Theory 1:

Introduction to basic C# and ASP. NET

Solution:

C# (C Sharp) and ASP.NET collectively form a dynamic duo in the realm of web development. C# stands as a modern, object-oriented programming language that seamlessly integrates with the expansive .NET framework. Renowned for its readability and versatility, C# is utilized for an array of applications, ranging from desktop software to web and cloud-based solutions. It embraces fundamental programming concepts such as variables, control structures, functions, and object-oriented principles, making it accessible for both novice and seasoned developers.

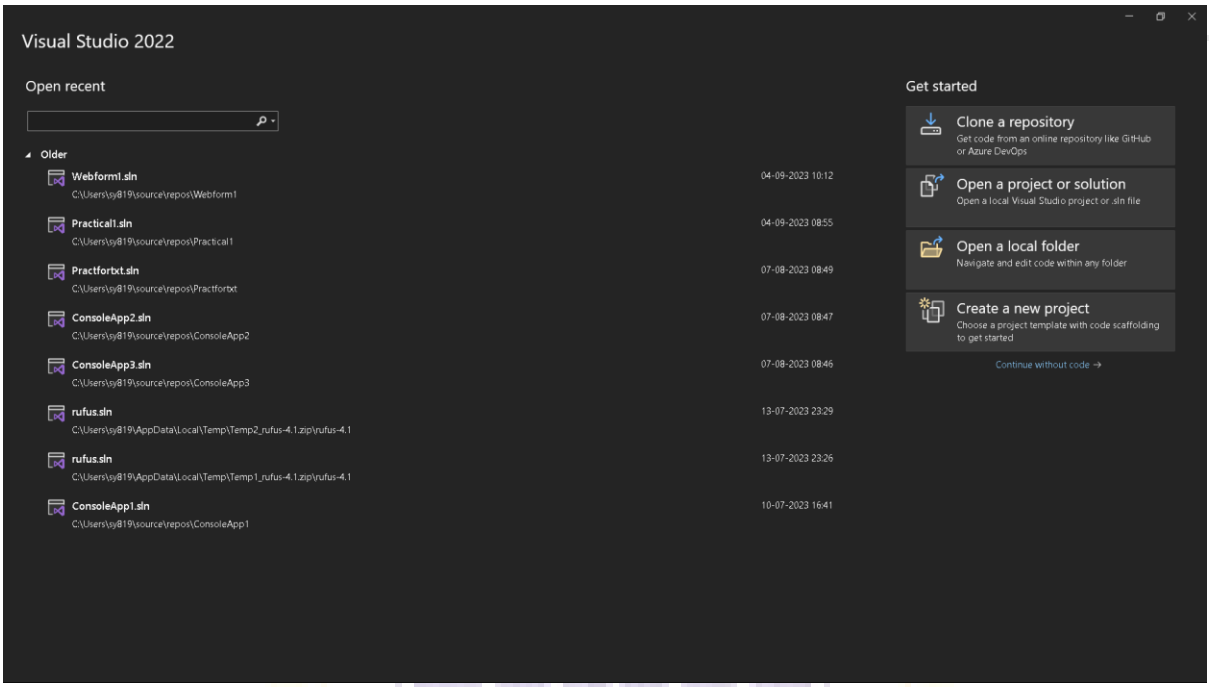
ASP.NET, an integral part of the .NET framework, is a robust web development framework that empowers developers to build interactive and scalable web applications. Offering two primary paradigms, Web Forms and MVC (Model-View-Controller), ASP.NET provides flexibility in designing web solutions. Web Forms facilitate rapid application development through a visual design approach, while MVC promotes a structured architectural pattern for greater separation of concerns. Both paradigms leverage the C# language, allowing developers to seamlessly integrate server-side logic with dynamic web content.Together, C# and ASP.NET, often harnessed within the Visual Studio integrated development environment, form a comprehensive ecosystem for crafting modern and responsive web applications. This powerful combination not only streamlines the development process but also ensures the creation of robust and feature-rich web solutions within the Microsoft technology stack.

Practical 1

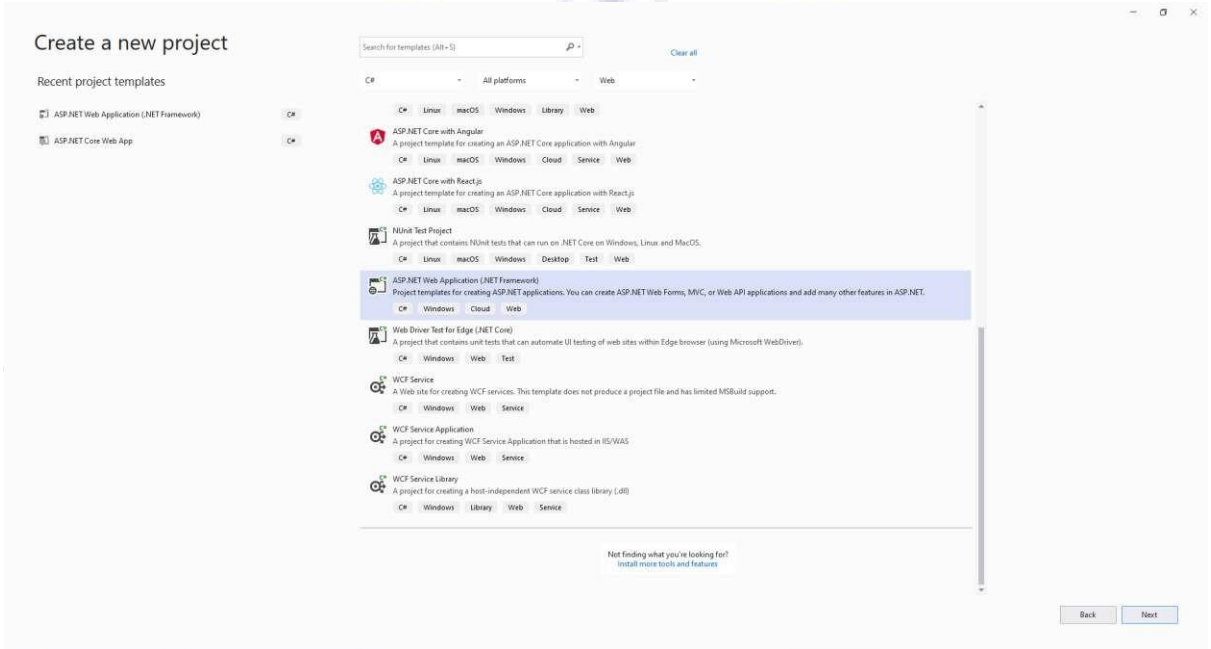
(a)

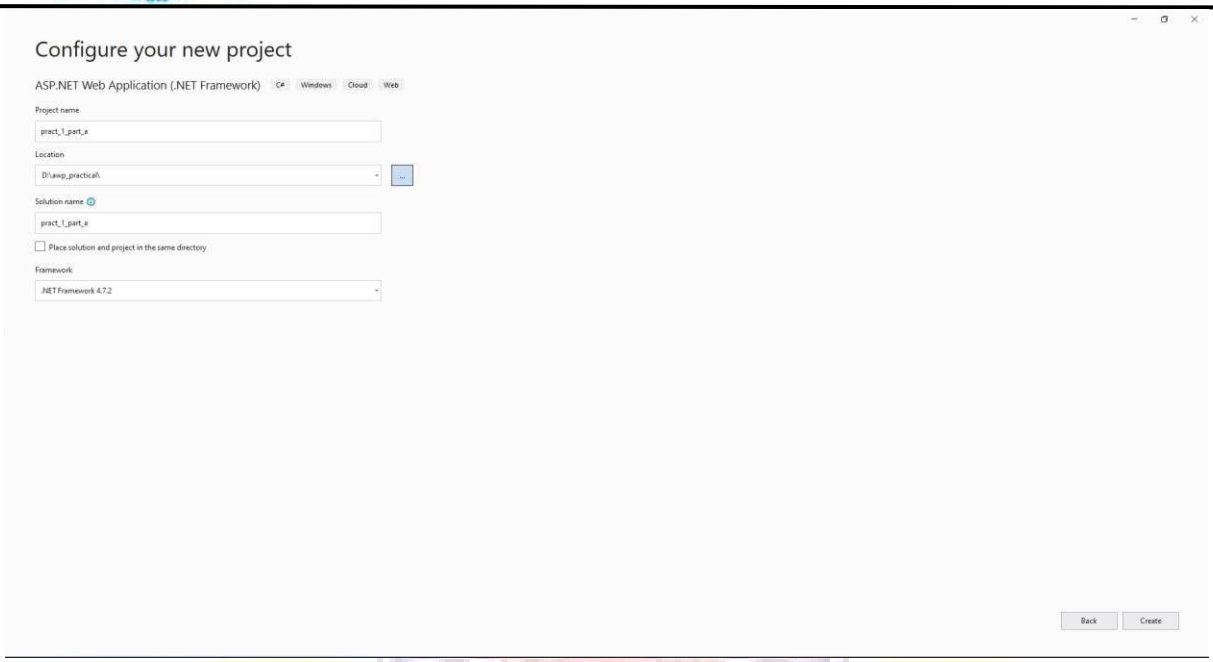
Aim. Create an application that obtains four int values from the user and displays the product.

