LIST OF EXPRIMENT

Ex.No:1 WORKING WITH CONSOLE APPLICATIONS

AIM

To understand about basics of C# and execute simple c# programs to perform the following actions:
(a). Create a simple Console Application Program to display a text message.
(b). Taking non-numerical data from keyboard into Console Application.
(c). Taking numerical data in Console Application.

ALGORITHM

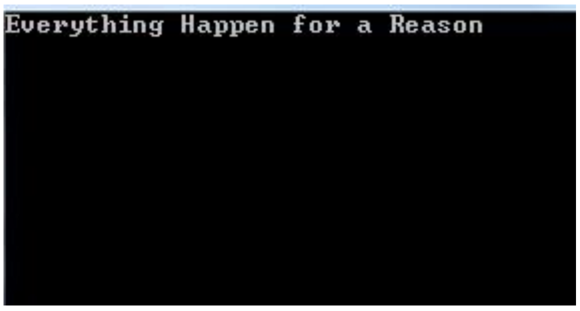
Step1: Open Visual Studio Express edition2010
Step2: Click File New project Select C# under installed tab and select console application
Step 3: Give name for your application and click OK
Step4: Give any class name and declare variables and write methods
Step 5: Create objects for classes to execute methods
Step6:Click save and click run button for execution

PROGRAM:

(a).Create simple Console Application Program to display a text message.

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

namespace lab1
{
   class Program
   {
      static void Main(string[] args)
      {
         Console.WriteLine("Everything Happens for a Reason");
         Console.ReadKey();
      }
   }
}
```

OUTPUT:



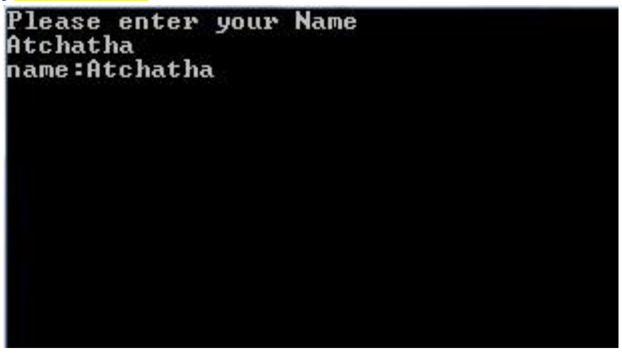
```
Everything Happen for a Reason
```

(b).Taking non numerical data from keyboard into Console Application.

PROGRAM:

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

namespace ConsoleApplication2
{
   class Program
   {
      static void Main(string[] args)
      {
         string name = "";

         Console.WriteLine("Please enter your name:");
         name = Console.ReadLine();

         Console.WriteLine("Name: " + name);
         Console.ReadKey();
      }
   }
}
```

OUTPUT:



```
Please enter your Name
Atchatha
name:Atchatha
```

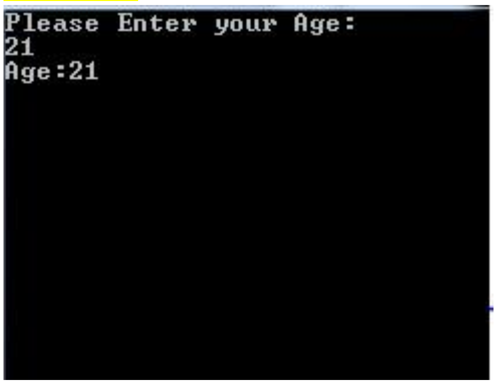
(c).Taking numerical data in Console Application

PROGRAM:

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

namespace ConsoleApplication3
{
   class Program
   {
      static void Main(string[] args)
      {
         int age = 0;
```

```
            Console.WriteLine ("Please enter your age:");
            age = Convert.ToInt16(Console.ReadLine());

            Console.WriteLine("Age: " + age);
            Console.ReadKey();
        }
    }
}
```

OUTPUT:



```
Please Enter your Age:
21
Age:21
```

RESULT:

Thus, to understand the basics of C# and execute simple C# programs has been verified.

Ex.No:2 Console Application using conditional and Looping statements

AIM:

To understand about basics of C# and execute simple c# programs to perform the following actions:
(a) Calculate the quadrant for the coordinates using if..else ladder.
(b) Check whether the alphabet is a vowel or not using switch..case.
(c) To understand about for..each loop and strings.

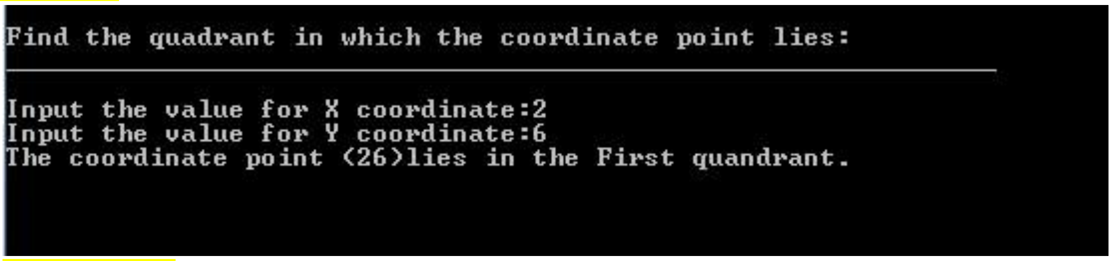
ALGORITHM:

Step 1: Open Visual Studio Express edition 2010
Step 2: Click File > New project. Select C# under installed tab and select console application
Step 3: Give name for your application and click OK
Step 4: Give any class name and declare variables and write methods
Step 5: Create objects for classes to execute methods
Step 6: Click save and click run button for execution

PROGRAM

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication4

{

    class Program

    {

        static void Main(string[] args)

        {

            int co1, co2;

            Console.Write("\n\n");

            Console.Write("Find the quadrant in which the coordinate point lies:\n");

            Console.Write("_\t");

            Console.Write("\n\n");


            Console.Write("Input the value for X coordinate: ");

            co1 = Convert.ToInt32(Console.ReadLine());

            Console.Write("Input the value for Y coordinate: ");

            co2 = Convert.ToInt32(Console.ReadLine());


            if (co1 > 0 && co2 > 0)
              Console.Write("The coordinate point ({0}, {1}) lies in the First quadrant.\n\n", co1, co2);
            else if (co1 < 0 && co2 > 0)
              Console.Write("The coordinate point ({0}, {1}) lies in the Second quadrant.\n\n", co1, co2);
            else if (co1 < 0 && co2 < 0)
              Console.Write("The coordinate point ({0}, {1}) lies in the Third quadrant.\n\n", co1, co2);
            else if (co1 > 0 && co2 < 0)
              Console.Write("The coordinate point ({0}, {1}) lies in the Fourth quadrant.\n\n", co1, co2);
            else if (co1 == 0 && co2 == 0)
              Console.Write("The coordinate point ({0}, {1}) lies at the origin.\n\n", co1, co2);
            Console.ReadKey();
        }
    }
}
```

OUTPUT:

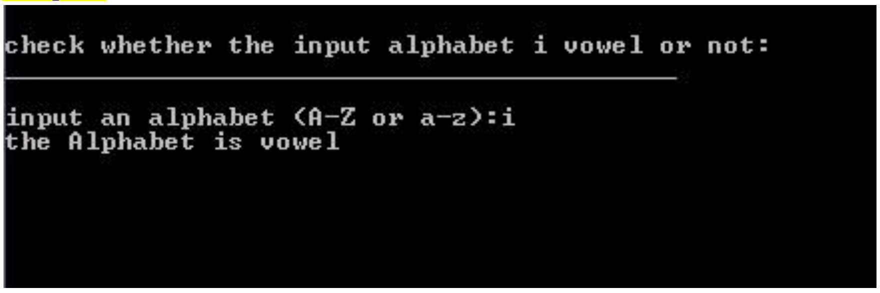


```
Find the quadrant in which the coordinate point lies:
----------------------------------------------------------------

Input the value for X coordinate:2
Input the value for Y coordinate:6
The coordinate point (26)lies in the First quandrant.
```

B)Program:

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

namespace ConsoleApplication5
{
    class Program
    {
        static void Main(string[] args)
        {
            char ch;

            Console.Write("\n\n");
            Console.Write("Check whether the input alphabet is a vowel or not:\n");
            Console.Write("_\t");
            Console.Write("\n\n");

