properties->Connection String ->Copy paste the entire line and paste in the src code with string connection.

Source Code on Clicking on Button:

```
usingSystem;
```

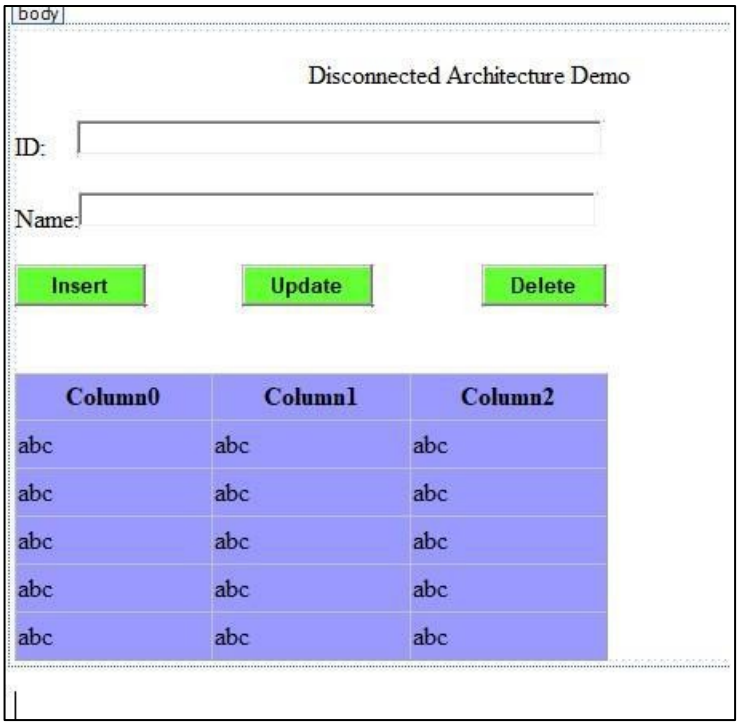
```
usingSystem.Collections.Generic;
usingSystem.Linq;
usingSystem.Web;
using System.Web.UI;
using System.Web.UI.WebControls; using System.Data.SqlClient;
namespace DataBase_Connectivity
{
public partial class Practical7 : System.Web.UI.Page
{
//creation of instance of connection,command and reader
SqlConnection myConn;
SqlCommand myCmd; SqlDataReader myRdr;
protected void Page_Load(object sender, EventArgs e)
{
//Initiatization or creation of connection
myConn = new SqlConnection();
String connstr = @"Data Source=JSKBS-08\SQLEXPRESS;Initial Catalog=College1;Integrated Security=True";
myConn.ConnectionString = connstr;
//open connection
myConn.Open();
}

protected void Button1_Click(object sender, EventArgs e) {
TextBox1.Text = null;
TextBox2.Text = null;
myCmd=new SqlCommand("select * from student1",myConn);
//Execute the query
myRdr=myCmd.ExecuteReader();
//fetch one record at a time and add it to text area while(myRdr.Read())
{
String s_id=myRdr[0].ToString();
String s_name=myRdr[1].ToString();
TextBox1.Text += s_id +" "+ Environment.NewLine;
TextBox2.Text += s_name + " " + Environment.NewLine; }
//close connection myConn.Close();
}
}
}
```

Output:

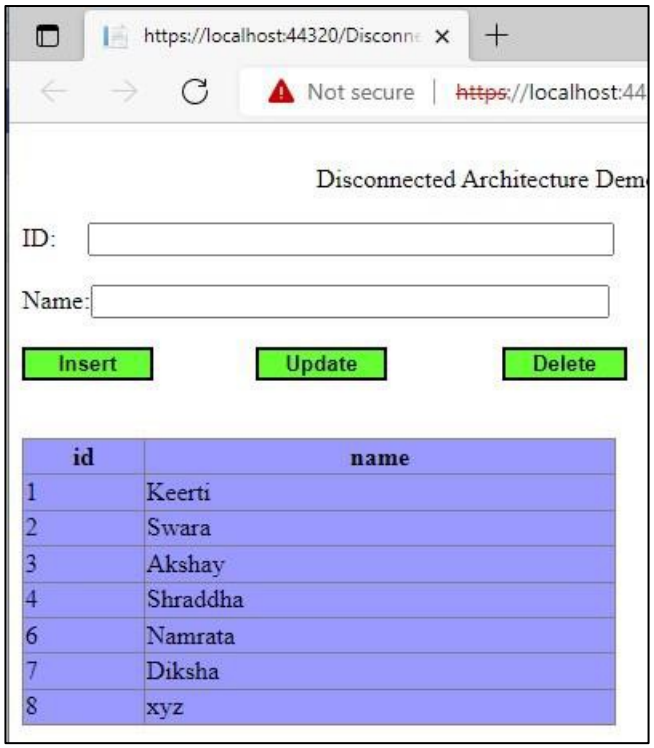

https://localhost:44332/Practical
localhost:44332/Practical7
CONNECTED ARCHITECTURE DEMO
Show Student Data
S_ID
S_NAME
101
102
103
104
105
null
Diksha
Jyoti
Keerti
Namrata
Vaishnavi

2. Design a webpage to demonstrate a disconnected architecture.

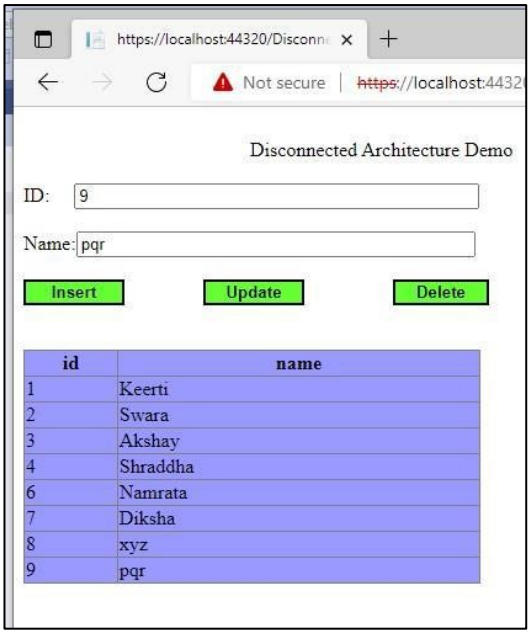
Design



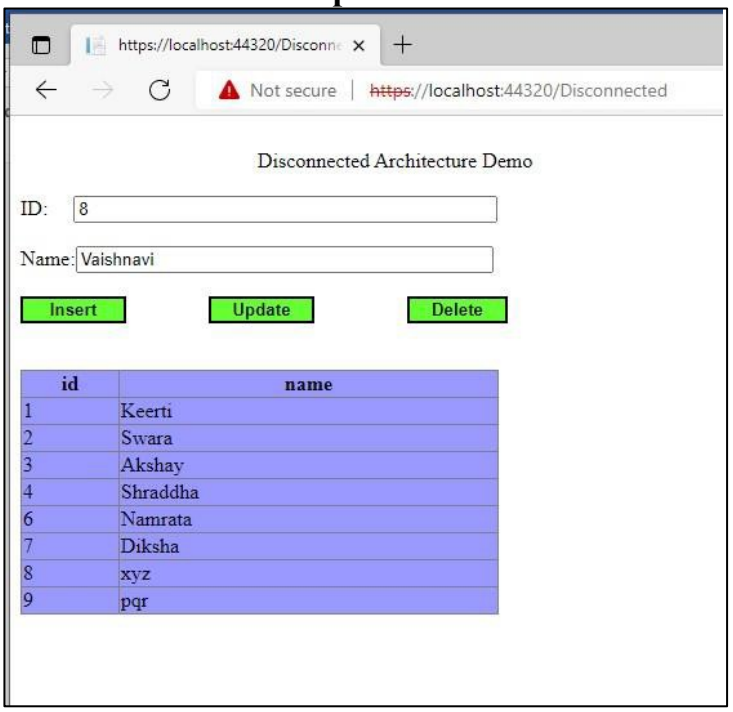
Output:

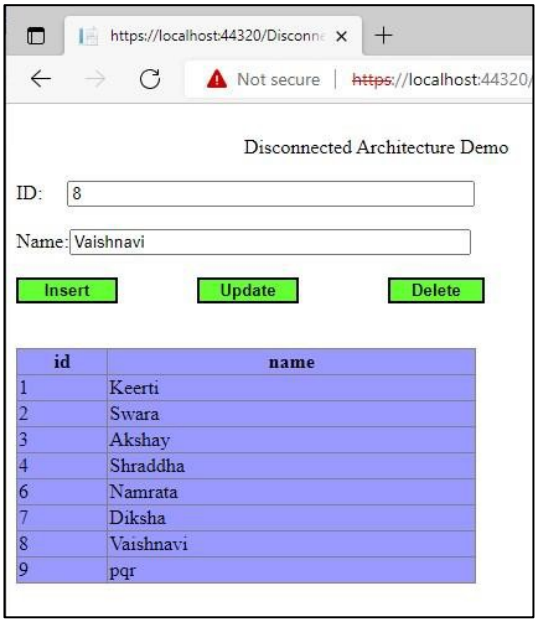


Insert

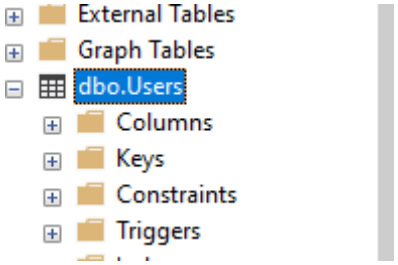


Update:




https://localhost:44320/Disconne
Not secure | https://localhost:44320/
Disconnected Architecture Demo
ID: 8
Name: Vaishnavi
Insert
Update
Delete
id name
1 Keerti
2 Swara
3 Akshay
4 Shraddha
6 Namrata
7 Diksha
8 Vaishnavi
9 pqr

3.Create a webpage that demonstrates the use of data bound controls of ASP.NET.



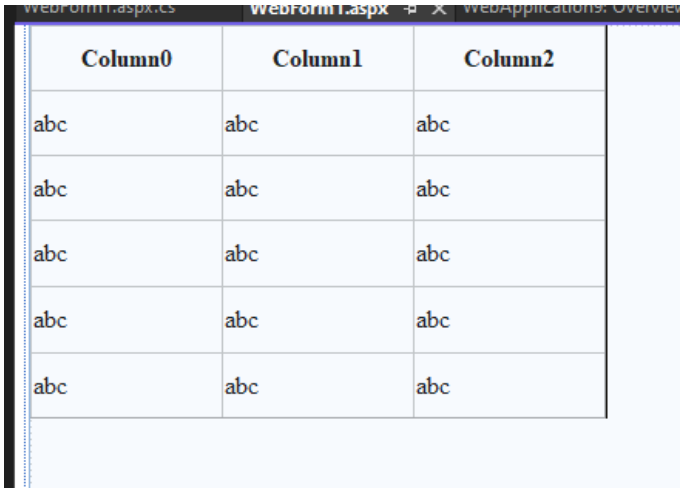
create procedure GetUsers

AS
begin
select * from Users

end

execute>>save

Webform1.aspx:
Design:



Source:
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="WebApplication9.WebForm1" %>

<!DOCTYPE html>

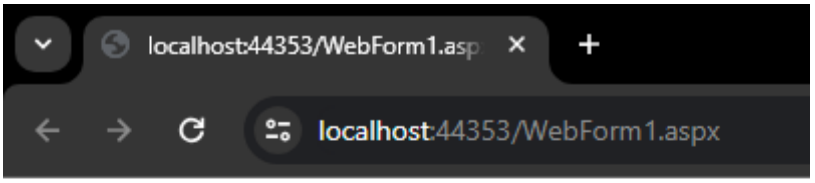
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
<style type="text/css">

```
        #form1 {
            height: 582px;
        }
    </style>
</head>
<body>
    <form id="form1" runat="server">
        <div style="height: 493px">
            <asp:GridView ID="GridView1" runat="server" Height="255px"
OnSelectedIndexChanged="GridView1_SelectedIndexChanged" Width="373px">
            </asp:GridView>
        </div>
    </form>
</body>
</html>
```

Webform1.aspx.cs:

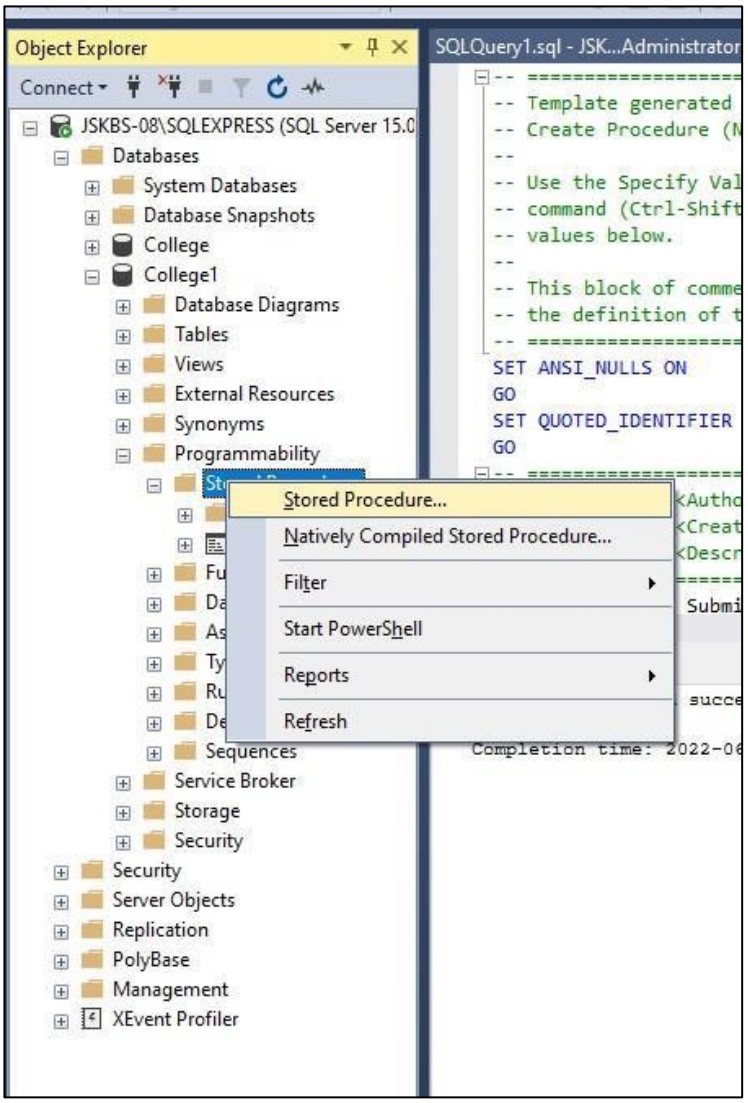
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;

namespace WebApplication9
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if(!IsPostBack)
            {
                BindGridView();
            }
        }

        private void BindGridView()
        {
            string connectionstring = @"Data Source=DESKTOP-AS3SIG1\SQLEXPRESS;Initial
Catalog=userdb;Integrated Security=True";
            string storedprocedure = "GetUsers";
            using(SqlConnection connection = new SqlConnection(connectionstring))
            {
                using(SqlCommand command = new SqlCommand(storedprocedure,connection))
                {
                    command.CommandType = CommandType.StoredProcedure;

                    using (SqlDataAdapter adapter = new SqlDataAdapter(command))
                    {
                        DataTable dt = new DataTable();
                        adapter.Fill(dt);
                        GridView1.DataSource = dt;
```

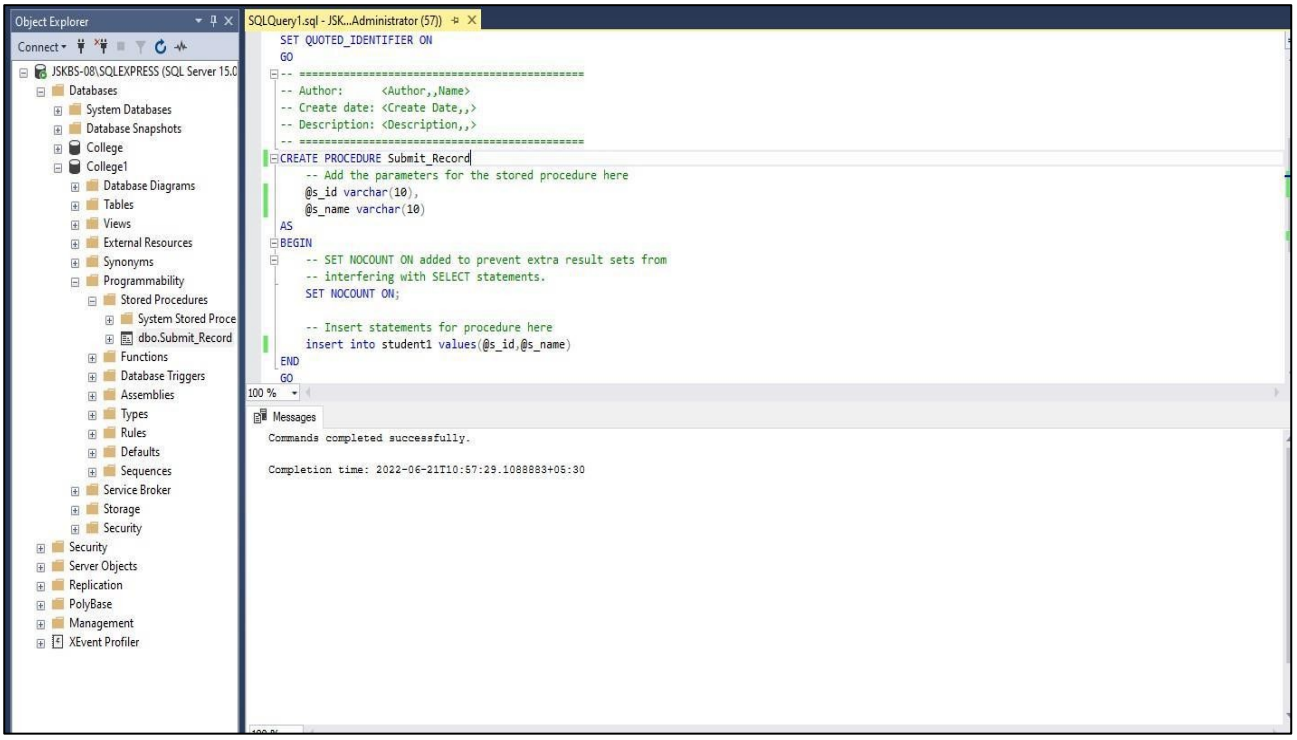
```
                GridView1.DataBind();
            }
        }
    }
}