Start >search –Visual Studio

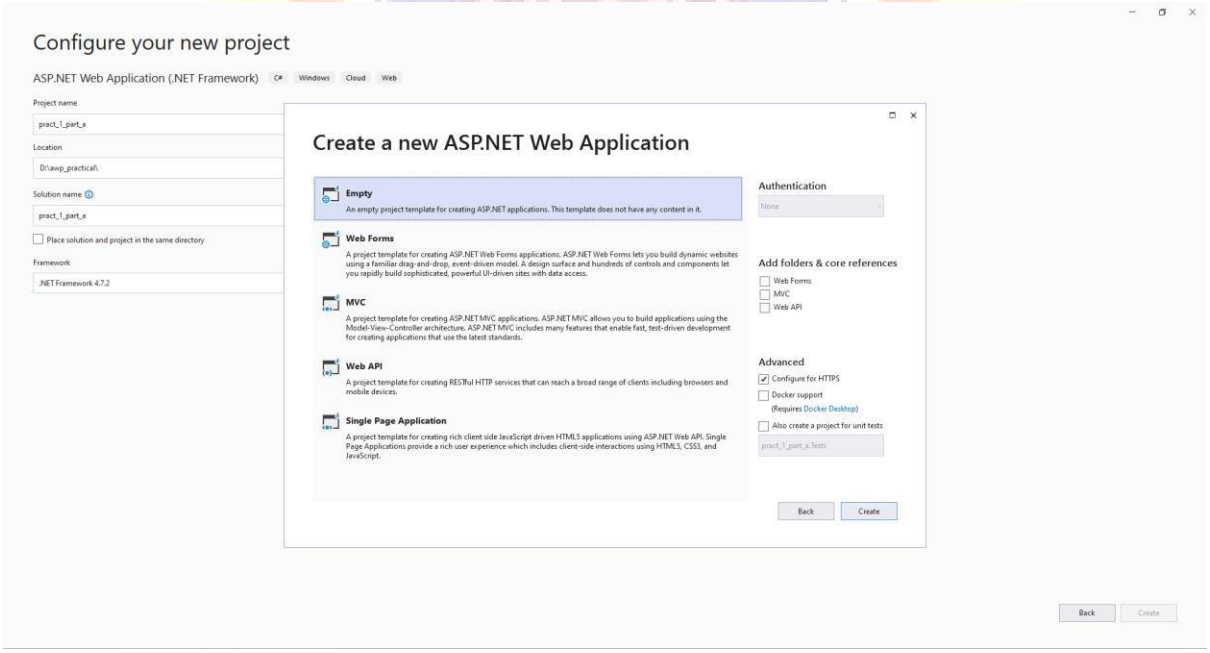


Click on create new project

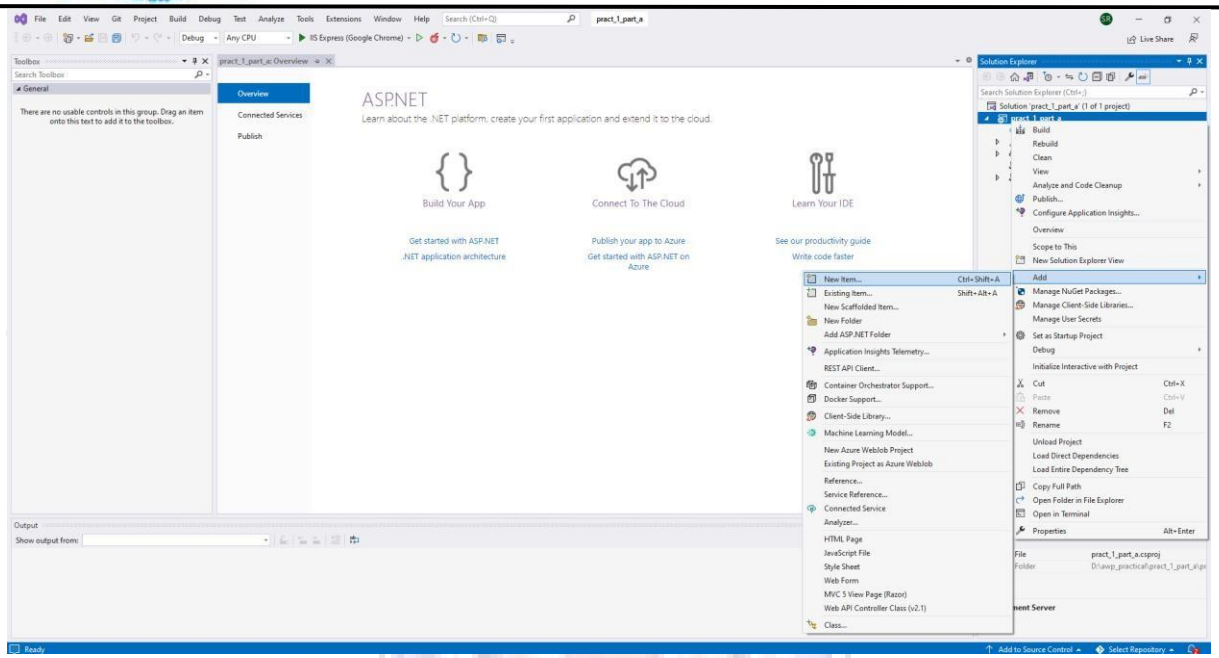




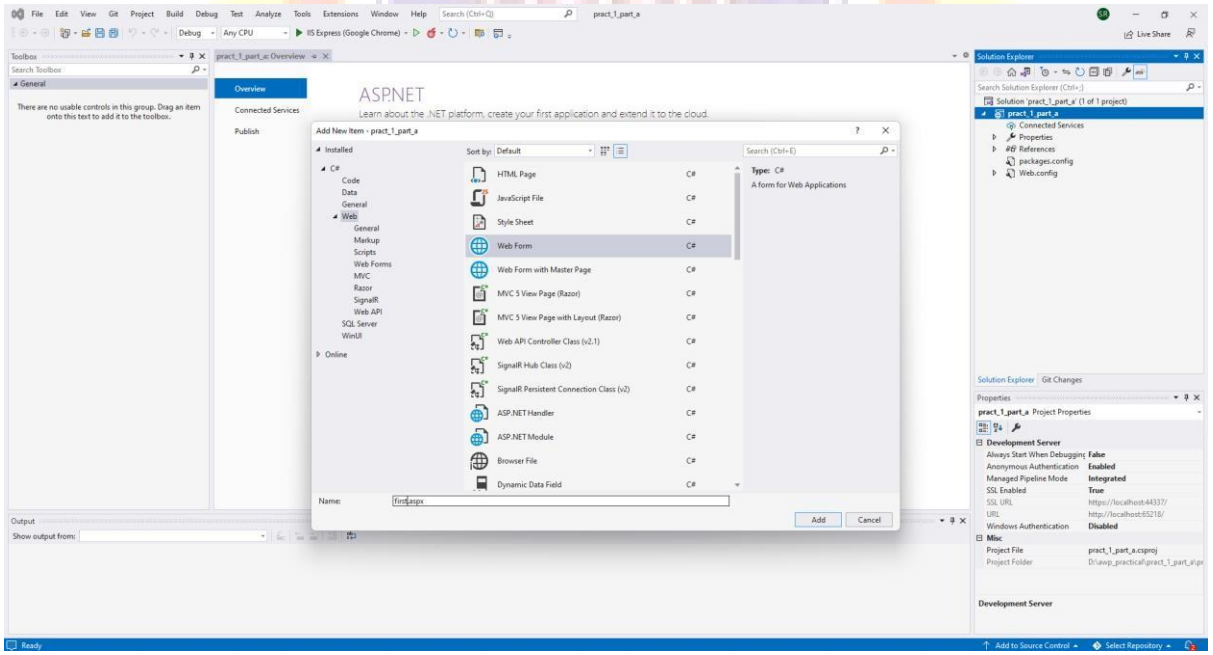
Then pop window show select empty

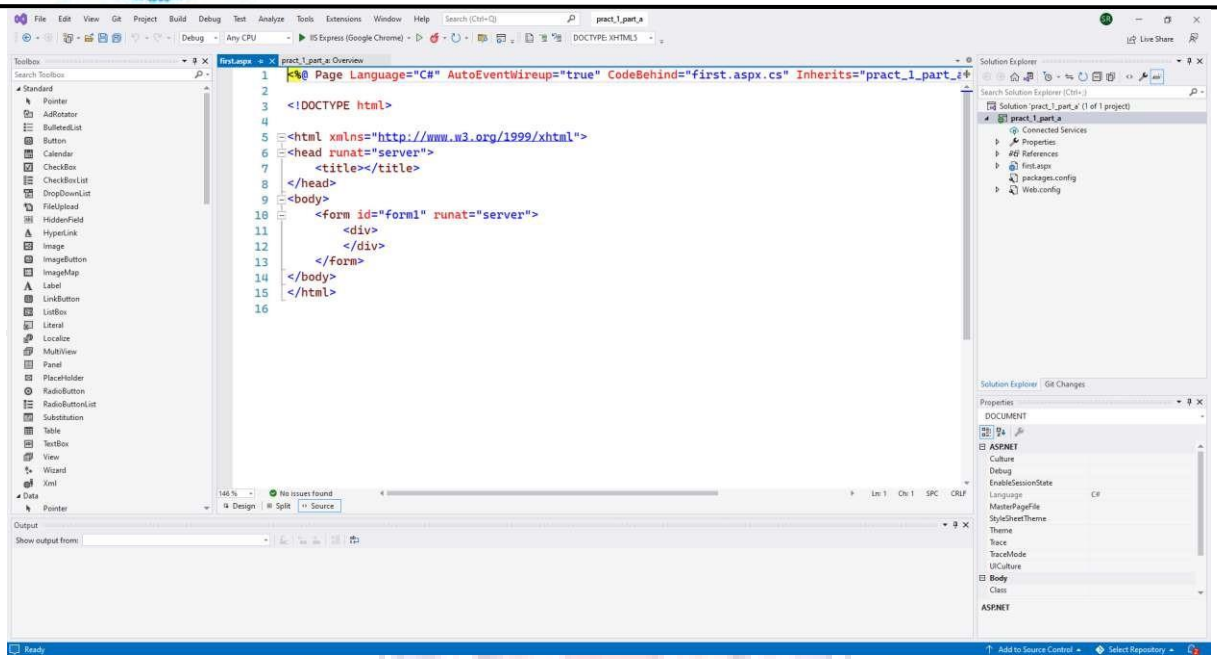


Adding new item



Add web form and change name of webform

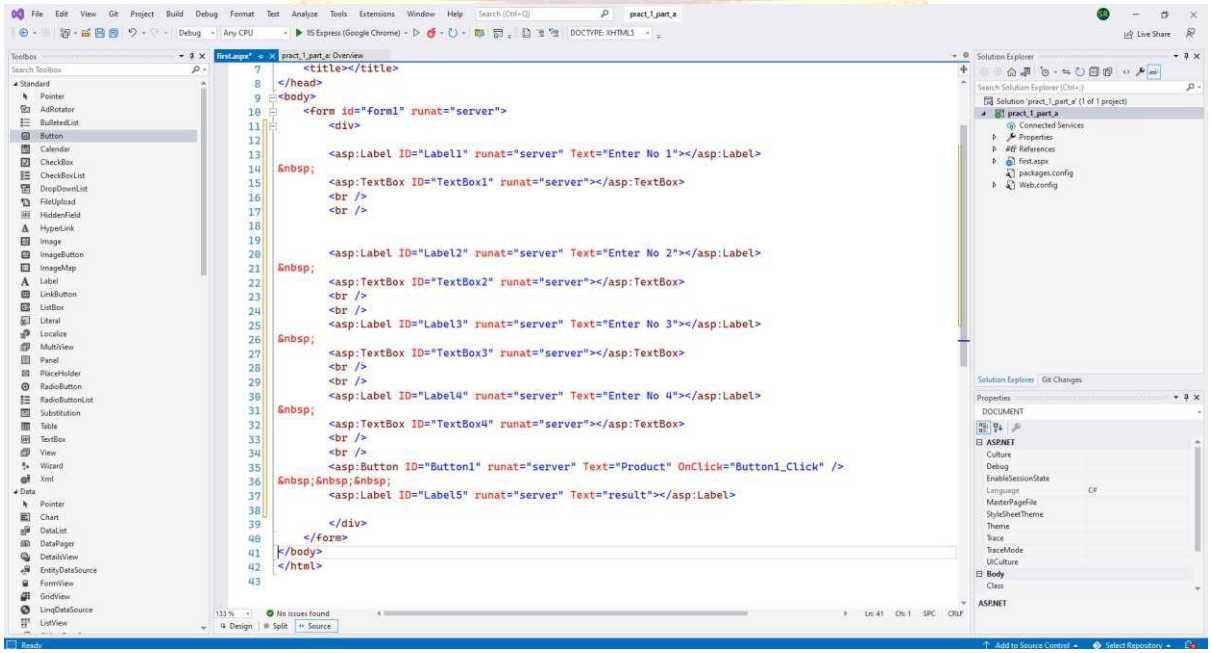




```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="first.aspx.cs" Inherits="pract_1_part_a" %>

<!DOCTYPE html>

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

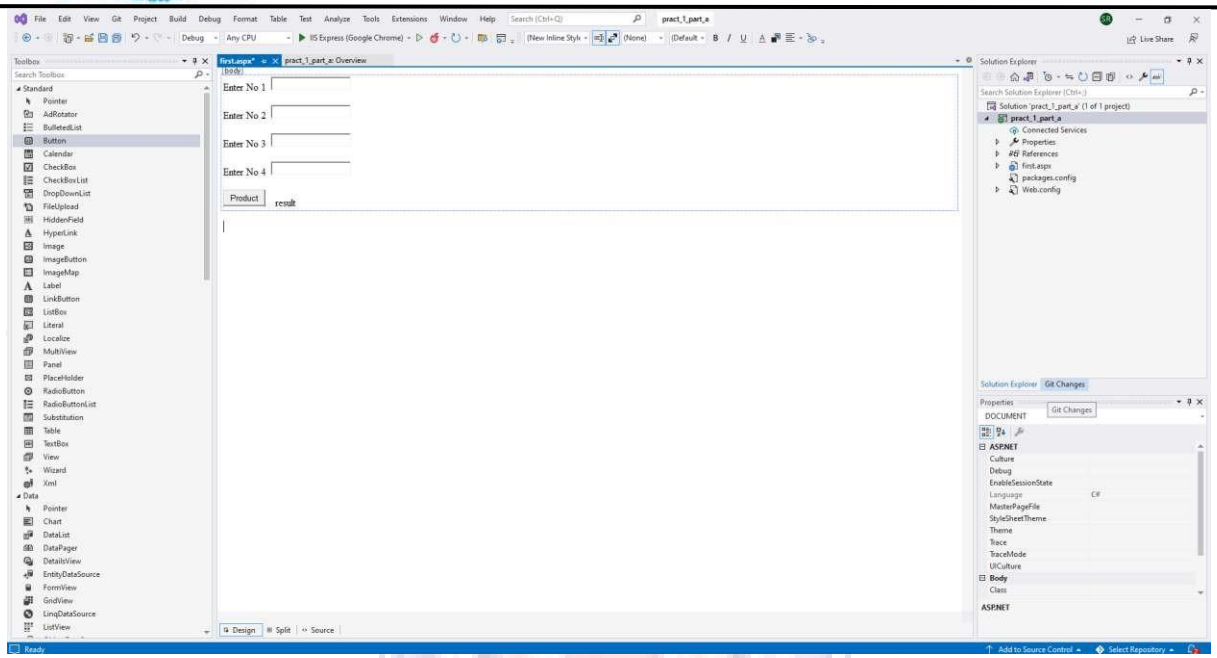
Source code interface

Drag and drop label and textbox from **Toolbox**

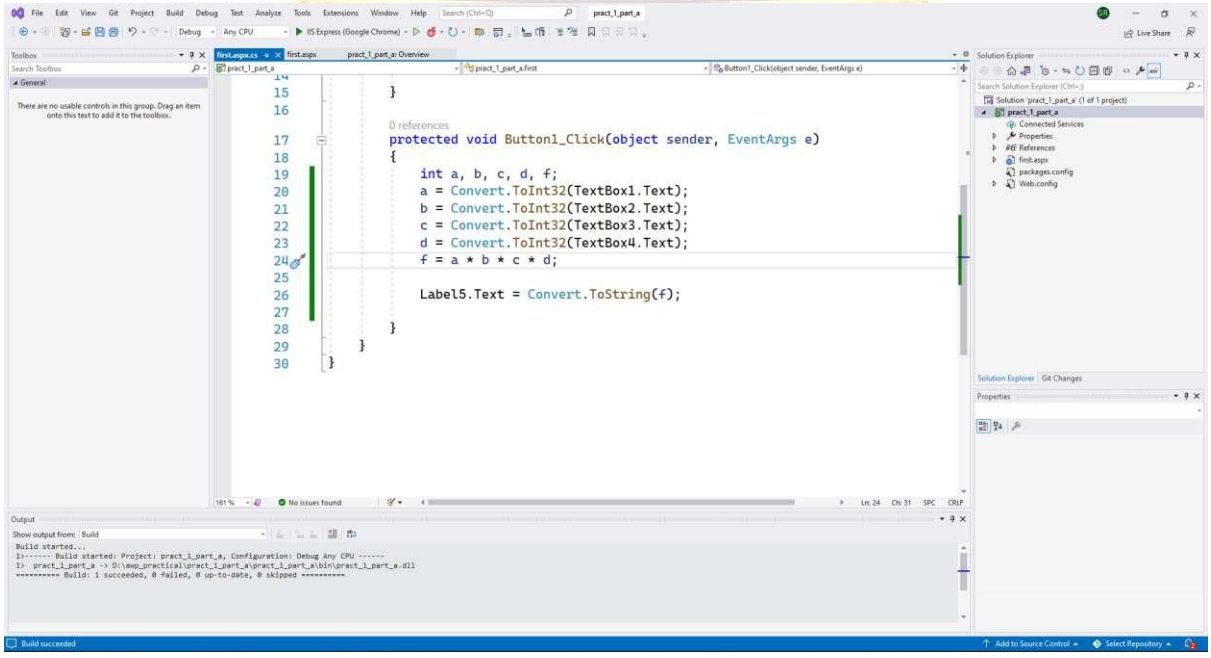


```
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>

        <asp:Label ID="Label1" runat="server" Text="Enter No 1"></asp:Label>
 
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <br />
        <br />


        <asp:Label ID="Label2" runat="server" Text="Enter No 2"></asp:Label>
 
        <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
        <br />
        <br />
        <asp:Label ID="Label3" runat="server" Text="Enter No 3"></asp:Label>
 
        <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>
        <br />
        <br />
        <asp:Label ID="Label4" runat="server" Text="Enter No 4"></asp:Label>
 
        <asp:TextBox ID="TextBox4" runat="server"></asp:TextBox>
        <br />
        <br />
        <asp:Button ID="Button1" runat="server" Text="Product" OnClick="Button1_Click" />
   
        <asp:Label ID="Label5" runat="server" Text="result"></asp:Label>

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

Design Interface

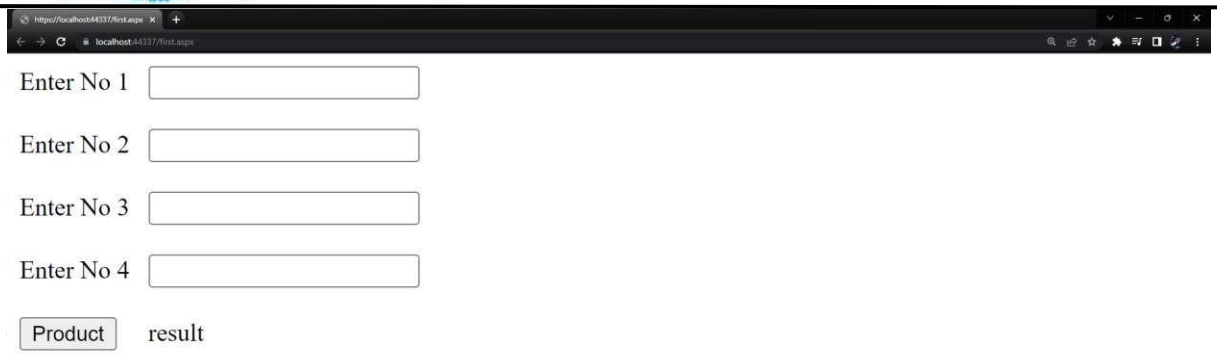


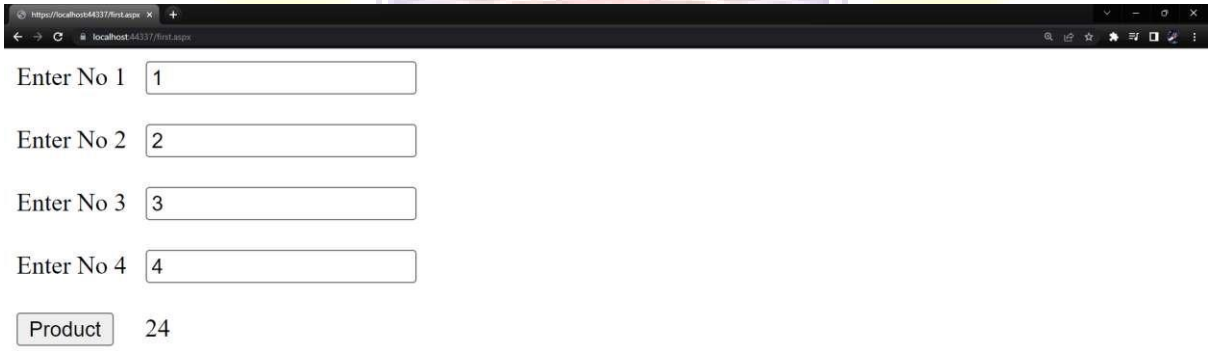
After double click on button cs page open where we can write cs code



```
protected void Button1_Click(object sender, EventArgs e)
{
    int a, b, c, d, f;
    a = Convert.ToInt32(TextBox1.Text);
    b = Convert.ToInt32(TextBox2.Text);
    c = Convert.ToInt32(TextBox3.Text);
    d = Convert.ToInt32(TextBox4.Text);
    f = a * b * c * d;

    Label5.Text = Convert.ToString(f);

}
```

Output:


Enter No 1
Enter No 2
Enter No 3
Enter No 4
Product
result


Enter No 1
1
Enter No 2
2
Enter No 3
3
Enter No 4
4
Product
24

Practical 1

Part (b)

Aim. Create an application to demonstrate string operations.

.aspx page

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="pract_1B.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
      <div>
         <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

</div
>
<div>
      <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />

</div
>
<div>
      <asp:Label ID="Label1" runat="server" Text="String length"></asp:Label>
      <br />
      <asp:Label ID="Label2" runat="server" Text="Substring"></asp:Label>
      <br />
      <asp:Label ID="Label3" runat="server" Text="Upper string"></asp:Label>
      <br />
       <asp:Label ID="Label4" runat="server" Text="Lower string"></asp:Label>
      <br />
      <asp:Label ID="Label5" runat="server" Text="Reverse string"></asp:Label>
      <br />
      <asp:Label ID="Label6" runat="server" Text="Pad left"></asp:Label>
      <br />
      <asp:Label ID="Label7" runat="server" Text="Pad right"></asp:Label>
       <br />
      <asp:Label ID="Label8" runat="server" Text="Insert"></asp:Label>
      <br />
      <asp:Label ID="Label9" runat="server" Text="Remove"></asp:Label>
       <br />
```

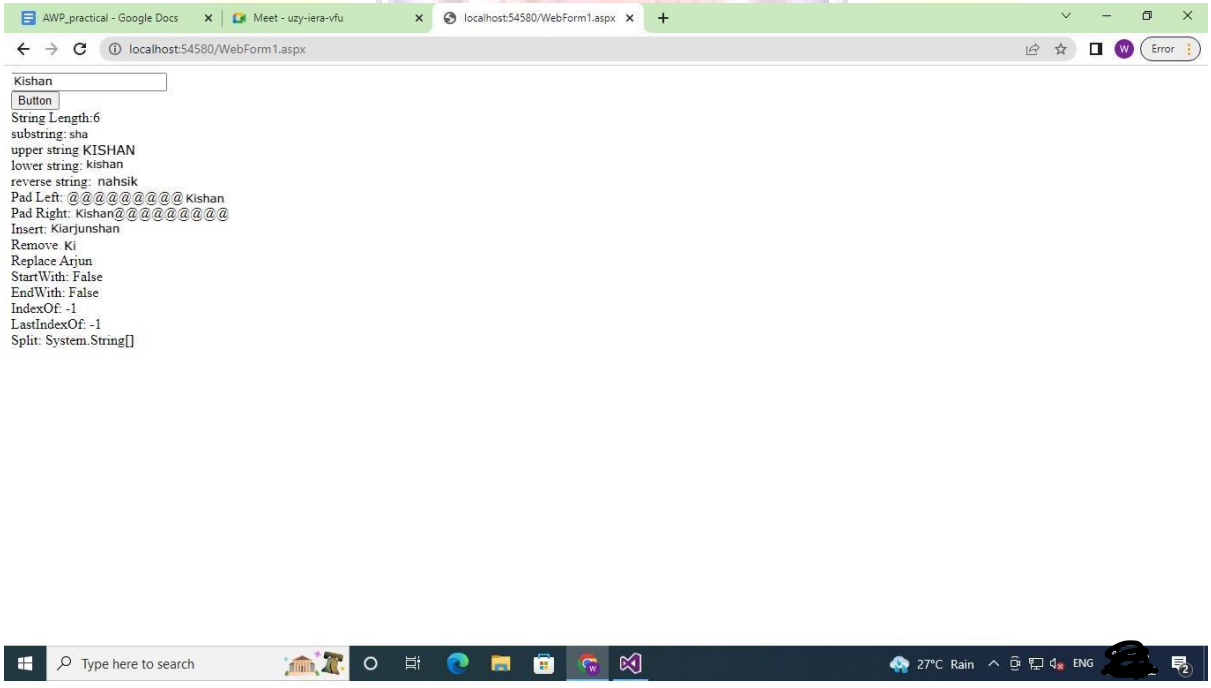
```
        <asp:Label ID="Label10" runat="server" Text="Replace"></asp:Label>
        <br />
        <asp:Label ID="Label11" runat="server" Text="stat with"></asp:Label>
         <br />
        <asp:Label ID="Label12" runat="server" Text="end with"></asp:Label>
        <br />
        <asp:Label ID="Label13" runat="server" Text="index Of"></asp:Label>
         <br />
        <asp:Label ID="Label14" runat="server" Text="Last index Of"></asp:Label>
        <br />
        <asp:Label ID="Label15" runat="server" Text="Split"></asp:Label>

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

.cs page

```
using System;
using
System.Collections.Ge
neric; using
System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls;

namespace pract_1B
{     public partial class WebForm1 :
System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            string s = TextBox1.Text;
            Label1.Text = "string length:" + s.Length;
Label2.Text = "substring:" + s.Substring(4, 3);
            Label3.Text = "upper string:" +
s.ToUpper();            Label4.Text = "lower
string:" + s.ToLower();            string rev =
"";            for (int i = s.Length - 1; i >= 0; i-
-)
            {
                rev = rev + s[i];
```

```
        }
        Label5.Text = "reverse string:" + rev.ToString();
Label6.Text = "Pad Left:" + s.PadLeft(18, '@');
        Label7.Text = "Pad Right:" + s.PadRight(18, '@');
        Label8.Text = "Insert:" + s.Insert(2, "Yadav");
        Label9.Text = "Remove:" + s.Remove(2, 5);
        Label10.Text = "Replace:" + s.Replace("bieber", "gupta");
        Label11.Text = "StartWith:" + s.StartsWith("Pre");
        Label12.Text = "EndWith:" + s.EndsWith("ber");
        Label13.Text = "IndexOf:" + s.IndexOf(" jus");
        Label14.Text = "LastIndexOf:" + s.LastIndexOf("tin");
        Label15.Text = "Split:" + s.Split('#');
    }
  }
}
```

Output.