            Console.Write("Input an alphabet (A-Z or a-z): ");
            ch = Convert.ToChar(Console.ReadLine().ToLower());
            int i = ch;

            if (i >= 48 && i <= 57)
            {
                Console.Write("You entered a number, please enter an alphabet.");
            }
            else
            {
                switch (ch)
                {
                    case 'a':
                        Console.WriteLine("The alphabet is a vowel.");
                        break;
                    case 'i':
                        Console.WriteLine("The alphabet is a vowel.");
                        break;
                    case 'o':
                        Console.WriteLine("The alphabet is a vowel.");
                        break;
                    case 'u':
                        Console.WriteLine("The alphabet is a vowel.");
                        break;
                    case 'e':
                        Console.WriteLine("The alphabet is a vowel.");
                        break;
                    default:
                        Console.WriteLine("The alphabet is a consonant.");
                        break;
                }
                Console.ReadKey();
            }
        }
    }
}
```

Output:

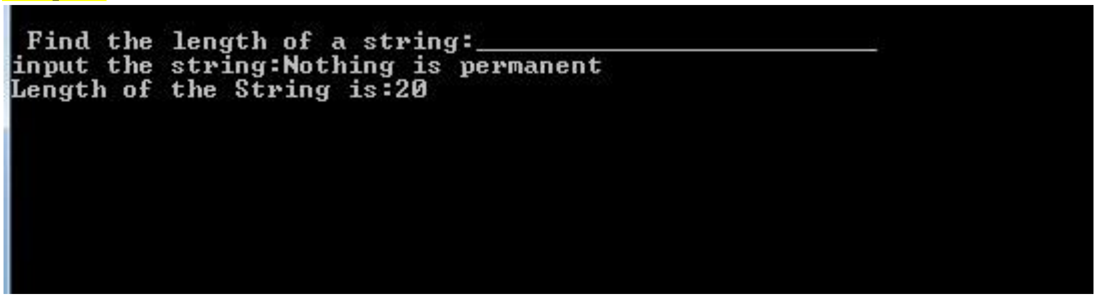


```
check whether the input alphabet i vowel or not:
------------------------------------------------

input an alphabet (A-Z or a-z):i
the Alphabet is vowel
```

2.C. String length Program:

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

namespace ConsoleApplication6
{
    class Program
    {
        static void Main(string[] args)
        {
            string str;
            int length = 0;

            Console.Write("\n\nFind the length of a string: ");
            Console.Write("_\t\n");
            Console.Write("Input the string: ");
            str = Console.ReadLine();

            foreach (char chr in str)
            {
                length += 1;
            }

            Console.Write("Length of the string is: {0}\n\n", length);
            Console.ReadKey();
        }
    }
}
```

Output:



```
 Find the length of a string:__________________________
input the string:Nothing is permanent
Length of the String is:20
```

Result:

Thus, to understand the basics of C# and execute simple C# programs has been verified

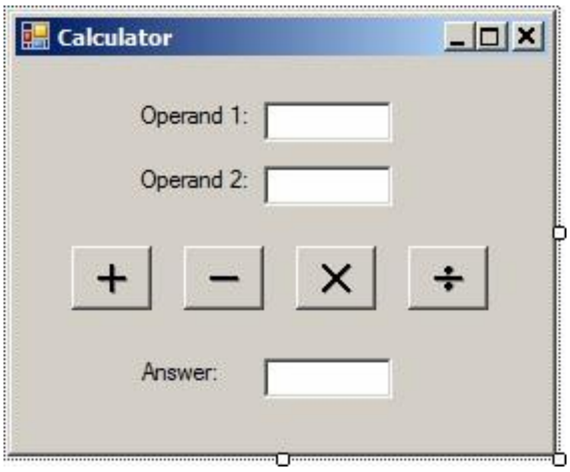
Ex.No:3 **Simple calculator program using C#.Net windows Application**

AIM

To build a simple calculator that performs addition, subtraction, multiplication, and division using C# .NET Windows Application.

ALGORITHM:

1. Create a new C# Windows Forms Application named *MyCalculator*. Name the form class and the associated file *Calculator*. Save the solution.
2. Design the form window controls (from Toolbox) for the four arithmetic operations.
3. Set the properties of each control.
4. Trap the *Click* event for each of the four buttons that specify math operations.
5. In each handler, write code to convert the string data in each textbox to a floating-point value. Perform the appropriate math operation for the button. Finally, place the result back in the textbox that holds the answer. Compile and run the program.



FORM DESIGN:

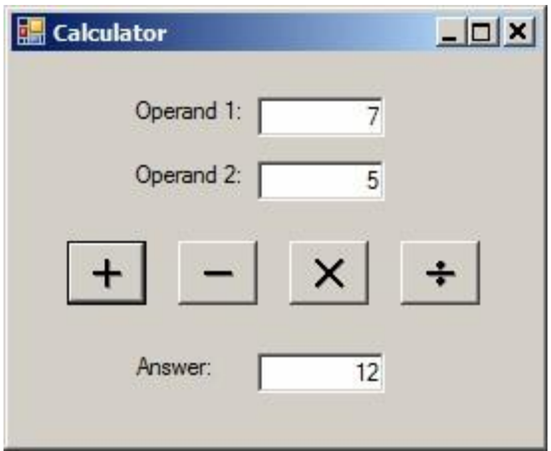
PROGRAM CODING:

a)Simple calculator program using#.net windows form application.

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

```
namespace WindowsFormsApplication1
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void Form1_Load(object sender, EventArgs e)
      { }

      private void button1_Click(object sender, EventArgs e)
      {
         var a = Convert.ToInt32(textBox1.Text);
         var b = Convert.ToInt32(textBox2.Text);
         var c = a + b;
         textBox3.Text = c.ToString();
      }

      private void button2_Click(object sender, EventArgs e)
      {
         var a = Convert.ToInt32(textBox1.Text);
         var b = Convert.ToInt32(textBox2.Text);
         var c = a - b;
         textBox3.Text = c.ToString();
      }
      private void button3_Click(object sender, EventArgs e)
      {
         var a = Convert.ToInt32(textBox1.Text);
         var b = Convert.ToInt32(textBox2.Text);
         var c = a * b;
         textBox3.Text = c.ToString();
      }
      private void button4_Click(object sender, EventArgs e)
      {
         var a = Convert.ToInt32(textBox1.Text);
         var b = Convert.ToInt32(textBox2.Text);
         var c = a / b;  // Changed from '%' to '/' for division
         textBox3.Text = c.ToString();
      }
   }
}
```

OUTPUT:



Operand 1: 7
Operand 2: 5
Answer: 12

B)USING CHECKBOX

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

namespace WindowsFormsApplication1
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void checkBox1_CheckedChanged(object sender, EventArgs e)
      {
         label1.Font = new Font(label1.Font, FontStyle.Bold);
      }

      private void checkBox2_CheckedChanged(object sender, EventArgs e)
      {
         label1.Font = new Font(label1.Font, FontStyle.Italic);
      }
   }
}
```

RESULT:

Thus, to build a C# .NET Windows application and access various controls has been verified

Ex.No:4 Working with various Controls such as timer, calendar, etc.,

AIM:

To create a DateTimePicker control to display the current date and time using C# .NET Windows Forms Application.

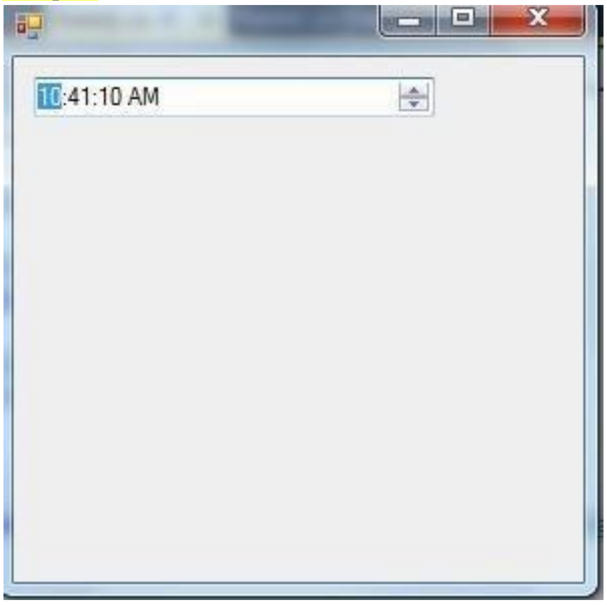
ALGORITHM:

1 Create a new project -> Windows Application -> Name -> OK
2 Design the form window controls (from Toolbox) and drag and drop the DateTimePicker control.
3 Set the properties of the control.
4 Write the code to display the system date and time in the Form Load event.
5 Finally, compile and run the program.

PROGRAM:

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

namespace WindowsFormsApplication1
{
  public partial class Form1 : Form
  {
    private DateTimePicker timePicker;

    public Form1()
    {
      InitializeComponent();
    }

    private void Form1_Load(object sender, EventArgs e)
    {
      timePicker = new DateTimePicker();
      timePicker.Format = DateTimePickerFormat.Time;
      timePicker.ShowUpDown = true;
      timePicker.Width = 100;
      Controls.Add(timePicker);
    }

    [STAThread]
    static void Main()
    {
      Application.EnableVisualStyles();
      Application.Run(new Form1());
    }
    }
}
```

Output



10:41:10 AM

Result:

Thus, to build a C# .NET Windows application and use DateTimePicker controls has been verified.