protected void GridView1_SelectedIndexChanged(object sender, EventArgs e)
{

}
}
}
```

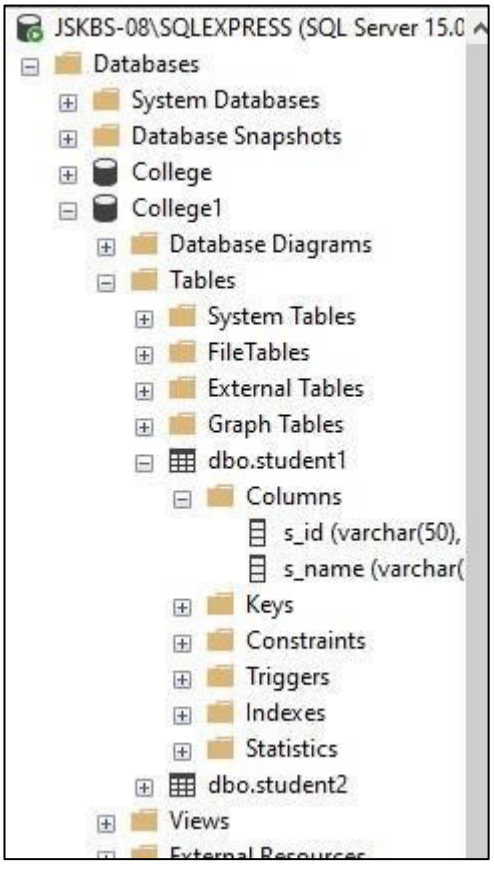


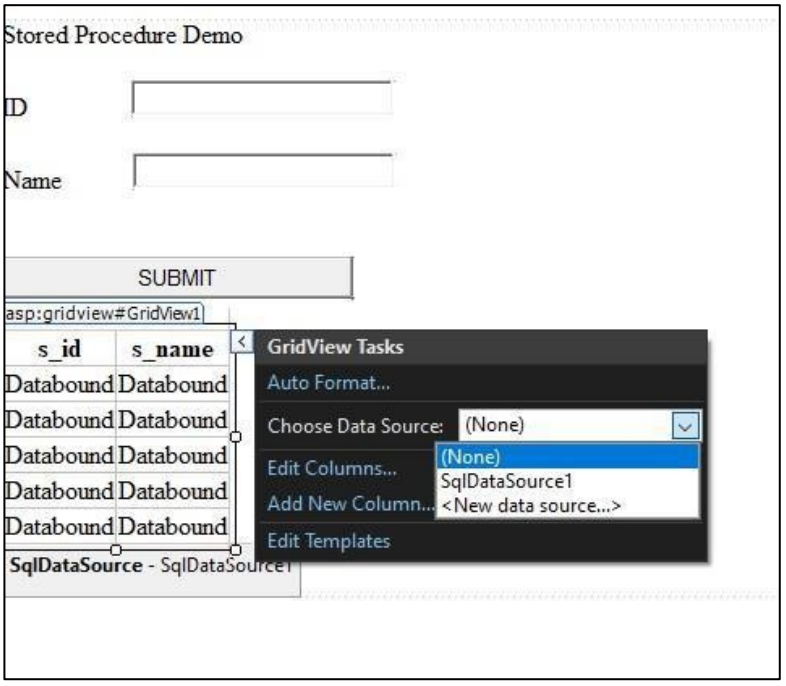
Username	Password
sanjana	sanju
neha	neha
yahika	yashu
dhanashree	dhano
shruti	

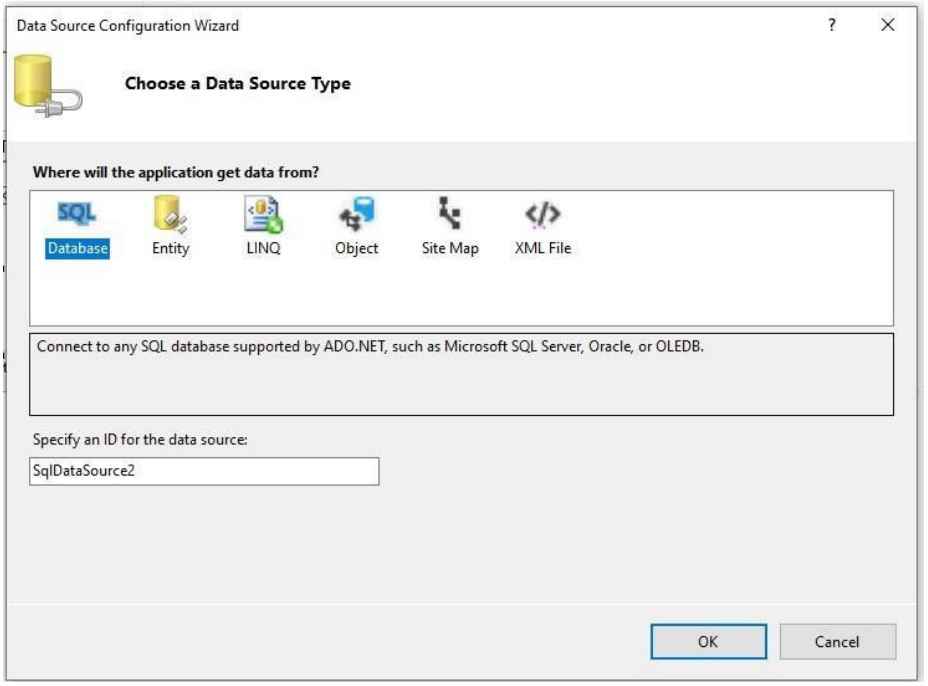
4.Design a webpage to demonstrate the working of a simple stored procedure.

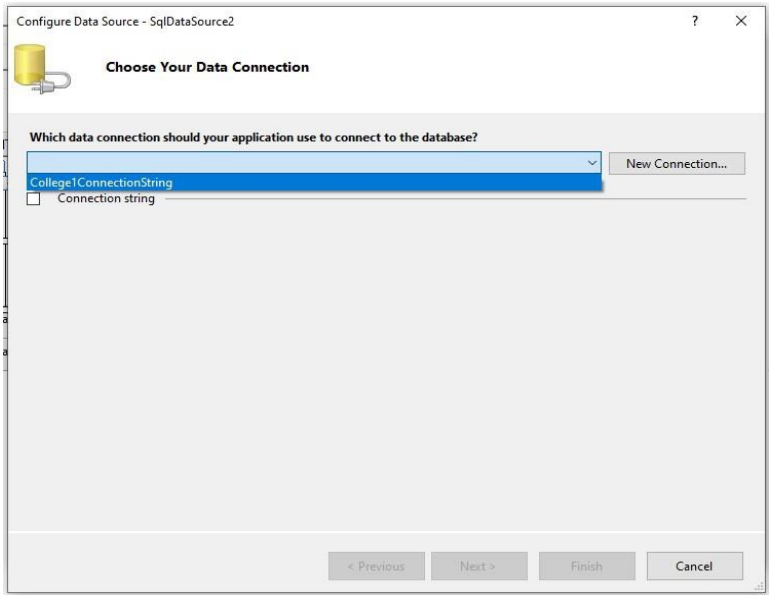


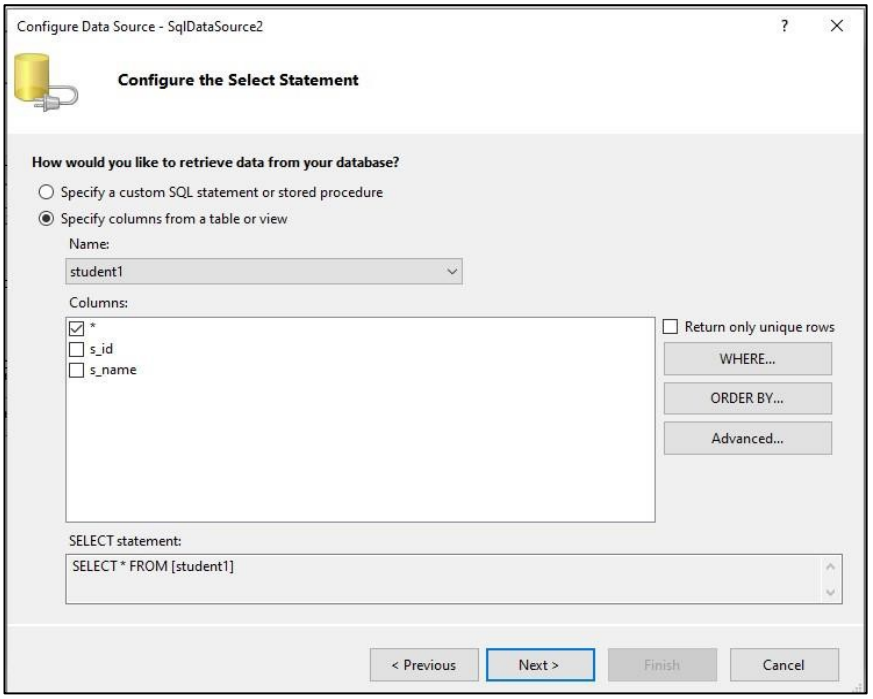

Object Explorer
Connect
JSKBS-08\SQLEXPRESS (SQL Server 15.0
Databases
System Databases
Database Snapshots
College
College1
Database Diagrams
Tables
Views
External Resources
Synonyms
Programmability
Stored Procedures
System Stored Proce
dbo.Submit_Record
Functions
Database Triggers
Assemblies
Types
Rules
Defaults
Sequences
Service Broker
Storage
Security
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler
SQLQuery1.sql - JSK...Administrator (57))
SET QUOTED_IDENTIFIER ON
GO
-- ==
-- Author: <Author,,Name>
-- Create date: <Create Date,,>
-- Description: <Description,,>
-- ==
CREATE PROCEDURE Submit_Record
-- Add the parameters for the stored procedure here
@s_id varchar(10),
@s_name varchar(10)
AS
BEGIN
-- SET NOCOUNT ON added to prevent extra result sets from
-- interfering with SELECT statements.
SET NOCOUNT ON;
-- Insert statements for procedure here
insert into student1 values(@s_id,@s_name)
END
GO
100 %
Messages
Commands completed successfully.
Completion time: 2022-06-21T10:57:29.1088883+05:30

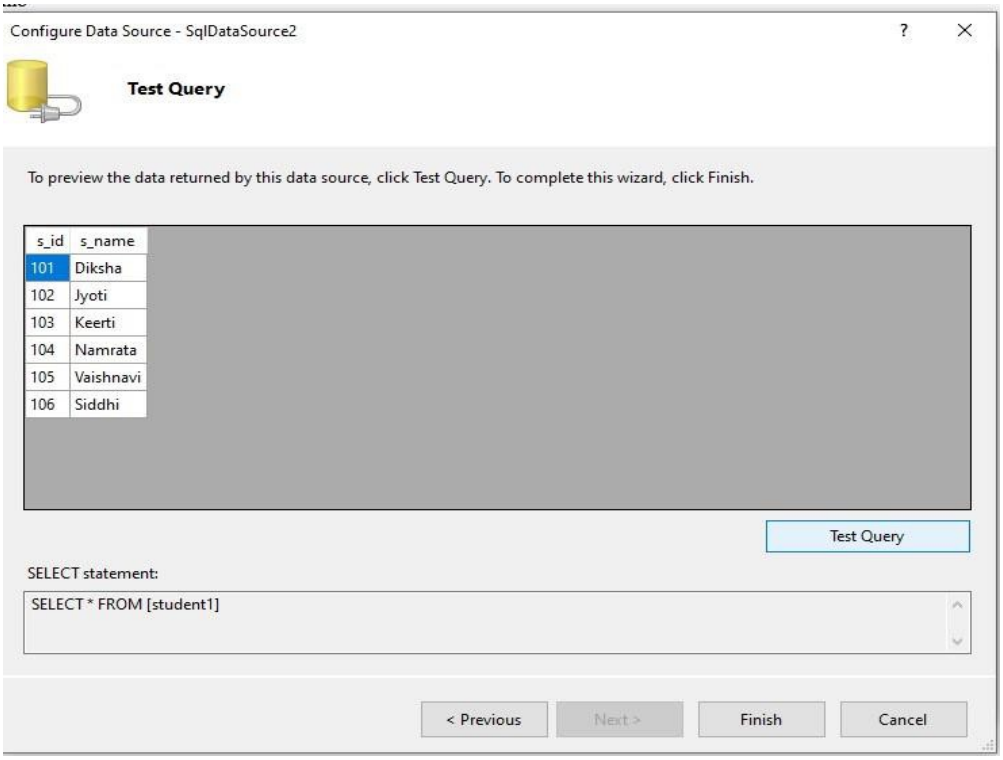

JSKBS-08\SQLEXPRESS (SQL Server 15.0
Databases
System Databases
Database Snapshots
College
College1
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.student1
Columns
s_id (varchar(50),
s_name (varchar(
Keys
Constraints
Triggers
Indexes
Statistics
dbo.student2
Views


Stored Procedure Demo
ID
Name
SUBMIT
asp:gridview#GridView1
s_id
s_name
Databound Databound
Databound Databound
Databound Databound
Databound Databound
Databound Databound
SqlDataSource - SqlDataSource1
GridView Tasks
Auto Format...
Choose Data Source:
(None)
Edit Columns...
Add New Column...
Edit Templates
(None)
SqlDataSource1
<New data source...>


Data Source Configuration Wizard
Choose a Data Source Type
Where will the application get data from?
Database
Entity
LINQ
Object
Site Map
XML File
Connect to any SQL database supported by ADO.NET, such as Microsoft SQL Server, Oracle, or OLEDB.
Specify an ID for the data source:
SqlDataSource2
OK
Cancel


Configure Data Source - SqlDataSource2
Choose Your Data Connection
Which data connection should your application use to connect to the database?
College1ConnectionString
New Connection...
Connection string
< Previous
Next >
Finish
Cancel

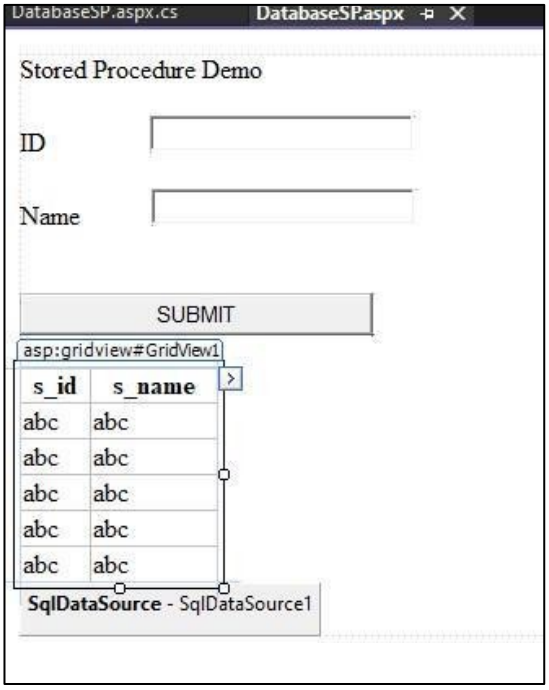

Configure Data Source - SqlDataSource2
Configure the Select Statement
How would you like to retrieve data from your database?
Specify a custom SQL statement or stored procedure
Specify columns from a table or view
Name:
student1
Columns:
*
s_id
s_name
Return only unique rows
WHERE...
ORDER BY...
Advanced...
SELECT statement:
SELECT * FROM [student1]
< Previous
Next >
Finish
Cancel



In VSC

DatabaseSp.apcx

DESIGN

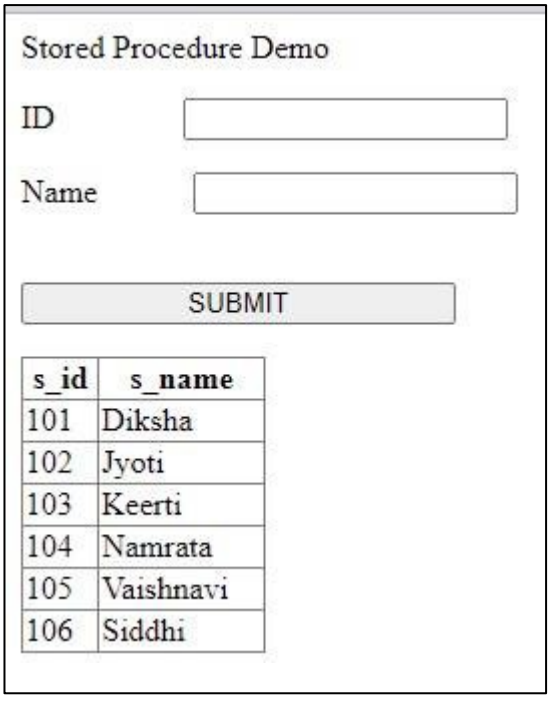


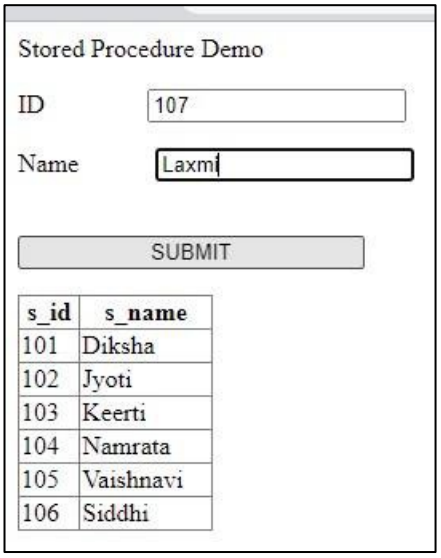
Click on the submit button

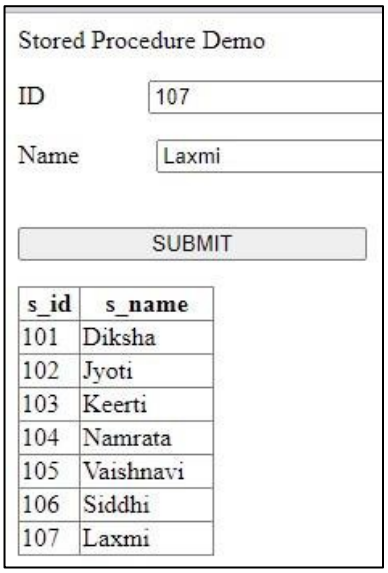
Code:

```
Using System;
Using System.Collections.Generic;
```

```
Using System.Linq;
Using System.Web;
Using System.Web.UI;
Using System.Web.UI.WebControls;
Using System.Data.SqlClient;
Using System.Data;
Namespace WebApplication2
{
Public partial class DatabaseSP : System.Web.UI.Page
{
SqlConnection con;
SqlCommand cmd;
Protected void Page_Load( object sender, EventArgs e)
{
}
Protected void Button1_Click( object sender, EventArgs e)
{

con= new SqlConnection( @"Data Source=JSKBS-08\SQLEXPRESS;Initial Catalog=College1;Integrated
Security=True");
con.Open();
cmd = new SqlCommand("Submit_Record", con);
cmd.CommandType= CommandType.StoredProcedure;
SqlParameter param = cmd.Parameters.Add("@s_id", SqlDbType.VarChar);
param.Value = TextBox1.Text;
SqlParameter param1 = cmd.Parameters.Add("@s_name",SqlDbType.VarChar);
param1.Value = TextBox2.Text;
cmd.ExecuteNonQuery();
GridView1.DataBind();
con.Close();
}
}
}
```



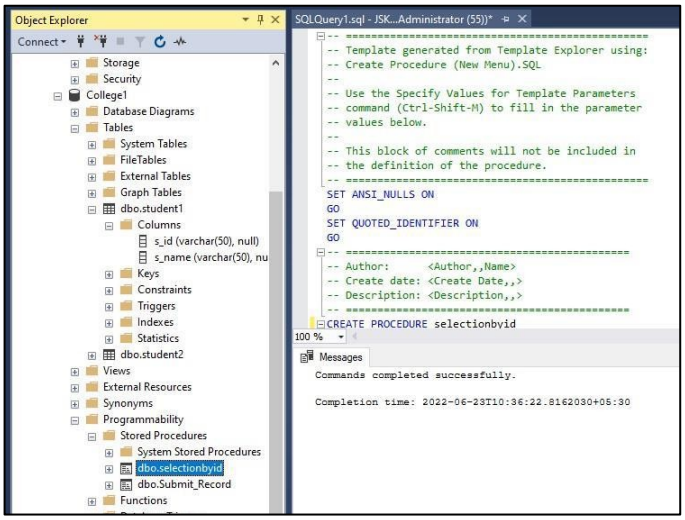

Stored Procedure Demo
ID
107
Name
Laxmi
SUBMIT
s_id
s_name
101
Diksha
102
Jyoti
103
Keerti
104
Namrata
105
Vaishnavi
106
Siddhi


Stored Procedure Demo
ID
107
Name
Laxmi
SUBMIT
s_id
s_name
101
Diksha
102
Jyoti
103
Keerti
104
Namrata
105
Vaishnavi
106
Siddhi
107
Laxmi

5.Design a webpage to demonstrate the working of parameterized stored procedure.

SEARCH By ID

CREATE NEW Stored Procedure and execute it

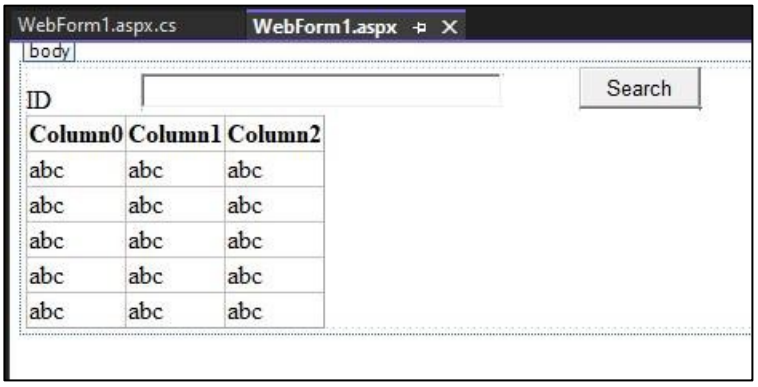


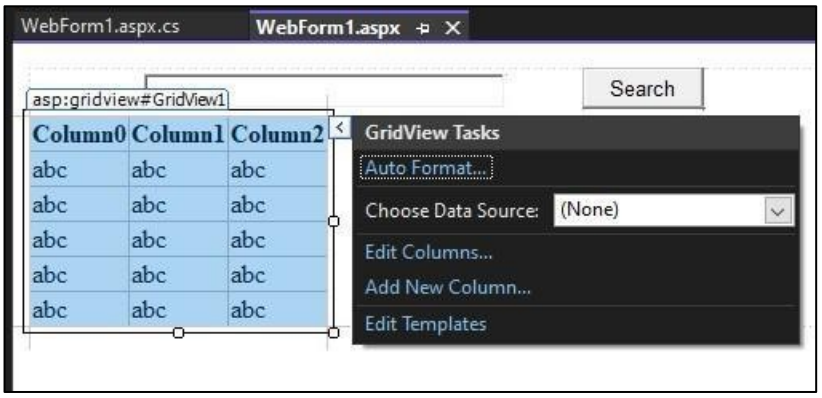

```
--=================================================
--Template generated from Template Explorer using:
--Create Procedure (New Menu).SQL
--
--Use the Specify Values for Template Parameters
--command (Ctrl-Shift-M) to fill in the parameter
--values below.
--
--This block of comments will not be included in
--the definition of the procedure.--=================================================
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--=============================================
--Author:<Author,,Name>
--Create date: <Create Date,,>
--Description:	<Description,,>
--=============================================
CREATE PROCEDURE selection by id
--Add the parameters for the stored procedure here
@s_id int=0
AS
BEGIN
--SET NOCOUNT ON added to prevent extra result sets from
--interfering with SELECT statements.
SET NO COUNT ON;
--Insert statements for procedure here
```

```
SELECT s_id ,s_name
FROM student1 WHERE s_id = @s_id
END
GO
```

CREATE New Web Form

DESIGN:



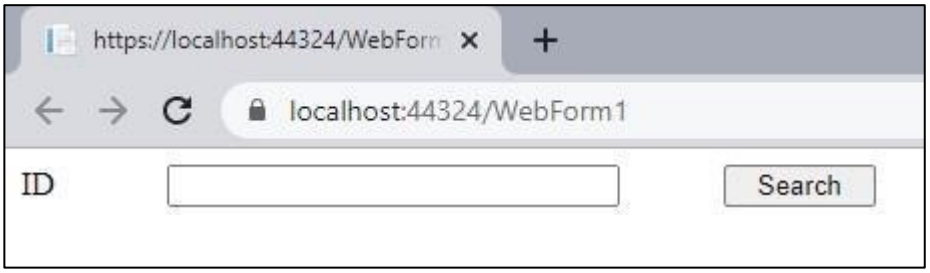


Webform1.aspx.cs

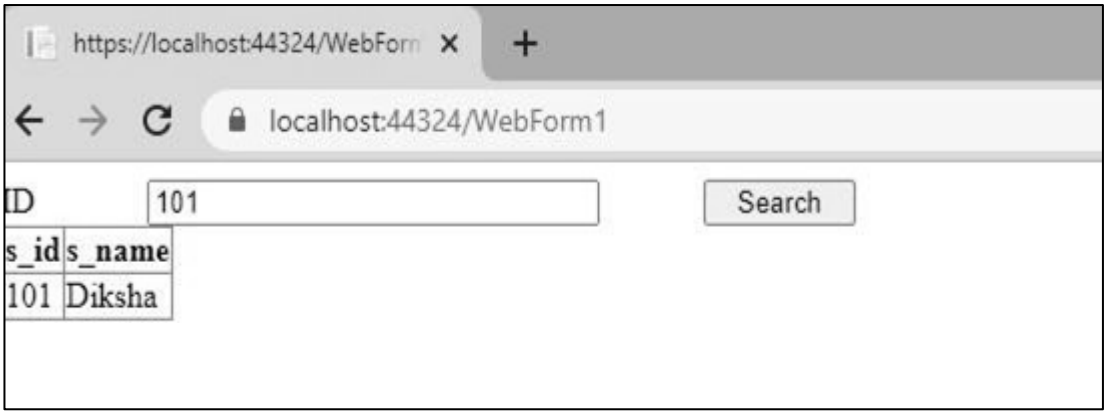
Code:

```
using System;
using System.Collections.Generic;
using System.Linq; using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Data;
namespace WebApplication2
{
public partial class WebForm1 : System.Web.UI.Page
{
SqlConnection conn; SqlCommand cmd;
protected void Page_Load(object sender, EventArgs e) {

}
protected void Button1_Click(object sender, EventArgs e)
{
conn = new SqlConnection(@"Data Source=JSKBS-08\SQLEXPRESS;Initial
Catalog=College1;Integrated Security=True"); SqlCommand cmd = new SqlCommand();
cmd.CommandType = CommandType.StoredProcedure; cmd.CommandText = "selectionbyid";
cmd.Parameters.Add("@S_ID", SqlDbType.Int).Value = TextBox1.Text.Trim();
cmd.Connection = conn;
try {
```

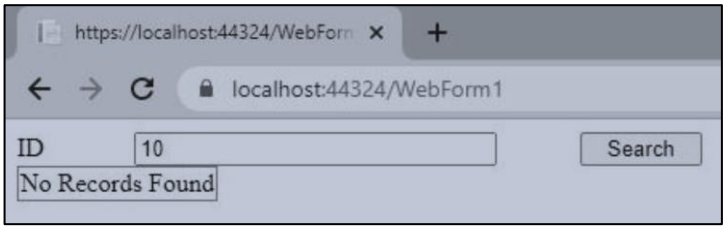
```
conn.Open();
GridView1.EmptyDataText = "No Records Found";
GridView1.DataSource = cmd.ExecuteReader();
GridView1.DataBind();
}
catch (Exception ex)
{
throw ex;
}
} }}
```



Searching Record that are already present



Records that are not Present



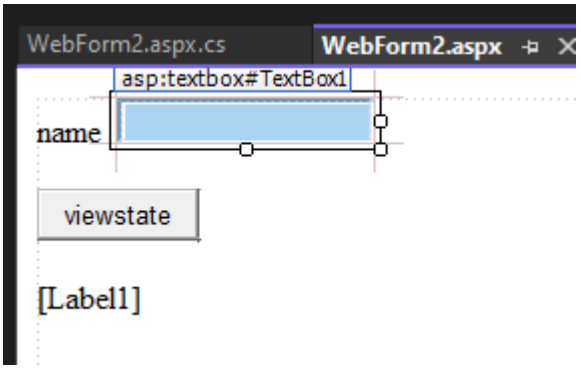
Experiment Number 4: STATE MANAGEMENT

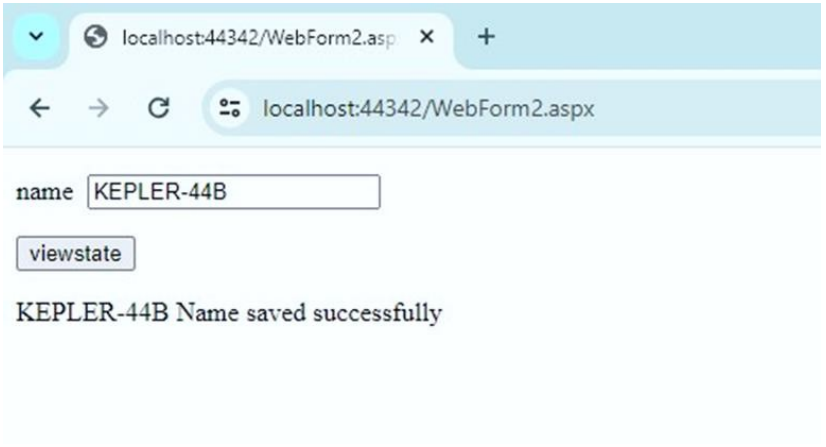
1. View State Example

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Reflection.Emit;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication8
{
   public partial class WebForm2 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if (!IsPostBack)
         {
            if (ViewState["Name"] != null)
            {
               TextBox1.Text = ViewState["Name"].ToString();
            }
         }
      }


      protected void Button1_Click(object sender, EventArgs e)
      {
         ViewState["Name"] = TextBox1.Text;
         Label1.Text = ViewState["Name"] + " " + "Name saved
successfully";
      }

      protected void TextBox1_TextChanged(object sender, EventArgs e)
      {

      }
```

```
    }
}
```

designer


WebForm2.aspx.cs
WebForm2.aspx
asp:textbox#TextBox1
name
viewstate
[Label1]

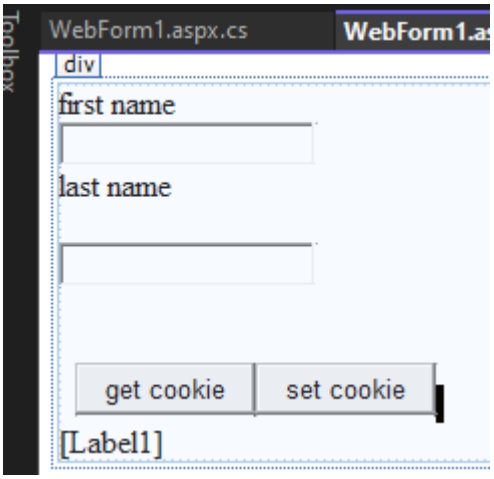

localhost:44342/WebForm2.asp
localhost:44342/WebForm2.aspx
name KEPLER-44B
viewstate
KEPLER-44B Name saved successfully

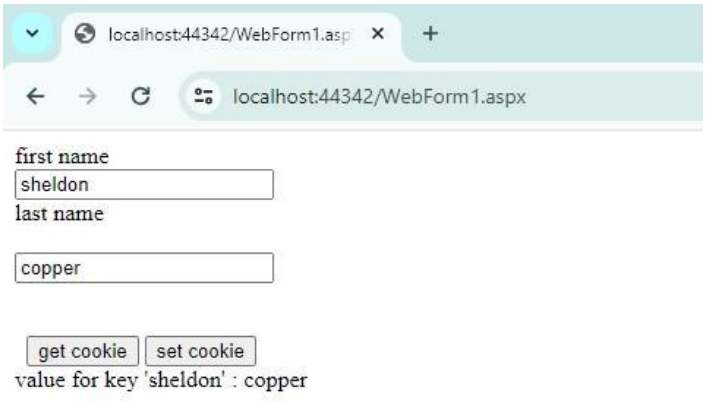
2. Cookies with multiple keys

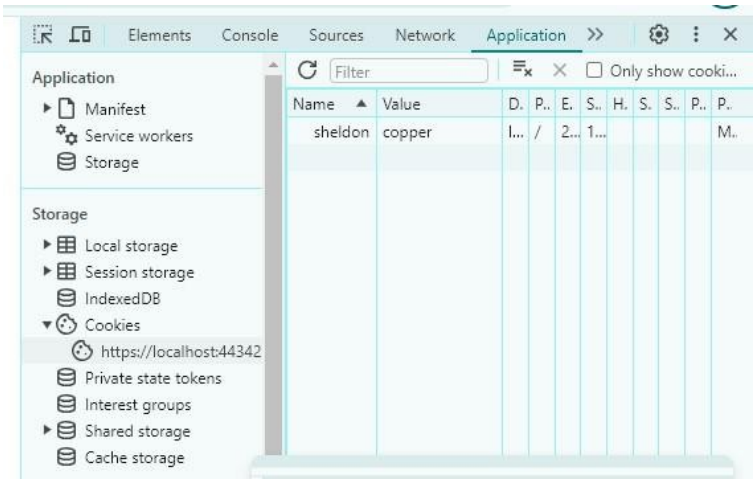
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Reflection.Emit;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication8
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void TextBox1_TextChanged(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         if (!string.IsNullOrEmpty(TextBox1.Text) &&
!string.IsNullOrEmpty(TextBox2.Text))
         {
            Response.Cookies[TextBox1.Text].Value = TextBox2.Text;
            Response.Cookies[TextBox1.Text].Expires =
DateTime.Now.AddDays(1);
         }
      }

      protected void Button2_Click(object sender, EventArgs e)
      {
```

```
        string key = TextBox1.Text;
        if (!string.IsNullOrEmpty(key) && Request.Cookies[key] != null)
        {
            string value = Request.Cookies[key].Value;
            Label1.Text = $"value for key '{key}' : {value}";
        }
        else
        {
            Label1.Text = "Cookie not found.";
        }

    }
  }
}
```

designer






Elements
Console
Sources
Network
Application
Application
Manifest
Service workers
Storage
Storage
Local storage
Session storage
IndexedDB
Cookies
https://localhost:44342
Private state tokens
Interest groups
Shared storage
Cache storage
Filter
Only show cooki...
Name
Value
sheldon
copper

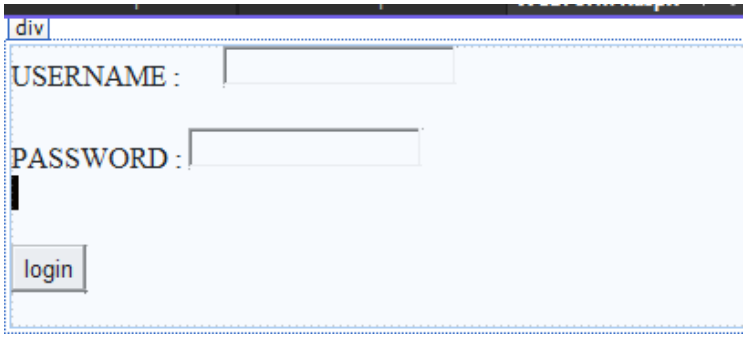
3. Login Application with Cookies

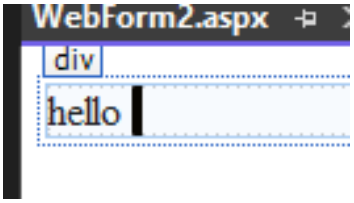
webForm1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication10
{
   public partial class WebForm1 : System.Web.UI.Page
   {

      string uname = "admin";
      string password = "pass";
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         if (TextBox1.Text == uname && TextBox2.Text == password)
         {
            // Redirect to WebForm1 after successful login
            Response.Cookies["authcookie"]["username"] = TextBox1.Text;
            Response.Cookies["authcookie"]["password"] = TextBox2.Text;
            Response.Redirect("WebForm2.aspx"); // Assuming the name of
the page is WebForm1.aspx
         }
         else
```

```
            {
                // Handle incorrect login
                // For simplicity, just displaying a message
                Response.Write("hello!!.");
            }
        }
    }
}
```

design

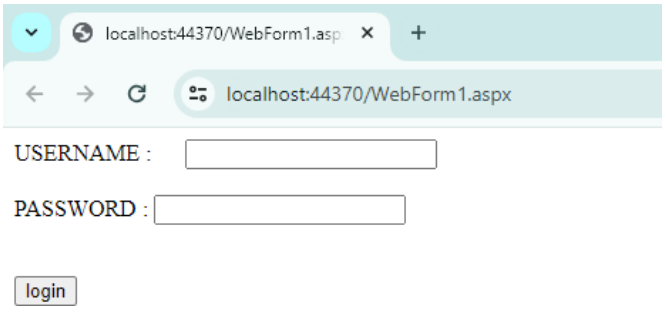


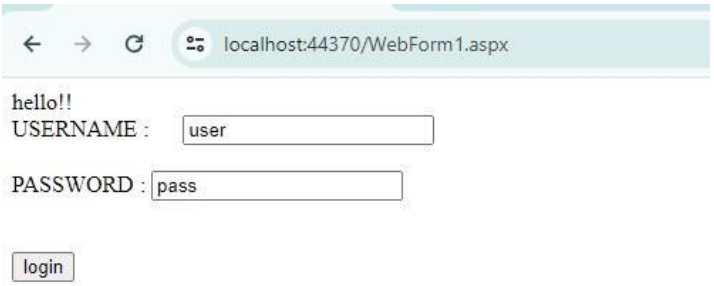


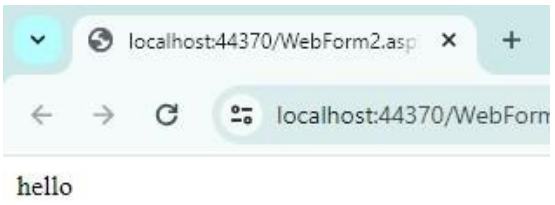

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm2.aspx.cs"
Inherits="WebApplication10.WebForm2" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            hello
```

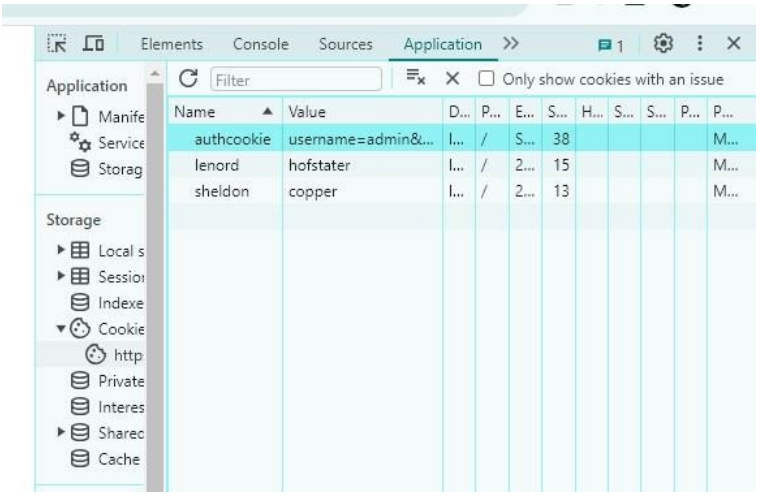
```
        </div>
    </form>
</body>
</html>
```

OUTPUT








Elements
Console
Sources
Application
Application
Filter
Only show cookies with an issue
Storage
Name
Value
authcookie
username=admin&...
lenord
hofstater
sheldon
copper

4. Login Application with Session

Webform1

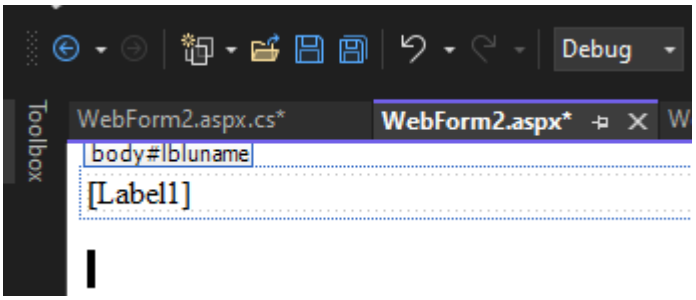
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication12
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         if (TextBox2.Text=="user" && TextBox3.Text=="pass")
         {
            Session["uname"] = TextBox2.Text;
            Response.Redirect("Webform2.aspx");
         }
      }
   }
}
```

Webform2

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

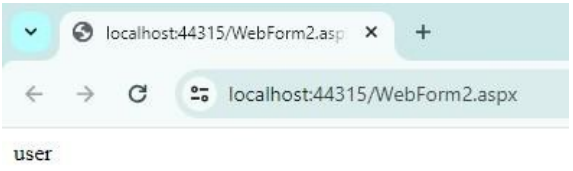
```
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication12
{
   public partial class WebForm2 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
            lbluname.Text= (string)Session["unname"];
      }
   }
   }
```

Design



Output


localhost:44315/WebForm1.asp
localhost:44315/
user name
user
password
pass
login


localhost:44315/WebForm2.asp
localhost:44315/WebForm2.aspx
user

5. Out-of-process Session State

Login.aspx:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="OutOfProcess.Login" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div style="height: 457px; font-size: large;">
            <br />
   UserName:
            <asp:TextBox ID="TextBox1" runat="server" Height="29px"></asp:TextBox>
            <br />
            <br />
                     
            <asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text="Submit" />
        </div>
    </form>
</body>
</html>
```

Login.aspc.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace OutOfProcess
{
    public partial class Login : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            Session["UserName"] = TextBox1.Text.ToString();
            Response.Redirect("Home.aspx");

        }
    }
}
```

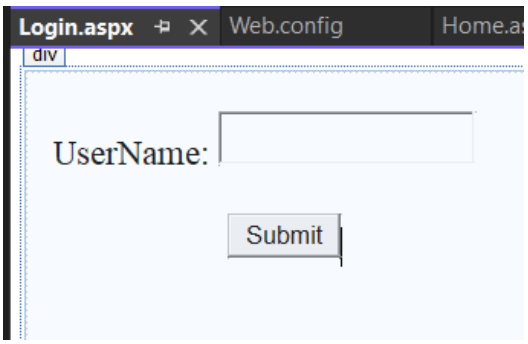
Home.aspx:

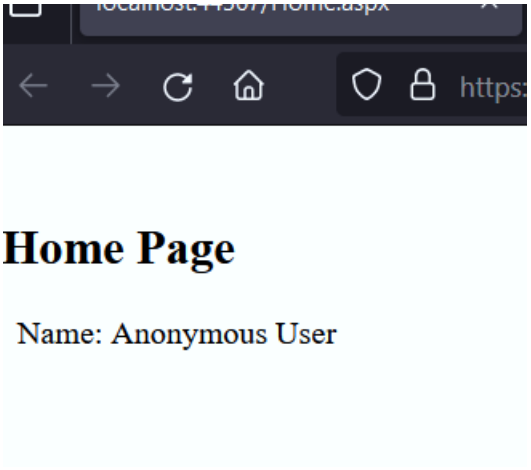
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Home.aspx.cs" Inherits="OutOfProcess.Home" %>
```

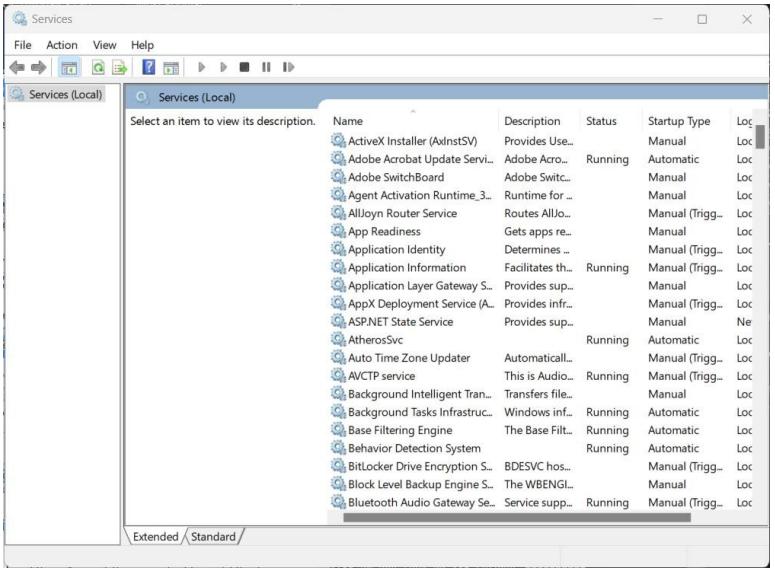
```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div style="height: 421px">
            <br />
            <h2>Home Page</h2>
 
            Name:
            <asp:Label ID="Label1" runat="server"></asp:Label>
        </div>
    </form>
</body>
</html>
```

Home.aspx.cs:

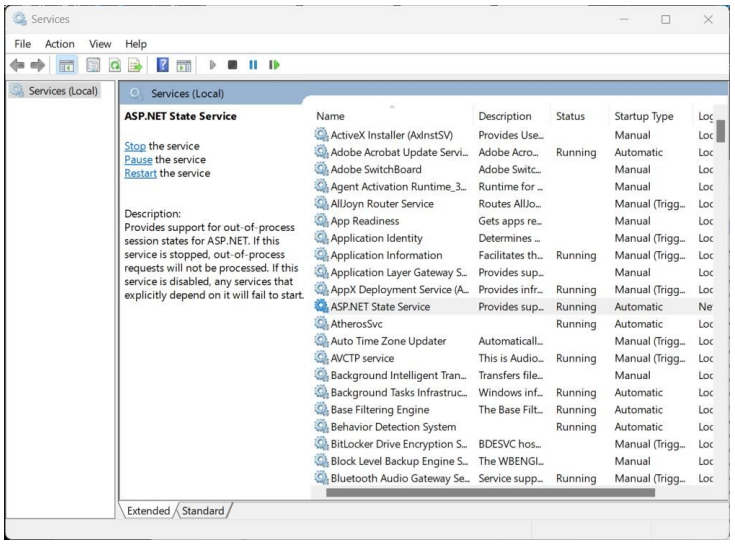
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace OutOfProcess
{
    public partial class Home : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (Session["UserName"] != null && Session["UserName"] != "")
            {
                Label1.Text = Session["UserName"].ToString();
            }
            else
            {
                Label1.Text = "Anonymous User";
            }

        }
    }
}
```





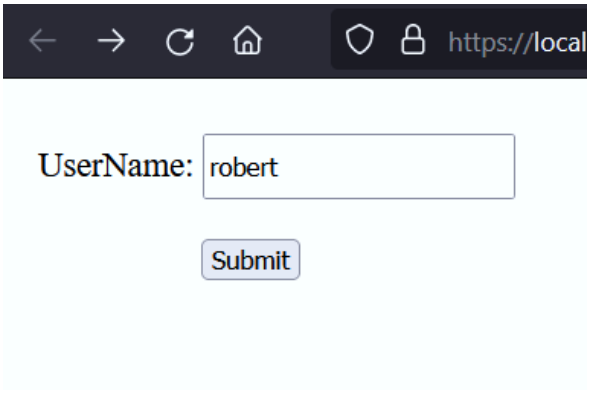


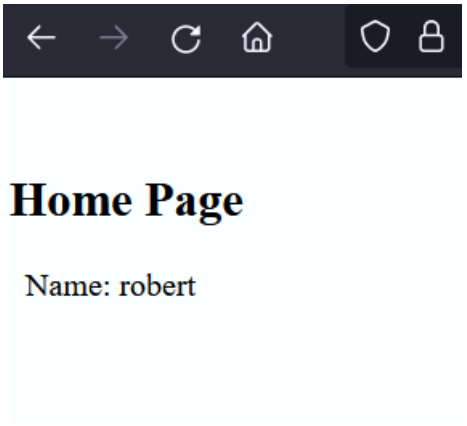
ASP.NET State Services → Start (Right Click) → Automatic(Properties)