Practical_1

Part (c)

Aim. Create an application to demonstrate following operations

i. Generate Fibonacci series

.aspx page

```
<body>
   <form id="form1" runat="server">
      <div>
      <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

<b
r
/>
<b
r
/>
      <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />

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

.cs page

```
using System;
using
System.Collections.Ge
neric; using
System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls;

namespace pract_1B
{    public partial class fibonacci :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
```

```
    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        int f1 = 0, f2 = 1, f3, n, co;
        n = int.Parse(TextBox1.Text);
        co = 3;
        Response.Write("Fibonaaci
Series");
Response.Write(f1 + "\t" + f2);
while (co <= n)
        {
            f3 = f1 + f2;
            Response.Write("\t" + f3);

f1 = f2;
f2 = f3;
co++;
        }
    }

  }
}
```

Output.

ii) Test for prime

numbers Cs code:

```
protected void Button1_Click(object sender, EventArgs e)

{
int
n, i,
c;
        n = int.Parse(TextBox1.Text);
        for (c = 2; c <= n - 1; c++)
        {
           if ((n % c)
== 0)
break;
```

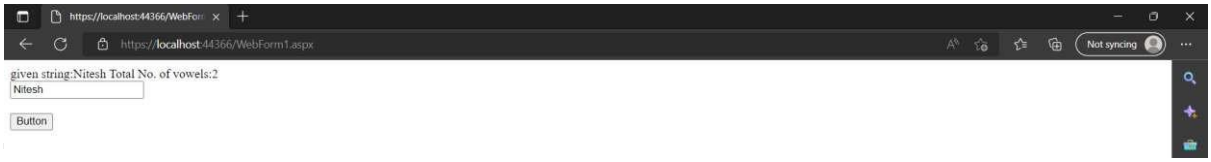
```
}
if (n
== 1)
            Response.Write(n + " is niether prime nor
commposite");            else if (c < n - 1)
            Response.Write(n + " is not prime number");
          else Response.Write(n + " is prime number");

      }
```

iii. Test for vowels

.cs code

```
protected void Button1_Click(object sender, EventArgs e)
      {
         string ch; int count = 0;
         ch = TextBox1.Text;

         for (int i = 0; i < ch.Length; i++)
         {
            if ((ch.Substring(i, 1) == "a") || (ch.Substring(i, 1) == "e") || (ch.Substring(i, 1)
== "i") || (ch.Substring(i, 1) == "o") || (ch.Substring(i, 1) == "u"))
            {
               count++;
            }
         }
```

```
        Response.Write("given string:" + ch);
        Response.Write("Total No. of vowels:" + count);
    }
```



iv) use for each

loop arrays Cs

code:

```
protected void Button1_Click(object sender, EventArgs e)
    {
        int[] a = { 5, 6, 7, 8
};          foreach (int x in
a)
Response.Write(x);

    }
```

Axpx code:

```
<form id="form1" runat="server">
    <div>

    <br />
    <br />
    <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />     </div>    </form> Output.
```

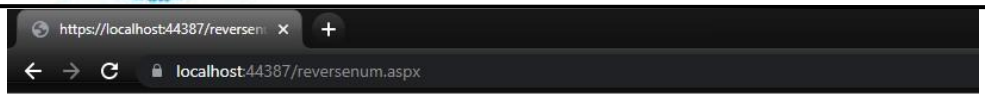
5678

Button

v. Reverse a number and find sum of digits of a number Cs code:

```
protected void Button1_Click(object sender, EventArgs e)
      {
         int n, m, r = 0, d,
sum = 0;           n =
int.Parse(TextBox1.Text);

m = n;
while (n
> 0)
         {
d = n % 10;
r = r * 10 + d;
sum = sum +
d;              n
= n / 10;

         }
         Response.Write("Reverse of " + m + "=" + r +
"<br>");          Response.Write(" Sum of its digits:"
+ sum);       }
```

Output.



Reverse of 1234=4321

Sum of its digits:10

1234

Button

Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is .NET?

2-What is . architecture?

3-What is C#?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 2

Introduction to Object Oriented concepts in C# and ASP.NET.

Solution

Object-Oriented Programming (OOP) principles serve as the bedrock for software design and development in both C# and ASP.NET, offering a systematic and efficient approach to building complex systems. In C#, an object-oriented language, these concepts are integral to the language's design philosophy. Central to OOP is the notion of classes and objects, encapsulating data and behavior into cohesive units.

In C#, developers use classes to define blueprints for objects, which are instances of those classes. Encapsulation ensures that the internal workings of a class are hidden, promoting modular and maintainable code. Inheritance allows for the creation of new classes that inherit properties and methods from existing ones, fostering code reuse. Polymorphism enables objects of different types to be treated as objects of a common base type, facilitating flexibility and extensibility.

ASP.NET, as an extension of the .NET framework, seamlessly incorporates these OOP principles. Object-oriented concepts play a crucial role in structuring ASP.NET applications, particularly in the context of ASP.NET MVC (Model-View-Controller) architecture. Models encapsulate the application's data and business logic, Views handle the user interface, and Controllers manage user input and orchestrate the interaction between models and views.

The application of OOP principles in C# and ASP.NET promotes code organization, reusability, and scalability. Through abstraction, polymorphism, and encapsulation, developers can create modular and maintainable codebases, enhancing the efficiency and flexibility of the software development process. This foundational understanding of object-oriented concepts in C# and ASP.NET empowers developers to design robust and extensible applications, aligning with the best practices of modern software engineering.

Practical 2

Part (a)

Aim. Create simple application to demonstrate use of following concepts

i.Function Overloading

Cs code:

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


   namespace pract2
   {
   class Program1
   {       public int
Area(int a, int b)
       {
return a
* b;
       }
public int
Area(int a)
       {
return
a * a;
       }
       public double Area(double a)
       {
return 3.14 * a
* a;
       }
   }
   internal class Program
       {
          static void Main(string[] args)
          {
             Program1 p = new Program1(); int res = p.Area(6, 7); int squ = p.Area(5);
double circ = p.Area(5.7);
             Console.WriteLine("area of React=" + res + "\n"); Console.WriteLine("area of
Square=" + squ + "\n"); Console.WriteLine("area of circle=" + circ + "\n");
Console.ReadLine();
```

```
    }
  }
}
```

ii) inheritance all type

single

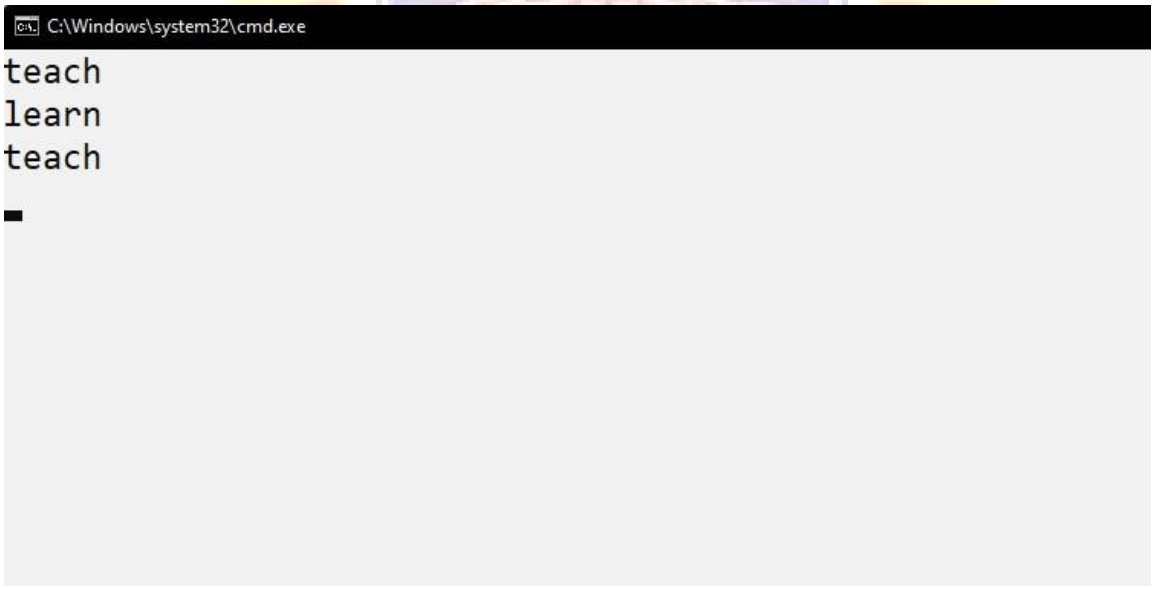
inheritan

ce Cs

code:

```
namespace pract2
{
  //single
inheritance
  class Teacher
  {
public void
Teach()
    {
      Console.WriteLine("teach");
    }
  }
  class Student : Teacher
  {
public void
Learn()
    {
```

```
            Console.WriteLine("learn");
        }
    }

    internal class inheritance
    {
        static void Main(string[] args)
        {
            Teacher T = new Teacher();
            T.Teach();
            Student S = new Student();
            S.Learn();
            S.Teach();
            Console.ReadLine();
        }

    }
}
```



ii)Multilevel

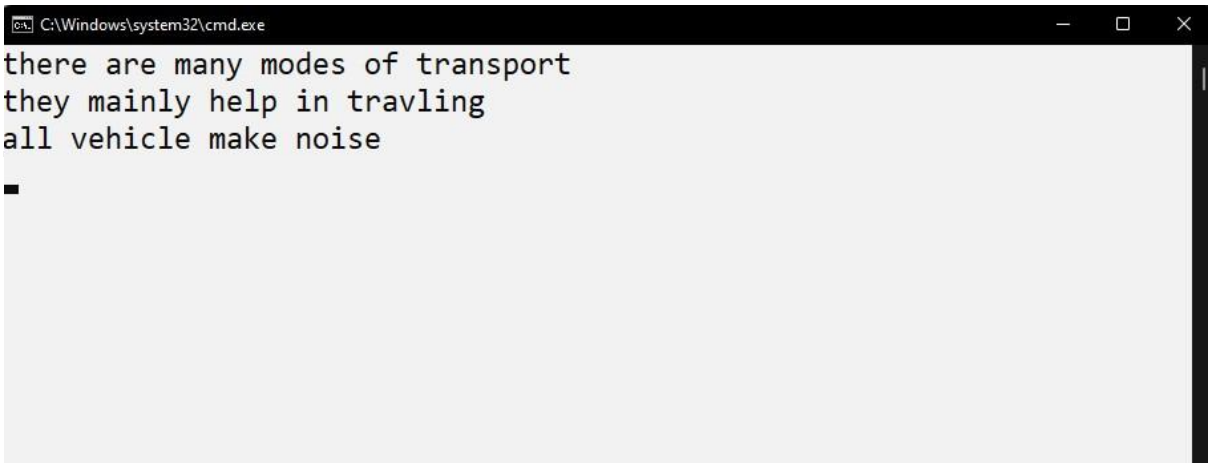
inheritance .cs

code:

```
using System;
//multilevel
inheritance
namespace
pract2
{
    class Mode
```

```
    {
        public void mode()
        {
            Console.WriteLine("there are many modes of transport");
        }
    }

    class vehicle : Mode
    {
        public void feature()
        {
            Console.WriteLine("they mainly help in travling");
        }
    }
    class inherit : vehicle
    {
public void
noise()
        {
            Console.WriteLine("all vehicle make noise");
        }
    }
    internal class multilevelinheritance
    {
        static void Main(string[] args)
        {
            inherit I = new inherit(); I.mode();
            I.feature();
            I.noise(); Console.ReadLine();
        }

    }
}
```

Output.



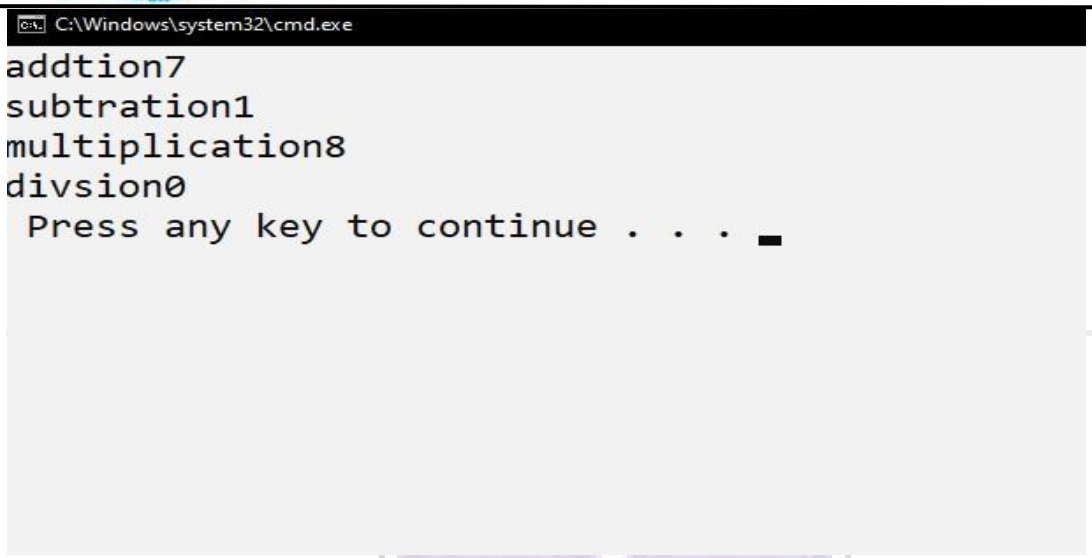
iii.hierarchical inheritance

Cs code:

```
using System;

namespace pract2
{
inter
face
calc
1
   {
      int add(int a, int b);

   }
inter
face
calc
2
   {
      int sub(int x, int y);

   }
inter
face
calc
3
   {
      int mul(int r, int s);

   }
inter
face
calc
4
   {       int
div(int c, int
d);

}
   class calulation : calc1, calc2, calc3, calc4
   {
public int
result1;
      public int add(int a, int b)
      {
return result1 = a
+ b;
```

```
        }
public int
result2;
        public int sub(int a, int b)
        {
return result2 = a
- b;

        }
public int
result3;
        public int mul(int a, int b)
        {
return result3 = a
* b;
        }
public int
result4;
        public int div(int a, int b)
        {
return result4 = a
/ b;
        }

    }       internal
class hierarchical
    {
        static void Main(string[] args)
        {
            calulation C = new calulation(); C.add(3, 4);
            C.div(4, 8);
            C.sub(3, 2);
            C.mul(4, 2);
            Console.WriteLine("addtion" + C.result1); Console.WriteLine("subtration" +
C.result2); Console.WriteLine("multiplication" + C.result3);
Console.WriteLine("divsion" + C.result4); Console.ReadKey();
        }
    }

}
```

Output.

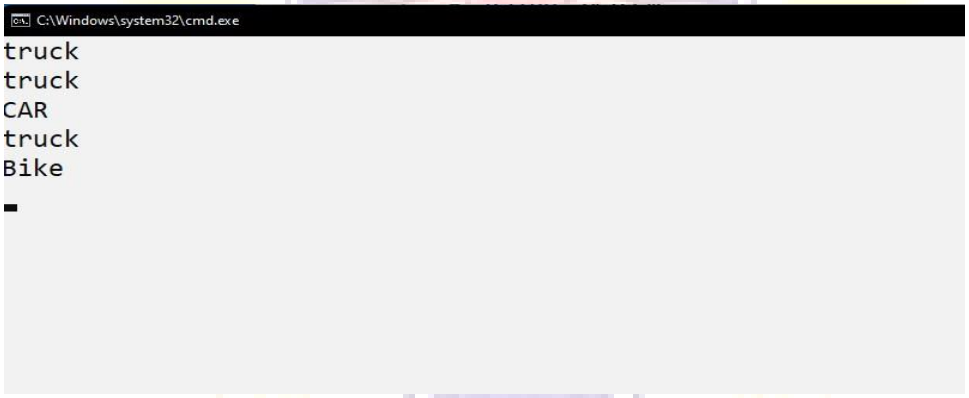


iv. Hybrid inheritance

Cs code:

```
using System;

namespace pract2
{
internal
class
hybrid
   {
      static void Main(string[] args)
      {
         grandfather g = new
grandfather();            g.truck();
dad d = new dad();
d.truck();
         d.car(); son s = new son();
         s.truck();
         s.bike();
         Console.ReadLine();

      }

   }
   class grandfather
   {
public void
truck()
      {
         Console.WriteLine("truck");
      }
```

```
    }
    class dad : grandfather
    {
        public void car()
        {
            Console.WriteLine("CAR");
        }
    }
    class son : grandfather
    {
        public void bike()
        {
            Console.WriteLine("Bike");
        }
    }

}
```



2. Constructor overloading

.cs

code

```
using
Syste
m;
name
space
pract
2
{    class
BankAcc
ount
  {
```

```
        private int cust_id, bal; private string
cust_name;        public BankAccount()
        {
            cust_id = 1000;
            cust_name = "not yet specifed"; bal = 0;
        }
        public BankAccount(int cust_id, string cust_name, int bal)
        {
            this.cust_id = cust_id; this.cust_name = cust_name; this.bal = bal;
        }
        public BankAccount(BankAccount obj)
        {
            cust_id = obj.cust_id; cust_name =
obj.cust_name;            bal = obj.bal;
        }
        public void showdata()
        {
            Console.WriteLine("cust id={0},cust name={1},cust bal={2}", cust_id,
cust_name, bal);

        }
    }

    internal class constructoroverloading
    {
        public static void Main(string[] args)
        {
            BankAccount a = new BankAccount();
            BankAccount b = new BankAccount(102, "ehtishaam", 200);
BankAccount c = new BankAccount(b); a.showdata(); b.showdata();
c.showdata(); Console.ReadLine();

        }

    }
}
```

```
C:\Windows\system32\cmd.exe
cust id=1000,cust name=not yet specifed,cust bal=0
cust id=102,cust name=ehtishaam,cust bal=200
cust id=102,cust name=ehtishaam,cust bal=200
```

3.Interfaces .cs code

```
using System;

namespace pract2
{    public
interface
Itransaction
   {
      string retcode();
      int amtfunction();

   }
   public class transaction : Itransaction
   {
      private string tcode; private int
amount;        public transaction()
      {
         tcode = ""; amount = 0;
      }
      public transaction(string c, int a)
      {
         tcode = c; amount = a;
      }
      public int amtfunction()
      {
return
amount;
      }

      public string retcode()
      {
         return tcode;
      }
   }

   internal class inter
   {       public static void
Main(string[] args)
      {

         transaction t1 = new transaction("Nitesh", 400);
transaction t2 = new transaction("Sanket", 600);
         Console.WriteLine("hello " + t1.retcode() + " your balance:" + t1.amtfunction());
Console.WriteLine("hello " + t2.retcode() + " your balance:" + t2.amtfunction());
      }

   }
}
```

Output.


Microsoft Visual Studio Debug Console
hello Nitesh your balance:400
hello Kishan your balance:600
C:\Users\sushil\source\repos\Practical_2\Practical_2\bin\Debug\net6.0\Practical_2.exe (process 11328) exited with code 0
.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .

Practical_2

Part (C)

Aim.Create simple application to demonstrate use of following concepts

i. Using Delegates and events

.c

s

code

```
using
Syste
m;

namespace pract2
{    internal
class
deligate
   {
      public delegate void Print(int value);
      static void Main(string[] args)
      {
         Print printDel = PrintNumber;

         printDel(100000);
         printDel(200);

         printDel = PrintMoney;

         printDel(100000);
printDel(200);
Console.ReadLine();

      }
      public static void PrintNumber(int num)
      {
         Console.WriteLine("Number: {0}", num);

      }
      public static void PrintMoney(int money)
      {
         Console.WriteLine("Money: {0}", money);
```

```
    }
  }
}
```

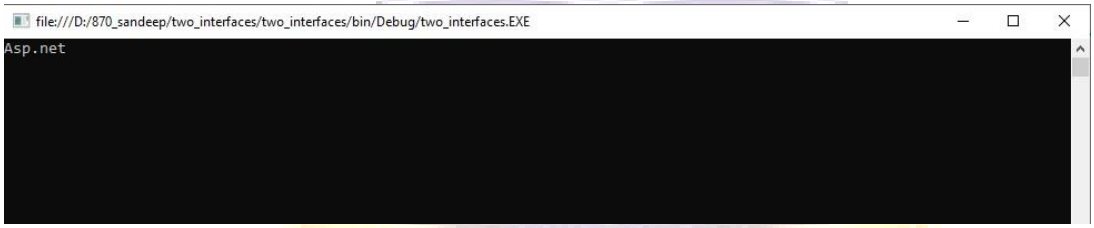
ii. Exception handling

Cs code:

```
using System;
namespace two_interfaces
{

  interface gut
  {       void setTut(int pid,
string pname);
    string getTut();
  }


  class Program : gut
  {
    protected int tutid;
protected string tutname;
public void setTut(int pid, string
pname)
    {
      tutid = pid;
      tutname = pname;
    }
    public string getTut()
    {
```

```
        return tutname;
    }
    static void Main(string[] args)
    {
        Program ptoutr = new Program();
ptoutr.setTut(1, "Asp.net");
Console.WriteLine(ptoutr.getTut());
        Console.ReadLine();
    }
  }
}
```

Output.



Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is List?

2-What is Master page?