Ex.No:5 AccessingDatawith ADO.NET

AIM:

To create a C# .NET Console Application to connect to an MS Access database to display the table values using the OleDbConnection object.

ALGORITHM:

1 Create a new project -> Console Application -> Name -> OK

2 To select the *Tools* menu -> Connect to database

3 Select the database and select the dataset, click *Next*, click *New Connection*, click *Change* button, and select *Microsoft Access Data Source* -> OK button

4 Click the *Browse* button and select *Northwind* and click the *Open* button

5 Click *Test Connection* button and click *OK*, then select *Next* -> *Yes* button

6 Double-click *Tables* folder to view the list of tables available for the Northwind database

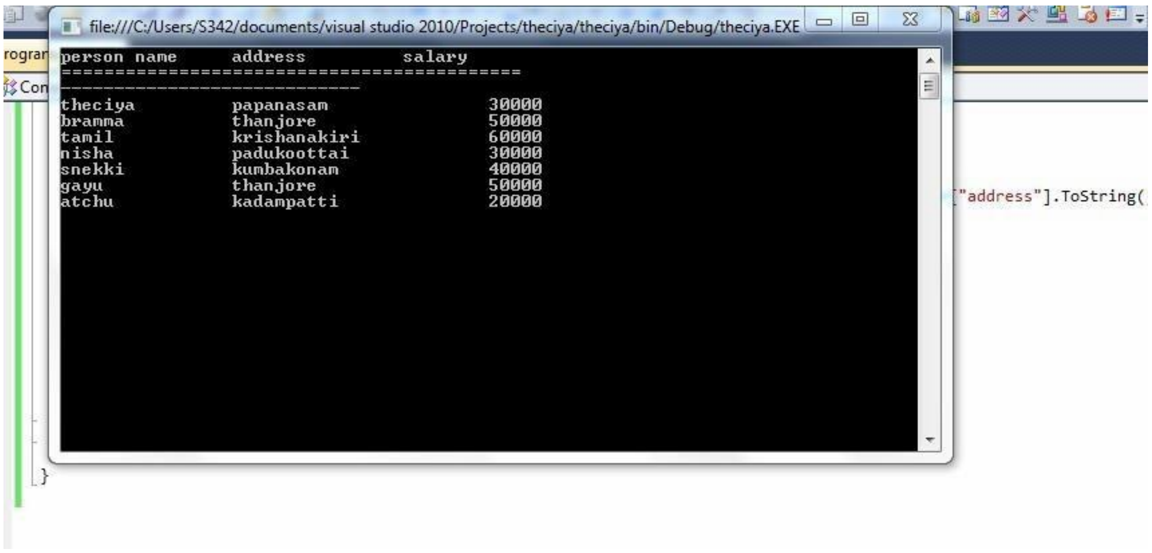
7 To display the *Employee* table in the Windows form

POGRAM CODING:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data.OleDb;

namespace ConsoleApplication19
{
   class Program
   {
      static void Main(string[] args)
      {
         string connectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:\\Users\\S342\\Documents\\theciyasiva.accdb";
         OleDbConnection conn = new OleDbConnection(connectionString);
         string sql = "SELECT name, address, salary FROM employee";
         OleDbCommand cmd = new OleDbCommand(sql, conn);

         Console.WriteLine("Person Name\tAddress\t\tSalary");
         Console.WriteLine("=========================================");

         try
         {
            conn.Open();
            using (OleDbDataReader reader = cmd.ExecuteReader())
            {
               while (reader.Read())
```

```
                {
                    Console.WriteLine("{0}\t\t{1}\t\t{2}",
                        reader["name"].ToString(),
                        reader["address"].ToString(),
                        reader["salary"].ToString());
                }
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
        finally
        {
            conn.Close();
        }

        Console.ReadKey();
    }
  }
}
```

Output:



file:///C:/Users/S342/documents/visual studio 2010/Projects/theciya/theciya/bin/Debug/theciya.EXE