Web.config:

```
<system.web>
  <sessionState mode="StateServer" stateConnectionString="tcpip=127.0.0.1:42424"></sessionState>
  <compilation debug="true" targetFramework="4.7.2" />
   <httpRuntime targetFramework="4.7.2" />
 </system.web>
```

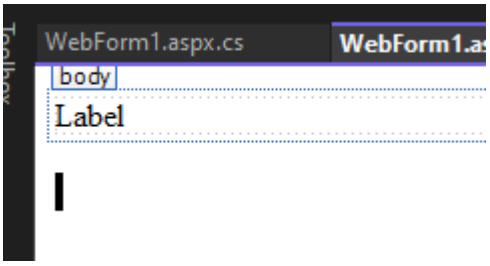




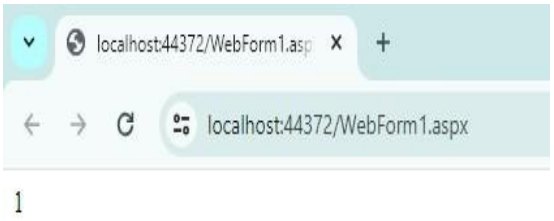
6. Application State

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication13
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         int count = 0;
         if (Application["pagecount"] != null)
         {
             count = Int32.Parse(Application["pagecount"].ToString());

         }
         {
            count= count + 1;
            Label1.Text= count.ToString();
            Application["pagecount"]=count.ToString();
         }
      }
   }
}
```

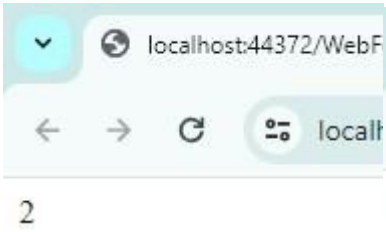
Designer

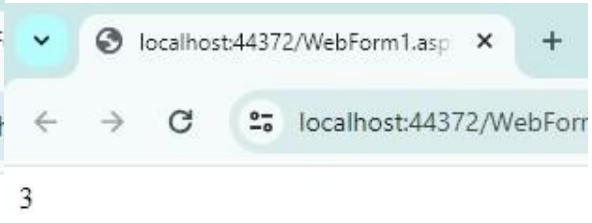


Output



Every time you click on refresh it shows the number of times you visited the webpage.



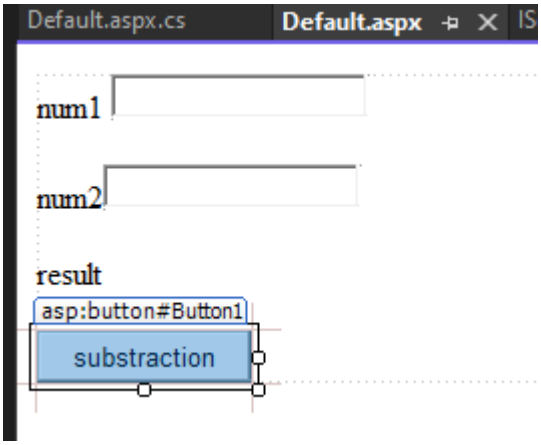


Experiment Number 5: WEB SERVICES AND WCF

1. Create an application to show the use of web services.

Default .aspx.cs

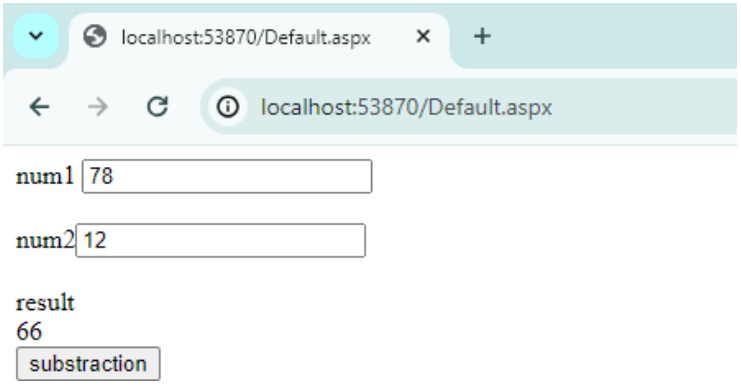
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

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

   }

   protected void Button1_Click(object sender, EventArgs e)
   {
      ServiceReference1.ServiceClient svc = new
ServiceReference1.ServiceClient();
      Double a = Convert.ToDouble(TextBox1.Text);
      Double b = Convert.ToDouble(TextBox2.Text);
      Double ans = svc.Addition(a, b);
      Label1.Text = ans.ToString();
   }
        }
```




```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService" in both code and config file together.
[ServiceContract]
public interface IService
{

        [OperationContract]

        string GetData(int value);
    [OperationContract]
    Double Addition(Double a, Double b);
    [OperationContract]
        CompositeType GetDataUsingDataContract(CompositeType composite);

        // TODO: Add your service operations here
}
```

```
// Use a data contract as illustrated in the sample below to add composite types to service operations.
[DataContract]
public class CompositeType
{
        bool boolValue = true;
        string stringValue = "Hello ";

        [DataMember]
        public bool BoolValue
        {
                get { return boolValue; }
                set { boolValue = value; }
        }

        [DataMember]
        public string StringValue
        {
                get { return stringValue; }
                set { stringValue = value; }
        }
}


using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the class name "Service" in code, svc and config file together.
public class Service : IService
{
        public string GetData(int value)
```

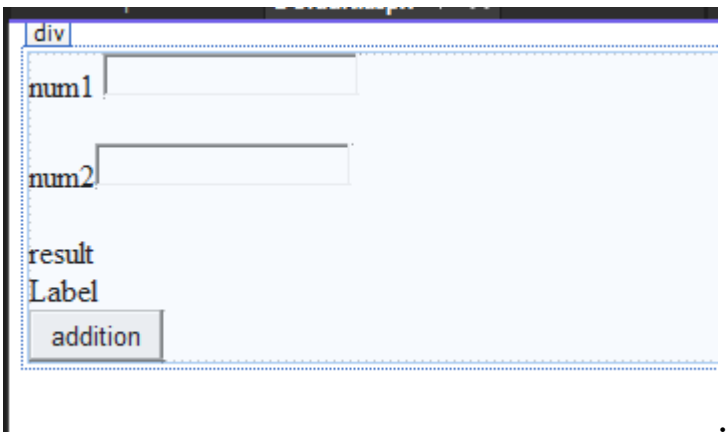
```
        {
            return string.Format("You entered: {0}", value);
        }
    public Double Addition(Double a, Double b)
    {
       return (a - b);
    }
    public CompositeType GetDataUsingDataContract(CompositeType
composite)
        {
            if (composite == null)
            {
                throw new ArgumentNullException("composite");
            }
            if (composite.BoolValue)
            {
                composite.StringValue += "Suffix";
            }
            return composite;
        }
}
```



2. Create an application to show the use of WCF

Default.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

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

    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        ServiceReference1.ServiceClient svc = new
ServiceReference1.ServiceClient();
        Double a = Convert.ToDouble(TextBox1.Text);
        Double b = Convert.ToDouble(TextBox2.Text);
        Double ans = svc.Addition(a, b);
        Label1.Text = ans.ToString();
    }
}
```



Iservice.cs

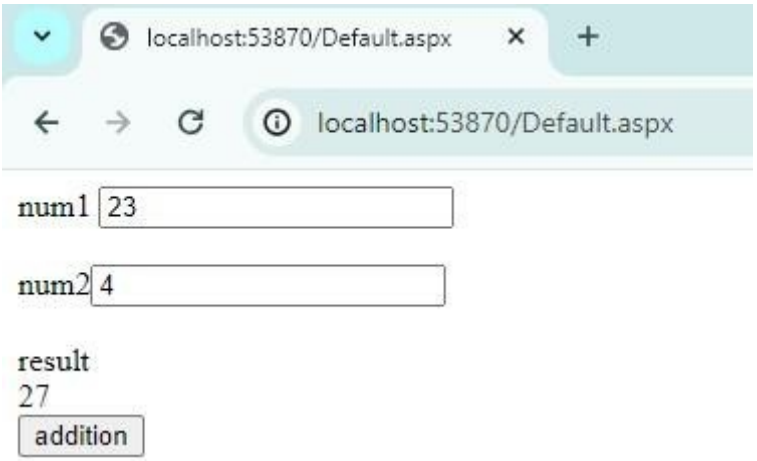
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService" in both code and config file together.
[ServiceContract]
public interface IService
{

        [OperationContract]

        string GetData(int value);
    [OperationContract]
    Double Addition(Double a, Double b);
    [OperationContract]
        CompositeType GetDataUsingDataContract(CompositeType composite);

        // TODO: Add your service operations here
}
```

```
// Use a data contract as illustrated in the sample below to add composite types to service operations.
[DataContract]
public class CompositeType
{
        bool boolValue = true;
        string stringValue = "Hello ";

        [DataMember]
        public bool BoolValue
        {
                get { return boolValue; }
                set { boolValue = value; }
        }

        [DataMember]
        public string StringValue
        {
                get { return stringValue; }
                set { stringValue = value; }
        }
}
```

Service .cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the class name "Service" in code, svc and config file together.
public class Service : IService
{
```

```
        public string GetData(int value)
        {
                return string.Format("You entered: {0}", value);
        }
    public Double Addition(Double a, Double b)
    {
       return (a + b);
    }
    public CompositeType GetDataUsingDataContract(CompositeType
composite)
        {
                if (composite == null)
                {
                        throw new ArgumentNullException("composite");
                }
                if (composite.BoolValue)
                {
                        composite.StringValue += "Suffix";
                }
                return composite;
        }
}
```



Experiment Number 6: MVC and LINQ

1.Create an MVC based application.

Index.cshtml

```
<form>
@model List<SimpleMvcApp1.Models.item>
<h2>items</h2>
<ul>
   @foreach(var item in Model)
        {
          <li>@item.Name</li>
        }
</ul>

   <h2>Add New Item</h2>
<form action="@Url.Action("Add")"method="post"></form>
<input type="submit"value="Add"/>



<h2>Index</h2>

   </form>
```

Item controller.cs

```
<form>
@model List<SimpleMvcApp1.Models.item>
<h2>items</h2>
<ul>
   @foreach(var item in Model)
        {
```

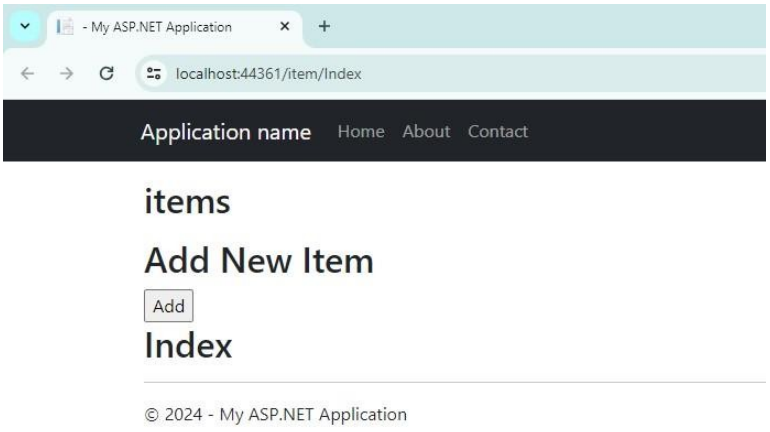
```
<li>@item.Name</li>
```

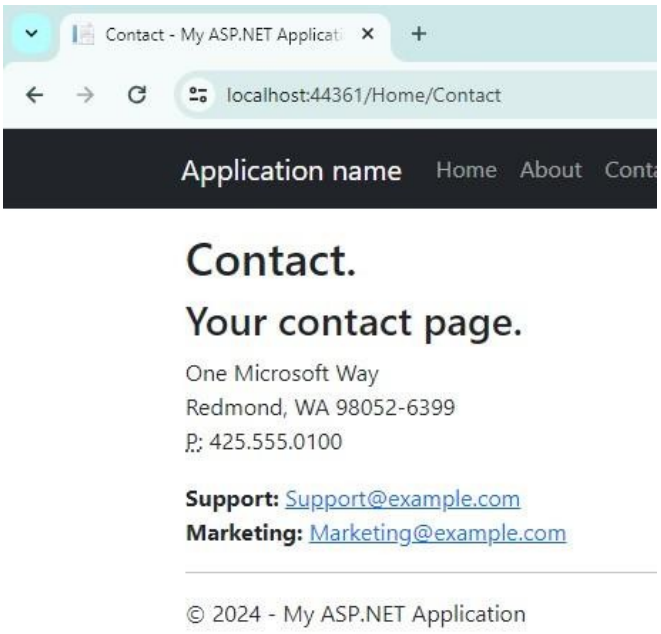
```
        }
</ul>
   <h2>Add New Item</h2>
<form action="@Url.Action("Add")"method="post"></form>
<input type="submit"value="Add"/>



<h2>Index</h2>

   </form>
```

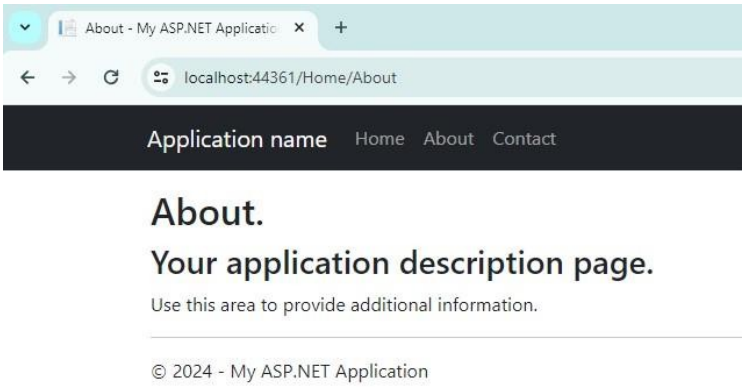
Item.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace SimpleMvcApp1.Models
{
   public class item
   {
      public int Id { get; set; }

      public string Name { get; set; }
   }
}
```

A sample web page(template) is shown we can add required features and customize our website


- My ASP.NET Application
localhost:44361/item/Index
Application name
Home
About
Contact
items
Add New Item
Add
Index
© 2024 - My ASP.NET Application

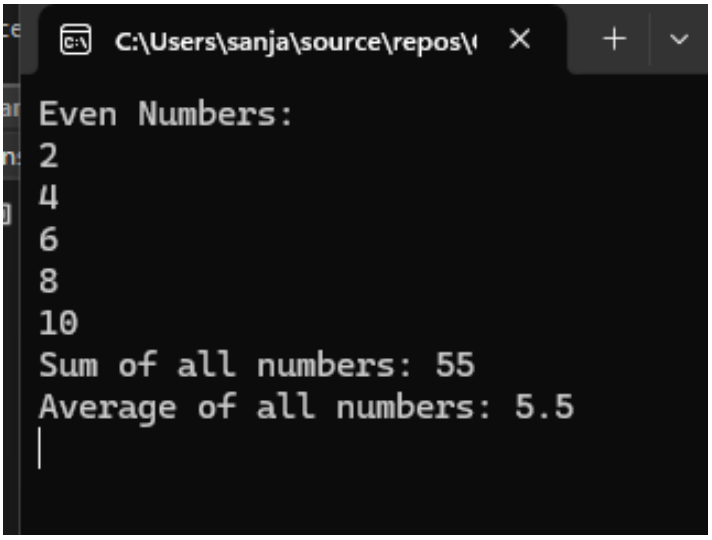

Contact - My ASP.NET Applicat
localhost:44361/Home/Contact
Application name
Home
About
Cont
Contact.
Your contact page.
One Microsoft Way
Redmond, WA 98052-6399
P: 425.555.0100
Support: Support@example.com
Marketing: Marketing@example.com
© 2024 - My ASP.NET Application


About - My ASP.NET Applicatic
localhost:44361/Home/About
Application name
Home
About
Contact
About.
Your application description page.
Use this area to provide additional information.
© 2024 - My ASP.NET Application

2. Create an application to show the use of LINQ

```
 using System;
using System.Collections.Generic;
using System.Linq;

namespace SimpleLINQConsoleApp
{
   class Program
   {
      static void Main(string[] args)
      {
         // Sample data
         List<int> numbers = new List<int> { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

         // LINQ query to filter even numbers
         var evenNumbers = from num in numbers
                           where num % 2 == 0
                           select num;

         // Display even numbers
         Console.WriteLine("Even Numbers:");
         foreach (var number in evenNumbers)
         {
            Console.WriteLine(number);
         }

         // LINQ query to find the sum of all numbers
         var sum = numbers.Sum();

         // Display sum
         Console.WriteLine($"Sum of all numbers: {sum}");

         // LINQ query to find the average of all numbers
         var average = numbers.Average();

         // Display average
         Console.WriteLine($"Average of all numbers: {average}");
```

```
            // Wait for user input before closing the console window
            Console.ReadLine();
        }
    }
}
```



Assignments

Assignment 1

PART - I : CONSOLE APPLICATION

Aim: Write a program to create a class department which has department information and also create an employee class which inherits the department and displays all the information.

A console application, in the context of C#, is an application that takes input and displays output at a command line console with access to three basic data streams: standard input, standard output and standard error.

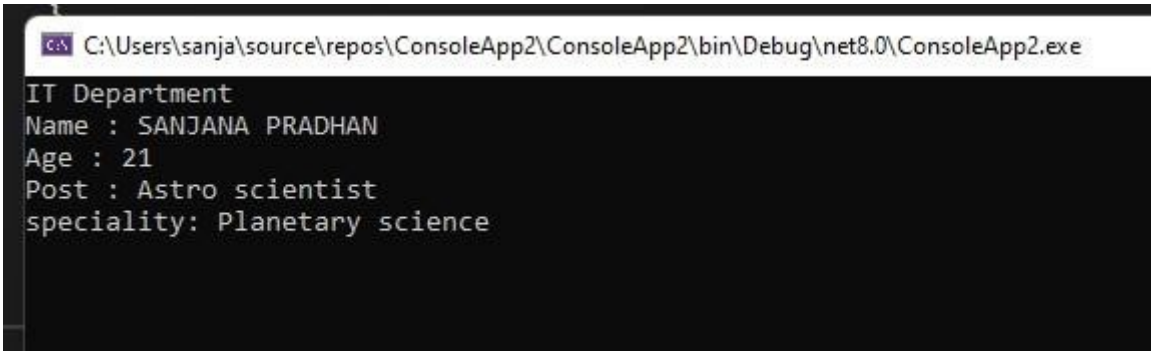
A console application facilitates the reading and writing of characters from a console – either individually or as an entire line. It is the simplest form of a C# program and is typically invoked from the Windows command prompt. A console application usually exists in the form of a stand-alone executable file with minimal or no graphical user interface (GUI).

Console Application example

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Console_Department
{
   class Department
   {
```

```
        public void display()
        {
            Console.WriteLine("IT Department");
        }
    }
    class Employee : Department
    {
        public void disemp()
        {
            Console.WriteLine("Name : SANJANA PRADHAN");
            Console.WriteLine("Age : 21");
            Console.WriteLine("Post : Astro scientist ");
            Console.WriteLine("speciality: Planetary science");

            Console.ReadLine();
        }
        class Program
        {
            private static void Main(string[] args)
            {
                Employee myobj = new Employee();
                myobj.display();
                myobj.disemp();
            }
        }
    }
}
```

Output



Aim : Write a program to create an abstract class named employee which has an abstract method salary and having 2 subclasses developer and non-developer. Calculate salary of both (developer and non-developer.)

Abstract Class:

Abstract class is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class). Abstract method: can only be used in an abstract class, and it does not have a body. The body is provided by the derived class (inherited from).

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

namespace Console_Abstract_Devloper_NonDeveloper
{

    public abstract class Employee
    {
        public abstract int salary();

    }

    public class developer : Employee
    {
        int sal;
        public developer(int s)
        {
            sal = s + 5000;
        }
        public override int salary()
        {
            return sal;
        }
```

```
        public class Nondeveloper : Employee
        {
            int sal;
            public Nondeveloper(int s)
            {
                sal = s + 1000;
            }
            public override int salary()
            {
                return sal;
            }


        class Program
        {
            static void Main(string[] args)
            {
                Nondeveloper non = new Nondeveloper(20000);
                developer dev = new developer(49000);

                Console.WriteLine("Non-Developer Salary: " + non.salary());
                Console.WriteLine("Developer Salary: " + dev.salary());
                Console.ReadLine();
            }
        }
    }
}
```

Aim: Write a program for interface.

Like a class, Interface can have methods, properties, events, and indexers as its members. But interfaces will contain only the declaration of the members. The implementation of the interface's members will be given by class who implements the interface implicitly or explicitly.

- Interfaces specify what a class must do and not how.
- Interfaces can't have private members.
- By default all the members of Interface are public and abstract.
- The interface will always defined with the help of keyword 'interface'.
- Interface cannot contain fields because they represent a particular implementation of data.
- Multiple inheritance is possible with the help of Interfaces but not with classes.

Objective: To demonstrate Interface

```
using System;
using System.Collections.Ge
neric; using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace ConsoleApp3
{


    interface IAnimal    // interface
    {
        void animaleat();
    }
```

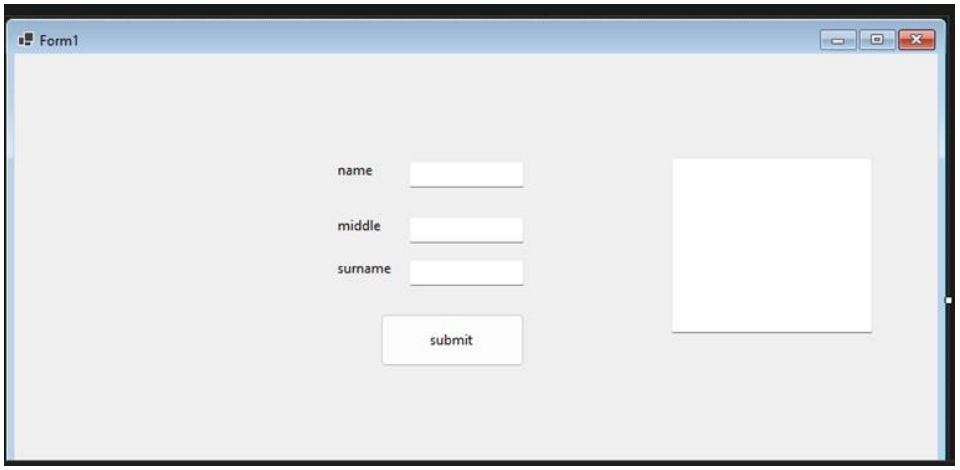
```
class Horse : IAnimal
{
    public void animaleat()
    {

        Console.WriteLine("The Horse eat
        Grass"); Console.ReadLine();
    }
}

class Program
{
    static void Main(string[] args)

    {
        Horse myhorse = ne
        w Horse();
        myhorse.animaleat();
    }
}
}
```

WINDOWS APPLICATION

Aim: Write a program to design a form that contains employee information (Id, Name, department, city). By clicking view data is displayed in the list box. Create an employee class which has different properties.