3-What is aggregation?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 3

Introduction to Web Forms and Controls

Web Forms and Controls are fundamental components within the ASP.NET framework, playing a pivotal role in the development of dynamic and interactive web applications. Web Forms provide a structured, event-driven model for building web pages, enabling developers to create visually appealing and user-friendly interfaces through a familiar drag-and-drop design approach. The essence of Web Forms lies in their abstraction of HTML, offering a higher-level abstraction that simplifies the development process.

At the heart of Web Forms are server controls, encapsulating various HTML elements and providing enhanced functionality. These controls range from basic input fields and buttons to complex data-bound components, allowing developers to seamlessly integrate dynamic content and interactivity into their web applications. The server controls automatically manage state, handle events, and facilitate a consistent user experience across different browsers.

The versatility of Web Forms and Controls is exemplified through their ability to support rapid application development. Developers can design and structure web pages using visual tools within the Visual Studio integrated development environment, minimizing the need for extensive manual coding. This abstraction layer simplifies the complexities of web development, making it accessible to a broader audience, including those with a background in desktop application development.

In summary, Web Forms and Controls in ASP.NET offer a powerful and intuitive framework for creating feature-rich, interactive web applications. By providing a higher-level abstraction and a wide array of pre-built controls, they empower developers to efficiently design and implement dynamic user interfaces while leveraging the robust capabilities of the ASP.NET framework.

Practical 3

Part (a)

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

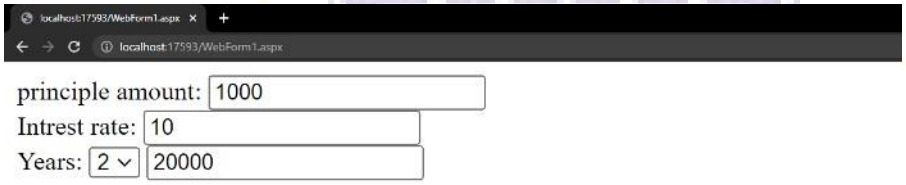
Default .aspx code

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="practical_3A._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:Label ID="Label1" runat="server" Text="Principal Amount"></asp:Label>
      <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox><br />
      <asp:Label ID="Label2" runat="server" Text="Interest Amount"></asp:Label>
      <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox><br />
      <asp:Label ID="Label3" runat="server" Text="Years"></asp:Label><br />
<asp:DropDownList ID="DropDownList1" runat="server">

         <asp:ListItem>1</asp:ListItem>
<asp:ListItem>2</asp:ListItem>
<asp:ListItem>3</asp:ListItem>
<asp:ListItem>4</asp:ListItem>
         <asp:ListItem>5</asp:ListItem>
      </asp:DropDownList><br />
      <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>
</div>
   </form>
</body>
</html>
```

.cs code

```
using System;
```

```
namespace
new_pract_3a
{    public partial class WebForm1 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void TextBox3_TextChanged(object sender, EventArgs e)
      {

      }

      protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
      {
         double p =
Convert.ToDouble(TextBox1.Text);          double r
= Convert.ToDouble(TextBox2.Text);
         double t = Convert.ToDouble(DropDownList1.SelectedValue);
         TextBox3.Text = Convert.ToString(p * (1 * r * t));
      }
   }
    }
```



Practical 3

Part (B)

Aim. Demonstrate the use of Calendar control to perform following operations.

a) **Display messages in a calendar control**

b) **Display vacation in a calendar control**

c) **Selected day in a calendar control using style**

1.Webform.aspx code:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="pract_3B.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:Calendar ID="Calendar1" runat="server" BackColor="White"
BorderColor="#3366CC" BorderWidth="1px" CellPadding="1" DayNameFormat="Shortest"
Font-Names="Verdana" Font-Size="8pt" ForeColor="#003399" Height="200px"
Width="220px" OnDayRender="Calendar1_DayRender">
         <DayHeaderStyle BackColor="#99CCCC" ForeColor="#336666" Height="1px"
/>
         <NextPrevStyle Font-Size="8pt" ForeColor="#CCCCFF" />
         <OtherMonthDayStyle ForeColor="#999999" />
         <SelectedDayStyle BackColor="#009999" Font-Bold="True"
ForeColor="#CCFF99" />
         <SelectorStyle BackColor="#99CCCC" ForeColor="#336666" />
         <TitleStyle BackColor="#003399" BorderColor="#3366CC" BorderWidth="1px"
Font-Bold="True" Font-Size="10pt" ForeColor="#CCCCFF" Height="25px" />
         <TodayDayStyle BackColor="#99CCCC" ForeColor="White" />
         <WeekendDayStyle BackColor="#CCCCFF" />
      </asp:Calendar>
   </div>
      <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label><br />
      <asp:Label ID="Label2" runat="server" Text="Labe2"></asp:Label><br />
```

```
        <asp:Label ID="Label3" runat="server" Text="Labe3"></asp:Label><br />
<asp:Label ID="Label4" runat="server" Text="Labe4"></asp:Label><br />
<asp:Label ID="Label5" runat="server" Text="Labe5"></asp:Label><br />
        <asp:Button ID="Button1" runat="server" Text="Result"
OnClick="Button1_Click" />     </form>
</body>
</html>
```

Webform.aspx.cs code:

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

namespace pract_3B
{     public partial class WebForm1 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }



      protected void Calendar1_DayRender(object sender, DayRenderEventArgs e)
      {
         if (e.Day.Date.Day == 5 && e.Day.Date.Month == 9)
         {
            e.Cell.BackColor =
System.Drawing.Color.Orange;              Label lbl =
new Label();            lbl.Text = "<br> Teacher
Day";             e.Cell.Controls.Add(lbl);
Image g1 = new Image();             g1.ImageUrl =
"dog.jpg";
            g1.Height =
20;              g1.Width =
20;
            e.Cell.Controls.Add(g1);
         }
         if (e.Day.Date.Day == 31 && e.Day.Date.Month == 0)
         {
            Calendar1.SelectedDate = new DateTime(2022, 8, 31);
            Calendar1.SelectedDates.SelectRange(Calendar1.SelectedDate,
Calendar1.SelectedDate.AddDays(10));
            Label lbl1 = new Label();
lbl1.Text = "<br>Ganpati ";
e.Cell.Controls.Add(lbl1);
```

```
        }
    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        Calendar1.Caption = "kalinary";
        Calendar1.FirstDayOfWeek = FirstDayOfWeek.Sunday;
        Calendar1.NextPrevFormat = NextPrevFormat.ShortMonth;
        Calendar1.TitleFormat = TitleFormat.Month;
        Label2.Text = "Today date" + Calendar1.TodaysDate.ToShortDateString();
        Label3.Text = "Ganpati vaction start: 8-31-2022";
        TimeSpan d = new DateTime(2022, 8, 31) - DateTime.Now;
        Label4.Text = "Day remaining for ganpati vaction " +
d.Days.ToString();          TimeSpan d1 = new DateTime(2022, 12, 31) -
DateTime.Now;          Label5.Text = "Day remaning for new year" +
d.Days.ToString();          if
(Calendar1.SelectedDate.ToShortDateString() == "8-31-2022")

            Label3.Text = "<b>ganpati festival</b>";
if(Calendar1.SelectedDate.ToShortDateString()=="9-9-2022")
Label3.Text = "ganpati festival end";

    }
  }
    }
```

kalinary

Aug			September			Oct
Su	Mo	Tu	We	Th	Fr	Sa
28	29	30	31	1	2	3
4	5 Teacher Day	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	1
2	3	4	5	6	7	8

Label
Today date13-09-2022
Ganpati vaction start: 8-31-2022
Day remaining for ganpati vaction -13
Day remaning for new year-13
Result

Practical 3

Part (c)

Aim. Demonstrate the use of Treeview control perform following operations.
Treeview control and datalist

a) Tree view control and data list

.aspx page

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="practical3c._Default" %>


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">


<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:DataList ID="DataList1" runat="server">
         <ItemTemplate>
            <table style="width:100%;">
               <tr>
                  <td>Roll No: <%# Eval("sid") %></td>
                  <td>Name: <%# Eval("sname") %></td>
                  <td>Class: <%# Eval("sclass") %></td>
               </tr>
```

```
            </table>
          </ItemTemplate>
        </asp:DataList>
```

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

Student_detail.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<studentdetail>
  <student>
    <sid>1</sid>
    <sname>Ashish</sname>
    <sclass>TYIT</sclass>
  </student>
  <student>
    <sid>2</sid>
    <sname>Vighnesh</sname>
    <sclass>TYIT</sclass>
  </student>
  <student>
    <sid>3</sid>
    <sname>Nitesh</sname>
    <sclass>TYIT</sclass>
  </student>
  <student>
```

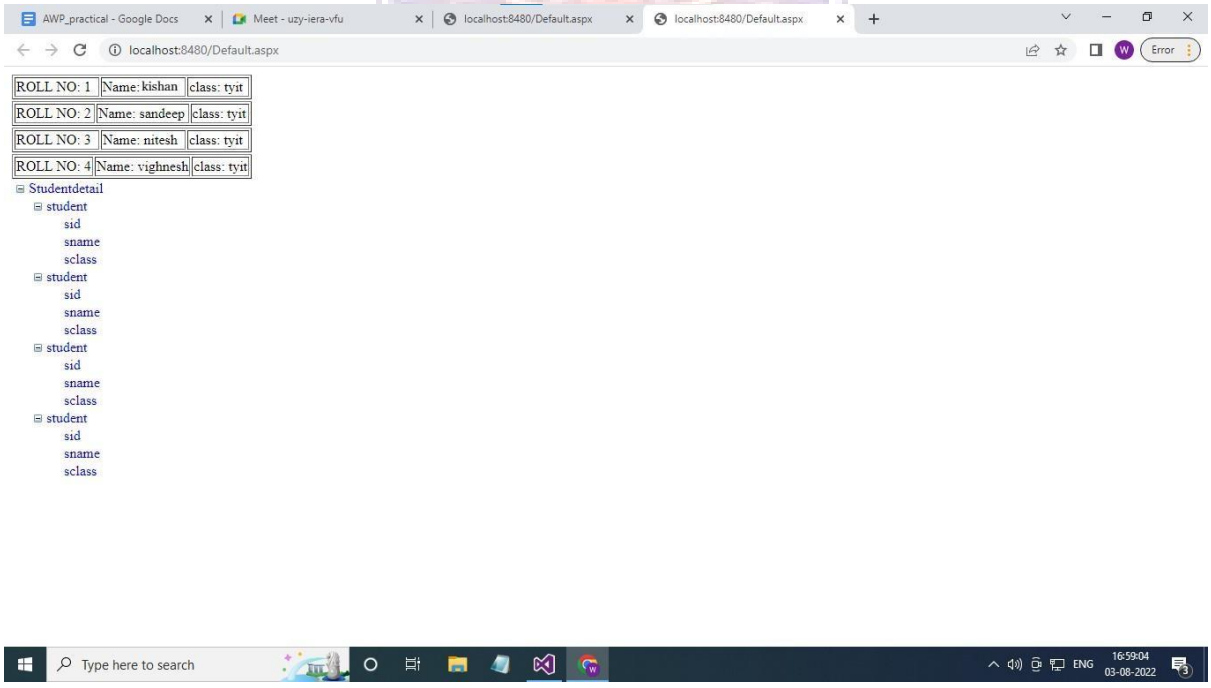
```
  <sid>4</sid>
  <sname>Sandeep</sname>
  <sclass>TYIT</sclass>
 </student>
 <student>
  <sid>5</sid>
  <sname>Kishan</sname>
  <sclass>TYIT</sclass>
 </student>
</studentdetail>
```

Default.aspx.cs

```
using System; using
System.Collections.Gen
eric; using
System.Data; using
System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls;

namespace practical3c_1
{
   public partial class _Default : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if (!IsPostBack)
         {
            BindData();


}
      }
      protected void BindData()
      {
         DataSet ds = new DataSet();
ds.ReadXml(Server.MapPath("Student.xml"));
if (ds != null && ds.HasChanges())
         {
            DataList1.DataSource = ds;
```

```
            DataList1.DataBind();

}

e
l
s
e
        {
            DataList1.DataBind();
        }
    }

  }
}
```

Output:



b. Tree view operations

Page1:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="page1.aspx.cs"
Inherits="practical_4A.WebForm1" %>

<!DOCTYPE html>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
   <h1>tree view</h1>
      <asp:TreeView ID="TreeView1" runat="server">
<Nodes>
            <asp:TreeNode NavigateUrl="~/page2.aspx" Text="home" Value="home">
<asp:TreeNode NavigateUrl="~/page3.aspx" Text="about us" Value="about
us"></asp:TreeNode>
            </asp:TreeNode>
            <asp:TreeNode NavigateUrl="~/page4.aspx" Text="gallery"
Value="gallery">
               <asp:TreeNode NavigateUrl="~/page5.aspx" Text="photo"
Value="photo"></asp:TreeNode>
            </asp:TreeNode>
            <asp:TreeNode NavigateUrl="~/page6.aspx" Text="video"
Value="video"></asp:TreeNode>
         </Nodes>
      </asp:TreeView>
      <h1>menu view</h1>
      <asp:Menu ID="Menu1" runat="server">
         <Items>
            <asp:MenuItem          NavigateUrl="~/page2.aspx"          Text="home"
Value="home">                        <asp:MenuItem NavigateUrl="~/page3.aspx"
Text="about us" Value="about us"></asp:MenuItem>
            </asp:MenuItem>
            <asp:MenuItem NavigateUrl="~/page4.aspx" Text="gallery"
Value="gallery">
               <asp:MenuItem NavigateUrl="~/page5.aspx" Text="photo"
Value="photo"></asp:MenuItem>
            </asp:MenuItem>
            <asp:MenuItem NavigateUrl="~/page6.aspx" Text="video"
Value="video"></asp:MenuItem>
         </Items>
      </asp:Menu>
   </div>
   </form>
</body>
</html>
```

Page2:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="page2.aspx.cs"
Inherits="practical_4A.WebForm2" %>
```

```
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <h1>Welcome to home </h1>
    </div>
    </form>
</body>
</html>
```

Page 3:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="page3.aspx.cs"
Inherits="practical_4A.page3" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <h1>This is about us section</h1>
    </div>
    </form>
</body>
</html>
```

Page 4:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="page4.aspx.cs"
Inherits="practical_4A.page4" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <h1>Welcome to my gallery</h1>
```

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

Page 5:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="page5.aspx.cs"
Inherits="practical_4A.page5" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <h1>This my photo section</h1>
    </div>
    </form>
</body>
</html>
```

Page 6:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="page6.aspx.cs"
Inherits="practical_4A.page6" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
    <h1>this is a video section</h1>
    </div>
    </form>
</body></html>
```

Output.

tree view

- home
  - about us
- gallery
  - photo
  - video

menu view

home ▸ about us
gallery ▸
video

Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is . NET Framework?

2-What is ADO?

3-What is Sql ?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 4

Introduction to Form Controls

Form controls are essential elements in web development, serving as the interactive building blocks that enable user input and engagement within HTML forms. These controls play a pivotal role in gathering data from users and facilitating various types of interactions on web pages. From simple text input fields and checkboxes to more complex dropdown menus and file upload buttons, form controls provide the means to create dynamic and user-friendly interfaces.

In the context of web development, HTML forms act as containers for these form controls, allowing users to input information that can be submitted to a server for processing. Each form control possesses unique characteristics and functionalities tailored to different types of data and user interactions. For instance, text input fields are commonly used for short pieces of information, while radio buttons and checkboxes allow users to make selections.

The versatility of form controls extends beyond basic HTML with the integration of JavaScript, CSS, and server-side technologies. JavaScript enables the creation of dynamic and responsive forms by manipulating form controls in real-time, validating input, and enhancing user experience. Cascading Style Sheets (CSS) contribute to the visual styling and layout of form controls, ensuring a cohesive and aesthetically pleasing design.

In summary, form controls are the fundamental components that empower developers to create interactive and data-capturing elements within web applications. Whether building a simple contact form or a complex data entry interface, understanding and effectively utilizing form controls is crucial for crafting engaging and functional web experiences.

Practical 4

Part A

Aim. Create a registration form to demonstrate use of various Validation controls.

Default.aspx page code:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="pract_4a._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
   <style type="text/css">
.auto-style1 {
        height: 24px;
      }
   </style>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <table style="width: 100%;">
         <tr>
            <td colspan="3" ><b><center>Registration Form</center></b></td>
         </tr>
         <tr>
            <td>
               <asp:Label ID="Label1" runat="server" Text="Full
Name"></asp:Label></td>
            <td>
               <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox></td>
            <td>
               <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
ErrorMessage="Enter full name"
ControlToValidate="TextBox1"></asp:RequiredFieldValidator>
               <asp:CustomValidator ID="CustomValidator1" runat="server"
ControlToValidate="TextBox1" ErrorMessage="CustomValidator"
OnServerValidate="CustomValidator1_ServerValidate"></asp:CustomValidator>
<asp:Label ID="Label8" runat="server"></asp:Label>
            </td>
         </tr>
         <tr>
            <td>
```

```
            <asp:Label ID="Label2" runat="server" Text="Age"></asp:Label></td>
          <td>
            <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox></td>
          <td>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"
ErrorMessage="Enter your age"
ControlToValidate="TextBox2"></asp:RequiredFieldValidator>
            <asp:RangeValidator ID="RangeValidator1" runat="server"
ControlToValidate="TextBox2" ErrorMessage="18-30" MaximumValue="30"
MinimumValue="18"></asp:RangeValidator>
          </td>
        </tr>
        <tr>
          <td>
            <asp:Label ID="Label3" runat="server"
Text="Contact"></asp:Label></td>
          <td>
            <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox></td>
          <td>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server"
ErrorMessage="contact details"
ControlToValidate="TextBox3"></asp:RequiredFieldValidator></td>
        </tr>
        <tr>
          <td class="auto-style1">
            <asp:Label ID="Label4" runat="server"
Text="Email"></asp:Label></td>          <td class="auto-style1">
            <asp:TextBox ID="TextBox4" runat="server"></asp:TextBox></td>
          <td class="auto-style1">
            <asp:RequiredFieldValidator ID="RequiredFieldValidator4" runat="server"
ErrorMessage="Enter your emails"
ControlToValidate="TextBox4"></asp:RequiredFieldValidator>
            <asp:RegularExpressionValidator ID="RegularExpressionValidator1"
runat="server" ControlToValidate="TextBox4" ErrorMessage="Enter proper mail"
ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-
.]\w+)*"></asp:RegularExpressionValidator>
          </td>
        </tr>
        <tr>
          <td>
            <asp:Label ID="Label6" runat="server" Text="Password"></asp:Label>
<asp:Label ID="Label5" runat="server" Text="Password"></asp:Label></td>
          <td>
            <asp:TextBox ID="TextBox5" runat="server"
TextMode="Password"></asp:TextBox></td>
          <td>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator5" runat="server"
ErrorMessage="Type your password"
```

```
ControlToValidate="TextBox5"></asp:RequiredFieldValidator></td>
        </tr>
        <tr>
          <td>
            <asp:Label ID="Label7" runat="server" Text="Re-Type
password"></asp:Label></td>
          <td>
            <asp:TextBox ID="TextBox6" runat="server"
TextMode="Password"></asp:TextBox></td>
          <td>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator6" runat="server"
ErrorMessage="Re-type password"
ControlToValidate="TextBox6"></asp:RequiredFieldValidator>
            <asp:CompareValidator ID="CompareValidator1" runat="server"
ControlToCompare="TextBox5" ControlToValidate="TextBox6"
ErrorMessage="Passowrd should be same"></asp:CompareValidator>
          </td>
        </tr>
        <tr colspan="3">
          <td>
            <asp:Button ID="Button1" runat="server" Text="Submit"
OnClick="Button1_Click" /></td>
          <td>
             </td>
          <td>

 </td>
</tr>
        <tr>
          <td></td>
<td></td>
          <td></td>
        </tr>

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

Aspx.cs code:

```
using System; using
System.Collections.Ge
neric; using
System.Linq;
using
System.Text.RegularExpressi
```

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

namespace pract_4a
{    public partial class _Default :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void CustomValidator1_ServerValidate(object source,
ServerValidateEventArgs args)
      {
         if (!Regex.IsMatch(TextBox1.Text, @"^[a-zA-Z]+$"))
         {
            Label8.Text = "Enter character only";
         }
      }

      protected void Button1_Click(object sender, EventArgs e)
      {

      }
   }
}
```

Output.