```
person name     address          salary
===========================================
------------------------------
theciya         papanasam               30000
bramma          thanjore                50000
tamil           krishanakiri            60000
nisha           padukoottai             30000
snekki          kumbakonam              40000
gayu            thanjore                50000
atchu           kadampatti              20000
```

RESULT:

Thus, to build a C# .NET Windows application and MS Access database connection has been verified.

Ex.No:6 Insert Update Delete Select Search Operation Using Oledb Connection

AIM:

To create a C# .NET Windows Forms application to perform insert, update, delete, and select operations using the OleDbConnection object.

ALGORITHM:

1 Create a new project -> Windows Application -> Name -> OK

2 Design your form with necessary labels and pictures

3 From the toolbox, select the "DataGridView" control and place it on the form

4 Select the database and select the dataset, click *Next*, click *New Connection*, and click *Change* button. Then select *Microsoft Access Data Source* -> OK button

5 Click *Test Connection* button and click *OK*

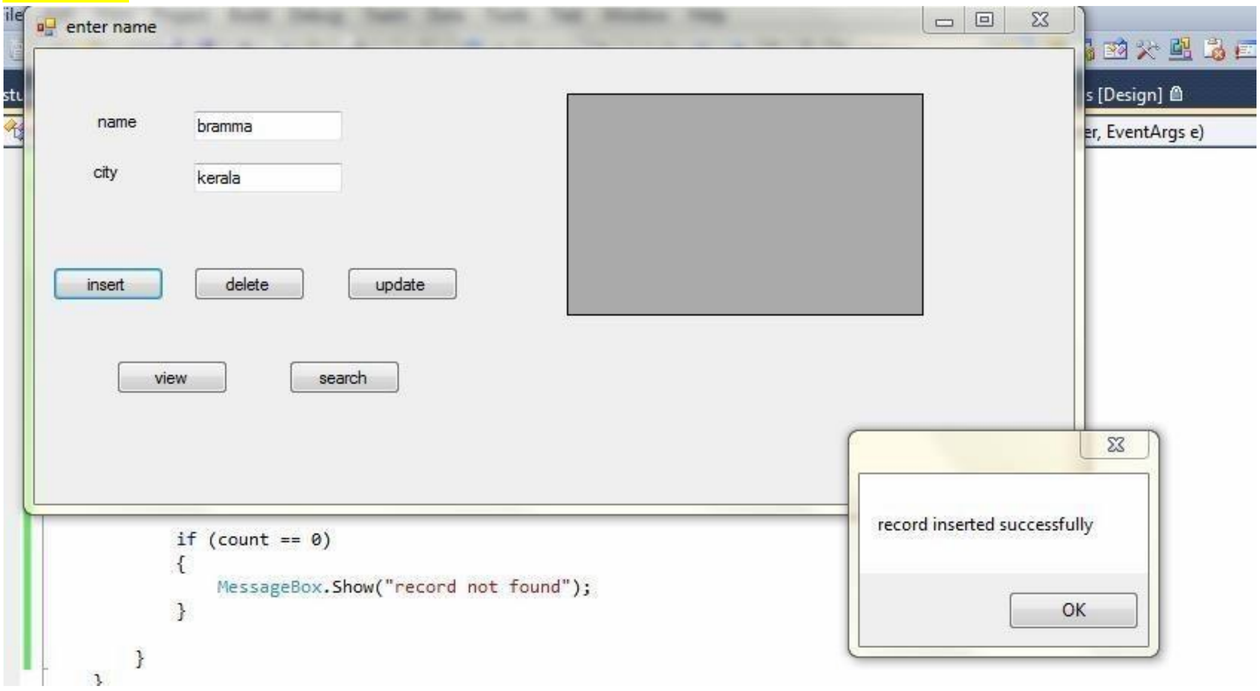
6 Run the application

7 The result will be displayed on the form

PROGRAM

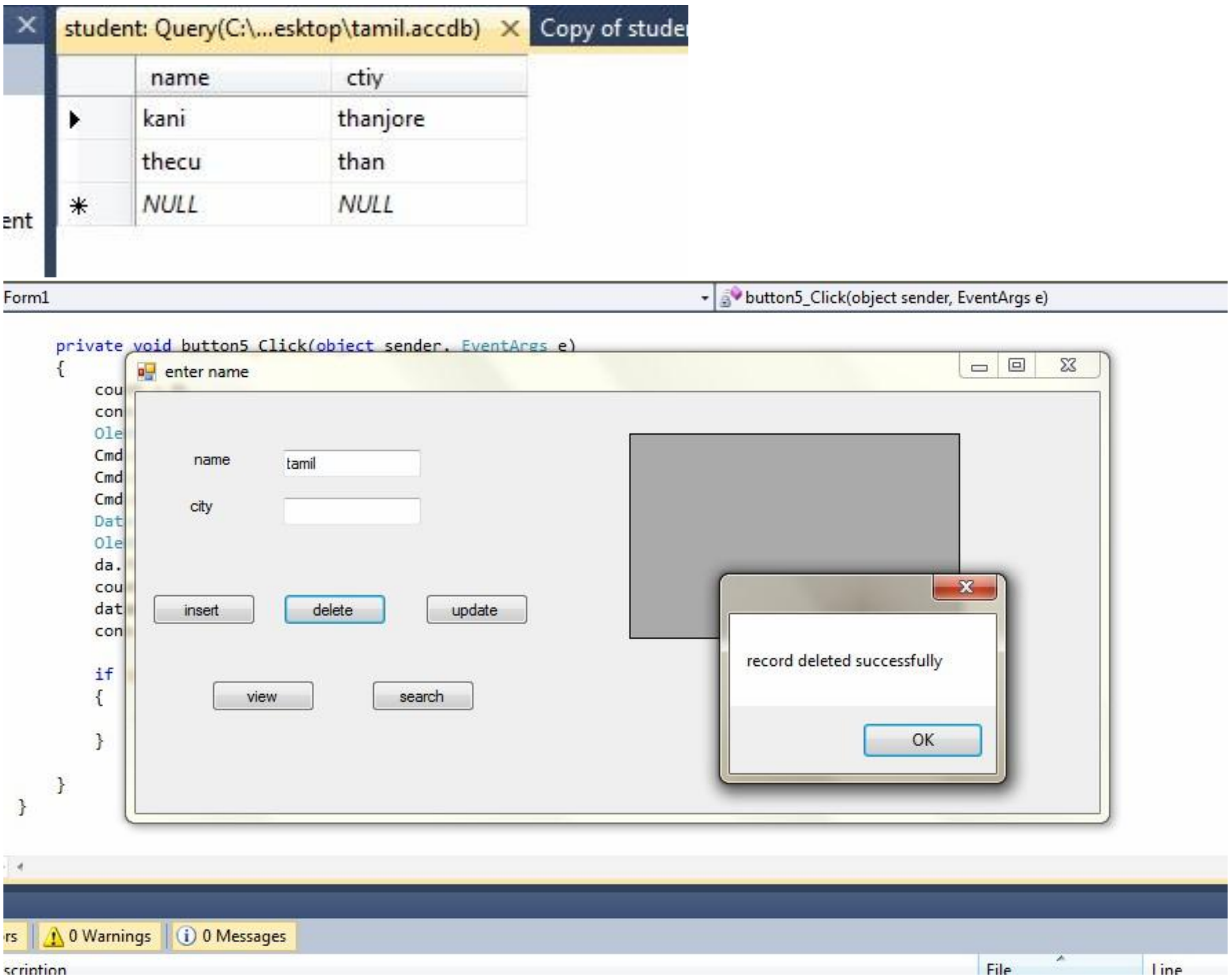
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.OleDb;

namespace thecu
{
   public partial class Form1 : Form
   {
      int count = 0;
      OleDbConnection conn = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=C:\\Users\\S342\\Desktop\\tamil.accdb");

      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         conn.Open();
```

```
            OleDbCommand Cmd = conn.CreateCommand();
            Cmd.CommandType = CommandType.Text;
            Cmd.CommandText = "insert into student values('" + textBox1.Text + "','" + textBox2.Text + "')";
            Cmd.ExecuteNonQuery();
            conn.Close();
            MessageBox.Show("Record inserted successfully");
        }

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

        private void button4_Click(object sender, EventArgs e)
        {
            conn.Open();
            OleDbCommand Cmd = conn.CreateCommand();
            Cmd.CommandType = CommandType.Text;
            Cmd.CommandText = "select * from student";
            Cmd.ExecuteNonQuery();
            conn.Close();
            DataTable dt = new DataTable();
            OleDbDataAdapter da = new OleDbDataAdapter(Cmd);
            da.Fill(dt);
            dataGridView1.DataSource = dt;
            MessageBox.Show("Record viewed successfully");
        }

        private void button2_Click(object sender, EventArgs e)
        {
            conn.Open();
            OleDbCommand Cmd = conn.CreateCommand();
            Cmd.CommandType = CommandType.Text;
            Cmd.CommandText = "delete from student where name='" + textBox1.Text + "'";
            Cmd.ExecuteNonQuery();
            conn.Close();
            MessageBox.Show("Record deleted successfully");
        }

        private void button3_Click(object sender, EventArgs e)
        {
            conn.Open();
            OleDbCommand Cmd = conn.CreateCommand();
            Cmd.CommandType = CommandType.Text;
            Cmd.CommandText = "update student set name='" + textBox2.Text + "' where name='" +
textBox1.Text + "'";
            Cmd.ExecuteNonQuery();
            conn.Close();
            MessageBox.Show("Record updated successfully");
        }

        private void button5_Click(object sender, EventArgs e)
        {
            count = 0;
            conn.Open();
            OleDbCommand Cmd = conn.CreateCommand();
            Cmd.CommandType = CommandType.Text;
            Cmd.CommandText = "select * from student where name='" + textBox1.Text + "'";
            Cmd.ExecuteNonQuery();
            DataTable dt = new DataTable();
            OleDbDataAdapter da = new OleDbDataAdapter(Cmd);
            da.Fill(dt);
            count = Convert.ToInt32(dt.Rows.Count.ToString());
            dataGridView1.DataSource = dt;
            conn.Close();
            if (count == 0)
            {
                MessageBox.Show("Record not found");
            }
        }
    }
}
```
OUTPUT



record inserted successfully

```
            if (count == 0)
            {
                MessageBox.Show("record not found");
            }

        }
```



student: Query(C:\...esktop\tamil.accdb)

	name	ctiy
▶	kani	thanjore
	thecu	than
*	NULL	NULL

Form1 | button5_Click(object sender, EventArgs e)