Windows Forms is a Graphical User Interface(GUI) class library which is bundled in .Net Framework. Its main purpose is to provide an easier interface to develop the applications for desktop, tablet, PCs. It is also termed as the **WinForms**. The applications which are developed by using Windows Forms or WinForms are known as the **Windows Forms Applications** that runs on the desktop computer. WinForms can be used only to develop the Windows Forms Applications not web applications. WinForms applications can contain the different type of controls like labels, list boxes, tooltip etc.

Design



Form1.CS

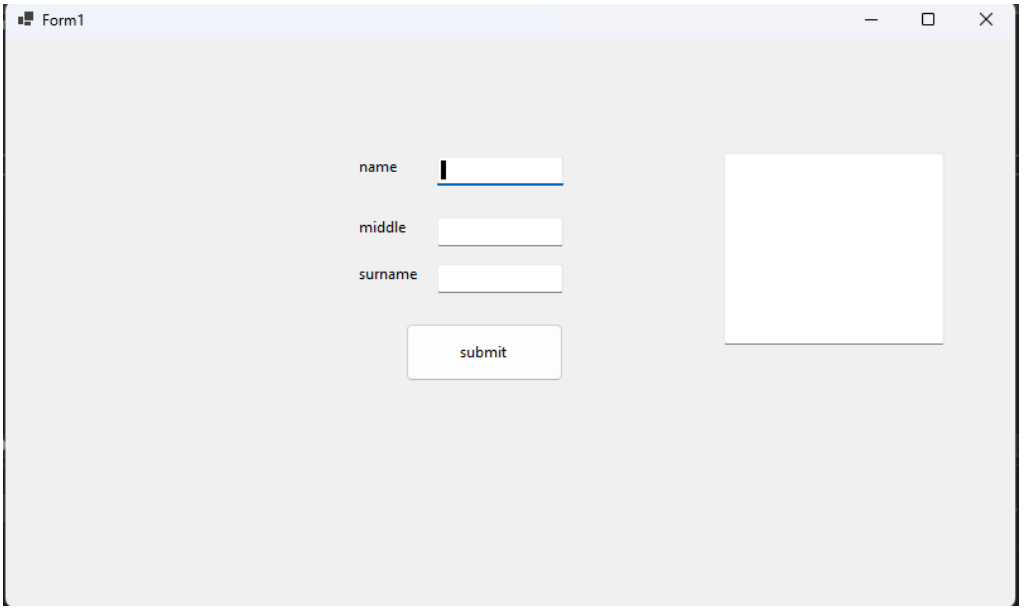
```
using System;
using System.Collections.Gen
eric;
using System.ComponentMo
del; using System.Data;
using System.Dra
wing;
using System.Linq
;
using System.Text
;
using System.Threading.Ta
sks;
```

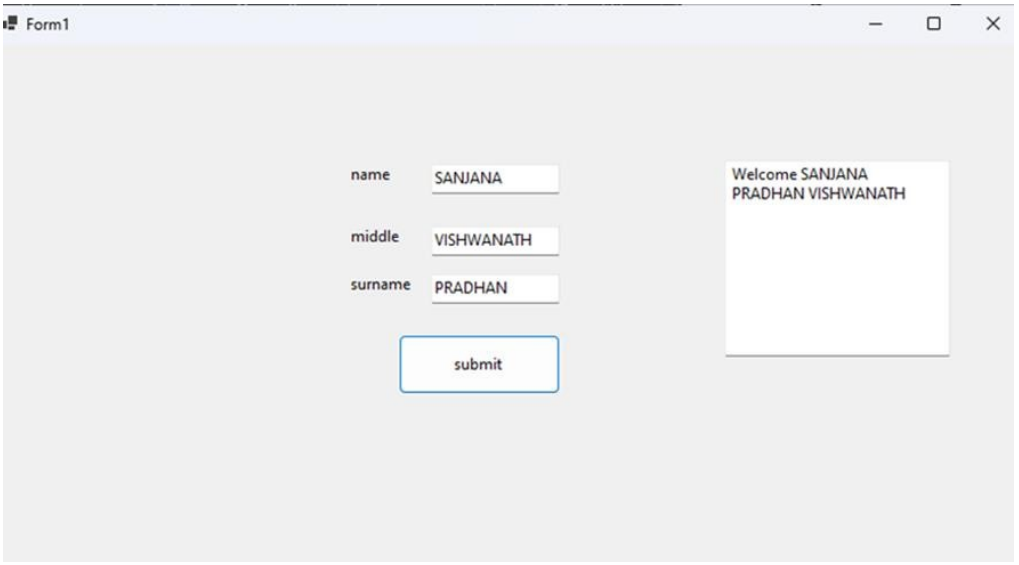
```
using System.Windows.For
ms;

namespace WindowsForms_EmployeeRcrd
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Class1 personobject = new Class1
            (); personobject.Dept_ID
                = textBox1.Text;
            personobject.Name = textBox2.Tex
            t;
            personobject.Department = textBox
            3.Text;
            personobject.City = textBox4.Text;


            listBox1.Items.Add(personobject.Dept_ID);
            listBox1.Items.Add(personobject.Name);
            listBox1.Items.Add(personobject.Department);
            listBox1.Items.Add(personobject.City);
        }
    }
}
```

Class1.CS

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

namespace WindowsForms_EmployeeRcrd
{
    class Class1
    {
```

```
        private string dept_id ;
        private string name;
        private string departme
        nt ; private string city;

        public string Dept_ID
        {
            get { return dept_id ;
             } set { dept_id
                = value; }
        }

        public string Name
        {
            get { return name
            ; }
            set { name = val
            ue; }
        }

        public string Department
        {
            get { return department
             ; }
            set { department = val
            ue; }
        }

        public string City
        {
            get { return city;
            }
            set { city = value
            ; }
        }

    }
}
```


Form1
name
middle
surname
submit


Form1
name
SANJANA
middle
VISHWANATH
surname
PRADHAN
submit
Welcome SANJANA
PRADHAN VISHWANATH

2. Aim: Write a program to design and perform following operations.

a. **Create multiple(at least 2 form) inherited form**
b. **First form for general information of MCA course.**
c. **Second form for information of first year.**

Form1.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsForms_Inheritance_A1_2
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
}

private void button1_Click(object sender, EventArgs e)
{
GeneralMcaInformation g = new GeneralMcaInformation(); g.Show();
}

private void button2_Click(object sender, EventArgs e)
{
FirstYearMca2 f = new FirstYearMca2(); f.Show();
}
}
}
```

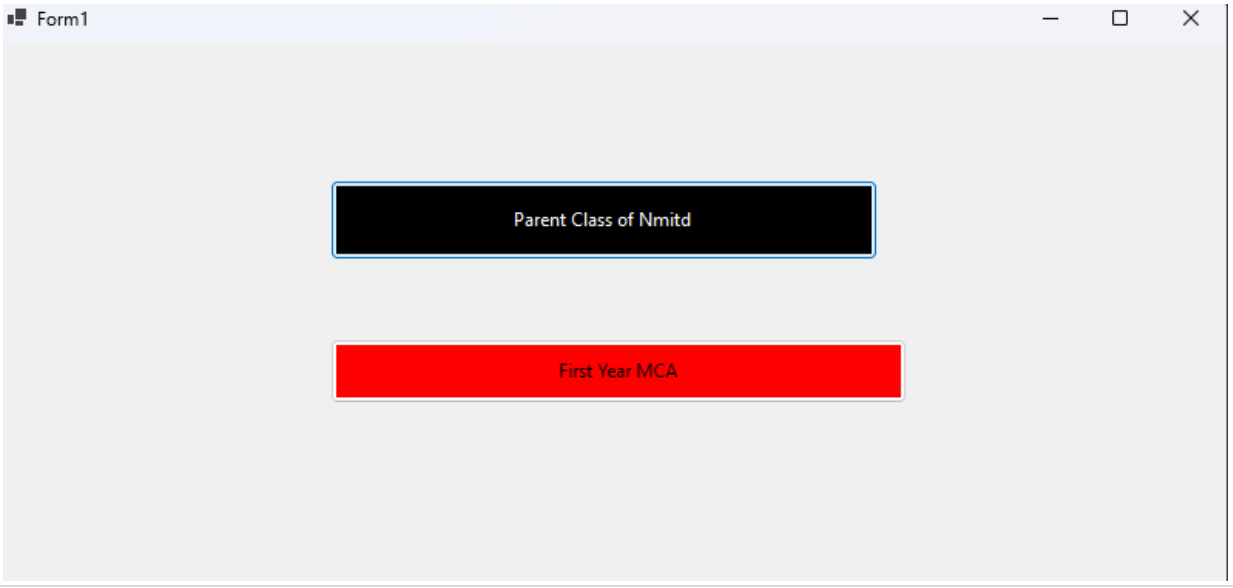
GeneralMcaInformation.CS (Parent Form) :

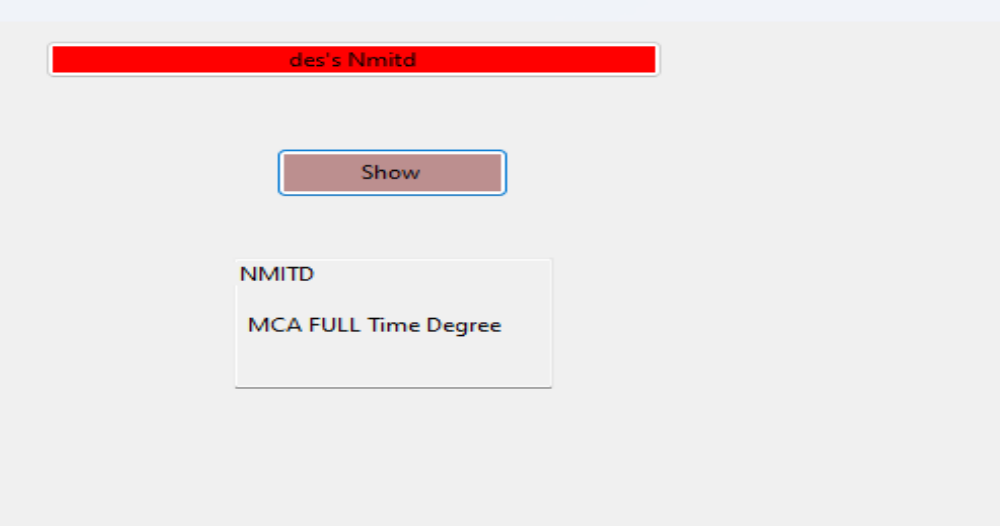
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;+
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsForms_Inheritance_A1_2
{
    public partial class GeneralMcaInformation : Form
    {
        public GeneralMcaInformation()
        {
            InitializeComponent();
        }

        private void label1_Click(object sender, EventArgs e)
        {
```

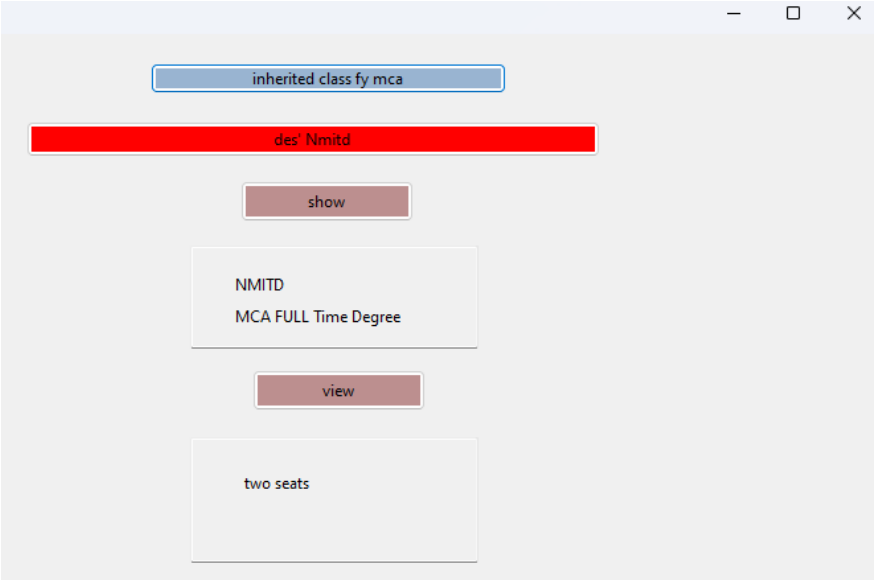
```
        }

        private void button1_Click(object sender, EventArgs e)
        {
            listBox1.Items.Add("des'Nmitd"); listBox1.Items.Add("
                "); listBox1.Items.Add("Mca Full time Degree");
            listBox1.Items.Add("      ");
    }
}
```

FirstYearMca.CS (Inherited Form) :

```
using System;
using System.Collections.Generic; using System.ComponentModel;
using System.Data;
  using System.Drawg;
  using System.Text;
  using System.Windows.Forms;

  namespace WindowsForms_Inheritance_A1_2
  {
      public partial class FirstYearMca2 : WindowsForms_Inheritance_A1_2.GeneralMcaInformation
      {
          public FirstYearMca2()
          {
              InitializeComponent();
          }

          private void button2_Click(object sender, EventArgs e)
          {
              listBox2.Items.Add("two seats ");

          }
      }
}
```




des's Nmitd
Show
NMITD
MCA FULL Time Degree


inherited class fy mca
des' Nmitd
show
NMITD
MCA FULL Time Degree
view
two seats

PART – III : WEB APPLICATION

Web Application:

ASP.NET is an open-source, server-side web-application framework designed for web development to produce dynamic web pages. It was developed by Microsoft to allow programmers to build dynamic web sites, applications and services. NET Framework and is the successor to Microsoft's Active Server Pages (ASP) technology.
Web Application: In C# ASP.NET, a web application refers to a dynamic and interactive software solution that incorporates server-side logic to process user requests and generate dynamic content. It typically follows a multi-tier architecture, involving separate layers for presentation, business logic, and data access

Web.SiteMap

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
<siteMapNode url="~/Home.aspx" title="Gmail"  description="Login to Gmail Account">
<siteMapNode url="~/FirstPage.aspx" title="Facebook"  description="Login to
Facebbok Account" >
<siteMapNode url="~/SecondPage.aspx" title="Insta"  description="Login to
Instagram Account" />
</siteMapNode>
<siteMapNode url="~/LastPage.aspx" title="Yahoo"  description="Welcome To
Yahoo" />
</siteMapNode>
</siteMap>
```

WebConfig :-

```
<?xml version="1.0" encoding="utf-8"?>
<!--
For more information on how to configure your ASP.NET application, please visit
https://go.microsoft.com/fwlink/?LinkId=169433
-->
<configuration>
<system.web>
<compilation debug="true" targetFramework="4.7.2" />
<httpRuntime targetFramework="4.7.2" />
```

Web.SiteMap

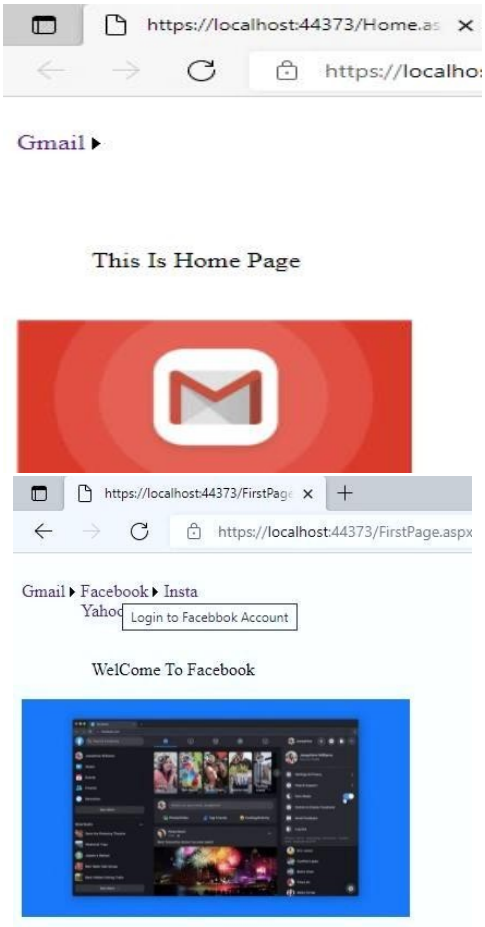
```
<?xml version="1.0" encoding="utf-8" ?>

<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
```

```
<siteMapNode url="~/Home.aspx" title="Gmail" description="Login to Gmail Account">
<siteMapNode url="~/FirstPage.aspx" title="Facebook" description="Login to Facebbok Account"
>
<siteMapNode url="~/SecondPage.aspx" title="Insta" description="Login to Instagram Account"
/>
</siteMapNode>
<siteMapNode url="~/LastPage.aspx" title="Yahoo" description="Welcome To Yahoo" />
</siteMapNode>
</siteMap>
```

WebConfig :-

```
<?xml version="1.0" encoding="utf-8"?>
<!--
For more information on how to configure your ASP.NET application, please visit
https://go.microsoft.com/fwlink/?LinkId=169433
-->
<configuration>
<system.web>
<compilation debug="true" targetFramework="4.7.2" />
<httpRuntime targetFramework="4.7.2" />
<siteMap>
<providers>
<remove name="MySqlSiteMapProvider"/>
</providers>
</siteMap>
</system.web>
<system.codedom>
```

```
<compilers>

<compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default
/nowarn:1659;1699;1701" />

<compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default
/nowarn:41008 /define:_MYTYPE=\&quot;Web\&quot;

/optionInfer+" />

</compilers>

</system.codedom>

</configuration>
```




https://localhost:44373/SecondP
https://localhost:44373/SecondP
Gmail Facebook Insta
Yahoo
Login to Instagram Account
Instagram Home
https://localhost:44373/LastPage
https://localhost:44
Gmail Facebook
Yahoo
Welcome To Yahoo
Welcome to Yahoo
YAHOO!

2. **Aim: Create a website using the master page concept and content pages also use navigation control on that.**

Objective: To demonstrate Master Pages Concept.

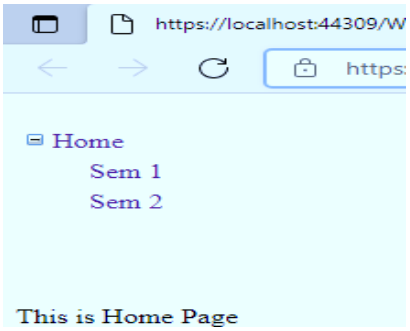
Web.SiteMap

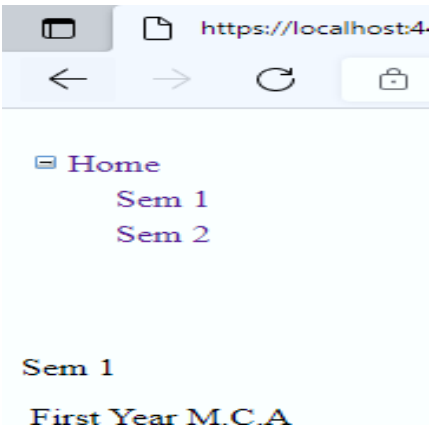
```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
  <siteMapNode url="~/WebForm1.aspx" title="Home"
  description="First Year of Mca">
    <siteMapNode url="~/WebForm2.aspx" title="Sem 1"
  description="Sem 1 of First Year" />
    <siteMapNode url="~/WebForm3.aspx" title="Sem 2"
  description="Sem 2 of First Year" />
  </siteMapNode>
</siteMap>
```

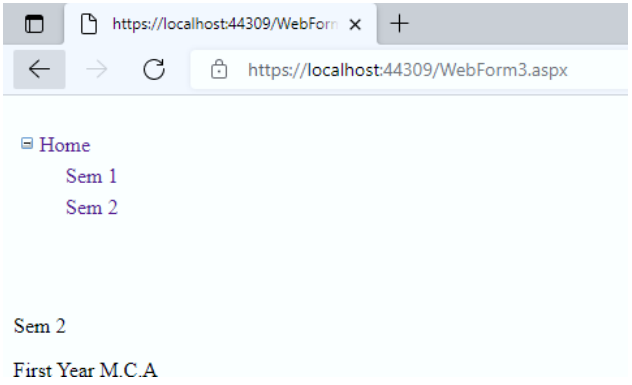
WebConfig :

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  https://go.microsoft.com/fwlink/?LinkId=169433
  -->
 <configuration>
   <system.web>
     <compilation debug="true" targetFramework="4.7.2" />
     <httpRuntime targetFramework="4.7.2" />
          <siteMap>
                <providers>
                      <remove name="MySqlSiteMapProvider"/>
                </providers>
          </siteMap>
   </system.web>
   <system.codedom>
     <compilers>
       <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutra
l, PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:1659;1699;1701" />
       <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutra
l, PublicKeyToken=31bf3856ad364e35" warningLevel="4"
compilerOptions="/langversion:default /nowarn:41008 /define:_MYTYPE=\&quot;Web\&
quot;
/optionInfer+" />
```