Practical 4

Part (b)

Aim. Create Web Form to demonstrate use of Ad rotator Control.

.aspx file :

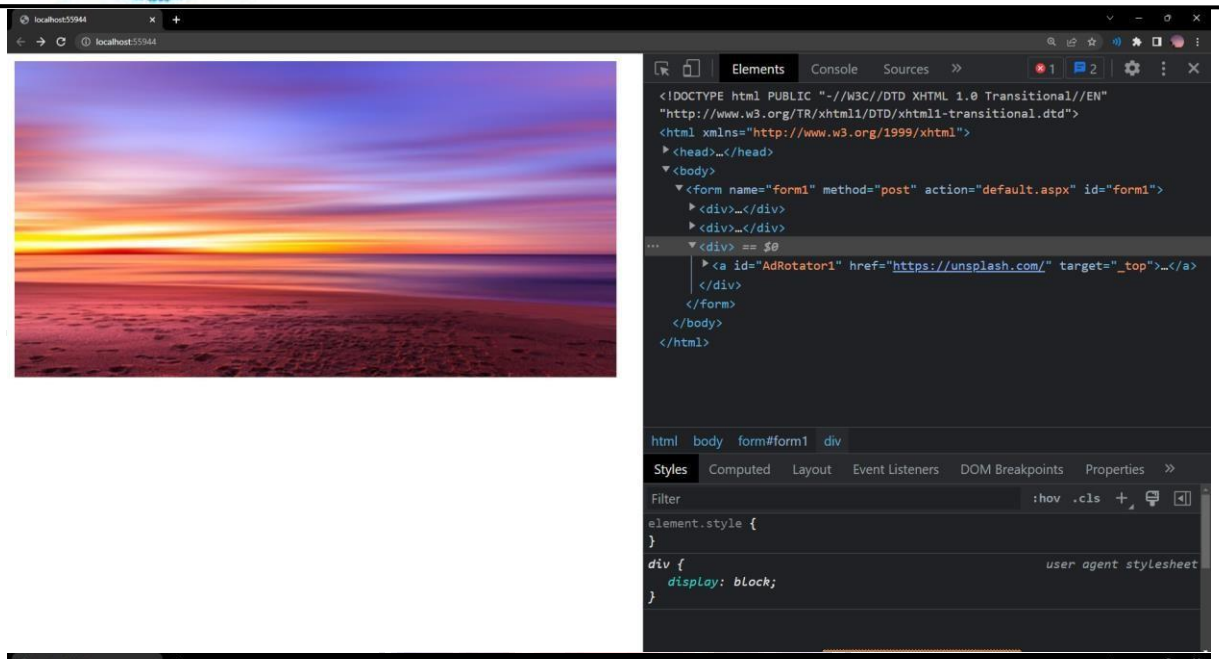
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="Pract4A._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:AdRotator ID="AdRotator1" runat="server"
DataSourceID="XmlDataSource1" />
      <asp:XmlDataSource ID="XmlDataSource1" runat="server"
DataFile="~/Pract4BXML.xml"></asp:XmlDataSource>
   </div>
   </form>
</body>
</html>
```

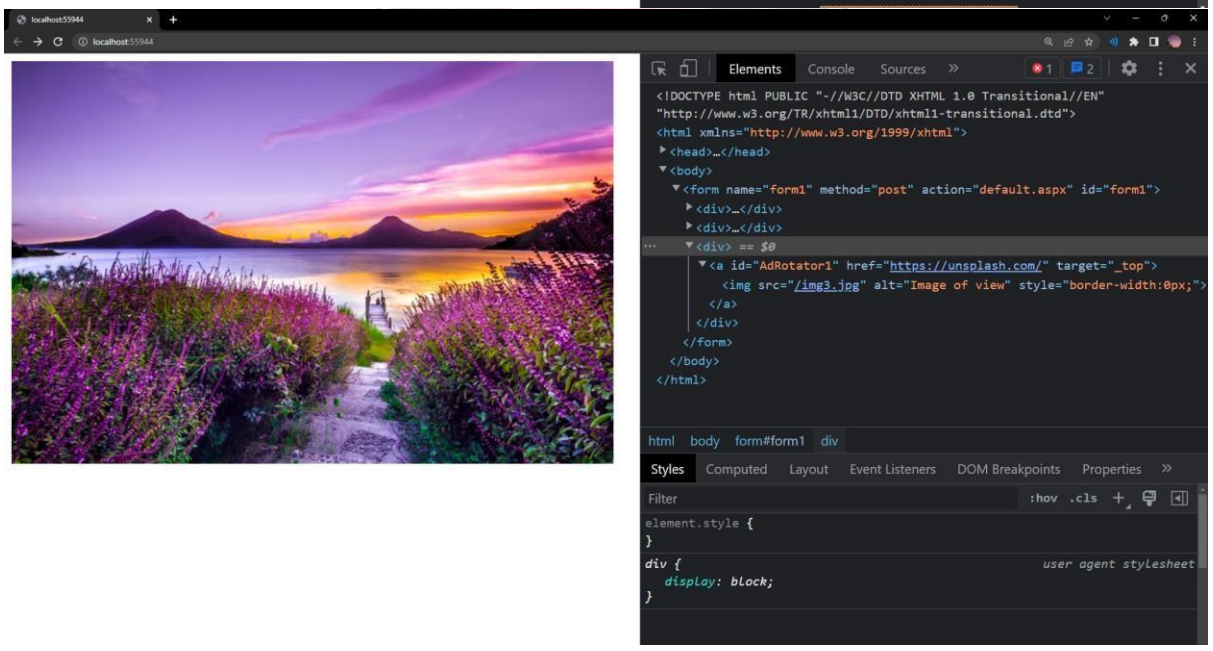
.xml file:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="Pract4A._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

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

```
    <div>
        <asp:AdRotator ID="AdRotator1" runat="server"
DataSourceID="XmlDataSource1" />
        <asp:XmlDataSource ID="XmlDataSource1" runat="server"
DataFile="~/Pract4BXML.xml"></asp:XmlDataSource>
    </div>
    </form>
</body>
</html>
```

Output:





Practical 4

Part(c)

Aim. Create Web Form to demonstrate use User Controls.

.aspx page:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Pract4C.aspx.cs"
Inherits="Pract4A.Pract4C" %>
<%@ Register Src="~/WebUserControl1.ascx" TagName="Route"
TagPrefix="Routes"%>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:Label ID="Label1" runat="server" Text="Welcome to My
Page"></asp:Label>    </div>
      <br />
      <div>
         <Routes:Route runat="server" />
      </div>
   </form>
</body>
</html>
```

Webusercontrol.ascx page:

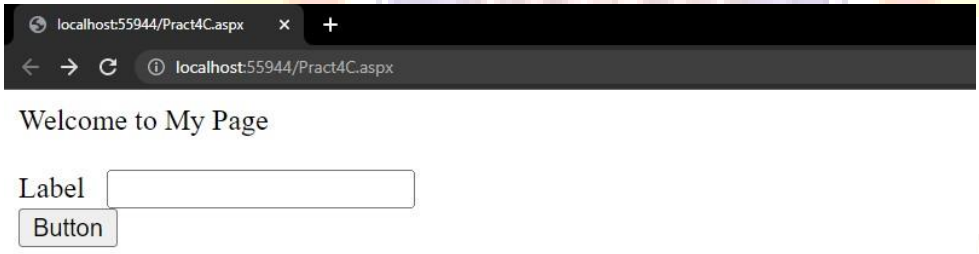
```
<%@ Control Language="C#" AutoEventWireup="true"
CodeBehind="WebUserControl1.ascx.cs" Inherits="Pract4A.WebUserControl1" %>


<asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>  

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>  <br />

<asp:Button ID="Button1" runat="server" Text="Button" OnClick="Button1_Click" />
```

Webusercontrol.ascx.cs page:

```
using System;
using
System.Collections.Ge
neric; using
System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls;
```

```
namespace Pract4A
{    public partial class WebUserControl1 :
System.Web.UI.UserControl
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         Response.Redirect("Default.aspx");
      }

}
}
```

Output:



Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is Cookies?

2-What is CSS?

3-What is View?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 5

Introduction to Navigation, Beautification and Master page.

Theory **Navigation:** Effective navigation is paramount for a positive user experience. It involves the organization and presentation of links and menus that guide users through different sections of a website. Well-designed navigation enhances accessibility and ensures that users can easily locate information or features. Common navigation elements include menus, breadcrumbs, and hyperlinks, each serving to create an intuitive and structured browsing experience.

Beautification: The visual appeal of a website significantly influences user engagement. Beautification involves the use of design elements, colors, typography, and multimedia to create an aesthetically pleasing interface. Consistent styling across pages enhances brand identity and user recognition. Cascading Style Sheets (CSS) play a crucial role in beautification, allowing developers to apply styles consistently and efficiently throughout the website. Attention to detail in design contributes to a polished and professional appearance.

Master Page: A master page is a template that defines the common structure and elements of a website, ensuring a consistent layout across multiple pages. It serves as a framework for maintaining uniformity in headers, footers, navigation menus, and other shared components. By using master pages, developers can streamline the design process, make global updates effortlessly, and enforce a cohesive look and feel. This not only saves time but also enhances the maintainability and scalability of the website.

In summary, navigation, beautification, and master pages are crucial components in crafting a successful web presence. Seamless navigation enhances user-friendliness, beautification enhances visual appeal, and master pages promote consistency and ease of maintenance. A holistic approach to these elements ensures that web developers can create websites that are both aesthetically pleasing and functionally robust, resulting in a positive and engaging user experience.

Practical 5

Part (a)

Aim. Create Web Form to demonstrate use of Website Navigation controls and Site Map.

Home page:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="home.aspx.cs"
Inherits="WebApplication1.home" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>Home page</title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:TreeView ID="TreeView1" runat="server"
DataSourceID="SiteMapDataSource1"></asp:TreeView>
      <asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server" />
</div>
   </form>
</body>
</html>
```

About page:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="aboutus.aspx.cs"
Inherits="WebApplication1.aboutus" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>About us</title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
   <img src="img2.jpg" height="500px"
width="720px">     </div>
   </form>
</body>
</html>
```

Contact page:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="contactus.aspx.cs"
Inherits="WebApplication1.contactus" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>Contact us</title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <img src="img1.jpg" height="500px"
width="720px">     </div>
   </form>
</body>
</html>
```

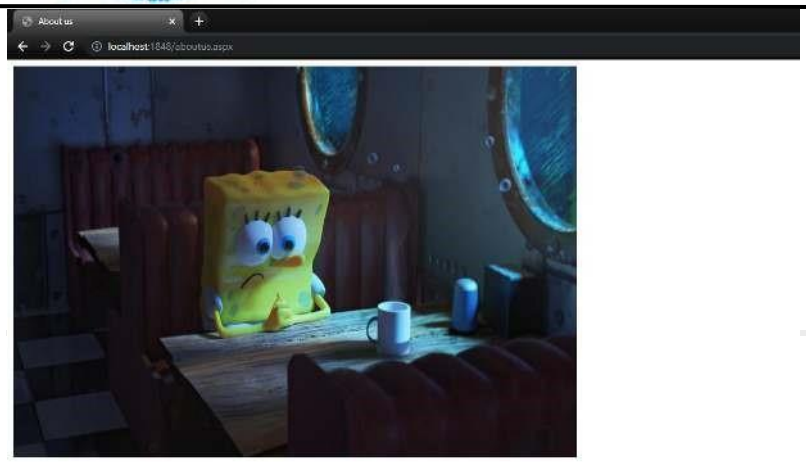
Output:

home page

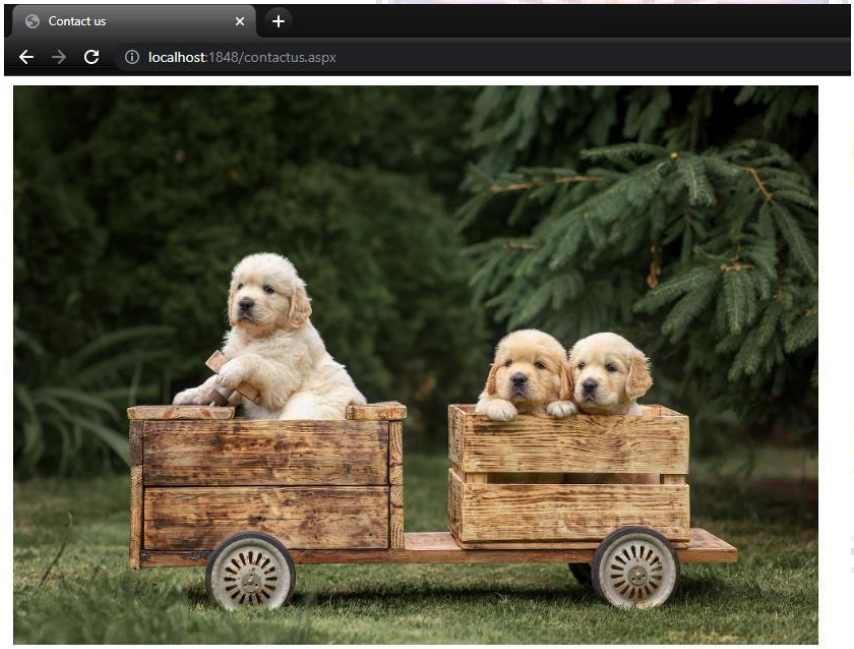
about page

contact page

After clicking about page



After clicking contact page

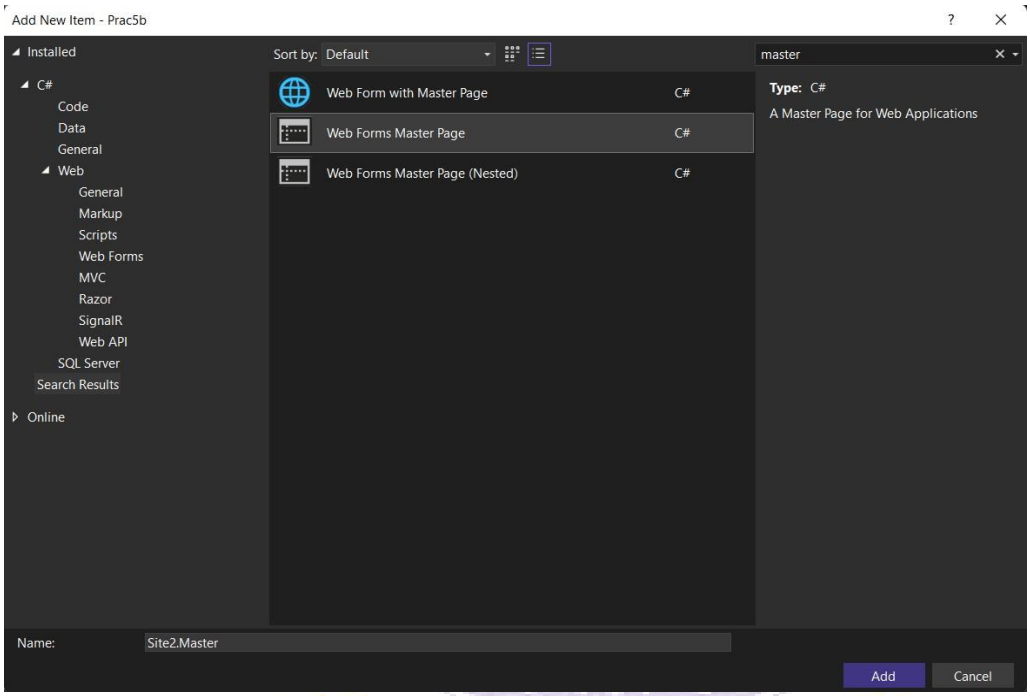


Practical 5

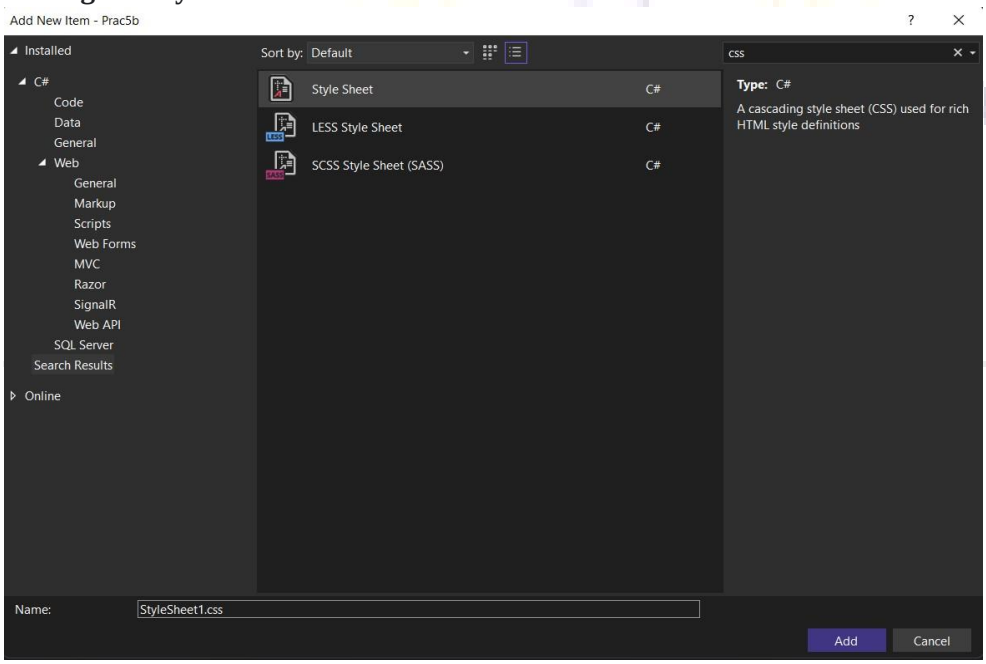
Part (b)

Aim. Create a web application to demonstrate use of Master Page with applying Styles and Themes for page beautification.

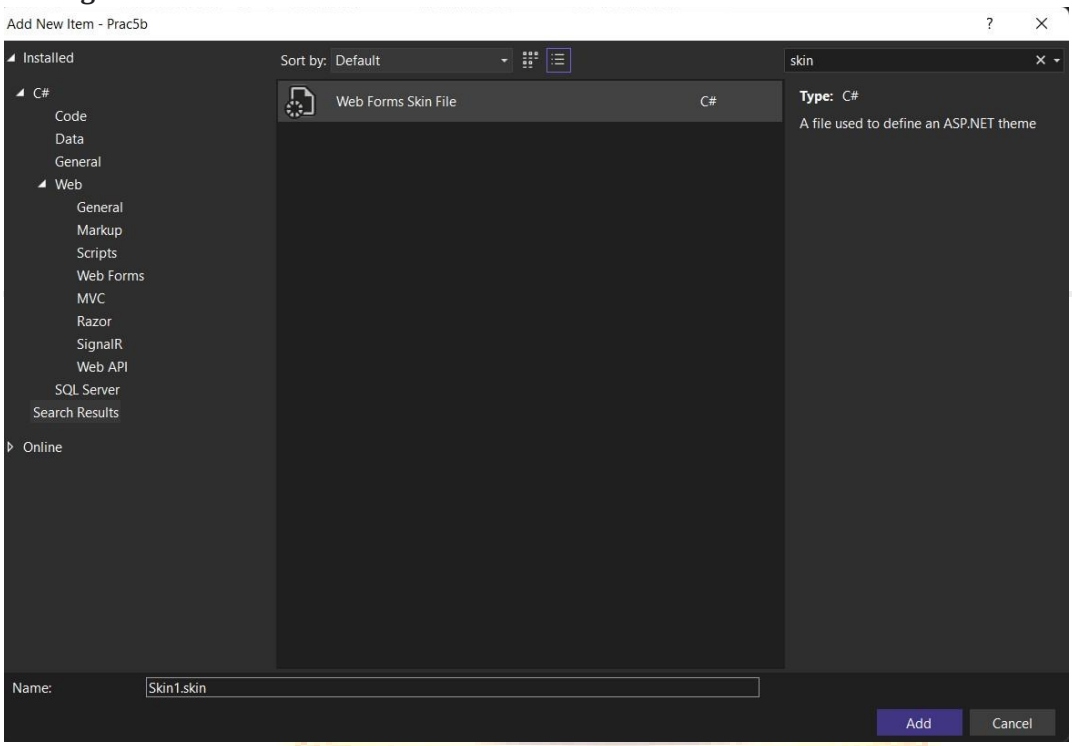
Adding Master page :



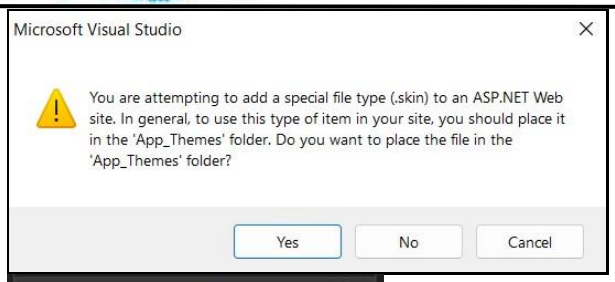
Adding Css style sheet :

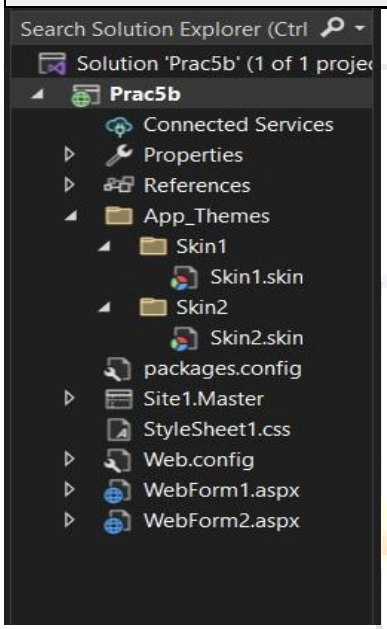


Adding skin :



This message appears because the skin file should be added in the themes folder , we click on yes .





WebForm1.aspx code:

WebForm1.aspx code:

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master"
AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="Pract5b.WebForm1"
Theme="Skin1"%>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
runat="server">
   <asp:Label ID="Label1" runat="server" Text="kishan"></asp:Label>
   <br />
   <asp:Label ID="Label2" runat="server" Text="amit"></asp:Label>
   <br />
   <asp:Label ID="Label3" runat="server"
Text="kartik"></asp:Label>     <br />
   <asp:Button ID="Button1" runat="server"
Text="Submit" />     <br />
</asp:Content>
```