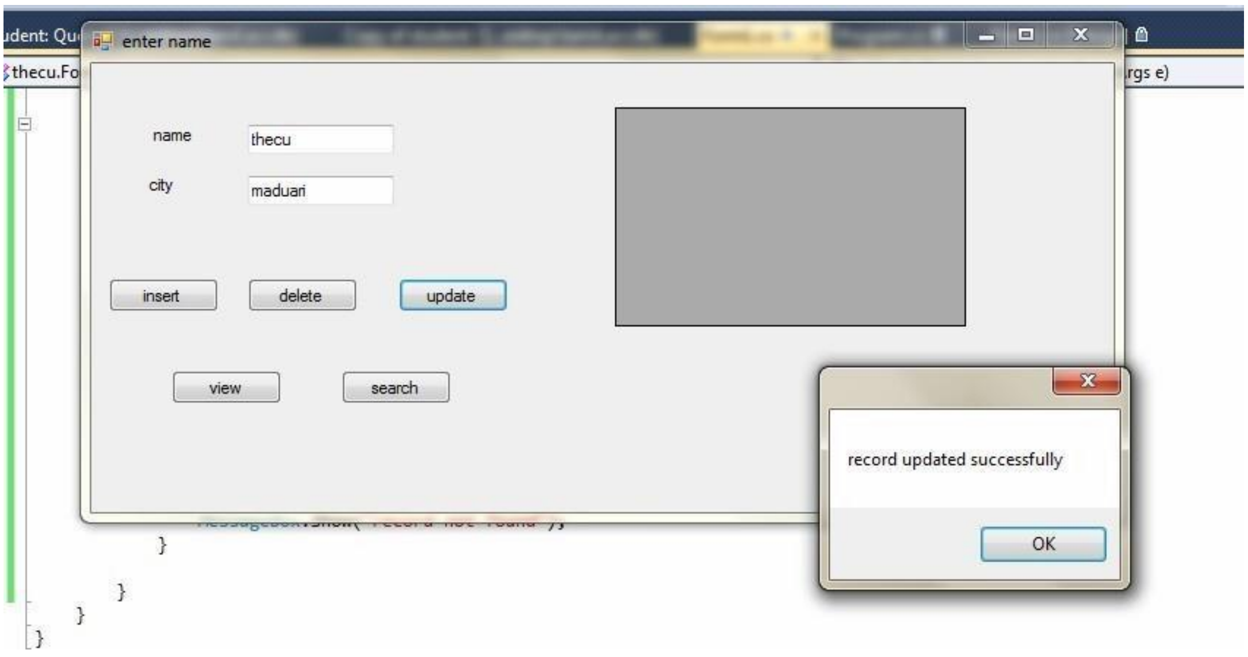
```
private void button5_Click(object sender, EventArgs e)
{
```

enter name

name: tamil

city:

record deleted successfully

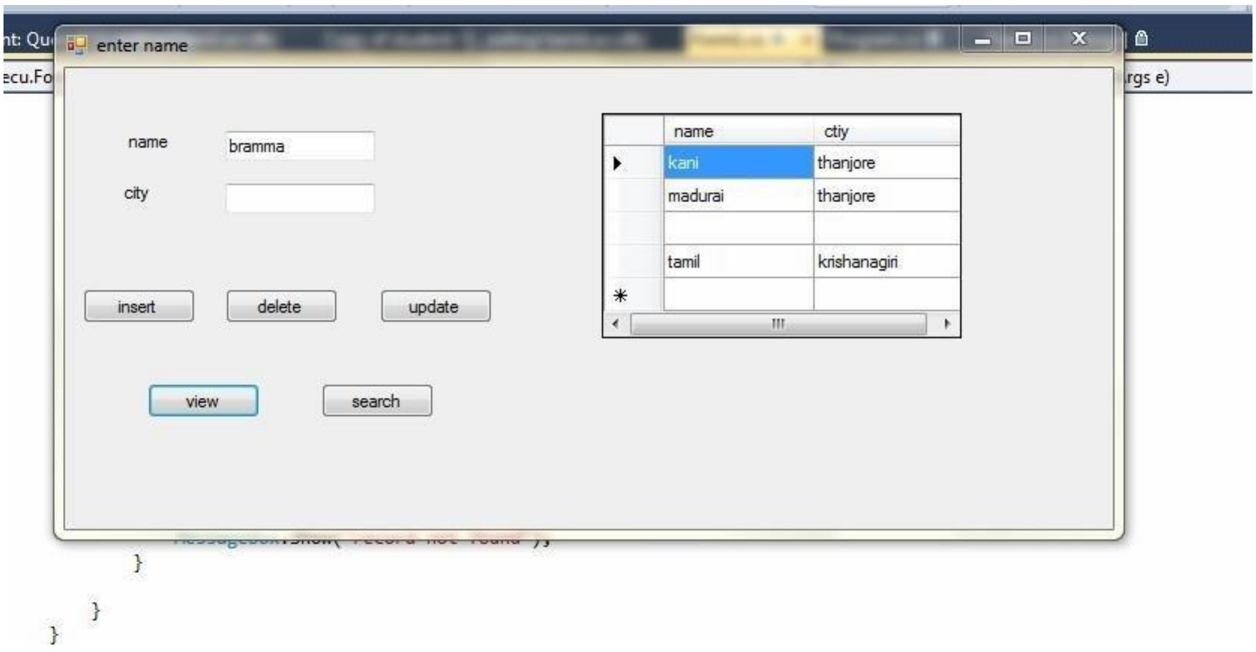


enter name

name: thecu

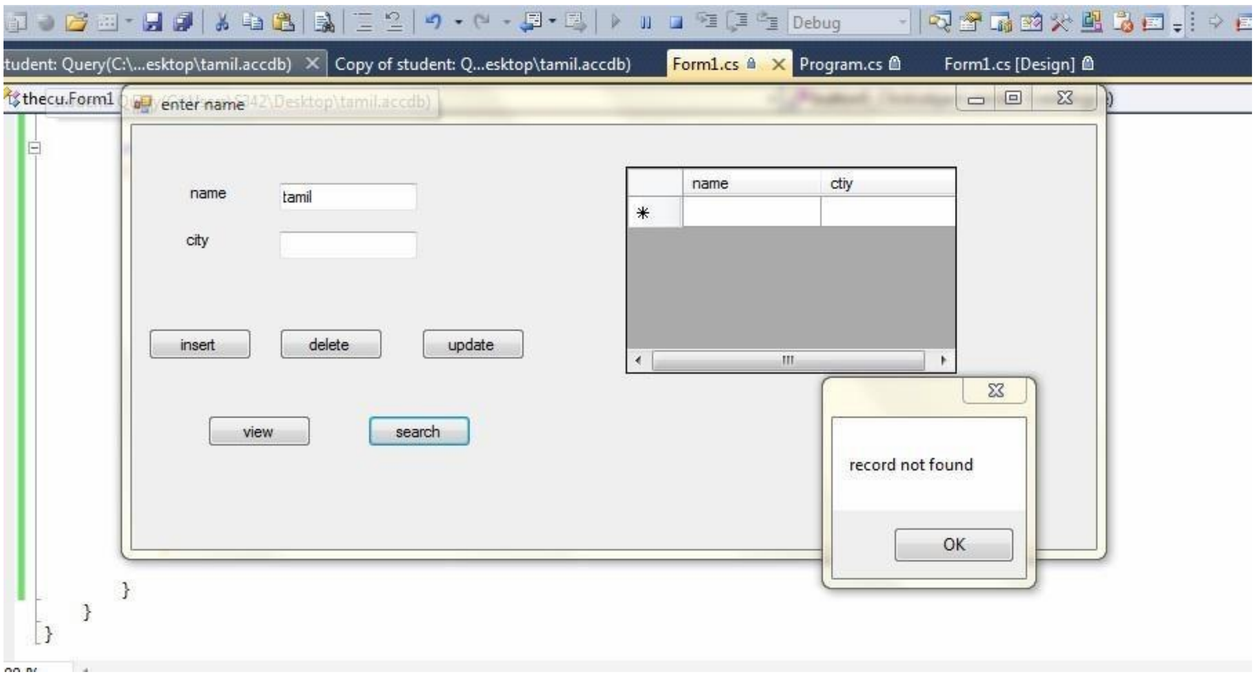
city: maduari

record updated successfully



	name	ctiy
▶	kani	thanjore
	madurai	thanjore
	tamil	krishanagiri
*		

```
            }
        }
    }
```



tudent: Query(C:\...esktop\tamil.accdb) Copy of student: Q...esktop\tamil.accdb) Form1.cs Program.cs Form1.cs [Design]

	name	ctiy
*		

record not found

```
            }
        }
    }
```

RESULT:

Thus, to create a C# .NET Windows Forms application to insert, update, delete, and select operations in OleDb Connection object has been verified

Ex.No:7 Working with various Controls in ASP .NET

AIM:

To create ASP.NET web application using server controls.