```
    </compilers>
  </system.codedom>
</configuration>
```







3. Design an ASP.NET Application to Display Random Advertisements using ADRotator Control. Use XML DataSource and image folder.

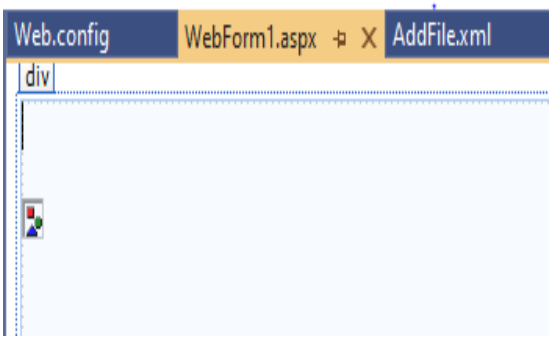
Objective: To demonstrate AdRotator Control

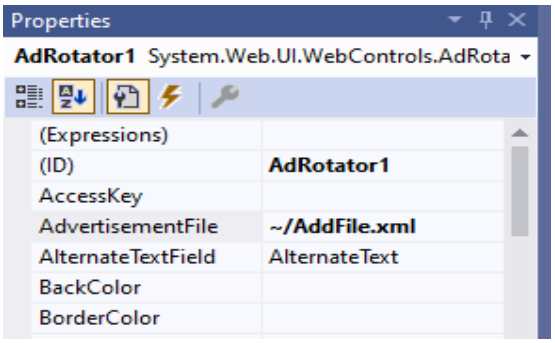
AdRotator:

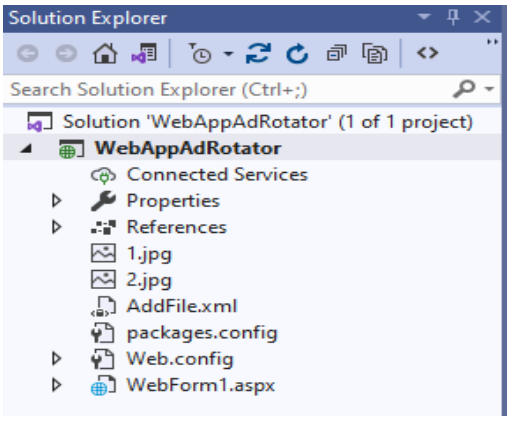
The AdRotator control randomly selects banner graphics from a list, which is specified in an external XML schedule file. This external XML schedule file is called the advertisement file.

The AdRotator control allows you to specify the advertisement file and the type of window that the link should follow in the AdvertisementFile and the Target property respectively.

DESIGN :







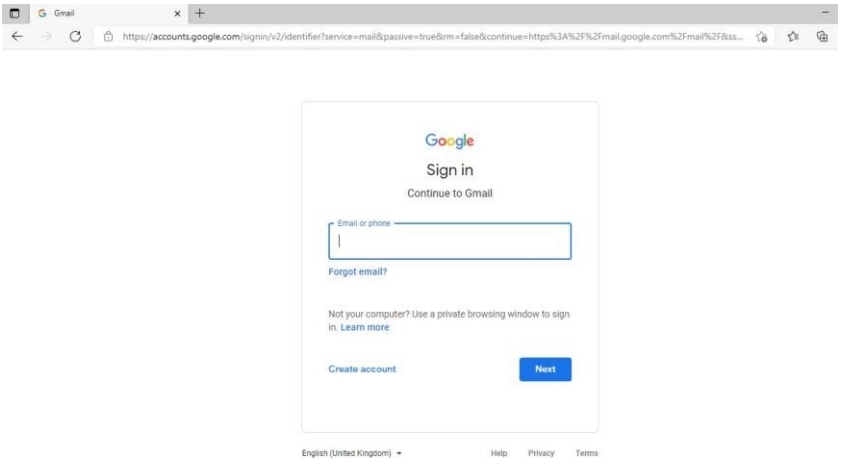
Code :

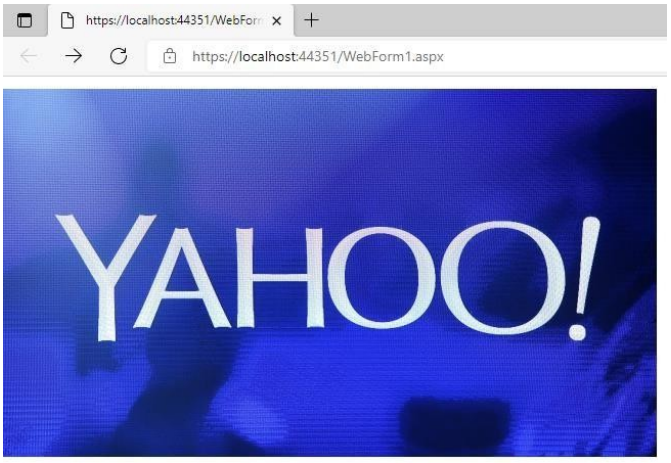
AddFile.Xml :

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>
        <Ad>
                <ImageUrl>1.jpg</ImageUrl>
                <NavigateUrl>http://www.gmail.com</NavigateUrl>
                <AlternateText>Gmail.com</AlternateText>
                <Impressions>20</Impressions>
                <Keyword>Gmail</Keyword>
        </Ad>

        <Ad>
                <ImageUrl>2.jpg</ImageUrl>
                <NavigateUrl>http://www.yahoo.com</NavigateUrl>
                <AlternateText>Yahoo.com</AlternateText>
                <Impressions>20</Impressions>
                <Keyword>David</Keyword>
        </Ad>

</Advertisements>
```

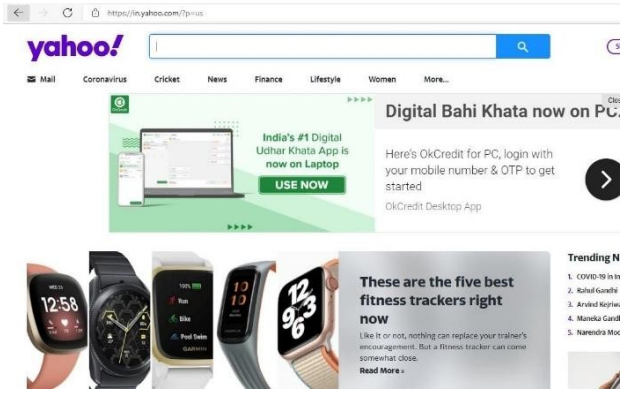
Output :-

After Click Image Gmail this will Open :





After Click Image Yahoo! this Open :



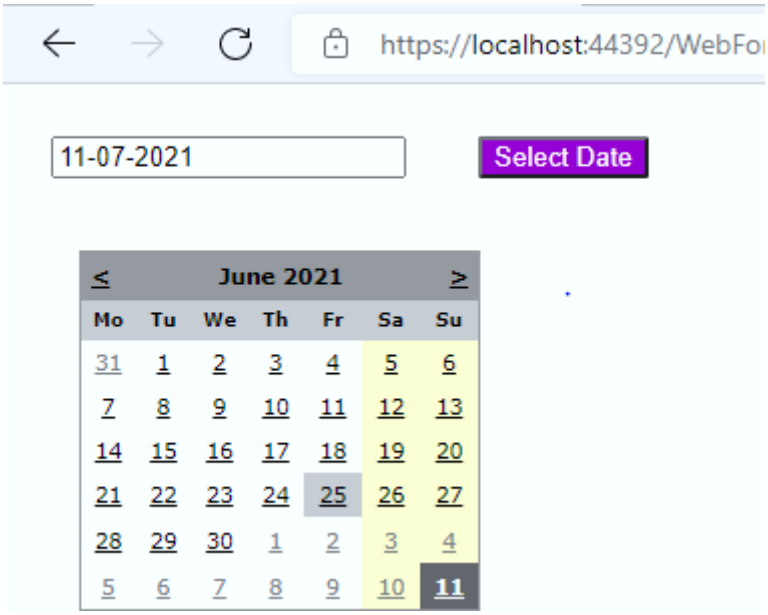
4 Aim: Design an ASP.NET application to Display Current Month's Calendar. Render the calendar to week days and current months days only.

Objective: To demonstrate Calendar Control

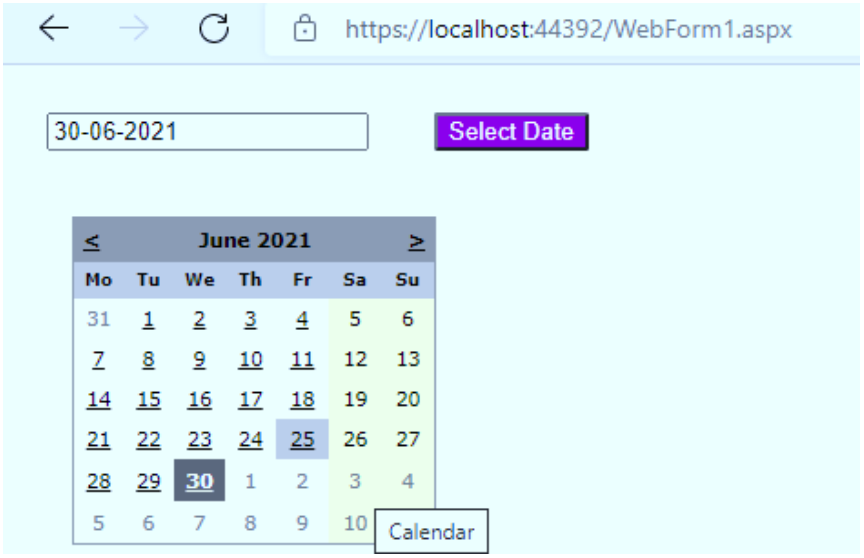
ASP.NET provides a Calendar control that is used to display a calendar on a Web page. It can be used to display or take birthdays, anniversaries, appointments, holidays, bill payments, and project deadlines. ASP.NET Calendar control displays a month calendar that allows user to select dates and move to the next and pr

```
using System;
using System.Collections.Generic; using
System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApp_Calender
{
    public partial class calender : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if(!IsPostBack)
            {
                Calendar1.Visible = false;
            }
        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            if (Calendar1.Visible)
            {
                Calendar1.Visible = false;
            }
            else
            {
                Calendar1.Visible = true;
            }
        }


        protected void Calendar1_SelectionChanged(object sender, EventArgs e)
        {
            TextBox1.Text = Calendar1.SelectedDate.ToString("d"); Calendar1.Visible
            = false;
        }

        protected void Calendar1_DayRender(object sender, DayRenderEventArgs e)
        {
            if(e.Day.IsOtherMonth ||      e.Day.IsWeekend)
            {
                e.Day.IsSelectable = false;
            }
        }
    }
}
```

Button Click & Day Select Events :



Day Rendor Events :



2. Aim: Design a Web Application for an Organization with Registration forms and advanced controls

a) Registration form must i) Emp_Name ii) Emp_ID iii) DOB iv) photograph

v) Password vi) new password

b) Apply validation control on form field

c) Apply calendar on DOB to select date

d) file upload on photograph.

Objective: To demonstrate form Validation

Validation Control:

There are Six Servers as well as client-side validation controls in ASP.Net that are as follows:

1. RequiredFieldValidator control:

This control ensures that the control it is used for validation is not empty when the form is submitted. In other words suppose their is one Text Box control and you have used a RequiredFieldValidator to validate that text box; then before submitting the data on the server it checks if the text box is not empty.

2. RangeValidator:

Checks that the value of the associated control is within a specified range. The value and the range can be numerical, a date or a string. In other words suppose their is one text box and you want to allow only 10 digits or any strings with a specified range using RangeValidator then before submitting the data on the server it ensure that the value is within a specified range.

3. CompareValidator:

Checks that the value of the associated control matches a specified comparison (less than, greater than, and so on) against another constant value or control.

4. RegularExpressionValidator

Checks if the value of the control it has to validate matches the specified regular expression.

5. CustomValidator

Allows specification of any client-side JavaScript validation routine and its server-side counterpart to perform your own custom validation logic .

6. ValidationSummary

Shows a summary with the error messages for each failed validator on the page (or in a pop-up message box).

Code :-

WebConfig :

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  https://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.7.2" />
    <httpRuntime targetFramework="4.7.2" />
  </system.web>
  <system.codedom>
    <compilers>
      <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default
/nowarn:1659;1699;1701" />
      <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default
/nowarn:41008 /define:_MYTYPE=\&quot;Web\&quot;
/optionInfer+" />
    </compilers>
    </system.codedom>
        <appSettings>
            <add key="ValidationSettings:UnobtrusiveValidationMode"
value="None"></add>
        </appSettings>
</configuration>
```

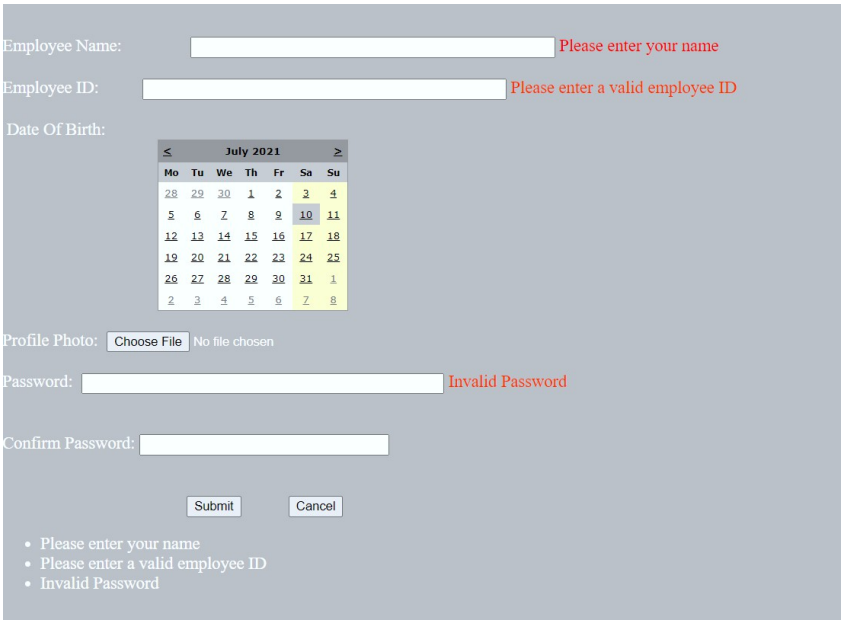
Form1.aspx.cs :

```
using System;
using System.Collections.Generic; using
System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Asp_Validation_AdvanceControl_ResgistraionForm
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            RangeValidator1.MinimumValue = DateTime.Now.AddYears(- 60).ToShortDateString();
            RangeValidator1.MaximumValue = DateTime.Now.AddYears(-
6).ToShortDateString();        //.AddMonths

            if (!IsPostBack)
            {
                Calendar1.Visible = false;
            }
        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            if (Calendar1.Visible)
            {
                Calendar1.Visible = false;
            }
            else
            {
                Calendar1.Visible = true;
            }
        }



        protected void Calendar1_SelectionChanged(object sender, EventArgs e)
        {
            TextBox3.Text = Calendar1.SelectedDate.ToString("d"); Calendar1.Visible
            = false;
        }

        protected void Button2_Click(object sender, EventArgs e)
        {
            if (FileUpload1.HasFile)
            {
                string ext = System.IO.Path.GetExtension(FileUpload1.FileName); if
                (ext == ".jpg" || ext == ".png")
                {
                    string path = Server.MapPath("Photograph//");
                    FileUpload1.SaveAs(path + FileUpload1.FileName);
                }
                else
                {
                    Response.Write("Select only '.jpg' and '.png' formate");
                }
            }
```

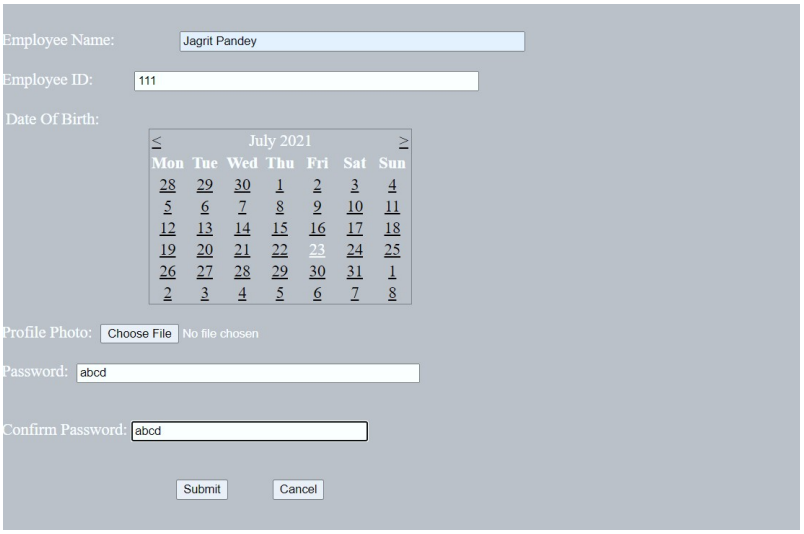
```
                else
                {
                    Response.Write("Select a file");
                }

                Response.Write("Record Added successfully");
            }
        }
    }
```

Output :-

Without Fill up Form :



After Click Submit Button :-



Assignment - 02

Practical No	Problem Statement
1	Design UI based applications using basic Windows forms Controls
2	Design Applications using Classes and Objects
3	Design Applications using Inheritance and Abstract Classes

1. Design UI based applications using basic windows forms Controls

Code:

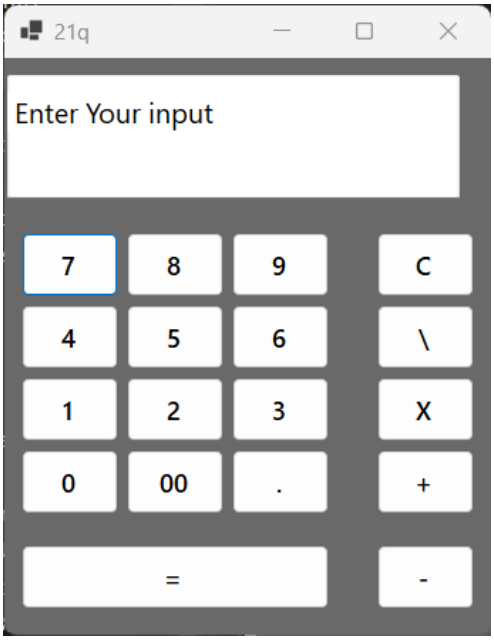
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Calculator
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }
      float num1, ans;
      int count;


      private void textBox1_TextChanged(object sender, EventArgs e)
      {

      }

      private void button2_Click(object sender, EventArgs e)
      {
         label2.Text = null;
         textBox1.Text = textBox1.Text + 2;
      }

      private void button1_Click(object sender, EventArgs e)
      {
         label2.Text = null;
         textBox1.Text = textBox1.Text + 1;
      }
```

```
private void button3_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 3;
}

private void button4_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 4;
}

private void button5_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 5;
}

private void button6_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 6;
}

private void button7_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 7;
}

private void button8_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 8;
}

private void button9_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 9;
}

private void button10_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 0;
}

private void button11_Click(object sender, EventArgs e)
{
   label2.Text = null;
   textBox1.Text = textBox1.Text + 0+0;
}

private void button13_Click(object sender, EventArgs e)
{
   label2.Text = "Enter your input";
   textBox1.Text = null;
}
```

```
private void button16_Click(object sender, EventArgs e)
{
   num1 = float.Parse(textBox1.Text);
   textBox1.Clear();
   textBox1.Focus();
   count = 2;
}

private void button17_Click(object sender, EventArgs e)
{
   compute(count);

}
public void compute(int count)
{
   switch (count)
   {
      case 1:
         ans = num1 - float.Parse(textBox1.Text);
         textBox1.Text = ans.ToString();
         break;
      case 2:
         ans = num1 + float.Parse(textBox1.Text);
         textBox1.Text = ans.ToString();
         break;
      case 3:
         ans = num1 * float.Parse(textBox1.Text);
         textBox1.Text = ans.ToString();
         break;
      case 4:
         ans = num1 / float.Parse(textBox1.Text);
         textBox1.Text = ans.ToString();
         break;
      default:
         break;
   }
}

private void button18_Click(object sender, EventArgs e)
{


      num1 = float.Parse(textBox1.Text);
      textBox1.Clear();
      textBox1.Focus();
      count = 1;

}

private void button15_Click(object sender, EventArgs e)
{
   num1 = float.Parse(textBox1.Text);
   textBox1.Clear();
   textBox1.Focus();
   count = 3;
}

private void button14_Click(object sender, EventArgs e)
```

```
        {
            num1 = float.Parse(textBox1.Text);
            textBox1.Clear();
            textBox1.Focus();
            count = 4;
        }

        private void button12_Click(object sender, EventArgs e)
        {
            label2.Text = null;
            textBox1.Text = textBox1.Text + ".";
        }

        private void label2_Click(object sender, EventArgs e)
        {

        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }
    }
}
```

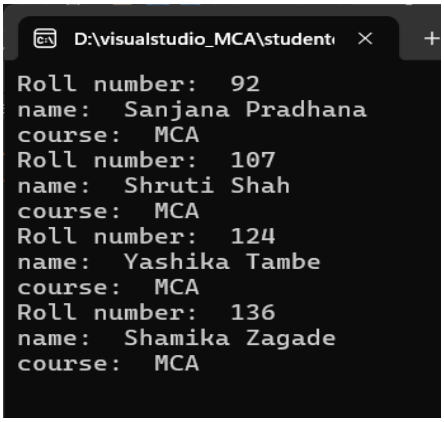
OUTPUT:



2. Design Application using Classes and Objects.

```
using System;


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

namespace Class_and_object
{
   public class Student
   {
      int rollnumber;
      string name;
      string course;


      public void insertdata(int r, string n, string c)
      {
         rollnumber = r;
         name = n;
         course = c;
      }

      public void display()
      {
         Console.WriteLine("Roll number:  " + rollnumber);
         Console.WriteLine("name:  " + name);
         Console.WriteLine("course:  " + course);
      }

   }
   class Program
   {
      static void Main(string[] args)
      {
         Student s = new Student();
         s.insertdata(92, "Sanjana Pradhana", "MCA");
         s.display();
         s.insertdata(107, "Shruti Shah", "MCA");
         s.display();
         s.insertdata(124, "Yashika Tambe", "MCA");
         s.display();
         s.insertdata(136, "Shamika Zagade", "MCA");
         s.display();
         Console.ReadKey();
      }
   }
}
```

OUTPUT:

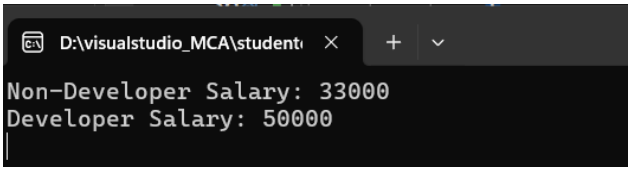


3. Design Applications using Inheritance and Abstract Classes.

Code:

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

namespace Console_abstract_inhertance
{
   public abstract class Employee
   {
      public abstract int salary();
   }

   public class Developer : Employee
   {
      int sal;
      public Developer(int s)
      {
         sal = s + 5000;
      }
      public override int salary()
      {
         return sal;
      }

      public class Nondeveloper : Employee
      {
         int sal;
         public Nondeveloper(int s)
         {
            sal = s + 2000;
         }
         public override int salary()
         {
            return sal;
         }

         class Program
         {
            static void Main(string[] args)
            {
               Nondeveloper non = new Nondeveloper(31000);
               Developer dev = new Developer(45000);

               Console.WriteLine("Non-Developer Salary: "+non.salary());
               Console.WriteLine("Developer Salary: "+ dev.salary());
               Console.ReadLine();
            }
         }
      }
   }
}
```

OUTPUT:


D:\visualstudio_MCA\student
Non-Developer Salary: 33000
Developer Salary: 50000

Assignment-3

1) Design a webpage to display the use of LINQ.
a. Create a Database.
b. Perform functions like Filtering, Ordering, and Joining.

To design a webpage and display use of LINQ

Objective :

To create a database and perform functions to it.

LINQ (Language Integrated Query) is uniform query syntax in C# and VB.NET to retrieve data from different sources and formats. LINQ queries return results as objects. It enables you to uses object-oriented approach on the result set and not to worry about transforming different formats of results into objects. LINQ provides functions to query cached data from all kinds of data sources.

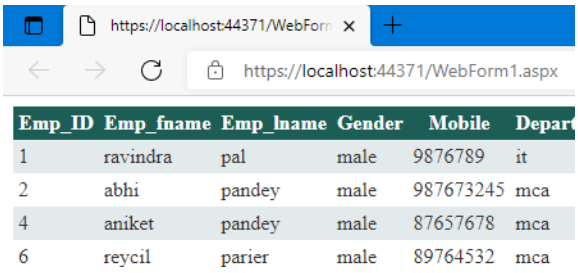
.

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace LinqD22
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         DataClasses1DataContext context = new DataClasses1DataContext();
         var result = from Employee in context.Employees
           join dist in context.dists on Employee.Gender equals dist.Gender
           select new { Employee = Employee.Emp_fname, distGender = dist.Gender };
         GridView1.DataSource = result;
         GridView1.DataBind();



      }
   }
}
```

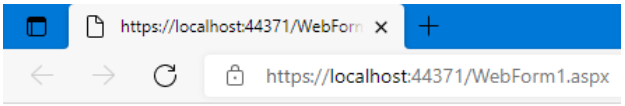
OutPut:

1. Where employee.gender= =”male



Emp_ID	Emp_fname	Emp_lname	Gender	Mobile	Depart
1	ravindra	pal	male	9876789	it
2	abhi	pandey	male	987673245	mca
4	aniket	pandey	male	87657678	mca
6	reycil	parier	male	89764532	mca

2. Where Employee.Department==”mca”/Ascending



Emp_ID	Emp_fname	Emp_lname	Gender	Mobile	Depar
2	abhi	pandey	male	987673245	mca
4	aniket	pandey	male	87657678	mca
5	mansi	patil	female	876534567	mca
6	reycil	parier	male	89764532	mca

3. Joining the names between table dist & Employee

EmployeeName	distGender
Akshu	female
Mansi	female
Pranav	male
Aniket	male
Jay	male
Abhi	male
Akshu	female
Mansi	female

2. Demonstrate the working of entity framework in dot net with Insert, Update, Delete

Aim:

To design entity framework in dot net .