WebForm2.aspx code:

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master"
AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="Pract5b.WebForm2"
Theme="Skin2" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
runat="server">
    <asp:Label ID="Label1" runat="server" Text="kishan"></asp:Label> <br />
    <asp:Label ID="Label2" runat="server" Text="amit"></asp:Label> <br />
    <asp:Label ID="Label3" runat="server" Text="kartik"></asp:Label> <br />
    <asp:Button ID="Button1" runat="server" Text="Button" />
</asp:Content>
```

Site1.Master code:

```
<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Site1.master.cs"
Inherits="Pract5b.Site1" %>

<!DOCTYPE html>

<html>
<head runat="server">
    <title>Pract5B</title>
    <link href="StyleSheet1.css" rel="stylesheet"/>
    <asp:ContentPlaceHolder ID="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <h1>This is Master Webpage</h1>
            <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
</asp:ContentPlaceHolder>
        </div>
    </form>
</body>
</html>
```

Skin1.skin code:

```
<%--
Default skin template. The following skins are provided as examples
only. 1. Named control skin. The SkinId should be uniquely defined
because    duplicate SkinId's per control type are not allowed in the
same theme.
```

```
<asp:GridView runat="server" SkinId="gridviewSkin" BackColor="White" >
  <AlternatingRowStyle BackColor="Blue" />
</asp:GridView>

2. Default skin. The SkinId is not defined. Only one default
control skin per control type is allowed in the same theme.
<asp:Image runat="server"
ImageUrl="~/images/image1.jpg" />
--%>
<asp:Label runat="server" ForeColor="Red" Font-Name="Verdana"/>
<asp:Button runat="server" Borderstyle="Solid" Borderwidth="5px"/>
```

Skin2.skin code:

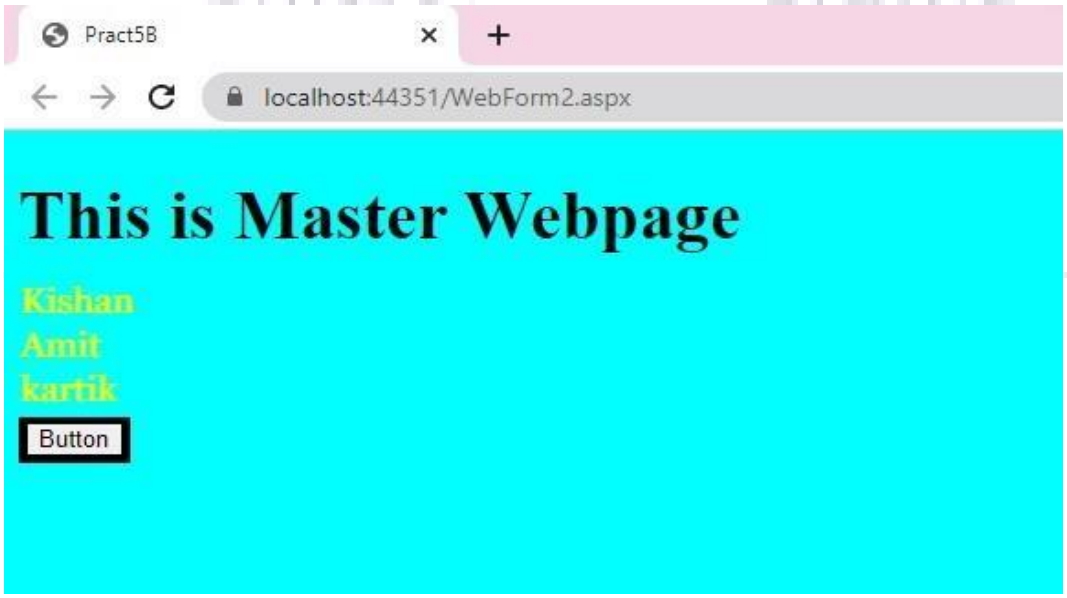
```
<asp:Label runat="server" ForeColor="Yellow" Font-Name ="Pacifico" />
<asp:Button runat="server" Borderstyle="Solid" Borderwidth="5px"/>
```

StyleSheet1.css code:

```
body {        background-
color:  cyan;          font-
family:    'Times    New
Roman';
   font-size: larger;
}
```

Output:

Webform1 :



Webform2:

Output:

Practical 5

Part (C)

Aim. Create a web application to demonstrate various states of ASP.NET Pages.

1.view state

Aspx code

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="PRACT_5c_A.aspx.cs"
Inherits="WebApplication1.PRACT_5c_A" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>     view state<asp:Button ID="Button1"
runat="server" Text="Button"
OnClick="Button1_Click" />
      <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
</div>
   </form>
</body>
</html>
```

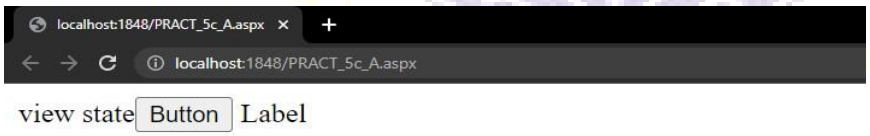
Aspx.c

s code

```
using

System

;

using
System.Collections.Ge
neric; using
System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication1
{    public partial class PRACT_5c_A :
System.Web.UI.Page
```

```
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                String str =
"sandeep";                  if
(ViewState["new"] == null)
{
                    ViewState["new"] = str;
                }
            }

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            Label1.Text = ViewState["new"].ToString();

        }

}
}
```

Output.



After button pressed

2. cookies

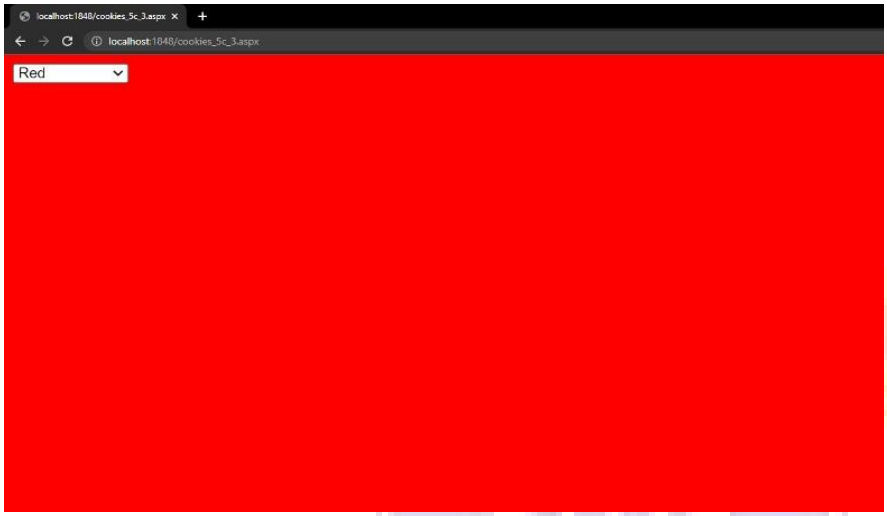
Aspx code:

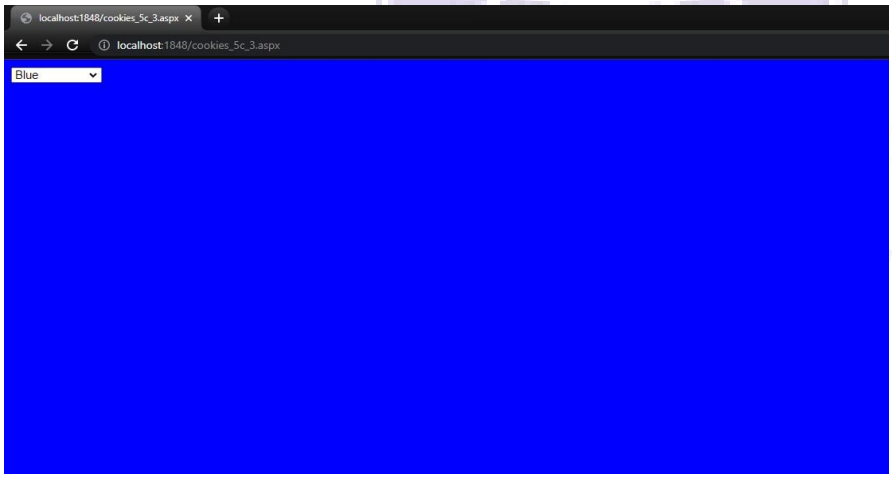
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="cookies_5c_3.aspx.cs"
Inherits="WebApplication1.cookies_5c_3" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body id="BodyTag" runat="server">
   <form id="form1" runat="server">
   <div>
      <asp:DropDownList ID="DropDownList1" runat="server" AutoPostBack="true"
OnSelectedIndexChanged="DropDownList1_SelectedIndexChanged">
         <asp:ListItem Selected="True" Value="White">Select Colour</asp:ListItem>
         <asp:ListItem>Red</asp:ListItem>
         <asp:ListItem>Green</asp:ListItem>
         <asp:ListItem>Blue</asp:ListItem>
         <asp:ListItem>Orange</asp:ListItem>

      </asp:DropDownList>
   </div>
   </form>
</body>
</html>
```

.aspx.cs page

```
using System;
using
System.Collections.Ge
neric; using
System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication1
{    public partial class cookies_5c_3 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if(Request.Cookies["BackgroundColor"]!=null)
         {
            DropDownList1.SelectedValue =
Request.Cookies["BackgroundColor"].Value;              BodyTag.Style["background-
color"] = DropDownList1.SelectedValue;

         }

      }

      protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
      {
         BodyTag.Style["background-color"] =
DropDownList1.SelectedValue;          HttpCookie cookie = new
HttpCookie("BackgroundColor");          cookie.Value =
DropDownList1.SelectedValue;          cookie.Expires =
DateTime.Now.AddMilliseconds(30);
         Response.SetCookie(cookie);
      }

}
}
```

Output:





3. query string

q1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="q1.aspx.cs"
Inherits="WebApplication15._Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:Label ID="Label1" runat="server" Text="UserId"></asp:Label>
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
      <br />
      <asp:Label ID="Label2" runat="server" Text="Name"></asp:Label>
<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
      <br />
      <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="SUBMIT" />
   </div>
   </form>
</body>
</html>
```

q1.aspx.cs

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

namespace WebApplication15
{    public partial class _Default :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
      }
      protected void Button1_Click(object sender, EventArgs e)
      {
         Response.Redirect("q2.aspx?UserId= " + TextBox1.Text + "&Name=" +
TextBox2.Text);
      }
   }
}
```

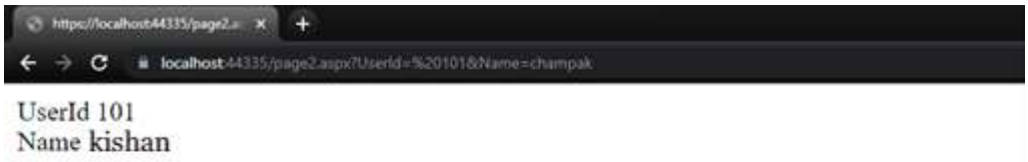
q2.aspx:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="q2.aspx.cs"
Inherits="WebApplication15.WebForm1" %>

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

      UserId<asp:Label ID="Label1" runat="server" Text="UserId"></asp:Label>
      <br />
      Name<asp:Label ID="Label2" runat="server" Text="Name"></asp:Label>
</div>
   </form>
</body>
</html>
```

q2.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Web; using
System.Web.UI; using
System.Web.UI.WebCont
rols;
namespace WebApplication15
{    public partial class WebForm1 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         Label1.Text = " " + Request.QueryString["UserId"];
         Label2.Text = " " + Request.QueryString["Name"];
      }
   }
}
```

Output:



Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is Timer?

2-What is Tex Reader?

3-What is XML?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 6

Working of Database with several web controls.

he integration of a database with various web controls is a fundamental aspect of web development, enabling dynamic and data-driven web applications. Web controls, such as textboxes, dropdown lists, and grids, serve as interfaces through which users interact with the application, and their functionality often relies on the seamless interaction with a database.

Database Interaction: Web controls facilitate the collection and display of data, and their functionality is closely tied to database operations. For instance, a textbox might capture user input, which needs to be stored or retrieved from a database. Dropdown lists may be populated with data queried from a database, providing users with selectable options dynamically. Grid controls can display tabular data retrieved from a database, allowing users to view, edit, or delete records.

Data Binding: Data binding is a key mechanism that connects web controls to a database. It establishes a link between the data in the database and the properties of the web controls. When properly configured, changes in the database reflect in the associated web controls, and vice versa. This synchronization enables real-time updates, ensuring that users interact with the most current data.

CRUD Operations: Web controls are often used to perform CRUD operations (Create, Read, Update, Delete) on a database. For example, a form with textboxes and buttons might be designed to insert new records into a database, update existing records, or delete entries. These controls trigger corresponding database queries or stored procedures to execute the necessary operations.

Practical 6

Part (A)

Aim. Create a web application to display records by using database.

.aspx code:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="practical6_b4._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:TextBox ID="TextBox1" runat="server" TextMode="MultiLine"
Width="205px"></asp:TextBox><br /> 
      <asp:Button ID="Button1" runat="server" Height="47px" Text="Button"
OnClick="Button1_Click" /> 
   </div>
      <asp:ListBox ID="ListBox1" runat="server" Height="130px" style="margin-
right:24px; margin-top:32px" Width="177px"></asp:ListBox>

   </form>
   <p>

 </
p>    <p>

 </
p>    <p>

 </
p>    <p>
       </p>
</body>
</html>
```

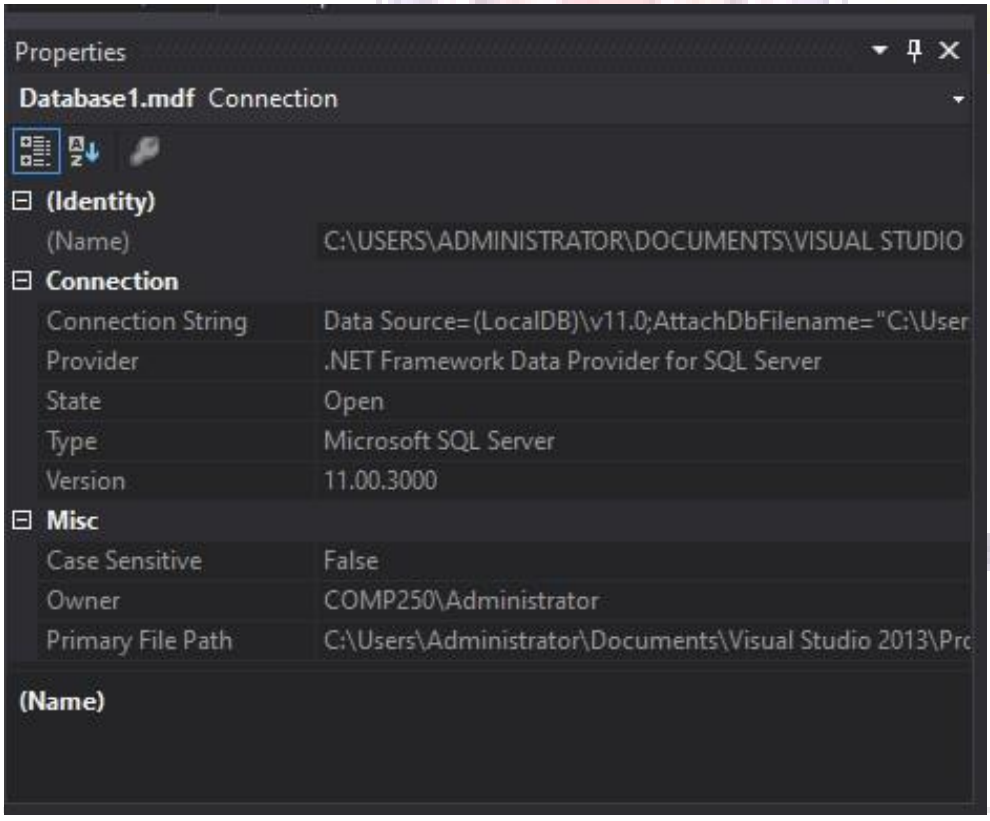
Buttonclick code:

```
using System;
using
System.Collections.Gener
ic; using System.Linq;
using System.Web; using
System.Web.UI; using
System.Web.UI.WebCont
rols; using
System.Configuration;
using
System.Data.SqlClient;
namespace practical6_b4
{    public partial class _Default :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         string d1 = ConfigurationManager.ConnectionStrings["d1"].ConnectionString;

         SqlConnection con = new SqlConnection(d1);
con.Open();
         SqlCommand cmd = new SqlCommand(TextBox1.Text, con);
         SqlDataReader reader = cmd.ExecuteReader();
         ListBox1.Items.Clear();
         while (reader.Read())
         {
            for (int i = 0; i < reader.FieldCount; i++)
            {
               ListBox1.Items.Add(reader[i].ToString());
            }
         }
         reader.Close();
con.Close();
      }
   }
}
```