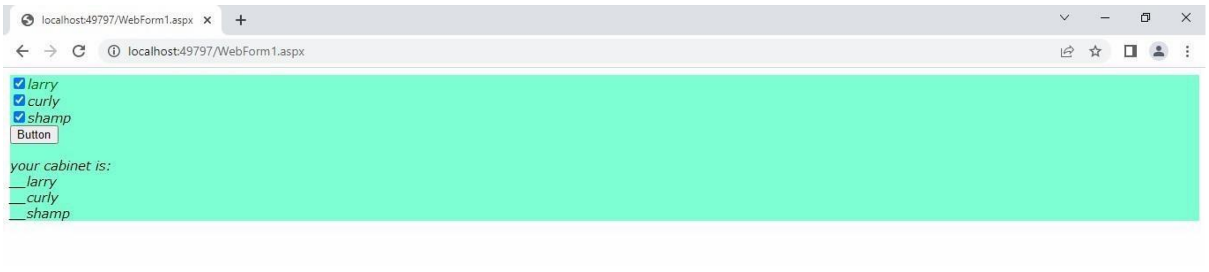
ProgramCoding

```
<%@PageLanguage="C#"AutoEventWireup="true"CodeBehind="WebForm1.aspx.cs"Inherits="WebApplication2.WebForm1

<!DOCTYPEhtml>

<htmlxmlns="http://www.w3.org/1999/xhtml">
<headrunat="server">
    <title></title>
    <styletype="text/css">#form1
        { font-style: italic; font-
            family:Verdana; font-
            size:11pt;
            background-color:aquamarine;

        }
    </style>
</head>
<body><div>
    <formid="form1"runat="server">
    &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nb
    sp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;


    Using Web Server Controls<br />


        <asp:CheckBoxID="CheckBox1"runat="server"Text="larry"BorderColor="#CC99FF"ForeColor="#006600" />
        <br/>

        <asp:CheckBoxID="CheckBox2"runat="server"Text="curly"OnCheckedChanged="CheckBox2_CheckedChanged" />
        <br/>

        <asp:CheckBoxID="CheckBox3"runat="server"Text="shamp"/><br/>

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


    </p>
    </form>

    </div>
    <p>

        <asp:ButtonID="Button1"runat="server"OnClick="Button1_Click1"Text="Button"/> </p>
</body>
</html>
```

C# CBox.aspx.cs

```csharp
using System;

public partial class Cbox : System.Web.UI.Page
{
    private string cabinet;

    protected void Button1_Click(object sender, EventArgs e)
    {
        cabinet = "Your cabinet is:<br/>";
        cabinet += CheckBox1.Checked == true ? "-" + CheckBox1.Text + "<br/>" : null;
        cabinet += CheckBox2.Checked == true ? "-" + CheckBox2.Text + "<br/>" : null;
        cabinet += CheckBox3.Checked == true ? "-" + CheckBox3.Text + "<br/>" : null;
        cabinet += CheckBox4.Checked == true ? "-" + CheckBox4.Text + "<br/>" : null;

        Label1.Text = cabinet;
    }
}
```

OUTPUT:



localhost:49797/WebForm1.aspx

```
☑ larry
☑ curly
☑ shamp
Button

your cabinet is:
__larry
__curly
__shamp
```

RESULT:

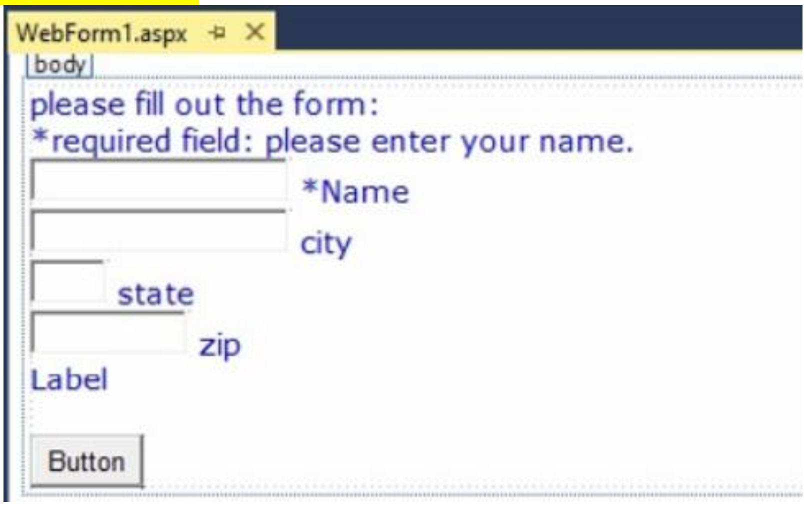
Thus, to create an ASP.NET web application using web server controls has been developed successfully.

Ex.No:8 Using Validation Controls

Aim:

To create ASP.NET web application using validation controls.

FORMDESIGN:


WebForm1.aspx

body

please fill out the form:
*required field: please enter your name.

*Name
city
state
zip
Label
Button

PROGRAM :

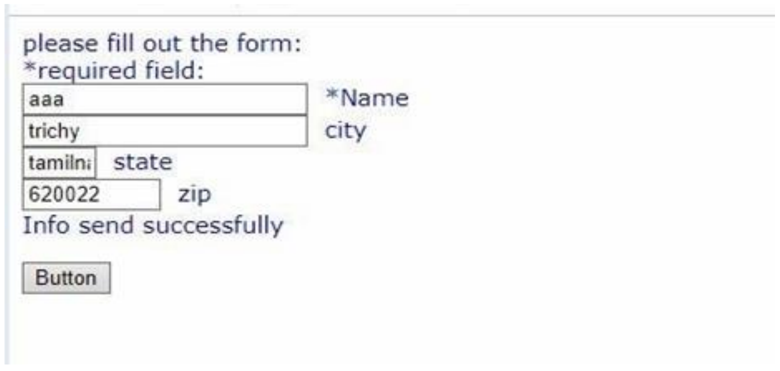
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs" Inherits="exe7.WebForm1"
%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <style type="text/css">
    div {
      font-family: verdana;
      font-size: 11pt;
      color: #0000cc;
    }
  </style>
  <title>Required Field Validation</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      Please fill out the form:<br/>
      * Required field:
      <asp:RequiredFieldValidator
ID="requiredFieldValidator2" runat="server"
ControlToValidate="textbox1" ErrorMessage="Please enter
your name." /><br/>
```

```
        <asp:TextBox ID="textbox1" runat="server" />&nbsp;*
Name<br/>
        <asp:TextBox ID="textbox2" runat="server"
/>&nbsp;City<br/>
        <asp:TextBox ID="textbox3" runat="server"
Width="38px" />&nbsp;State<br/>
        <asp:TextBox ID="textbox4" runat="server"
Width="78px" />&nbsp;Zip<br/>
        <asp:Label ID="Label1" runat="server"
Text="Label"></asp:Label>
      </div>
      <p>
        <asp:Button ID="Button1" runat="server"
OnClick="Button1_Click" Text="Submit" />
      </p>
    </form>
</body>
</html>
```

C#Code:

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

namespaceexe7
{
publicpartialclassWebForm1:System.Web.UI.Page
      { protectedvoidButton1_Click(objectsender,EventArgse)
            {
                  Label1.Text="Infosendsuccessfully";
            }
      }
```

OUTPUT:



```
please fill out the form:
*required field:
aaa                  *Name
trichy               city
tamilni  state
620022     zip
Info send successfully

Button
```

RESULT:

Thus, creating an ASP.NET web application using validation controls has been verified successfully.

Ex.No:9 Using stored Procedures

Aim:

To create SQL Server Stored Procedures by declaring parameters in an ASP.NET Web application..

ALGORITHM:

1 First, open **Microsoft SQL Server Management Studio** (SSMS).
2 Then, navigate to the database in which you want to create the stored procedure.
3 Select **New Stored Procedure**, then select **Stored Procedure Properties** for what to enter, and then click **OK**.
4 Now create an application named **Store Procedure** in .NET to use the above stored procedures.
5 Display the output. Stop the execution..