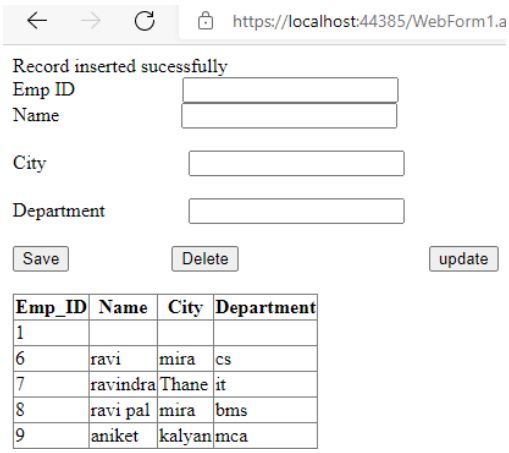
Entity framework is an Object Relational Mapping (ORM) framework that offers an automated mechanism to developers for storing and accessing the data in the database. An ORM takes care of creating database connections and executing commands, as well as taking query results and automatically materializing those results as your application objects. An ORM also helps to keep track of changes to those objects, and when instructed, it will also persist those changes back to the database for you.

Code:

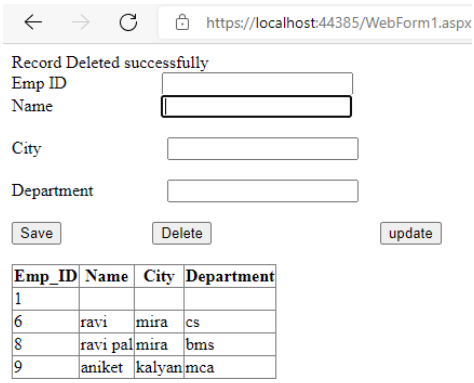
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.Entity;

namespace Entity1
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      emp model1 = new emp();
      protected void Page_Load(object sender, EventArgs e)
      {
         //clear();
      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         model1.Name = TextBox1.Text.Trim();
         model1.City = TextBox2.Text.Trim();
         model1.Department = TextBox3.Text.Trim();
         using (Database1Entities db = new Database1Entities())
         {
            db.emps.Add(model1);
            db.SaveChanges();
         }

         clear();
         Response.Write("Record inserted sucessfully");
         GridView1.DataBind();
      }
      void clear()
      {
         TextBox1.Text = TextBox2.Text = TextBox3.Text = " ";
         Button1.Text = "Save";
         model1.Emp_ID = 0;
      }
```

```
    protected void Button2_Click(object sender, EventArgs e)
    {
        model1.Name = TextBox1.Text.Trim();
        model1.City = TextBox2.Text.Trim();
        model1.Department = TextBox3.Text.Trim();
        model1.Emp_ID = Convert.ToInt32(TextBox4.Text);

        using (Database1Entities db = new Database1Entities())
        {
            db.Entry(model1).State = EntityState.Deleted;
            db.SaveChanges();
        }
        Response.Write("Record Deleted successfully");
        GridView1.DataBind();

    }

    protected void Button3_Click(object sender, EventArgs e)
    {
        model1.Name = TextBox1.Text.Trim();
        model1.City = TextBox2.Text.Trim();
        model1.Department = TextBox3.Text.Trim();
        model1.Emp_ID = Convert.ToInt32(TextBox4.Text);
        using (Database1Entities db = new Database1Entities())
        {
            db.Entry(model1).State = EntityState.Modified;
            db.SaveChanges();
        }
        Response.Write("Record Updated successfully");
        GridView1.DataBind();
    }
  }
}
```

Output:

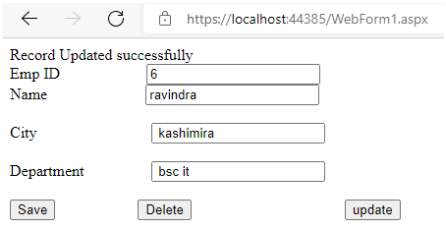


Delete record



Emp_ID	Name	City	Department
1			
6	ravi	mira	cs
8	ravi pal	mira	bms
9	aniket	kalyan	mca

Update record



Emp_ID	Name	City	Department
1			
6	ravindra	kashimira	bsc it
8	ravi pal	mira	bms
9	aniket	kalyan	mca

3. Design Web Application to produce and Consume a WCF Service for calculator.

To design web application for wcf service.

WCF stands for Windows Communication Foundation. The elementary feature of WCF is interoperability. It is one of the latest technologies of Microsoft that is used to build service-oriented applications. Based on the concept of message-based communication, in which an HTTP request is represented uniformly, WCF makes it possible to have a unified API irrespective of diverse transport mechanisms. WCF services offer enhanced reliability as well as security in comparison to ASMX (Active Server Methods) web services. It offers scalability and support for up-coming web service standards.

Webform1.aspx.

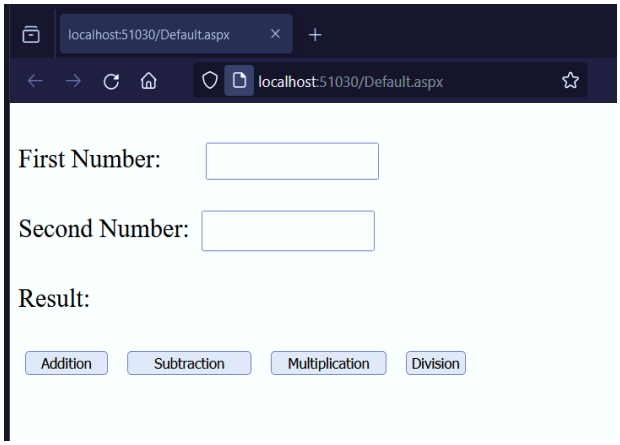
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace client_2._3
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      ServiceReference1.Service1Client cl = new ServiceReference1.Service1Client();
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         int result = cl.Addition(Convert.ToInt32(TextBox1.Text), Convert.ToInt32(TextBox2.Text));
         Label1.Text = result.ToString();
      }

      protected void Button2_Click(object sender, EventArgs e)
      {
         int result = cl.Subtraction(Convert.ToInt32(TextBox1.Text), Convert.ToInt32(TextBox2.Text));
         Label1.Text = result.ToString();
      }

      protected void Button3_Click(object sender, EventArgs e)
      {
         int result = cl.Multiplication (Convert.ToInt32(TextBox1.Text), Convert.ToInt32(TextBox2.Text));
         Label1.Text = result.ToString();
```

```
    }

    protected void Button4_Click(object sender, EventArgs e)
    {
        int result = cl.Division(Convert.ToInt32(TextBox1.Text), Convert.ToInt32(TextBox2.Text));
        Label1.Text = result.ToString();
    }
  }
}
```

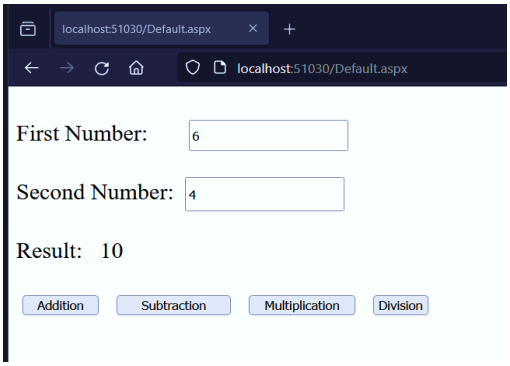
Iservice1.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

namespace wcf_calculator
{
  // NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService1" in both code and config file together.
  [ServiceContract]
  public interface IService1
  {

    [OperationContract]
    int Addition(int number1, int number2);

    [OperationContract]
    int Subtraction(int number1, int number2);

    [OperationContract]
    int Multiplication(int number1, int number2);

    [OperationContract]
    int Division(int number1, int number2);

  }
}
```

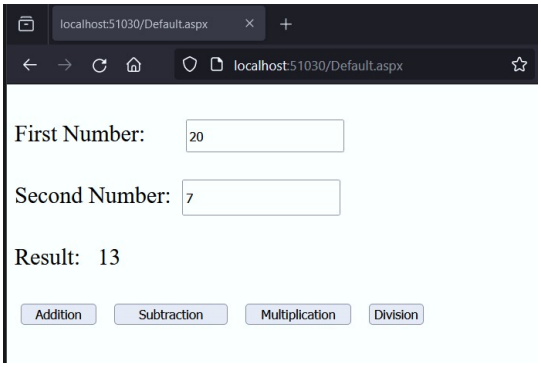
Output:



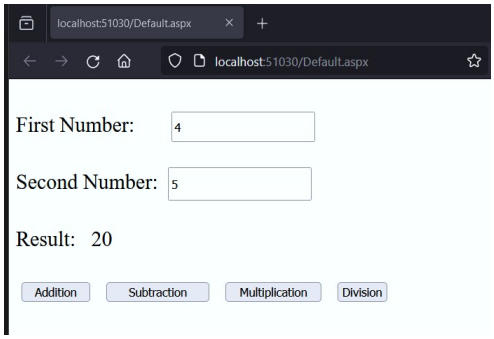
1. Addition



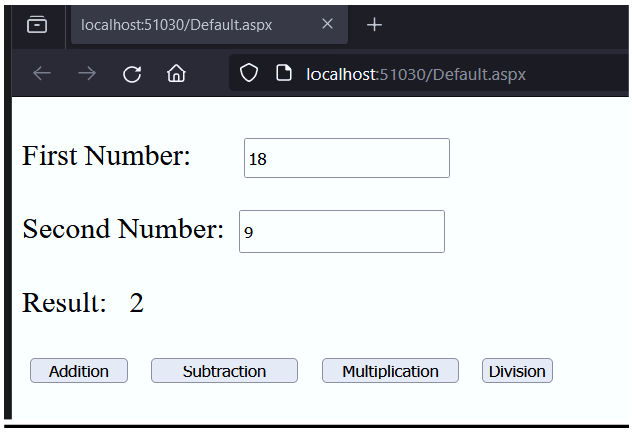
2. Substraction



3. Multiplication


localhost:51030/Default.aspx
localhost:51030/Default.aspx
First Number: 4
Second Number: 5
Result: 20
Addition
Subtraction
Multiplication
Division

4. Division


localhost:51030/Default.aspx
localhost:51030/Default.aspx
First Number: 18
Second Number: 9
Result: 2
Addition
Subtraction
Multiplication
Division

4. Design Web Application to produce and Consume a WCF Service, Perform in that insertion of record in database.

Aim:

To design web application and insert record in database.

WCF stands for Windows Communication Foundation. The elementary feature of WCF is interoperability. It is one of the latest technologies of Microsoft that is used to build service-oriented applications. Based on the concept of message-based communication, in which an HTTP request is represented uniformly, WCF makes it possible to have a unified API irrespective of diverse transport mechanisms. WCF services offer enhanced reliability as well as security in comparison to ASMX (Active Server Methods) web services. It offers scalability and support for up-coming web service standards.

1. Service.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;
using System.Data.SqlClient;

namespace Database3wcf
{
    // NOTE: You can use the "Rename" command on the "Refactor" menu to change the class name "Service1" in code, svc and config file together.
    // NOTE: In order to launch WCF Test Client for testing this service, please select Service1.svc or Service1.svc.cs at the Solution Explorer and start debugging.
    public class Service1 : IService1
    {
        public string Insert(InsertUser user)
        {
            string msg;
            SqlConnection con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\user\source\repos\Database3wcf\Database3wcf\App_Data\Database1.mdf;Integrated Security=True");

            con.Open();
            SqlCommand cmd = new SqlCommand("Insert into emp(Name,City) values(@Name,@City)", con);


            cmd.Parameters.AddWithValue("@Name", user.Name);

            cmd.Parameters.AddWithValue("@City", user.City);

            int result = cmd.ExecuteNonQuery();
```

```
            if (result == 1)
            {
                msg = "Record inserted Sucessfully";
            }
            else
            {
                msg = "Failed to insert";
            }

            return msg;
        }


    }
}
```

2. IService.cs

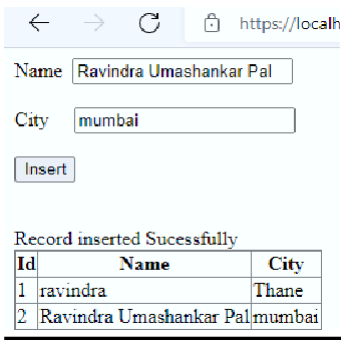
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

namespace Database3wcf
{
    // NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService1" in both code and config file together.
    [ServiceContract]
    public interface IService1
    {

        [OperationContract]
        string Insert(InsertUser user); //use class name InsertUser
        // TODO: Add your service operations here
    }


    // Use a data contract as illustrated in the sample below to add composite types to service operations.
    [DataContract]
    public class InsertUser
    {
        string name = string.Empty;
        string city = string.Empty;
        [DataMember]
        public string Name
        {
            get { return name; }
            set { name = value; }
        }
        [DataMember]
        public string City
        {
            get { return city; }
            set { city = value; }
        }
```

```
    }
}
```

1. WebForm1.aspx

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using WebApplication1.ServiceReference1;

namespace WebApplication1
{
    public partial class WebForm1 : System.Web.UI.Page
    {

        protected void Page_Load(object sender, EventArgs e)
        {

        }
        ServiceReference1.Service1Client client = new ServiceReference1.Service1Client();
        protected void Button1_Click(object sender, EventArgs e)
        {
            InsertUser u = new InsertUser();
            u.Name = TextBox1.Text;
            u.City = TextBox2.Text;
            string r = client.Insert(u);
            Label1.Text = r.ToString();
            GridView1.DataBind();

        }
    }
}
```

Output:



5. Demonstrate data consumption by web application using simple web service

Aim:

To design web application using simple web service.

Objective:

To demonstrate data consumption by designing web application for simple web service.

Entity framework is an Object Relational Mapping (ORM) framework that offers an automated mechanism to developers for storing and accessing the data in the database. An ORM takes care of creating database connections and executing commands, as well as taking query results and automatically materializing those results as your application objects. An ORM also helps to keep track of changes to those objects, and when instructed, it will also persist those changes back to the database for you.

Code:

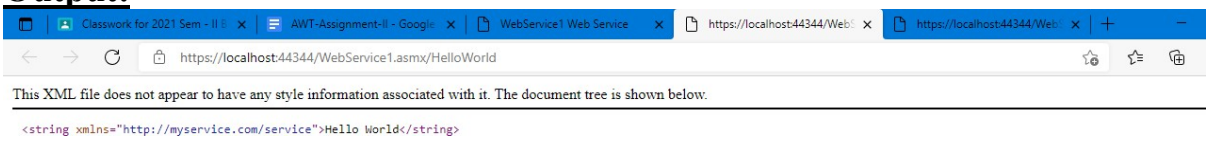
1. WebService1.asmx.cs

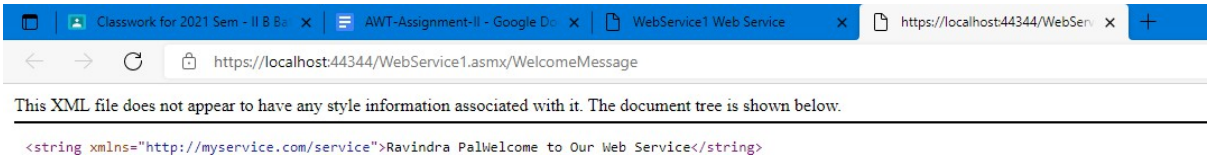
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace hellowebservices
{
    /// <summary>
    /// Summary description for WebService1
    /// </summary>
    [WebService(Namespace = "http://myservice.com/service")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
    // [System.Web.Script.Services.ScriptService]
    public class WebService1 : System.Web.Services.WebService
    {

        [WebMethod]
        public string HelloWorld()
        {
            return "Hello World";
        }
        [WebMethod]
        public string WelcomeMessage(string fname, string lname)
        {
            return string.Format("{0} {1}Welcome to Our Web Service",fname,lname);
        }
    }
}
```

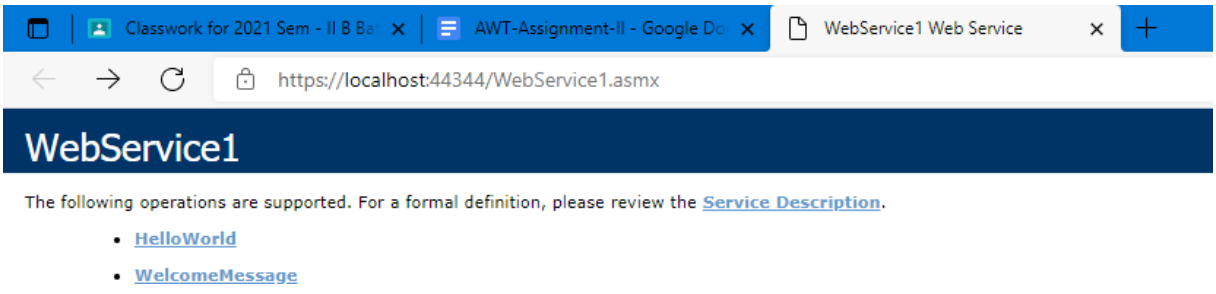
1. WebForm1.aspx.cs

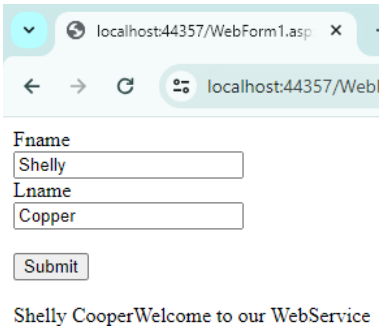
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication1
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }
      ServiceReference1.WebService1SoapClient client = new ServiceReference1.WebService1SoapClient();
      protected void Button1_Click(object sender, EventArgs e)
      {
         string result = client.WelcomeMessage(TextBox1.Text, TextBox2.Text);
         Label1.Text = result;
      }
   }
}
```

Output:








localhost:44357/WebForm1.asp
localhost:44357/Webl
Fname
Shelly
Lname
Copper
Submit
Shelly CooperWelcome to our WebService

6. Design web applications to produce and consume simple web services.

Aim:

To design web applications to produce and consume simple web services.

Objective:

To design simple web services.