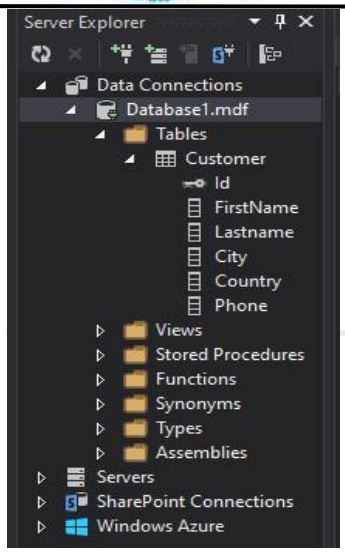
Web.config:

```
<connectionStrings>
                <add name="d1" connectionString="Data
Source=(LocalDB)\v11.0;AttachDbFilename='C:\Users\Administrator\Documents\Visua
l
Studio 2013\Projects\practical6_b4\practical6_b4\App_Data\Database1.mdf';Integrated
Security=True"/>
        </connectionStrings>
```

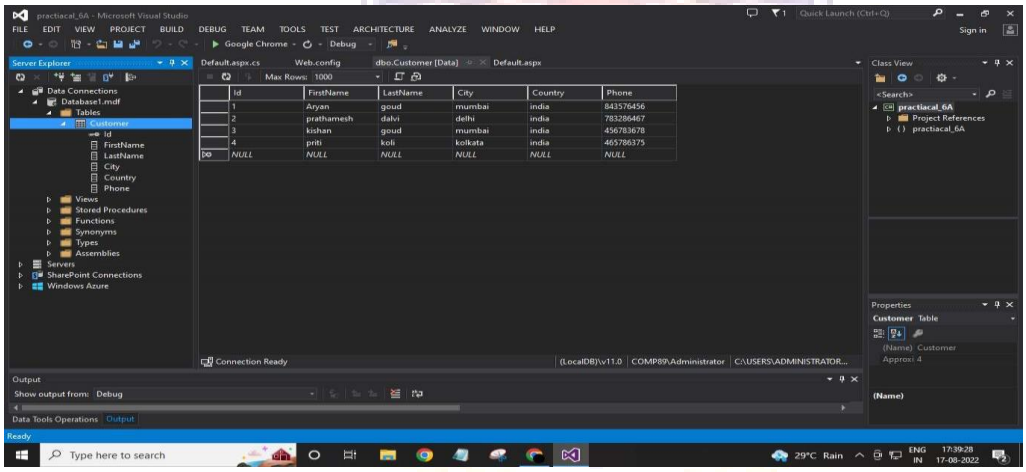
For pasting the database connection string.



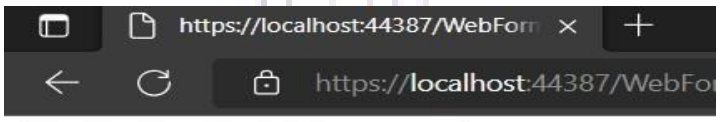
Database connection done and table created:



Data Entered



Output:




```
select * from customer;
```

Button

```
kishan
goud
bihar
india
89282
103
kartik
sharma
```

Practical 6

Part (B)

Aim. Create a web application to display records by using a database.

Aspx code:

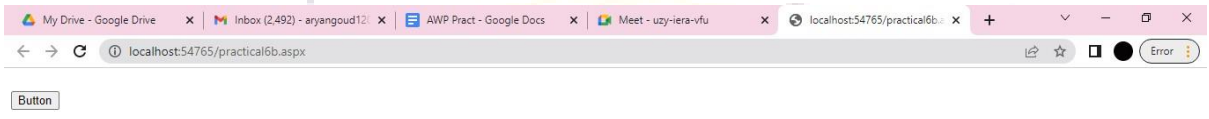
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="practical6_b4.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="Label1" runat="server"></asp:Label>
        <br />
        <br /> 
        <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />     </div>
    </form>
</body>
</html>
```

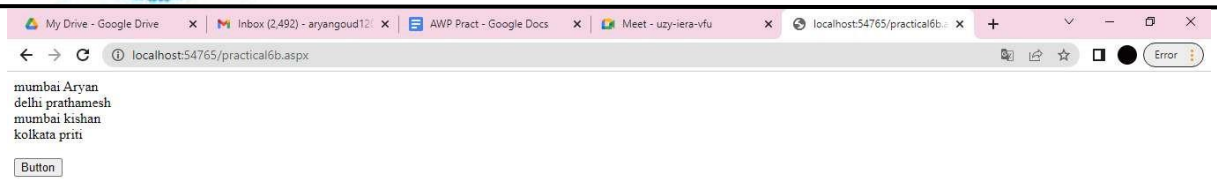
Button click code :

```
using System;
using
System.Collections.Ge
neric; using
System.Configuration;
using
System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace practical6_b4
```

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

      }

      protected void Button1_Click(object sender, EventArgs e)
      {
         string d1 = ConfigurationManager.ConnectionStrings["d1"].ConnectionString;

         SqlConnection con = new SqlConnection(d1);
con.Open();
         SqlCommand cmd = new SqlCommand("Select City,FirstName from Customer",
con);
         SqlDataReader reader = cmd.ExecuteReader();
         while (reader.Read())
         {
            Label1.Text += reader["City"].ToString() + "" +
reader["FirstName"].ToString() +
               "<br>";

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

Output:




My Drive - Google Drive
Inbox (2,492) - aryangoud12
AWP Pract - Google Docs
Meet - uzy-iera-vfu
localhost:54765/practical6b
localhost:54765/practical6b.aspx
Error
mumbai Aryan
delhi prathamesh
mumbai kishan
kolkata priti
Button


Type here to search
29°C Rain
ENG
US
17:58:13
17-08-2022

Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is Data Binding?

2-What is database?

3-What is Controls ?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 7

Introduction of Database with data binding with several web controls

Databases, coupled with data binding to web controls, form the backbone of dynamic and data-driven web applications, ushering in a new era of interactivity and responsiveness. Databases serve as repositories for storing and managing vast amounts of structured information. In the context of web development, they play a pivotal role in seamlessly connecting the frontend and backend, enabling the retrieval and manipulation of data.

Data binding, a powerful concept in web development, facilitates the automatic synchronization of user interface elements, or web controls, with data retrieved from a database. This synchronization allows for real-time updates, ensuring that changes in the underlying data are reflected instantaneously in the user interface. Web controls encompass a diverse range of elements, including textboxes, dropdown lists, grids, and more, each serving a unique purpose in displaying and interacting with data.

The integration of databases with data binding empowers developers to create dynamic, responsive, and user-friendly web applications. By establishing connections between web controls and the database, developers can effortlessly showcase information in a structured manner, implement sorting and filtering functionalities, and enable users to interact with and manipulate data seamlessly. This seamless interaction enhances the user experience, making web applications more intuitive and engaging.

Moreover, data binding reduces the need for manual code updates, as changes in the database automatically propagate to the associated web controls. This not only streamlines the development process but also ensures consistency and accuracy in presenting up-to-date information to users.

In summary, the synergy between databases and data binding with web controls forms the cornerstone of modern web application development. It empowers developers to create dynamic, data-driven interfaces that provide users with a rich and interactive experience, ultimately contributing to the success and effectiveness of web applications in diverse domains.

Top of Form

In addition to enabling you to bind the control to a set of data results, data-bound controls enable you to customize the layout of the control using templates. They also provide a convenient model for handling and canceling events.

This topic discusses how data Web server controls bind to data as well as the data-bound controls that are included with ASP.NET.

Practical 7

Part A

Aim. Create a web application to display Databinding using dropdownlist control.

Aspx code:

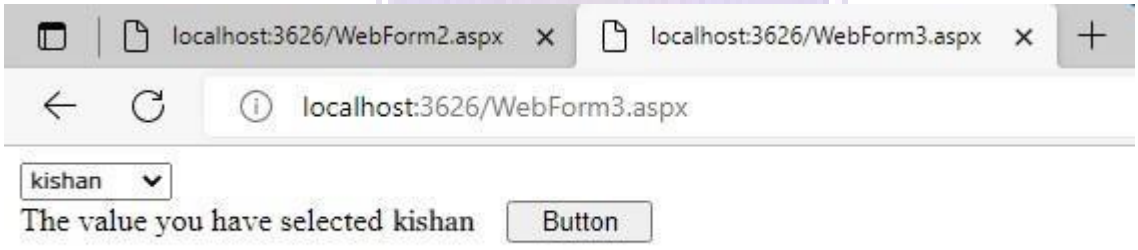
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="Practical6b.WebForm2" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
 <title></title>
</head>
   <body>
   <form id="form1" runat="server">
      <div>
         <asp:DropDownList ID="DropDownList1"
runat="server"></asp:DropDownList><br />
         <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label> 
         <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />
      </div>
   </form>
   </body>
</html>
```

Aspx.cs code:

```
using System;
using
System.Collections.Ge
neric; using
System.Configuration;
using
System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical6b
{    public partial class WebForm2 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
```

```
        if (IsPostBack == false)
        {
            string d1 = ConfigurationManager.ConnectionStrings["d1"].ConnectionString;
            SqlConnection con = new SqlConnection(d1); con.Open();
            SqlCommand cmd = new SqlCommand("Select City,FName from Customer",
con);
            SqlDataReader reader = cmd.ExecuteReader();
            DropDownList1.DataSource = reader;
            DropDownList1.DataTextField = "City";
            DropDownList1.DataBind();
            reader.Close();
con.Close();
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Label1.Text = "The value you have selected " + DropDownList1.SelectedValue;
    }
  }
}
```

Ouput:



Practical 7

Part (B)

Aim. Create a web application for to display the phone no of an author using database.

Web.config

```
<connectionStrings>
        <add name="d1" connectionString="Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename='C:\Users\Aryan
Sharma\Documents\Collage\AWP
pract\Practical7\Practical7\App_Data\Database1.mdf';Integrated
Security=True"/>
</connectionStrings>
```

Webform.aspx code:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="Practical7.WebForm1" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
 <title></title>
</head>
<body>
 <form id="form1" runat="server">
        <div>
          <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click"/><br
/> <br />
 <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label><br /><br />
 <asp:DropDownList ID="DropDownList1" runat="server" Height="164px"
Width="134px"> </asp:DropDownList>
        </div>
 </form>
</body>
</html>
```

Aspx.cs

```
using System;
using
System.Collections.Gener
ic; using
System.Configuration;
using
System.Data.SqlClient;
using System.Linq; using
System.Web; using
System.Web.UI; using
```

```
System.Web.UI.WebCont
rols;
namespace Practical7
{    public partial class WebForm1 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if (IsPostBack == false)
         {
string d1
=
            ConfigurationManager.ConnectionStrings["d1"].ConnectionString;
            SqlConnection con = new SqlConnection(d1);
            SqlCommand cmd = new SqlCommand("select distinct FName, LName, Phone
from P7",
           con);
con.Open();
            SqlDataReader reader = cmd.ExecuteReader();
            DropDownList1.DataSource = reader;
            DropDownList1.DataTextField = "FName";
            DropDownList1.DataValueField = "Phone";
            DropDownList1.DataBind();
            reader.Close();
con.Close();
         }
      }
      protected void Button1_Click(object sender, EventArgs e)
      {
         Label1.Text = "your number is:" + DropDownList1.SelectedValue;
      }