Declaring Parameters in SQL Server stored Procedures:

1. The name
2. The datatype
3. The default value
4. The direction (INPUT, OUTPUT, or INOUT)

The syntax is

@parameter_name [AS] datatype [= default | NULL] [VARYING] [OUTPUT | OUT]

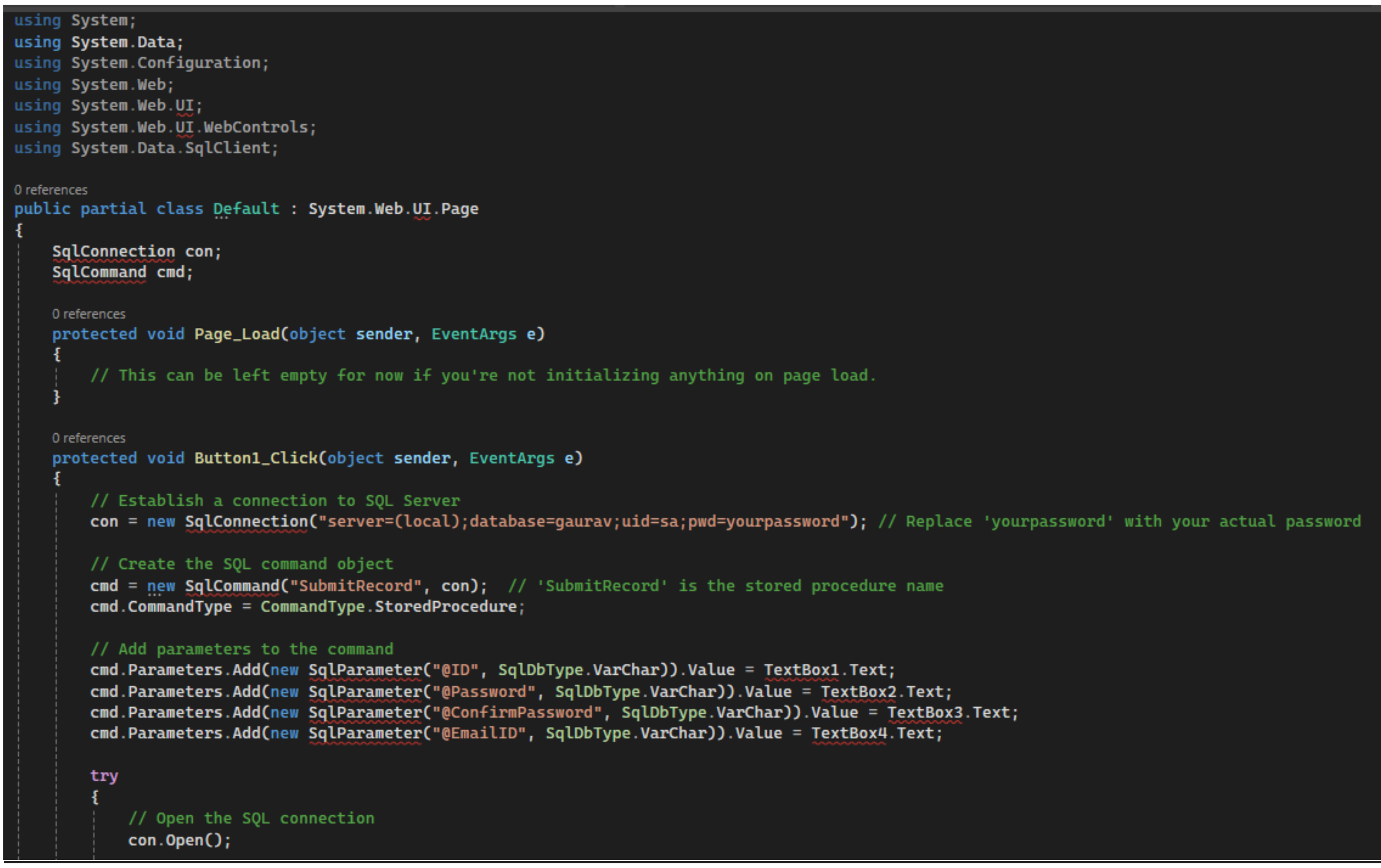
PROGRAM

Stored Procedure .aspx page code

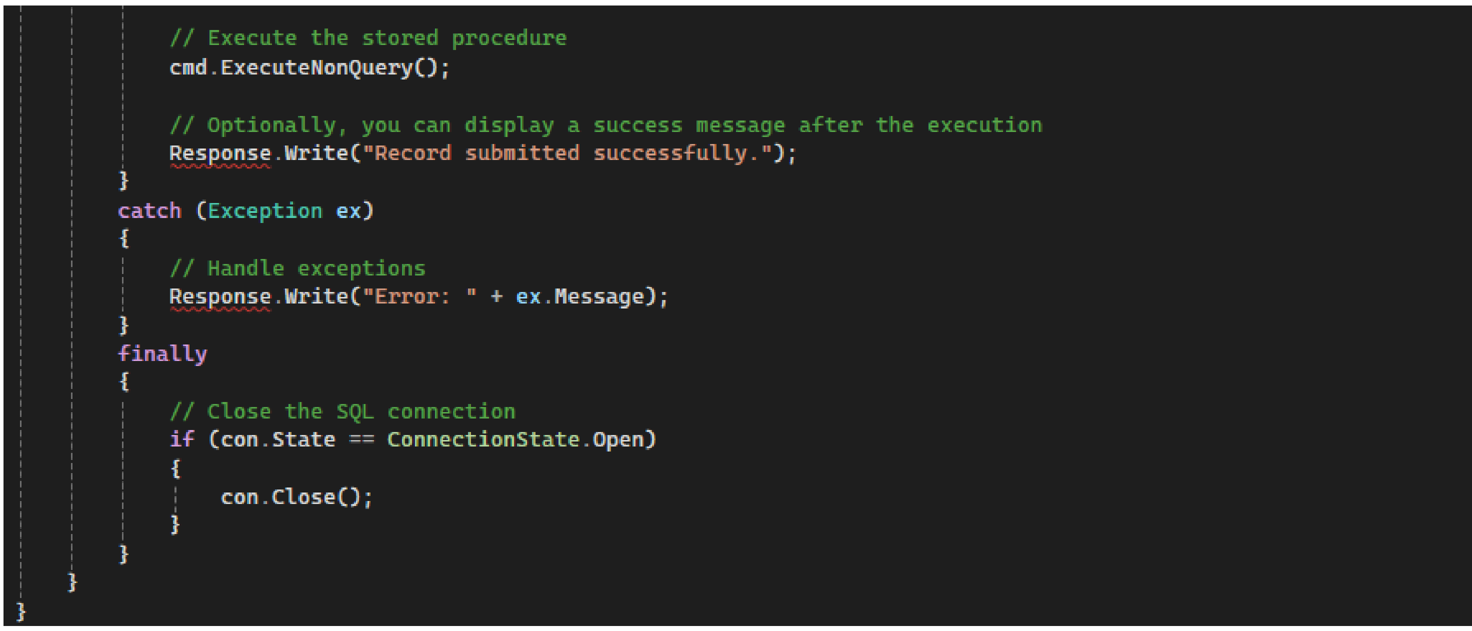
```
<%@PageLanguage="C#"AutoEventWireup="true"CodeFile="Default.aspx.cs"Inherits="_Defaul t"%>

<!DOCTYPE html>

<htmlxmlns="http://www.w3.org/1999/xhtml">

    <headrunat="server">

        <title>StoreProcedure</title>

    </head>

    <body>

        <formid="form1"runat="server">

            <div>

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

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

```
                    <asp:LabelID="Label2"runat="server"Text="Password"></asp:Label>

                    <asp:TextBoxID="TextBox2"runat="server"></asp:TextBox><br/><br/>


                    <asp:LabelID="Label3"runat="server"Text="ConfirmPassword"></asp:Label>

                    <asp:TextBoxID="TextBox3"runat="server"></asp:TextBox><br/><br/>


                    <asp:LabelID="Label4"runat="server"Text="EmailID"></asp:Label>

                    <asp:TextBoxID="TextBox4"runat="server"></asp:TextBox <br/><br/>


                <asp:ButtonID="Button1"runat="server"Text="SubmitRecord"OnClick="Button1_Click"/>
            </div>
        </form>