1. WebService1.asmx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace SimpleWebService6Rightwala
{
    /// <summary>
    /// Summary description for WebService1
    /// </summary>
    [WebService(Namespace = "http://Pranav.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
    // [System.Web.Script.Services.ScriptService]
    public class WebService1 : System.Web.Services.WebService
    {

        [WebMethod]
        public string Insert(string name, string city)
        {
            Class1 abc = new Class1
            {
                Name = name,
                City = city
            };
            string status = abc.Insert();
            if (status == "PASS")
                return "Record inserted Sucessfully";
            else
                return "Failed to insert";
        }
    }
}
```

1. Class1.cs

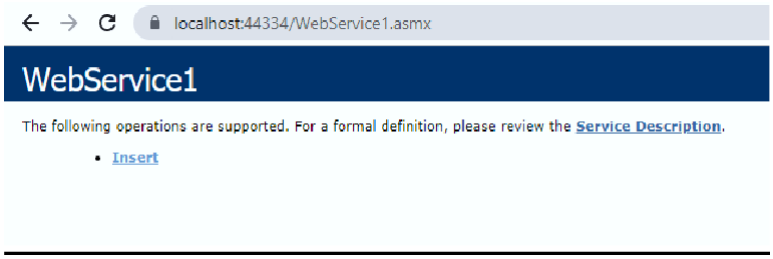
```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;

namespace SimpleWebService6Rightwala
{
    public class Class1
    {
        private string name;
        private string city;
        public string Name
        {
            get { return name; }
            set { name = value; }
        }
        public string City
        {
            get { return city; }
            set { city = value; }
        }
        public string Insert()
        {
            string msg;
            SqlConnection con = new SqlConnection(@"Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\mishr\source\repos\SimpleWebService6Rightwala\Sim
pleWebService6Rightwala\App_Data\Database1.mdf;Integrated Security=True");
            con.Open();
            SqlCommand cmd = new SqlCommand("Insert into Student(Name,City) values(@Name,@City)", con);

            cmd.Parameters.AddWithValue("@Name", Name);
            cmd.Parameters.AddWithValue("@City", City);

            int result = cmd.ExecuteNonQuery();
            if (result == 1)
            {
                msg = "PASS";
            }

            else
            {
                msg = "FAIL";
            }

            return msg;

        }
    }
}
```

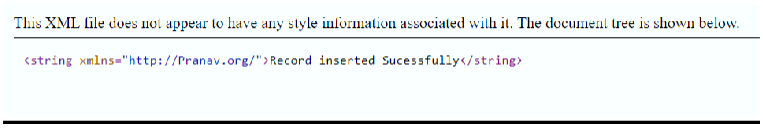
1. WebForm1.aspx.cs

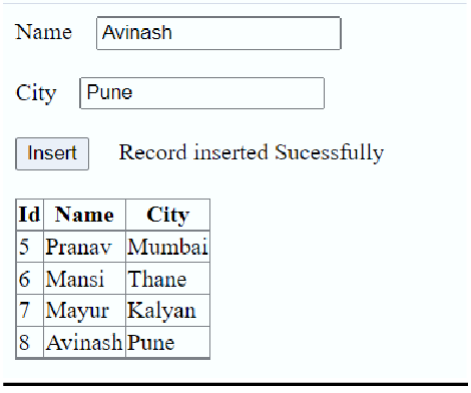
```
using System;
using System.Collections.Generic;
using System.Linq;
```

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

namespace Client
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      ServiceReference1.WebService1SoapClient client = new ServiceReference1.WebService1SoapClient();
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         string result = client.Insert(TextBox1.Text, TextBox2.Text);
         Label1.Text = result.ToString();
         GridView1.DataBind();
      }
   }
```

Output:







7. Write a program to demonstrate the ViewState.

Aim:

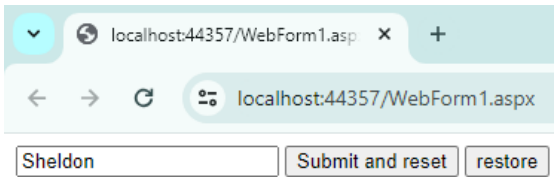
To demonstrate the viewstate.

View state is the method that the ASP.NET page framework uses to preserve page and control values between round trips. When the HTML markup for the page is rendered, the current state of the page and values that must be retained during postback are serialized into base64-encoded strings. This information is then put into the view state hidden field or fields

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace viewstate
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button2_Click(object sender, EventArgs e)
      {
         if (ViewState["myvalue"] != null)
            TextBox1.Text = ViewState["myvalue"].ToString();

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         ViewState["myvalue"] = TextBox1.Text.ToString();
         TextBox1.Text = " ";
      }
   }
      }
```

OUTPUT:



8. Write a program to demonstrate the Hidden Object.

Aim:

To demonstrate the hidden object.

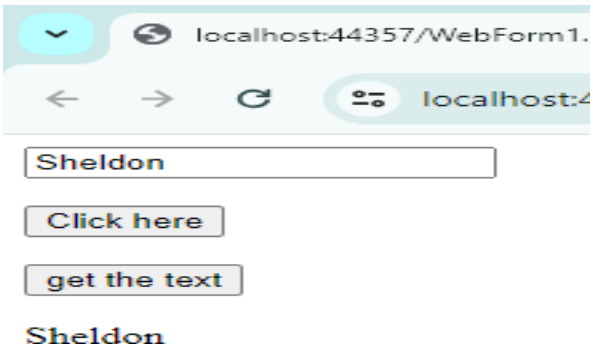
HiddenField, as name implies, is hidden. This is non visual control in ASP.NET where you can save the value. This is one of the types of client-side state management tools. It stores the value between the roundtrip

Code:

1. WebForm1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Hiddenfield
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         HiddenField1.Value = TextBox1.Text;
      }

      protected void Button2_Click(object sender, EventArgs e)
      {
         Label1.Text = HiddenField1.Value;
      }
   }
}
```

OUTPUT


localhost:44357/WebForm1.
localhost:
Sheldon
Click here
get the text
Sheldon

9. Write a program to demonstrate the query string.

Aim:

To demonstrate the query string.

A query string is one of the techniques in Web applications to send data from one webform to another through the URL. A query string consists of two parts, field and value, and each of pair separated by ampersand (&). The ?(question mark in a query string indicates the beginning of a query string and it's value.

1. WebForm1.aspx.cs

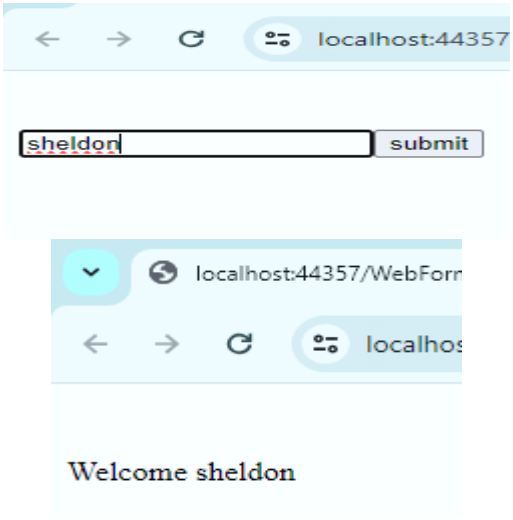
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace querystring
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         Response.Redirect("WebForm2.aspx?myname= " + TextBox1.Text.ToString());
      }
   }
}
```

]2.WebForm2.aspx.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace querystring
{
   public partial class WebForm2 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
```

```
        Label1.Text = "Welcome" + Request.QueryString["myname"];
    }
  }
}
```

OUTPUT:



10. Create a MVC application that displays data from the model.

Aim:

To demonstrate the query string.

Objective:

To demonstrate the query string.

A query string is one of the techniques in Web applications to send data from one webform to another through the URL. A query string consists of two parts, field and value, and each of pair separated by ampersand (&). The ?(question mark in a query string indicates the beginning of a query string and it's value.

Studentcontoler.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using MVC2.Models;

namespace MVC2.Controllers
{
   public class studentController : Controller
   {
      // GET: student
      public ActionResult Index()
      {
         student std = new student();
         std.ID = 1;
         std.Name = "ravindra";
         return View(std);
      }
   }
}
```

1. StudentController.cs

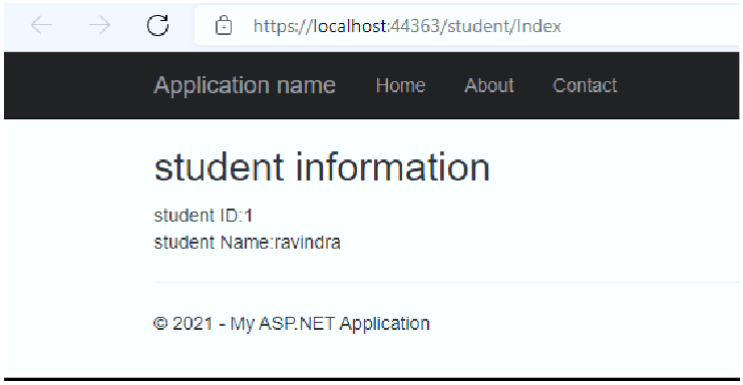
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace MVC2.Models
{
   public class student
   {
      public int ID
      {
         get;
```

```
        set;
    }
    public string Name
    {
        get;
        set;
    }
  }
}
```

2. Index.cshtml

```
@model MVC2.Models.student

@{
    ViewBag.Title = "Index";
}

<h2>student information</h2>
student ID:@Model.ID  </br>
student Name:@Model.Name
```

OUTPUT:



11. Write an application that accepts data from one page and displays it in another page using MVC (Passing data between views).

Aim:

To write an application that accepts data from one page and displays it in another page using MVC.

Objective:

To accept data from one page and display it in another page.

The Model component of MVC application represents the state of a particular data part or data portion of the application and it usually interacts with the file, database, and the web service etc. It actually represent the logic

Code:

1. Studentcontroller.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using MVCgetdatastudent23.Models;

namespace MVCgetdatastudent23.Controllers
{
    public class StudentController : Controller
    {
        // GET: Student
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult ShowStudent()
        {
            Student std = new Student();
            std.StudentId = int.Parse(Request.Form["sid"].ToString());
            std.StudentName = Request.Form["sname"].ToString();
            return View(std);

        }

    }
}
```

2. Student.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace MVCgetdatastudent23.Models
{
    public class Student
```

```
    {
        public int StudentId
        {
            get;
            set;
        }

        public string StudentName
        {
            get;
            set;
        }
    }
}
```

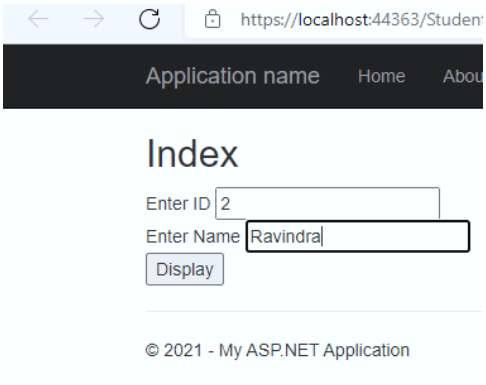
3. Showstudent.cshtml

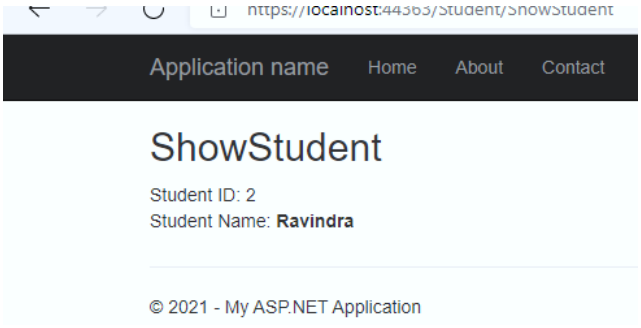
```
@model MVCgetdatastudent23.Models.Student

@{
    ViewBag.Title = "ShowStudent";
}

<h2>ShowStudent</h2>
<div>Student ID: @Model.StudentId</div>
<div>Student Name: @Html.Label("sname",Model.StudentName)</div>
```

4. index.cshtml

```
@model MVCgetdatastudent23.Models.Student

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>
@using (Html.BeginForm("ShowStudent", "Student", FormMethod.Post))
        {
            <div>Enter ID @Html.TextBox("sid")</div>
            <div>Enter Name @Html.TextBox("sname")</div>
            <input type="submit" value="Display" />

        }
```

OUTPUT:


https://localhost:44363/Studen
Application name
Home
Abou
Index
Enter ID 2
Enter Name Ravindra
Display
© 2021 - My ASP.NET Application


Application name
Home
About
Contact
ShowStudent
Student ID: 2
Student Name: Ravindra
© 2021 - My ASP.NET Application

12. Write a program that shows crud operation using MVC.

Aim:

To show crud operation using MVC

CRUD operation in MVC is the basic operations, where CRUD denotes create, read, update, and delete ..... MVC is the Model View Controller. MVC is a design pattern that is used to

differentiate the data from business logic and presentation logic. It gives a pattern that helps in designing the web application.

Product.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace crudmvc.Models
{
   public class product
   {
      public int id { get; set; }
      public string name { get; set; }
      public int price { get; set; }
   }
}
```

1. **ProductController.cs**

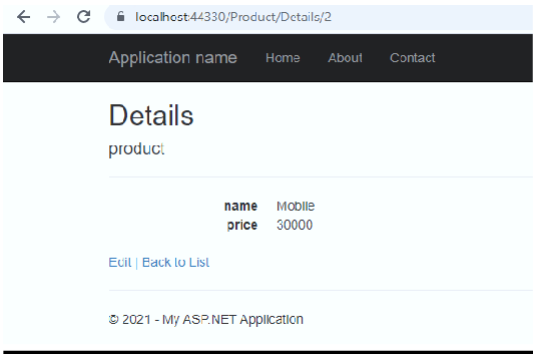
```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using crudmvc.Models;

namespace crudmvc.Controllers
{
   public class ProductController : Controller
   {
      SqlConnection con = new
SqlConnection(@"DataSource=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\jahnv\source\re
pos\crumvc\crudmvc\App_Data\Database1.mdf;Integrated Security=True");
    // GET: Product
    public ActionResult Index()
     {
       SqlDataAdapter da = new SqlDataAdapter("select * fromProductDetails", con);
      DataSet ds = new DataSet();
      da.Fill(ds);
       List<product> prod = new List<product>();
       for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
```

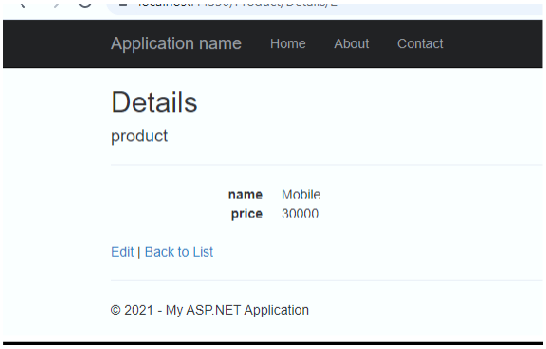
```
                {
                    product p1 = new product();
                    p1.id = int.Parse(ds.Tables[0].Rows[i][0].ToString());
                    p1.name = ds.Tables[0].Rows[i][1].ToString();
                    p1.price = int.Parse(ds.Tables[0].Rows[i][2].ToString());
                    prod.Add(p1);
                }
            return View(prod);

        }

        // GET: Product/Details/5
        public ActionResult Details(int id)
{
            SqlDataAdapter da = new SqlDataAdapter("Select * from ProductDetails where Id=" + id, con);

            DataSet ds = new DataSet();
            da.Fill(ds);
            product p = new product();
            p.id = int.Parse(ds.Tables[0].Rows[0][0].ToString());
            p.name = ds.Tables[0].Rows[0][1].ToString();
            p.price = int.Parse(ds.Tables[0].Rows[0][2].ToString());
            return View(p);
        }

        // GET: Product/Create{to show data}
        public ActionResult Create()
        {
            return View();
        }

        // POST: Product/Create{to enter data in database}
        [HttpPost]
        public ActionResult Create(product p)
{
            try
            {
                con.Open();
                SqlCommand cmd = new SqlCommand("Insert into ProductDetailsvalues('" + p.name + "'," +
    p.price + ")", con);
                cmd.ExecuteNonQuery();
            con.Close();
            return RedirectToAction("Index");
            // TODO: Add insert logic here
        }
        catch
            {
                return View();
        }
        }

        // GET: Product/Edit/5
        public ActionResult Edit(int id)
        {
            SqlDataAdapter da = new SqlDataAdapter("select * fromProductDetails where id=" + id, con);
           DataSet ds = new DataSet();
```

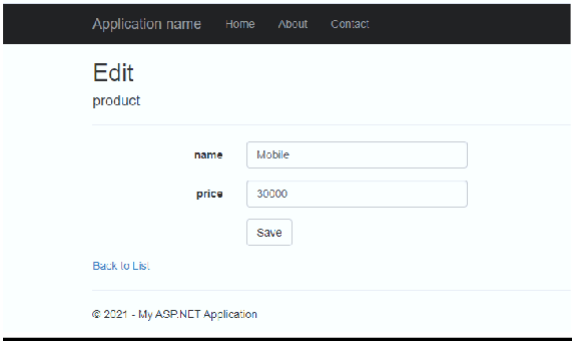
```
                da.Fill(ds);
                product p1 = new product();
    p1.id = int.Parse(ds.Tables[0].Rows[0][0].ToString());
                p1.name = ds.Tables[0].Rows[0][1].ToString();
                p1.price = int.Parse(ds.Tables[0].Rows[0][2].ToString()); ;
                return View(p1);
            }

            // POST: Product/Edit/5
            [HttpPost]
            public ActionResult Edit(int id, product p)
{
    try
                {
                    con.Open();
                    SqlCommand cmd = new SqlCommand("update ProductDetails setName='" + p.name + "', Price
     = " + p.price + " where Id =" + id, con);
                     cmd.ExecuteNonQuery();
                     con.Close();
                      return RedirectToAction("Index");
                      // TODO: Add update logic here
                }
                catch
    {
        return View();
                }
            }

            // GET: Product/Delete/5
            public ActionResult Delete(int id)
{
                 SqlDataAdapter da = new SqlDataAdapter("select * from ProductDetails where id=" + id, con);
                DataSet ds = new DataSet();
                da.Fill(ds);
                product p1 = new product();
                p1.id = int.Parse(ds.Tables[0].Rows[0][0].ToString());
                p1.name = ds.Tables[0].Rows[0][1].ToString();
                p1.price = int.Parse(ds.Tables[0].Rows[0][2].ToString()); ;
                return View(p1);

            }

            // POST: Product/Delete/5
            [HttpPost]
            public ActionResult Delete(int id, product p)
            {
                try
                {
                    con.Open();
                    SqlCommand cmd = new SqlCommand(" delete from ProductDetails where id=" + p.id , con);
                    cmd.ExecuteNonQuery();
                    con.Close();
        return RedirectToAction("Index");
                    // TODO: Add delete logic here
                }
                catch
```

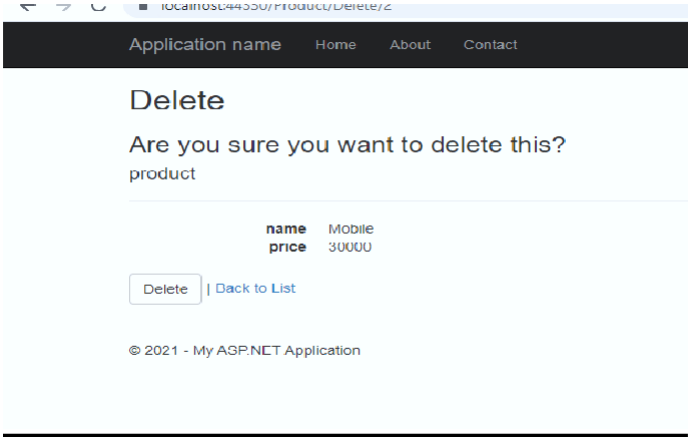
```
            {
                return View();
            }
        }
    }
}
```

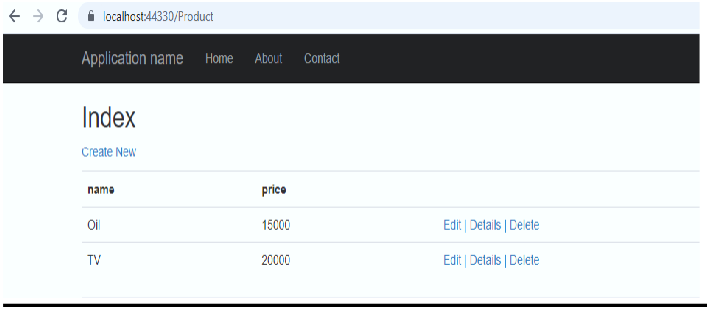
OUTPUT:








Application name
Home
About
Contact
Delete
Are you sure you want to delete this?
product
name Mobile
price 30000
Delete
| Back to List
© 2021 - My ASP.NET Application


localhost:44330/Product
Application name
Home
About
Contact
Index
Create New
name
price
Oil
15000
Edit | Details | Delete
TV
20000
Edit | Details | Delete

13. Write a program to count the number of live users in your web application

Aim:

To write a program to count the number of live users in web application.

Objective:

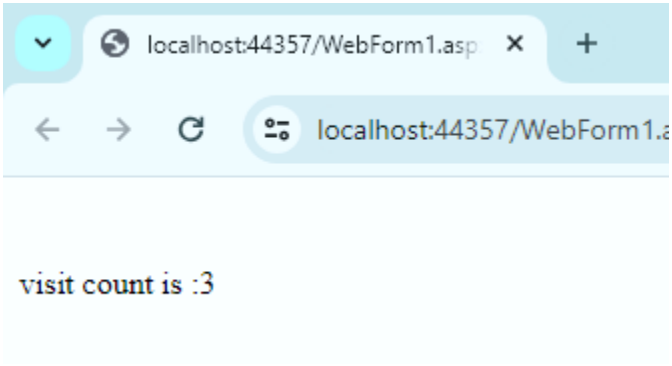
To count the number of live users in web application.

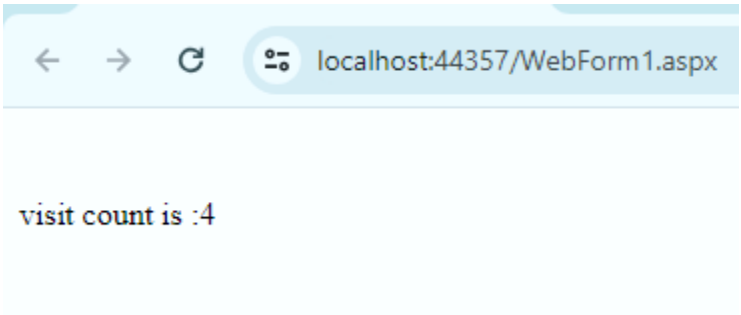
We have all seen many websites displaying the number of online users live currently on their web page so similarly we can count the number of person live on our website using the following code and implementing a counter.

WebForm1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace VisitorCount
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         int count = 0;
         if (Application["pcount"] != null)
         {
            count = Int32.Parse(Application["pcount"].ToString());
         }
         count = count + 1;
         Label1.Text = "Visitor Count is:" + count.ToString();
         Application["pcount"] = count.ToString();
      }
   }
}
```

OUTPUT:


localhost:44357/WebForm1.asp
localhost:44357/WebForm1.a
visit count is :3


localhost:44357/WebForm1.aspx
visit count is :4