}
}
```

Output:

Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is Grid?

2-What is class Query?

3-What is Data Source?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 8

Introduction to data controls

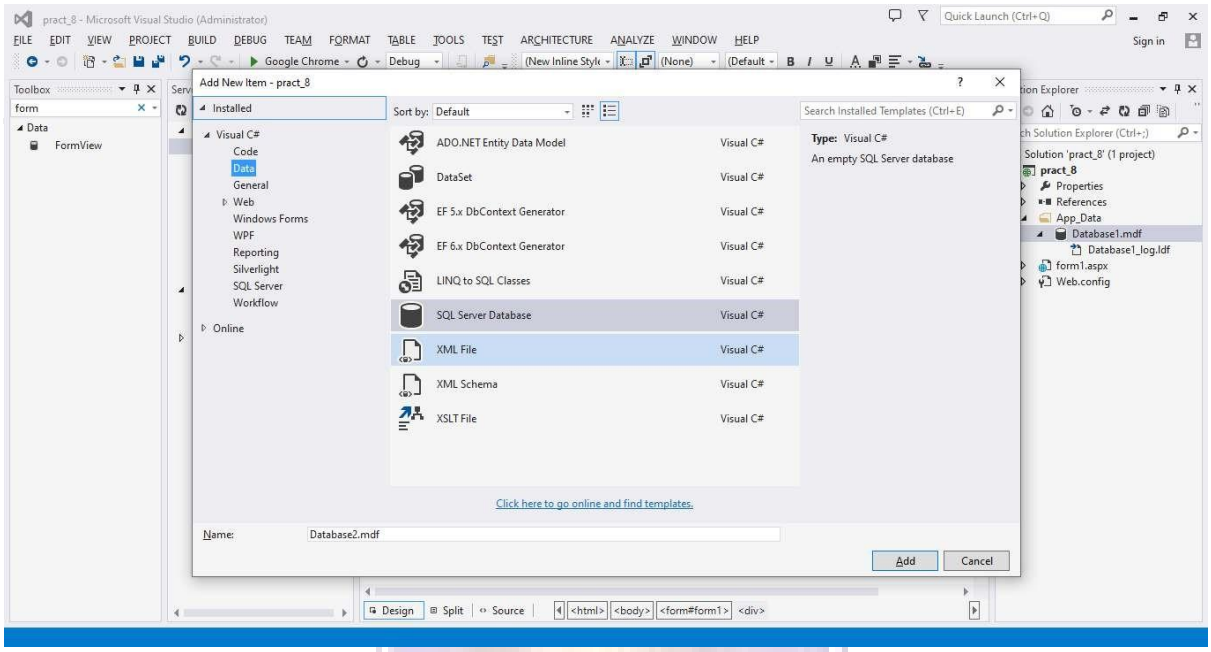
Data controls theory encompasses a set of principles and practices that guide the effective management and presentation of data within the context of software development. In the dynamic landscape of information technology, where data is a critical asset, the theory focuses on establishing robust controls to ensure the security, integrity, and accessibility of data. These controls extend across various layers of a system, from the storage and retrieval mechanisms to the user interfaces that interact with the data.

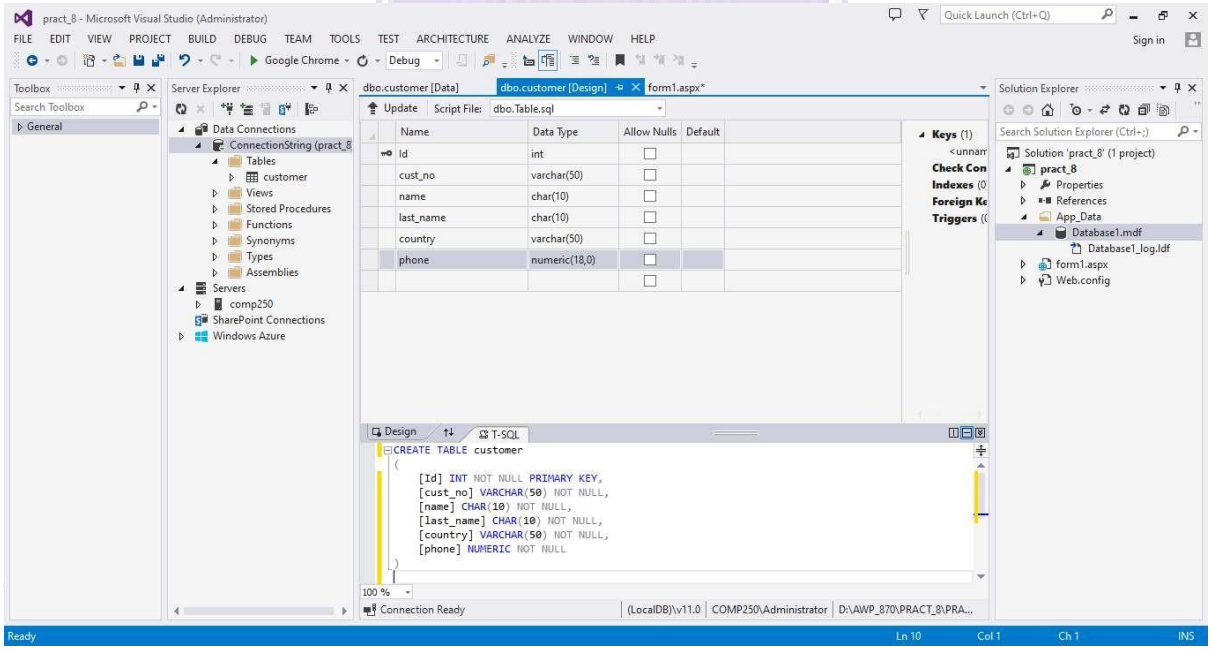
At its core, data controls theory addresses the challenges associated with handling data throughout its lifecycle. This involves implementing measures for data validation, encryption, access control, and auditing to safeguard against unauthorized access and ensure compliance with regulatory standards. The theory also emphasizes the importance of data presentation and user interaction, guiding the implementation of controls within user interfaces to enhance the user experience while maintaining data accuracy.

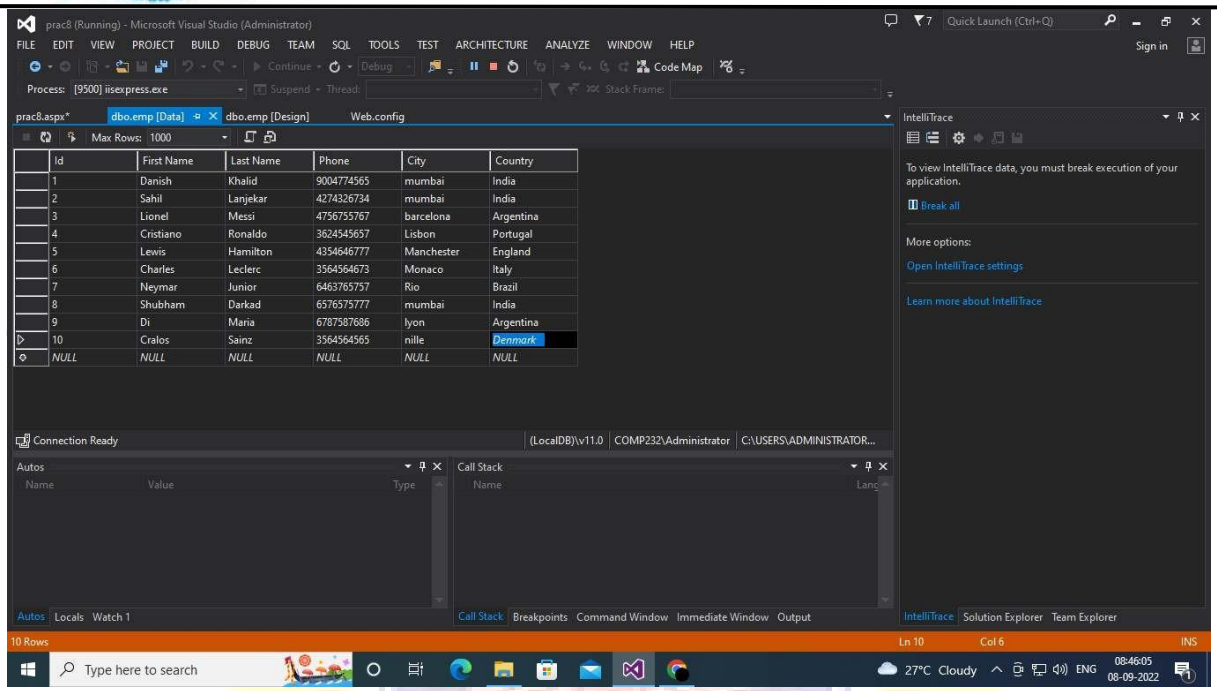
In the evolving landscape of technology, where data volumes are expanding exponentially, data controls theory becomes instrumental in mitigating risks associated with data breaches, ensuring data quality, and fostering trust in information systems. By adhering to these principles, developers and organizations can build robust and secure systems that effectively manage and harness the power of data in diverse applications, ranging from financial systems to healthcare platforms and beyond. In essence, data controls theory serves as a cornerstone in the foundation of reliable and secure data management within the realm of software development.

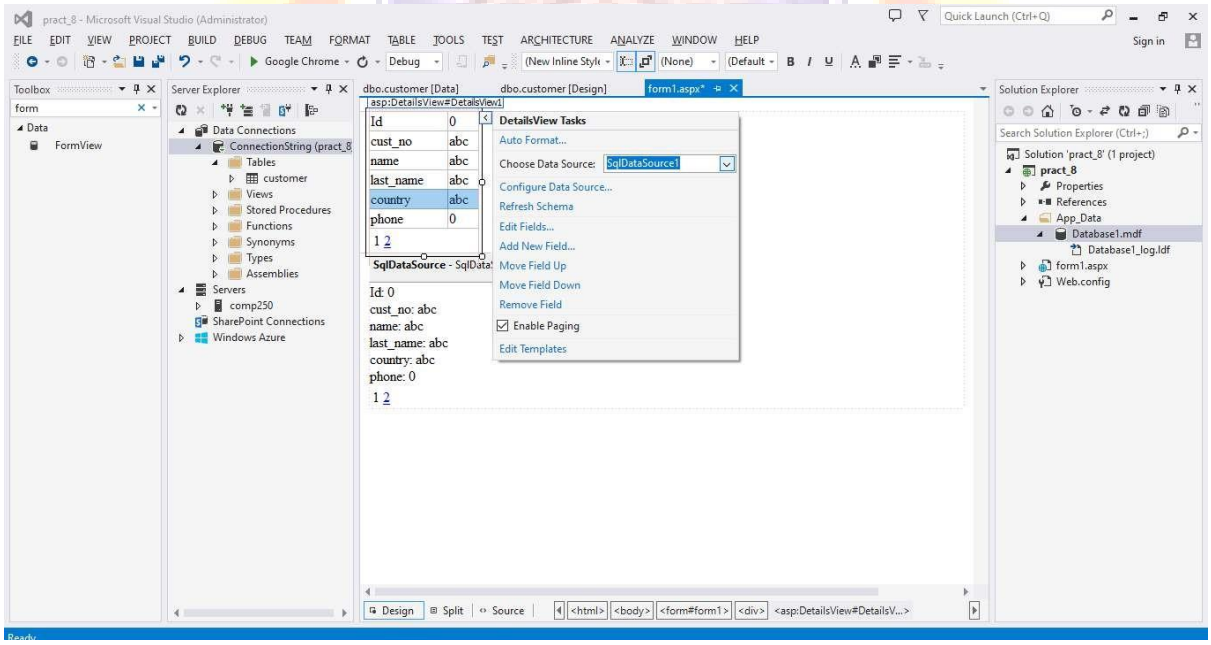
Practical 8

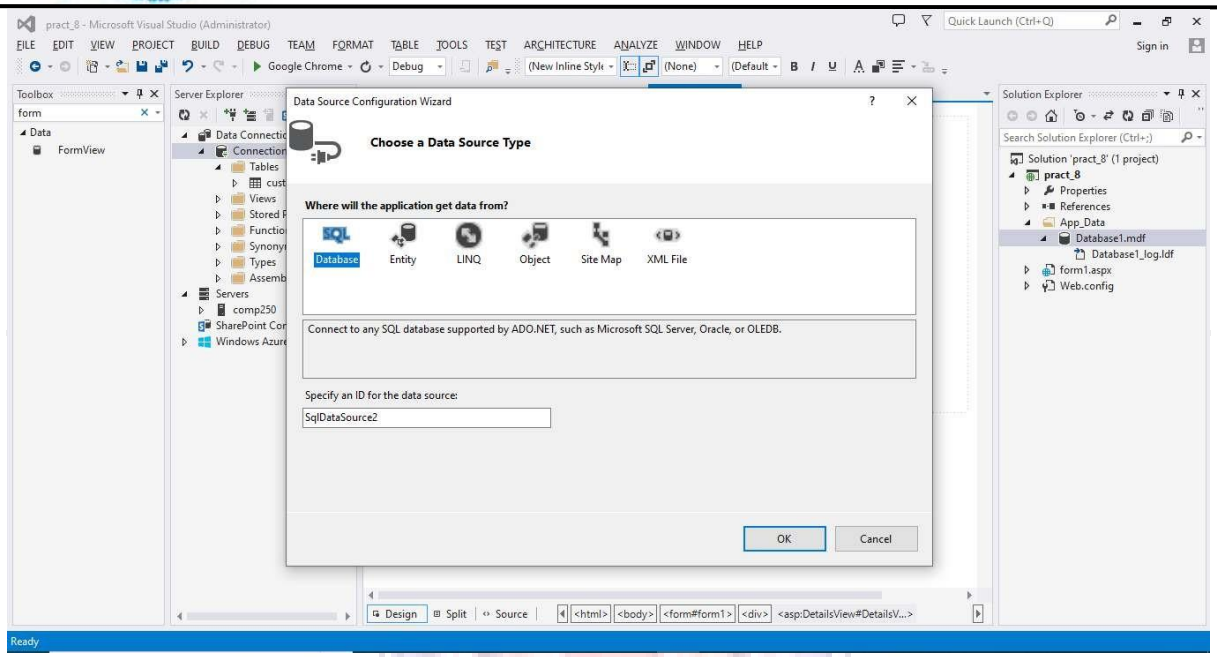
Aim. Create a web application to demonstrate data binding using DetailsView and FormView Control.

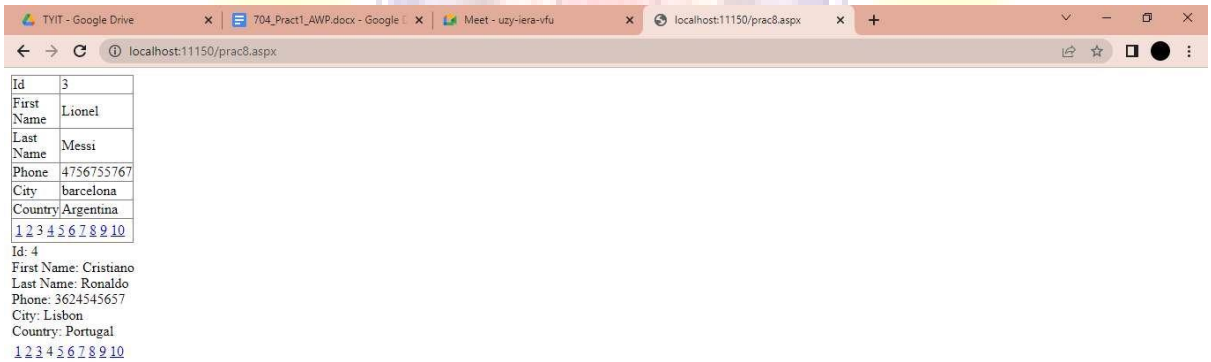


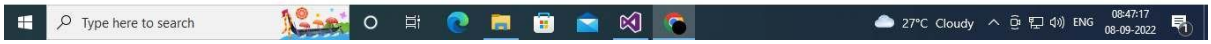



prac8 (Running) - Microsoft Visual Studio (Administrator)
prac8.aspx*
dbo.emp [Data]
dbo.emp [Design]
Web.config
Max Rows: 1000
Id First Name Last Name Phone City Country
1 Danish Khalid 9004774565 mumbai India
2 Sahil Lanjekar 4274326734 mumbai India
3 Lionel Messi 4756755767 barcelona Argentina
4 Cristiano Ronaldo 3624545657 Lisbon Portugal
5 Lewis Hamilton 4354646777 Manchester England
6 Charles Leclerc 3564564673 Monaco Italy
7 Neymar Junior 6463765757 Rio Brazil
8 Shubham Darkad 6576575777 mumbai India
9 Di Maria 6787587686 lyon Argentina
10 Cralos Sainz 3564564565 nille Denmark
NULL NULL NULL NULL NULL NULL
IntelliTrace
To view IntelliTrace data, you must break execution of your application.
Break all
More options:
Open IntelliTrace settings
Learn more about IntelliTrace
Connection Ready
Autos
Call Stack
Type here to search


pract_8 - Microsoft Visual Studio (Administrator)
Toolbox
Server Explorer
dbo.customer [Data]
dbo.customer [Design]
form1.aspx*
asp:DetailsView#DetailsView1
Id 0
cust_no abc
name abc
last_name abc
country abc
phone 0
1 2
DetailsView Tasks
Auto Format...
Choose Data Source: SqlDataSource1
Configure Data Source...
Refresh Schema
Edit Fields...
Add New Field...
Move Field Up
Move Field Down
Remove Field
Enable Paging
Edit Templates
SqlDataSource - SqlData
Id: 0
cust_no: abc
name: abc
last_name: abc
country: abc
phone: 0
1 2
Solution Explorer
Design
Split
Source


pract_8 - Microsoft Visual Studio (Administrator)
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
Solution Explorer
Ready


TYIT - Google Drive
704_Pract1_AWP.docx - Google
Meet - uzy-iera-vfu
localhost:11150/prac8.aspx
Id 3
First Name Lionel
Last Name Messi
Phone 4756755767
City barcelona
Country Argentina
1 2 3 4 5 6 7 8 9 10
Id: 4
First Name: Cristiano
Last Name: Ronaldo
Phone: 3624545657
City: Lisbon
Country: Portugal
1 2 3 4 5 6 7 8 9 10


Type here to search
27°C Cloudy
ENG
08:47:17
08-09-2022

Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is .NEt?

2-What is class C#?

3-What is Open Source?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 9

Introduction to Grid View control.

The GridView control is a crucial element in web development, designed to enhance the presentation and manipulation of tabular data within web applications. As part of the broader ASP.NET framework, the GridView facilitates the seamless integration of dynamic, database-driven content into a structured grid layout. This control empowers developers to create interactive and user-friendly interfaces by offering features like sorting, paging, and editing directly within the grid.

At its core, the GridView acts as a bridge between databases and web applications, allowing for the efficient retrieval and display of data in a visually organized format. By leveraging the GridView control, developers can design responsive tables that adapt to various screen sizes and devices, ensuring a consistent and optimal user experience.

One of the key strengths of the GridView lies in its configurability, enabling developers to customize the appearance and behavior of the grid to meet specific application requirements. Whether presenting product listings, user profiles, or any other tabular data, the GridView streamlines the process of data presentation and interaction, contributing to the overall effectiveness and user satisfaction of web applications.

Practical 9

Aim Create a webpage with GridView control.

Web.config file

```
<connectionStrings>
        <add name="d1" connectionString="Data Source=(LocalDB)\MSSQLLocalDB;
AttachDbFilename='C:\Users\Aryan Sharma\Documents\Collage\AWP
pract\Practical9\Practical9 \App_Data\Database1.mdf';Integrated Security=True"/>
</connectionStrings>
```

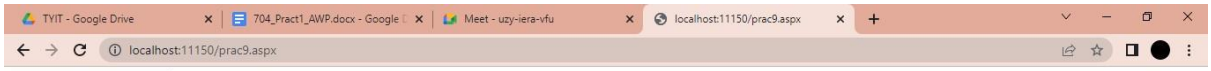
WebForm1.as

px.cs using

```
System;
using
System.Collections.Gener
ic; using System.Data;
using
System.Data.SqlClient;
using System.Linq; using
System.Web; using
System.Web.Configuratio
n; using System.Web.UI;
using
System.Web.UI.WebCont
rols;
namespace Practical9
{    public partial class WebForm1 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         string connectionString =

WebConfigurationManager.ConnectionStrings["d1"].ConnectionString;
string selectSQL = "SELECT Id, FName, Phone FROM P9";
SqlConnection con = new SqlConnection(connectionString);
         SqlCommand cmd = new SqlCommand(selectSQL, con);
         SqlDataAdapter adapter = new SqlDataAdapter(cmd);
         DataSet ds = new DataSet();
adapter.Fill(ds, "P9");
GridView1.DataSource = ds;
         GridView2.DataBind();
```

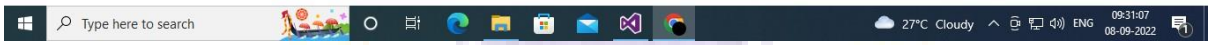
```
        }

}
}
```

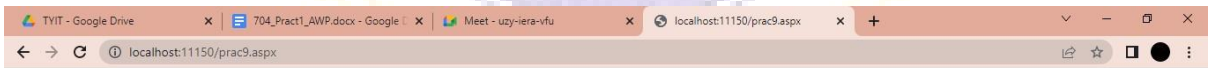
Output:



Id	FirstName	Phone
1	Sahil	9004774565
2	Danish	4274326734
3	Manas	4756755767
4	Abhishek	3624545657
5	Mangesh	4354646777
6	Kunal	3564564673
7	Subrato	6463765757
8	Shubham	6576575777
9	Kishan	6787587686
10	Rohan	3564564565

Id	FirstName	LastName	Phone	City	Country
4	Abhishek	Ajit	3624545657	Lisbon	Portugal
10	Rohan	Bacchav	3564564565	nille	Denmark
7	Subrato	Bhore	6463765757	Rio	Brazil
8	Shubham	Darkad	6576575777	mumbai	India
6	Kunal	Gohil	3564564673	Monaco	Italy
5	Mangesh	Gosavi	4354646777	Manchester	England
9	Kishan	Goud	6787587686	lyon	Argentina
3	Manas	Kawlekar	4756755767	barcelona	Argentina
2	Danish	Khalid	4274326734	mumbai	India
1	Sahil	Lanjekar	9004774565	mumbai	India





Id	FirstName	Phone
1	Sahil	9004774565
2	Danish	4274326734
3	Manas	4756755767
4	Abhishek	3624545657
5	Mangesh	4354646777
6	Kunal	3564564673
7	Subrato	6463765757
8	Shubham	6576575777
9	Kishan	6787587686
10	Rohan	3564564565

Id	FirstName	LastName	Phone	City	Country
3	Manas	Kawlekar	4756755767	barcelona	Argentina
4	Abhishek	Ajit	3624545657	Lisbon	Portugal
9	Kishan	Goud	6787587686	lyon	Argentina
5	Mangesh	Gosavi	4354646777	Manchester	England
6	Kunal	Gohil	3564564673	Monaco	Italy
1	Sahil	Lanjekar	9004774565	mumbai	India
2	Danish	Khalid	4274326734	mumbai	India
8	Shubham	Darkad	6576575777	mumbai	India
10	Rohan	Bacchav	3564564565	nille	Denmark
7	Subrato	Bhore	6463765757	Rio	Brazil

Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is CSS?

2-What is Master Page?

3-What is Theme?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]

Theory 10

Working with AJAX and XML

Working with AJAX (Asynchronous JavaScript and XML) involves utilizing a set of web development technologies to create dynamic and responsive user interfaces. AJAX enables asynchronous data exchange between the client (web browser) and the server, facilitating the updating of specific parts of a web page without requiring a full page reload. XML (eXtensible Markup Language) is often employed as the format for data interchange in AJAX applications, although JSON (JavaScript Object Notation) has become more prevalent in recent years.

The primary advantage of AJAX lies in its ability to enhance user experience by enabling real-time updates and interactions. Instead of reloading entire web pages, AJAX allows for partial updates, reducing latency and creating a smoother, more responsive interface. When working with AJAX and XML, developers typically employ JavaScript to make asynchronous requests to the server, retrieve XML data, and dynamically update the content on the client side.

XML serves as a structured and platform-independent format for data exchange in AJAX applications. It organizes data hierarchically with user-defined tags, making it easy to parse and manipulate. AJAX and XML together enable the seamless transmission of data between the client and server, supporting tasks such as form validation, auto-complete suggestions, and live data updates.

In contemporary web development, JSON has gained popularity over XML due to its lighter syntax and easier parsing with JavaScript. However, the principles of working with AJAX to achieve asynchronous communication for enhanced interactivity remain foundational, regardless of the data interchange format chosen. The combination of AJAX and XML (or JSON) is a powerful paradigm that has significantly contributed to the evolution of modern, dynamic web applications.

Practical 10

Aim. Create a web application to demonstrate reading and writing operation with XML

Webform.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="pract_10.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <div>
      <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label><br>
      <asp:Button ID="Button1" runat="server" Text="Write XML"
OnClick="Button1_Click" /><br>
      <asp:ListBox ID="ListBox1" runat="server"></asp:ListBox><br>
      <asp:Button ID="Button2" runat="server" Text="Read XML"
OnClick="Button2_Click" /><br>
   </div>
   </form>
</body>
</html>
```

Aspx.cs code:

```
using System;

using System.Xml;

namespace pract_10
{    public partial class WebForm1 :
System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void Button2_Click(object sender, EventArgs e)
      {
         String xmlNode = "D:\\for_awp\\demo.xml";
```

```
        XmlReader xReader = XmlReader.Create(xmlNode);
        while (xReader.Read())
        {
          switch (xReader.NodeType)
          {
            case XmlNodeType.Element:
              ListBox1.Items.Add("<" + xReader.Name
+ ">");                 break;                 case
XmlNodeType.Text:

ListBox1.Items.Add(xReader.Value);
break;                 case
XmlNodeType.EndElement:
              ListBox1.Items.Add("</" + xReader.Name +
">");                 break;
          }
        }
      }

      protected void Button1_Click(object sender, EventArgs e)
      {
        XmlTextWriter writer = new XmlTextWriter("D:\\for_awp\\demo.xml",
null);          writer.WriteStartDocument();
writer.WriteStartElement("Deatils", "");          writer.WriteStartElement("ID",
"1");          writer.WriteStartElement("firstname", "narendra");
writer.WriteStartElement("lastname", "modi");
writer.WriteStartElement("salary", "24000");
writer.WriteEndElement();          writer.WriteEndDocument();
        writer.Close();
        Label1.Text = "Data written successfully";

      }
    }
}
```

After click on write xml

Data written successfully

Write XML

Read XML

After click on read xml

Data written successfully

Write XML

<Deatils>
<ID>
<firstname>
<lastname>

Read XML

Result and Discussion:

Learning Outcome:

Course Outcome:

Conclusion:

Viva Question:

1-What is ASP.NET?

2-What is Tich control?

3-What is Exception?

For Faculty Use

Correction Parameter	Formative Assessment [ ]	Timely Completion Of practical [ ]	Attendance learning Attitude [ ]