</html>
</body>
```

Stored Procedure .aspx.cs page code

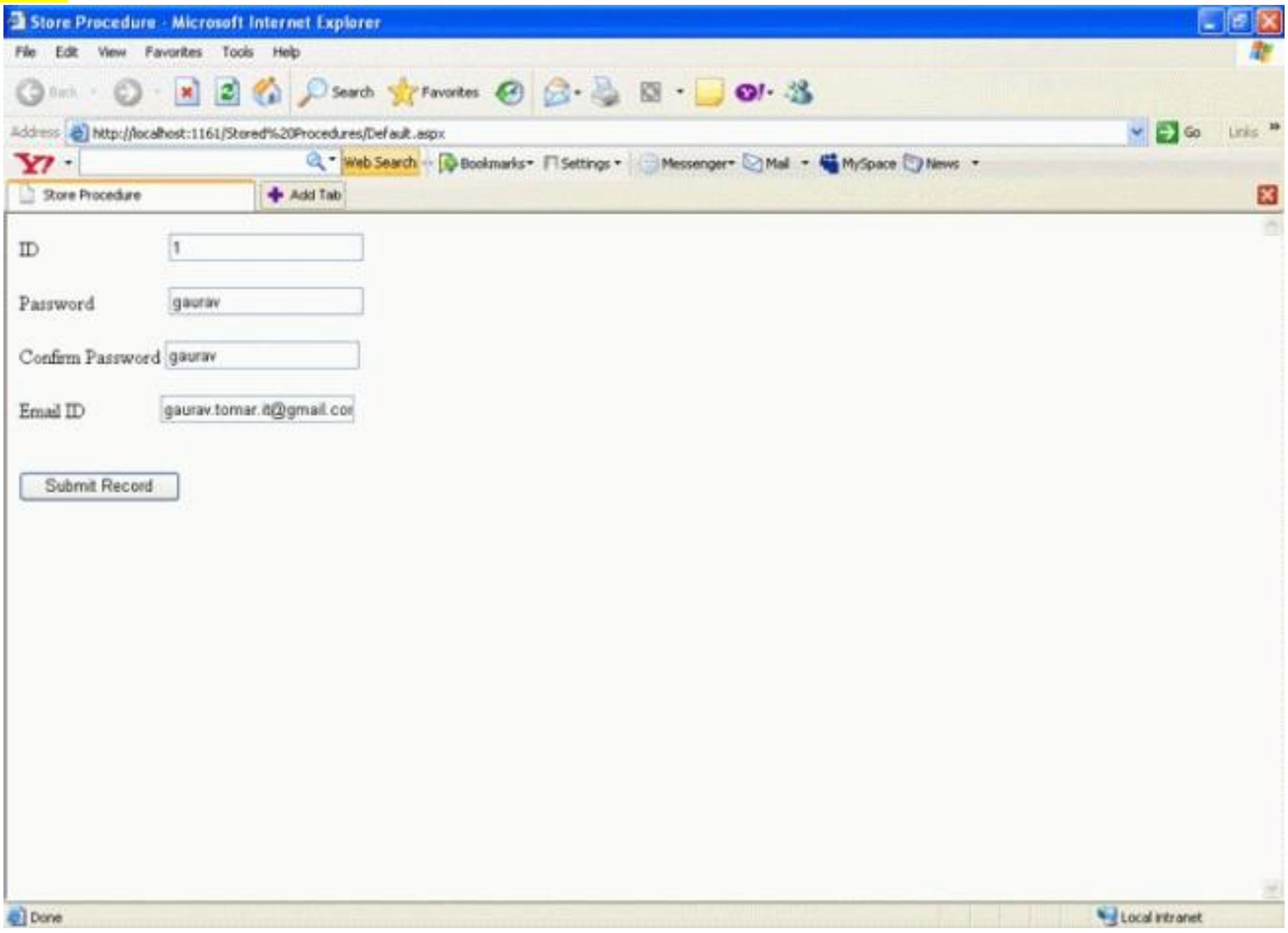


```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;

0 references
public partial class Default : System.Web.UI.Page
{
    SqlConnection con;
    SqlCommand cmd;

    0 references
    protected void Page_Load(object sender, EventArgs e)
    {
        // This can be left empty for now if you're not initializing anything on page load.
    }

    0 references
    protected void Button1_Click(object sender, EventArgs e)
    {
        // Establish a connection to SQL Server
        con = new SqlConnection("server=(local);database=gaurav;uid=sa;pwd=yourpassword"); // Replace 'yourpassword' with your actual password

        // Create the SQL command object
        cmd = new SqlCommand("SubmitRecord", con);  // 'SubmitRecord' is the stored procedure name
        cmd.CommandType = CommandType.StoredProcedure;

        // Add parameters to the command
        cmd.Parameters.Add(new SqlParameter("@ID", SqlDbType.VarChar)).Value = TextBox1.Text;
        cmd.Parameters.Add(new SqlParameter("@Password", SqlDbType.VarChar)).Value = TextBox2.Text;
        cmd.Parameters.Add(new SqlParameter("@ConfirmPassword", SqlDbType.VarChar)).Value = TextBox3.Text;
        cmd.Parameters.Add(new SqlParameter("@EmailID", SqlDbType.VarChar)).Value = TextBox4.Text;

        try
        {
            // Open the SQL connection
            con.Open();
```



```
            // Execute the stored procedure
            cmd.ExecuteNonQuery();

            // Optionally, you can display a success message after the execution
            Response.Write("Record submitted successfully.");
        }
        catch (Exception ex)
        {
            // Handle exceptions
            Response.Write("Error: " + ex.Message);
        }
        finally
        {
            // Close the SQL connection
            if (con.State == ConnectionState.Open)
            {
                con.Close();
            }
        }
    }
}
```

OUTPUT:


Store Procedure - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://localhost:1161/Stored%20Procedures/Default.aspx
Store Procedure
Done
Local intranet

ID 1

Password gaurav

Confirm Password gaurav

Email ID gaurav.tomar.it@gmail.com

Submit Record

After clicking the submit button the data is appended to the database as seen below in the SQL Server table record:

Result:

To create a SQL Server Stored Procedure declaring parameters in ASP.NET Web application has been verified.

Ex.No:10 Using Required Field Validation

Aim:

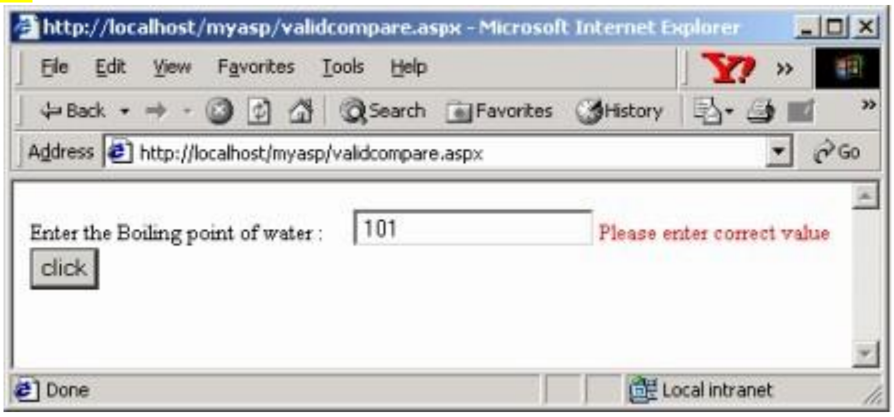
TocreateprogramusingReuiredFieldValidationcontrolinASP.NETWebapplication.

PROGRAM CODING:

```
<html>
<body>
<formrunat="server">
<asp:labelid="label" text="EntertheBoilingpointofwater:" runat="server"
/>&nbsp;&nbsp;&nbsp;&nbsp;
<asp:textboxid="text1"text=""runat="server"/>
<asp:comparevalidatorid="compboilpt"controltovalidate="text1"Type="Integer"
 ValueToCompare=100Operator="Equal"display="static"errormessage="Please enter
correct value" runat="server">
</asp:comparevalidator>
```

<asp:ValidationSummaryid=”sumErrors”runat=”server”
showSummary = true displayMode=“BulletList”
/>
<br>
<asp:buttonid=bt1runat=”server”text=”click”/>
</form>
</body>
</html>Exercise
: 1
LabSolutions
CentreforInformationTechnologyandEngineering,ManonmaniamSundaranarUniversity21 <html>
<body>
<h3>RequiredFieldValidation</h3>
<formrunat=server>
Name:<asp:Textboxid=”txtName” runat=”server”></asp:Textbox>
<asp:buttonid=”Button1"runat=”server”text=”Validate”/>
<p>
<asp:RequiredFieldValidatorid=”RequiredFieldValidator1"runat=”server”
ControlToValidate=”txtName”
ErrorMessage=”Nameisarequiredfield”
ForeColor=”Red”>
</asp:RequiredFieldValidator>
</form>
</body>
</html>

OUTPUT:


http://localhost/myasp/validcompare.aspx - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost/myasp/validcompare.aspx Go
Done
Local intranet

Enter the Boiling point of water : 101 Please enter correct value
click

Result:

ThustocreateprogramusingReuiredFieldValidation controlinASP.NETWebapplicationhasbeen verified successully.